802.15.3b™

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 Computer Society

Sponsored by the
LAN/MAN Standards Committee

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

5 May 2006

IEEE Std 802.15.3b™-2005
(Amendment to
IEEE Std 802.15.3™-2003)

Recognized as an
American National Standard (ANSI)

IEEE Std 802.15.3b™-2005
(Amendment to
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):

Amendment 1: MAC Sublayer

Sponsor

LAN/MAN Standards Committee
of the
IEEE Computer Society

Approved 28 March 2006

American National Standards Institute

Approved 6 December 2005

IEEE-SA Standards Board

Abstract: This amendment provides corrections and optimizations to IEEE Std 802.15.3-2003. The management service access points (SAPs) have been completely updated to create a consistent, logical interface. As a consequence, most of the message sequence charts (MSCs) in the medium access control (MAC) functional specification have been updated as well. Channel time usage is more efficient with the addition of multiple contention periods, releasing channel time, implied acknowledgment, and mullticast groups.
Keywords: channel time reuse, handover, implied acknowledgment, multicast, wireless personal area network, WPAN

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

Print: ISBN 0-7381-4872-5 SH95506
PDF: ISBN 0-7381-4873-3 SS95506

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 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 the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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. 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 addressed to:

Secretary, IEEE-SA Standards Board

445 Hoes Lane

Piscataway, NJ 08854

USA

NOTE—Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Introduction

This introduction is not part of 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 802.15.3b-2005 is an amendment to IEEE Std 802.15.3-2003 that fixes mistakes, clarifies the intent of the original standard, and adds enhancements that improve the efficiency of the base standard. The continuing goal of this standard is to enable low-cost, low-power, high-speed wireless multimedia applications for portable and stationary consumer electronic devices. In addition to the minor corrections, some of the key changes/additions in this amendment are as follows:

— A new definition of the medium access control (MAC) layer management entity (MLME) service access point (SAP) so that it now provides a consistent interface.

— A new acknowledgment (ACK) policy, implied-ACK, which allows polling and a more efficient use of channel time.

— An additional data frame, the logical link control/subnetwork access protocol (LLC/SNAP), which allows multiple protocols to share a single data connection.

— A method for relinquishing time in a channel time allocation (CTA) to allow another device (DEV) time to transmit data.

— The allowing of a DEV to return information about a signal quality of a received packet in an ACK frame.

— The ability to assign device identifiers (DEVIDs) to group addresses to allow multicast connections.

— Faster recovery of network operations when the piconet coordinator (PNC) abruptly disconnects with the conditional handover and the next PNC processes.

— A protocol by which a DEV can request that the PNC place an application-specific information element (ASIE) in the beacon.

— The allowing of multiple contention periods during a superframe.

Interest in working on an amendment to fix and enhance IEEE Std 802.15.3 began in the September 2003 meeting in Singapore. A study group was formed at the November 2003 meeting in Albuquerque, which completed a project authorization request that was approved in March 2004. Work progressed quickly, and the first task group letter ballot was successfully completed in March 2005. After one recirculation, the draft began sponsor ballot in August 2005. The ballot was successful, and after a recirculation ballot to validate some minor changes, IEEE Std 802.15.3b was approved by the IEEE Standards Board on 6 December 2005, just over two years after the study group started.

Notice to users

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 standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents or patent applications for which a license may be required to implement an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Conformance test methodology

An additional standards series, identified by the number 1802™, has been established to identify the conformance test methodology documents for the IEEE 802® family of standards. Thus the conformance test documents for IEEE 802.3™ are numbered 1802.3™, the conformance test documents for IEEE 802.5™ will be 1802.5™, and so on. Similarly, ISO will use 18802 to number conformance test standards for 8802 standards.

Participants

At the time this standard was completed, the IEEE 802.15 Working Group had the following officers:

Robert F. Heile, *Chair*
James D. Allen, *Vice-Chair*
Patrick Kinney, *Vice-Chair/Secretary*
James P. K. Gilb, *Technical Editor*

When the IEEE 802.15 Working Group approved this standard, Task Group 3b had the following membership:

John R. Barr, *802.15.3b Chair*
James D. Allen, *802.15.3b Vice Chair*
James P. K. Gilb, *802.15.3b Technical Editor*
John Sarallo, *802.15.3b Assistant Editor*

Roberto Aiello
Richard Alfvin
Kyu Hwan An
Mikio Aoki
Yasuyuki Arai
Takashi Arita
Larry Arnett
Naiel Askar
Arthur Astrin
Yasaman Bahreini
Jay Bain
Feng Bao
Anuj Batra
Phil Beecher
Bruce Bosco
Monique Bourgeois Brown
Mark Bowles
Charles Brabenac
David Brenner
Vern Brethour
Ronald Brown
Ed Callaway
Pat Carson
Kisoo Chang
Jonathon Cheah
Francois Chin
Kwan-Wu Chin
Yu-Chang Chiu
Sarm-Goo Cho
Sangsung Choi
Chia-Chin Chong
Manoj Choudhary
Celestino Corral
Robert Charles Cragie
Scott Davis
Joe Decuir
Javier Del Prado Pavon
Kai Dombrowski
Stefan Drude
Eryk Dutkiewicz
Michael Dydyk
Jason Ellis
Shahriar Emami
Yew Soo Eng
Frederick Enns
Mark W. Fidler
Reed Fisher
Kristoffer Fleming
Ricardo Gandia Sanchez
Ian Gifford
Sung-Wook Goh
Sorin Goldenberg
Paul Gorday
Bernd Grohmann
Rainer Hach
Julian Hall
Robert Hall
Shinsuke Hara
Jeff Harris
Allen Heberling

Robert Heile
Barry Herold
Karl Heubaum
Jin-Meng Ho
Patrick Houghton
Chi-Hao Huang
Robert Huang
Xiaojing Huang
Akira Ikeda
Tetsushi Ikegami
Oyvind Janbu
Yeong Min Jang
Adrian Jennings
Ho-In Jeon
Hong Jiang (Chiang)
Peter Johansson
Jeyhan Karaoguz
Michael Kelly
Stuart Kerry
Haksun Kim
Jae-Hyon Kim
Jaeyoung Kim
Yongsuk Kim
Young Hwan Kim
Youngsoo Kim
Kursat Kimyacioglu
Matthias Kindler
Patrick Kinney
Guenter Kleindl
Ryuji Kohno
Yasushi Kudo
Thomas Kuehnel
Haim Kupershmidt
Yuzo Kuramochi
Kyung Sup Kwak
Do-Hoon Kwon
Ismail Lakkis
John Lampe
Colin Lanzl
Dongjun Lee
Hyung Soo Lee
Kyung Kuk Lee
Myung Lee
Wooyong Lee
David Leeper
Israel Leibovich
Henry Li
Huan-Bang Li
Liang Li
Haixiang Liang
Darryn Lowe
Ian Macnamara
Tadahiko Maeda
Akira Maeki
Patricia Martigne
Frederick Martin
Abbie Mathew
Taisuke Matsumoto
Masafumi Matsumura
Michael McLaughlin
John McCorkle
Michael McInnis
Charlie Mellone
Jim Meyer
Klaus Meyer
Akira Miura
Hitoshi Miyasaka
Samuel Mo
Andreas Molisch
Mark Moore
Marco Naeve
Ken Naganuma
Yves Paul Nakache
Hiroyuki Nakase
Chiu Ngo
Erwin Noble
Mizukoshi Nobuyuki
Masaki Noda
Richard Noens
John (Jay) O'Conor
Knut Odman
Yasuyuki Okuma
Philip Orlick
Laurent Ouvry
John Pardee
Jonghun Park
Young Jin Park
Dave Patton
Miguel Pellon
Xiaoming Peng
Robert Poor
Clinton Powell
Vidyasagar Premkumar
Yihong Qi
Raad Raad
Ajay Rajkumar
Pekka Ranta
Dani Raphaeli
Gregg Rasor
Charles Razzell
Joseph Reddy
Ivan Reede
Yuko Rikuta
Benno Ritter
Terry Robar
Richard Roberts
Martin Rofheart
Jaeho Roh
Philippe Rouzet
Chandos Rypinski
Saeid Safavi
Zafer Sahinoglu
Tomoki Saito
Sidney Schrum
Erik Schylander
Alireza Seyedi
Sanjeev Sharma
Stephen Shellhammer
John (Chih-Chung) Shi
Shusaku Shimada
Yuichi Shiraki
Matthew Shoemake
Gadi Shor
William Shvodian
Thomas Siep
Michael Sim
Kazimierz Siwiak
Zachary Smith
V. Somayazulu
Carl Stevenson
Marinus Struik
Kazuaki Takahashi
Kenichi Takizawa
Teik-Kheong Tan
Mike Tanahashi
James Taylor
John Terry
Arnaud Tonnerre
Jarvis Tou
Jerry Upton
Robin Vaitonis
Bart Van Poucke
Nanci Vogtli
Jerry Wang
Jing Wang
Chris Weber
Matthew Welborn
Richard Wilson
Gerald Wineinger
Andreas Wolf
Marcus Wong
Timothy G. Wong
Patrick Worfolk
Tracy Wright
Xiaodong Wu
Yu-Ming Wu
Hirohisa Yamaguchi
Kamya Yekeh Yazdandoost
Su-Khiong Yong
Serdar Yurdakul
Honggang Zhang
Frank Xiaojun Zheng
Zheng Zhou
Chunhui Zhu

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

Toru Aihara
James D. Allen
Butch Anton
Mikio Aoki
Michael P. Baldwin
Charles L. Barest
John B. Barnett
John R. Barr
Gennaro Boggia
Matthew K. Burnburg
William A. Byrd
Sean S. Cai
James T. Carlo
Juan C. Carreon
Jay Catelli
Yi-Ming Chen
Danila Chernetsov
Aik Chindapol
Todor V. Cooklev
Tommy P. Cooper
Javier Del Prado Pavon
Thomas J. Dineen
Sourav K. Dutta
Carl Eklund
Marc Emmelmann
John W. Eng
Avraham Freedman
John N. Fuller
Devon L. Gayle
Michael D. Geipel
Fernando Genkuong
Theodore Georgantas
Ian C. Gifford
James P. K. Gilb
Nikhil Goel
Sergiu R. Goma
Randall C. Groves
Qiang Guo
C. G. Guy
Siamack Haghighi
Samudra E. Haque
Allen D. Heberling
Karl F. Heubaum
Shui H. Heung
Dennis Horwitz
Lalit R. Kotecha
Jeremy A. Landt
Colin Lanzl
Solomon Lee
Tsungyu Lee
Charles A. Lennon Jr
Daniel G. Levesque
Jan-Ray Liao
William Lumpkins
G. L. Luri
Steve Ma
Ben L. Manny
Kevin D. Marquess
Jonathon C. Mclendon
Francisco J. U. Melendez
George J. Miao
Gary L. Michel
William J. Mitchell
Ronald G. Murias
M. Narayanan
Michael S. Newman
Paul Nikolich
Erwin R. Noble
Robert O'Hara
Knut T. Odman
Satoshi Oyama
Glenn W. Parsons
Sunil Parthasarathy
Subburajan Ponnuswamy
Cam K. Posani
Vikram Punj
Jose P. Puthenkulam
Philip T. Robinson
John C. Sargent III
Michael Scholles
Stephen C. Schwarm
Perry L. Schwartz
George G. She
William M. Shvodian
Thomas M. Siep
Zachary B. Smith
Jung Je Son
Amjad A. Soomro
Robert T. Soranno
Thomas E. Starai
Mark A. Sturza
Joseph J. Tardo
David W. Thompson
Thomas A. Tullia
Mark-Rene Uchida
Thomas M. Wandeloski
Stanley S. Wang
Hung-Yu Wei
Jason J. Wilden
Chaehag Yi
Takahito Yoshizawa
Oren Yuen
Janusz Zalewski

Major technical contributions were received from the following individuals

James D. Allen
John R. Barr
Ian Gifford
James P. K. Gilb
Dan Grossman
Julian Hall
Allen Heberling
Karl Heubaum
Jin-Meng Ho
Peter Johansson
Chieh-Ying Kang
Charlie Fox
Colleen McGinn
Charlie Mellone
Knut Odman
Mike Rudnick
John Sarallo
Mark Schrader
Bill Shvodian
Larry Telle

When the IEEE-SA Standards Board approved this standard on 6 December 2005, it had the following membership:

Steve M. Mills, *Chair*
Richard H. Hulett, *Vice Chair*
Don Wright, *Past Chair*
Judith Gorman, *Secretary*

Mark D. Bowman
Dennis B. Brophy
Joseph Bruder
Richard Cox
Bob Davis
Julian Forster*
Joanna N. Guenin
Mark S. Halpin
Raymond Hapeman
William B. Hopf
Lowell G. Johnson
Herman Koch
Joseph L. Koepfinger*
David J. Law
Daleep C. Mohla
Paul Nikolich
T. W. Olsen
Glenn Parsons
Ronald C. Petersen
Gary S. Robinson
Frank Stone
Malcolm V. Thaden
Richard L. Townsend
Joe D. Watson
Howard L. Wolfman

*Member Emeritus

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

Satish K. Aggarwal, *NRC Representative*
Richard DeBlasio, *DOE Representative*
Alan H. Cookson, *NIST Representative*

Don Messina
IEEE Standards Project Editor

Contents

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

[This amendment is based on IEEE Std 802.15.3™-2003.]

NOTE—The editing instructions contained in this amendment define how to merge the material contained herein into the existing base standard to form the comprehensive standard.

The editing instructions are shown in ***bold italic***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strikethrough~~ (to remove old material) or underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Insertions may require renumbering. If so, renumbering instructions are given in the editing instructions. ***Replace*** is used to make changes in figures or equations by removing existing figure or equation and replacing it with a new one. Editorial notes will not be carried over into future editions because the changes will be incorporated into the base standard.

3. Definitions

Delete the following definition:

3.13 ~~device-host:~~ ~~The equipment that incorporates an 802.15.3™ device. The device-host may have more than one device incorporated in it as well as other networking connections, both wired and wireless.~~

Change the definition for 3.24 as shown:

3.24 parent piconet: A piconet which creates channel time allocations ~~allocates guaranteed time slots~~ for another piconet (child or neighbor types) operating in the same channel.

Insert the following definitions in alphabetical order:

3.43 implicit handover: A procedure by which a device (DEV) becomes the piconet coordinator (PNC) when the DEV determines that the old PNC either is no longer active or is out of range and without the old PNC explicitly indicating that the handover will take place.

3.44 preliminary handover: The process of transferring all of the necessary handover information from the piconet coordinator (PNC) to another device (DEV) while not handing over control of the piconet.

4. Acronyms and abbreviations

Insert the following acronyms:

CP	contention period
Imp-ACK	implied acknowledgment
NAK	negative acknowledgment
PAL	protocol adaptation layer
SNAP	subnetwork access protocol
STP	stream timeout period

Delete the following acronym:

DEV-host	device-host

6. Layer management

6.1 Overview of management model

Change the first and second paragraphs in 6.1 as shown:

Both MAC and PHY layers conceptually include management entities, called the MAC sublayer management entity and PHY layer management entity (MLME and PLME, respectively). These entities provide the layer management service interfaces for the layer management functions. Figure 3 depicts the relationship among the management entities. The protocol adaptation layer (PAL) includes both the FCSL and DME. The reference model depicts the interactions of a single PAL in a single piconet; support for multiple piconets and/or PALs is implementation-dependent.

~~In order to provide correct MAC operation, a device management entity (DME) should be present within each DEV. The DME is a layer-independent entity that may be viewed as residing in a separate management plane or as residing off to the side. The exact functionality of the DME is not specified in this standard, but in general this entity may be viewed as being responsible for such functions as the gathering of layer-dependent status from the various layer management entities, and similarly setting the value of layer-specific parameters. The DME typically performs such functions on behalf of the general system management entities and implements standard management protocols. Figure 3 depicts the relationship among the management entities.~~

Replace Figure 3 with the figure below:

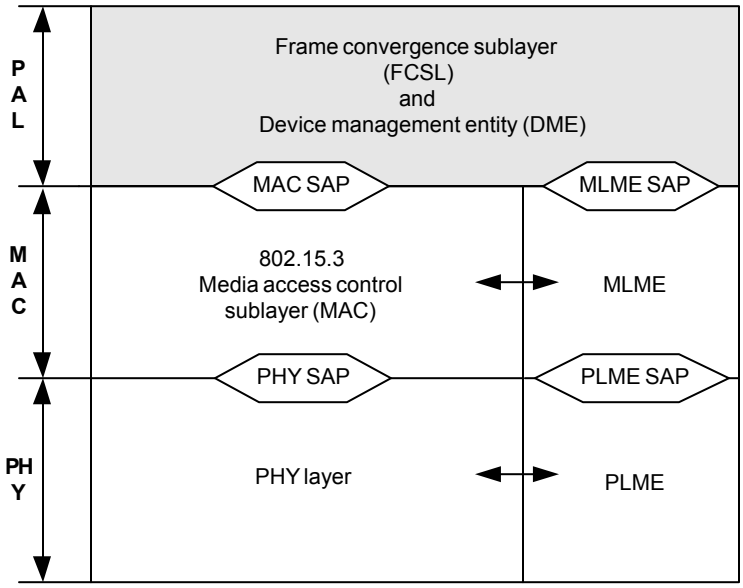

Figure 3—The reference model used in this standard

Change the fourth paragraph in 6.1 as shown and reletter the list:

The various SAPs within this model are the following:

a) ~~FCSL~~ PAL SAP (not shown)

b) MAC SAP

~~c) PHY SAP~~

d) MLME SAP

~~e) PLME SAP~~

~~f) MLME-PLME SAP~~

In 6.1, delete the fifth paragraph, which states "The latter two SAPs support ...." Insert the following paragraph in its place:

The PHY SAP and PLME SAP are not defined in this standard as they are rarely, if ever, exposed in a typical implementation. The PHY management objects and attributes are accessed through the MLME SAP with the generic management primitives defined in 6.2.

In 6.1, delete the eighth paragraph, which states "The split in functionality ...."

6.2 Generic management primitives

Insert the following sentence at the end of the second paragraph in 6.2:

An attempt to "GET" a PIB attribute identified as "write only" attribute is not a valid operation. An attempt to "SET" a PIB attribute identified as a "read only" attribute is not a valid operation.

Change the third paragraph in 6.2 as shown:

The GET and SET primitives are represented as requests with associated confirm primitives. ~~These primitives are prefixed by MLME or PLME depending upon whether the MAC or PHY layer management SAP is involved.~~ The DME uses the services provided by the MLME through the MLME SAP to access the management objects in either the MAC PIB or PHY PIB. The primitives are summarized in Table 1. ~~In Table 1 and in the subclauses that describe the primitives, XX denotes either MLME or PLME.~~

Change Table 1 as shown:

Table 1—Summary of generic management primitives

Name	Request	Confirm
~~XX~~MLME-GET	6.2.1	6.2.2
~~XX~~MLME-SET	6.2.3	6.2.4

Change Table 2 as shown:

Table 2—MLME ~~and PLME~~ generic management primitive parameters

Name	Type	Valid range	Description
PIBattribute	Octet string	Any PIB attribute as defined in 6.5 or 11.7	The name of the PIB attribute
PIBvalue	Variable	As defined in 6.5 or 11.7	The PIB value
ResultCode	Enumeration	SUCCESS, FAILURE ~~INVALID_PIB_ATTRIBUTE_NAME, INVALID_PIB_ATTRIBUTE_VALUE, READ_ONLY_PIB_ATTRIBUTE, WRITE_ONLY_PIB_ATTRIBUTE~~	Indicates the result of the MLME ~~or PLME~~ request.
ReasonCode	Enumeration	INVALID_PIB_ATTRIBUTE_NAME, INVALID_PIB_ATTRIBUTE_VALUE, READ_ONLY_PIB_ATTRIBUTE, WRITE_ONLY_PIB_ATTRIBUTE	Indicates the reason for a ResultCode of FAILURE

Change the title of 6.2.1 as shown:

6.2.1 MLME-GET.request ~~and PLME-GET.request~~

Change the primitive semantics in 6.2.1 as shown:

~~XX~~MLME-GET.request (
PIBattribute
)

Delete 6.2.1.1 and 6.2.1.2.

Change the title of 6.2.2 as shown:

6.2.2 MLME-GET.confirm ~~and PLME-GET.confirm~~

Change the primitive semantics in 6.2.2 as shown:

~~XX~~MLME-GET.confirm	(~~ResultCode,~~ PIBattribute, PIBvalue~~,~~ ResultCode, ReasonCode)

Delete 6.2.2.1 and 6.2.2.2.

Change the title of 6.2.3 as shown:

6.2.3 MLME-SET.request ~~and PLME-SET.request~~

Change the primitive semantics in 6.2.3 as shown:

~~XX~~MLME-SET.request	(PIBattribute, PIBvalue)

Delete 6.2.3.1 and 6.2.3.2.

Change the title of 6.2.4 as shown:

6.2.4 MLME-SET.confirm ~~and PLME-SET.confirm~~

Change the primitive semantics in 6.2.4 as shown:

~~XX~~MLME-SET.confirm	(~~ResultCode,~~ PIBattribute~~,~~ ResultCode, ReasonCode)

Delete 6.2.4.1 and 6.2.4.2.

6.3 MLME SAP interface

Change the paragraph in 6.3 as shown:

The services provided by the MLME to the DME are specified in this subclause. These services are described in an abstract way and do not imply any particular implementation~~ or exposed interface~~. MLME SAP primitives are of the general form ACTION.request followed by ACTION.confirm. An ACTION.indication provides information to the DME that originated either in the local MAC or from another DEV. The ~~DEV~~ DME optionally responds to the indication by issuing an ACTION.response. The DME uses the services provided by the MLME through the MLME SAP. The MLME interface models a single piconet

environment; support for multiple piconets is implementation-dependent. The primitives are summarized in Table 3.

Delete Table 3 and insert the following table in its place:

Table 3—Summary of MLME primitives

Name	Request	Confirm	Indication	Response
MLME-RESET	6.3.1.1	6.3.1.2	–	–
MLME-SCAN	6.3.2.1	6.3.2.2	6.3.2.3	–
MLME-START	6.3.3.1	6.3.3.2	–	–
MLME-STOP	6.3.4.1	6.3.4.2	–	–
MLME-ASSOCIATE	6.3.5.1	6.3.5.2	6.3.5.3	–
MLME-DISASSOCIATE	6.3.6.1	6.3.6.2	6.3.6.3	–
MLME-MEMBERSHIP-UPDATE	6.3.7.1	6.3.7.2	–	–
MLME-SECURITY-ERROR	–	–	6.3.7.3	–
MLME-SECURITY-MESSAGE	6.3.7.4	6.3.7.5	6.3.7.6	–
MLME-PNC-HANDOVER	–	–	6.3.8.1	6.3.8.2
MLME-NEW-PNC	–	–	6.3.8.3	–
MLME-DEV-INFO	6.3.9.1	6.3.9.2	6.3.9.3	–
MLME-SECURITY-INFO	6.3.10.1	6.3.10.2	6.3.10.3	6.3.10.4
MLME-APPLICATION-SPECIFIC	6.3.11.1	6.3.11.2	6.3.11.3	–
MLME-ANNOUNCE-SERVICE	6.3.12.1	6.3.12.2	–	–
MLME-PICONET-SERVICES	6.3.12.3	6.3.12.4	6.3.12.5	–
MLME-CREATE-STREAM	6.3.13.1	6.3.13.2	6.3.13.3	–
MLME-MODIFY-STREAM	6.3.13.4	6.3.13.5	–	–
MLME-TERMINATE-STREAM	6.3.13.6	6.3.13.7	6.3.13.8	–
MLME-BSID-CHANGE	6.3.14.1	6.3.14.2	–	–
MLME-PICONET-PARM-CHANGE	–	–	6.3.14.3	–
MLME-PS-SET-INFORMATION	6.3.15.1	6.3.15.2	–	–
MLME-SPS-CONFIGURE	6.3.15.3	6.3.15.4	–	–
MLME-PM-MODE-CHANGE	6.3.15.5	6.3.15.6	6.3.15.7	–
MLME-MONITOR-PM-MODE	6.3.15.8	6.3.15.9	6.3.15.10	–
MLME-MULTICAST-CONFIGURATION	6.3.16.1	6.3.16.2	–	–
MLME-MULTICAST-RX-SETUP	6.3.16.3	6.3.16.4	–	–
MLME-BEACON-EVENT	6.3.17.1	6.3.17.2	6.3.17.3	–

The MLME interface defined in IEEE Std 802.15.3-2003 did not provide a consistent interface. For example, one set of primitives expected information, such as the list of associated DEVs, to reside in the DME whereas other primitives expected these data to reside only in the MLME. When the task group developed a consistent model for the information and interface, it became clear that the edits were too extensive to be represented as delete and add without causing unnecessary confusion for the reader. Instead, to make it easier to read, the old interface has been deleted, and it is being replaced with the one that follows.

Delete 6.3.1 through 6.3.22.7.2 (including Table 4 through Table 30).

After 6.3, insert the following subclauses as 6.3.1 through 6.3.17.3 (including the new tables, Table 3a through Table 3w):

6.3.1 Resetting the MAC/MLME

These primitives support the process of resetting the MAC/MLME, which also resets the PHY/PLME. The parameters used for these primitives are defined in Table 3a.

Table 3a—MLME-RESET primitive parameters

Name	Type	Valid range	Description
SetDefaultPIB	Boolean	TRUE, FALSE	If TRUE, all PIB attributes are set to their default values. The default values are implementation-dependent. If FALSE, the MAC is reset, but all PIB attributes retain the values that were in place prior to the generation of the MLME-RESET.request primitive.
ResultCode	Enumeration	READY, ERROR	If READY, the MLME has successfully completed initialization and is ready to receive requests. If ERROR, the MLME was unable to successfully complete initialization and is unable to receive requests.

6.3.1.1 MLME-RESET.request

This primitive requests that the MAC entity be reset to its initial conditions. The semantics of this primitive are:

```
MLME-RESET.request          (
                            SetDefaultPIB
                            )
```

The primitive parameter is defined in Table 3a.

The MLME restores the MAC and its internal variables (other than PIB attributes) to their initial state and values. The SetDefaultPIB parameter governs whether PIB attributes are reinitialized or left unchanged.

6.3.1.2 MLME-RESET.confirm

This primitive is generated by the MLME when it completes its initialization process. The semantics of this primitive are:

```
MLME-RESET.confirm      (
                        ResultCode
                        )
```

The primitive parameter is defined in Table 3a.

6.3.2 Scanning for piconets

These primitives support the process of determining the presence or absence of piconets, as described in 8.2.1. The parameters used for these primitives are defined in Table 3b.

Table 3b—MLME-SCAN primitive parameters

Name	Type	Valid range	Description
ScanForBSID	Boolean	TRUE, FALSE	Indicates if the scan process should search for a specific BSID, as described in 8.2.1.
BSIDLength	Integer	As defined in 7.4.2	The number of octets in the BSID.
BSID	Octet string	As defined in 7.4.2	The text string of a specific piconet for which to scan. This parameter is not used if ScanForBSID is FALSE.
ScanForPNID	Boolean	TRUE, FALSE	Indicates if the scan process should search for a specific PNID, as described in 8.2.1.
PNID	Integer	0–65535	The ID of a specific piconet for which to scan. This parameter is not used if ScanForPNID is FALSE.
ScanForPNCAddress	Boolean	TRUE, FALSE	Indicates if the scan process should search for a PNC with a specific MAC address, as described in 8.2.1.
PNCAddress	MAC address	Any valid individual MAC address, as described in 7.1	The MAC address of a specific PNC for which to scan. This parameter is not used if ScanForPNCAddress is FALSE.
NumberOfPiconets	Integer	0–255	The number of piconets found during the scanning process.
PiconetDescriptionSet	Set of piconet descriptions, as defined in Table 3c.	A set containing zero or more instances of a PiconetDescription	The PiconetDescriptionSet is returned to indicate the results of the scan request.
NumberOfChannels	Integer	0 to the maximum number of PHY-dependent channels, as defined in 11.2.3	Indicates the number of channels scanned.

Table 3b—MLME-SCAN primitive parameters *(continued)*

Name	Type	Valid range	Description
ChannelRatingList	Ordered list of integers.	0 to the maximum number of PHY-dependent channels, as defined in 11.2.3	Specifies a list of the channels scanned ordered from the best to the worst in terms of interference.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	OTHER	Indicates the reason for a ResultCode of FAILURE.

In Table 3b, a PiconetDescriptionSet is a set of PiconetDescriptions. Each PiconetDescription consists of the elements shown in Table 3c.

Table 3c—Elements of PiconetDescription

Name	Type	Valid range	Description
BSIDLength	Integer	As defined in 7.4.2	The number of octets in the BSID.
BSID	Octet string	As defined in 7.4.2	The text string identifier of a discovered piconet.
PNID	Integer	0–65535	The PNID of a discovered piconet.
PNCAddress	MAC address	Any valid individual MAC address	The MAC address of the PNC of the piconet that was found.
ChannelIndex	Integer	0 to the maximum number of PHY-dependent channels, as defined in 11.2.3	A PHY-dependent channel number on which the piconet was found.
SECmode	Enumeration	MODE_0, MODE_1	The security mode of the piconet that was found, as described in 7.3.1.
SignalQuality	Integer	0–15	Indicates the quality of the received frame or beacon for this piconet. The value is implementation-dependent with 0 indicating the lowest quality and 15 the highest quality.
NumApplicationSpecificData	Integer	0–255	Indicates the number of ApplicationSpecificDataSets advertised in the beacon of a discovered piconet.
ApplicationSpecificDataSet	Set of application specific data, as defined in Table 3d.	A set containing zero or more instances of ApplicationSpecificData	The ApplicationSpecificDataSet is returned to indicate the ApplicationSpecificData offered by the discovered piconet.

In Table 3b, the ChannelRatingList is a set of N integer values, where N equals the number of channel numbers provided by the PHY. The elements of the set are channel numbers and they are ordered from best (least interference) at the lowest set index to worst (most interference) at the highest set index.

In Table 3c, an ApplicationSpecificDataSet is a set of ApplicationSpecificData. Each ApplicationSpecificData consists of the elements shown in Table 3d. The ApplicationSpecificDataSets are only the current values advertised in the beacon and potentially are different in subsequent beacons.

Table 3d—Elements of ApplicationSpecificData

Name	Type	Valid range	Description
OUI	Octet string	Any valid OUI, as defined in 7.4.7	The unique identifier for the data, as described in 7.4.7.
ApplicationDataLength	Integer	As defined in 7.4.7	The length of the ApplicationData in Octets.
ApplicationData	Octet string	Any valid octet string of length up to ApplicationDataLength	Application-dependent data, as described in 7.4.7.

Any security features of an existing piconet are ignored during the scan process.

It is not possible to obtain piconet services information, as described in 8.3.2, during the scan process.

6.3.2.1 MLME-SCAN.request

This primitive is used to initiate the passive scan procedure to search for either a specific piconet or any piconet. The semantics of this primitive are:

```
MLME-SCAN.request       (
                        ScanForBSID,
                        BSIDLength,
                        BSID,
                        ScanForPNID,
                        PNID,
                        ScanForPNCAddress,
                        PNCAddress,
                        Timeout
                        )
```

The primitive parameters are defined in Table 3b.

6.3.2.2 MLME-SCAN.confirm

This primitive is used to report the result of the request to initiate the passive scan procedure to search for either a specific piconet or any piconet. The semantics of this primitive are:

```
MLME-SCAN.confirm       (
                        NumberOfPiconets,
                        PiconetDescriptionSet,
                        NumberOfChannels,
                        ChannelRatingList,
                        ResultCode,
                        ReasonCode
                        )
```

The primitive parameters are defined in Table 3b. All of the piconets found during the scan will be reported in separate elements of the PiconetDescriptionSet, even if more than one piconet is found on a given channel.

6.3.2.3 MLME-SCAN.indication

This primitive is used to report the result of a passive scan that was initiated by the MAC. The semantics of this primitive are:

```
MLME-SCAN.indication    (
                        NumberOfPiconets,
                        PiconetDescriptionSet,
                        NumberOfChannels,
                        ChannelRatingList
                        )
```

The primitive parameters are defined in Table 3b.

6.3.3 Starting a piconet

These primitives support the process of creating a new piconet with the DEV acting as PNC, as described in 8.2.2. The parameters used for these primitives are defined in Table 3e.

Table 3e—MLME-START primitive parameters

Name	Type	Valid range	Description
BSIDLength	Integer	As defined in 7.4.2	The number of octets in the BSID.
BSID	Octet string	As defined in 7.4.2.	The BSID of the new piconet.
SECMode	Enumeration	MODE_0, MODE_1	The security mode of the piconet, as described in 7.3.1.
DEVID	Integer	Any valid DEVID, as defined in 7.2.3	The assigned DEVID for the DEV that is acting as the PNC, as described in 8.2.2.
MinDepSuperframe Percent	Integer	1–100	The minimum percent of the super-frame requested as a CTA for the dependent piconet, as described in 8.2.5 and 8.2.6.

Table 3e—MLME-START primitive parameters *(continued)*

Name	Type	Valid range	Description
DesiredDep SuperframePercent	Integer	1–100	The desired percent of the superframe requested as a CTA for the dependent piconet, as described in 8.2.5 and 8.2.6.
AllocatedSuperframe Percent	Integer	0–100	The percent of the superframe allocated to the new dependent piconet. If the channel time request was rejected, the value shall be set to zero. This parameter is ignored if the DEV is starting an independent piconet.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_PNC_CAPABLE, NO_CHANNELS_AVAILABLE, ALREADY_PNC, OTHER	Indicates the reason for a ResultCode of FAILURE.

6.3.3.1 MLME-START.request

This primitive is used to start a piconet. If the DEV is not a member of the piconet, this primitive causes the DEV to start an independent piconet. If the DEV is a member of the piconet, this primitive causes the DEV to start a child piconet. If the DEV is associated as a neighbor member of a piconet, this primitive causes the DEV to start a neighbor piconet. The semantics of this primitive are:

```
MLME-START.request      (
                        BSIDLength,
                        BSID,
                        SECMode,
                        MinDepSuperframePercent,
                        DesiredDepSuperframePercent,
                        )
```

The primitive parameters are defined in Table 3e.

6.3.3.2 MLME-START.confirm

This primitive reports the results of a piconet creation procedure. The semantics of this primitive are:

```
MLME-START.confirm      (
                        DEVID,
                        AllocatedSuperframePercent
                        ResultCode,
                        ReasonCode
                        )
```

The primitive parameters are defined in Table 3e.

6.3.4 Stopping a piconet

These primitives support the process of stopping operations as a PNC. The process may result in the shut-down of piconet operations, as described in 8.2.7, or the handover of PNC operations to another DEV in the piconet, as described in 8.2.3 and 8.2.4. The parameters used for these primitives are defined in Table 3f.

Table 3f—MLME-STOP primitive parameters

Name	Type	Valid range	Description
RequestType	Enumeration	SHUTDOWN, HANDOVER	If SHUTDOWN, the current piconet operations will be stopped. If HANDOVER, an attempt to handover PNC operations will be made.
AllowedHandoverTime	Duration	0–65535	If RequestType is HANDOVER, the time in milliseconds in which a handover attempt must be completed.
NumHandoverTargetDEV	Integer	0–mMaxNumValidDEVs	The number of DEVs in the HandoverTargetList
HandoverTargetList	List of DEVIDs	0 to maximum number of DEVIDs, as defined in 7.2.3	If RequestType is HANDOVER, specifies a list of Target DEVIDs for a handover attempt.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_A_PNC, HANDOVER_FAILED, OTHER	Indicates the reason for a ResultCode

In Table 3f, the HandoverTargetList is a set of N DEVIDs, where N equals the number of target DEVIDs for a handover provided by the DME. The elements of the set are DEVIDs and they are ordered from preferred target at the lowest set index to the least preferred target at the highest set index. If the HandoverTargetList contains only the BcstID, as defined in 7.2.3, the DME is not indicating any preference for the handover target.

6.3.4.1 MLME-STOP.request

This primitive initiates the piconet shutdown procedure or the piconet handover procedure. The semantics of this primitive are:

```
MLME-STOP.request       (
                        RequestType,
                        AllowedHandoverTime,
                        NumHandoverTargetDEV,
                        HandoverTargetList
                        )
```

The primitive parameters are defined in Table 3f.

6.3.4.2 MLME-STOP.confirm

This primitive reports the results of the request to stop operations as a PNC. The semantics of this primitive are:

```
MLME-STOP.confirm       (
                        ResultCode,
                        ReasonCode
                        )
```

The primitive parameters are defined in Table 3f.

6.3.5 Associating with a piconet

The following primitives support the process of a DEV associating with a PNC, as defined in 8.3.1. The parameters used for these primitives are defined in Table 3g.

Table 3g—MLME-ASSOCIATE primitive parameters

Name	Type	Valid range	Description
BSIDLength	Integer	As defined in 7.4.2	The number of octets in the BSID.
BSID	Octet string	As defined in 7.4.2	The BSID of the target PNC for the association.
PNID	Integer	0–65535	The PNID of the target PNC for the association, as defined in 7.2.2.
PNCAddress	MAC address	Any valid individual MAC address	The MAC address of the target PNC for the association.
ChannelIndex	Integer	0–255	A PHY-dependent channel number to search for the target PNC for the association.
PiconetServicesInquiry	Boolean	TRUE, FALSE	Requests that the PNC send the services information about the piconet, as described in 8.3.2.
DEVID	Integer	Any valid DEVID, as defined in 7.2.3	The DEVID assigned to a DEV as the result of an association or the DEVID of a DEV that has joined the piconet.
DEVAddress	MAC address	Any valid individual MAC address	The MAC address of a DEV that has joined the piconet.
NeighborPiconetRequest	Boolean	TRUE, FALSE	Indicates that the DEV will join as a neighbor PNC rather than as a member of the piconet.
VendorSpecificIE	Octet string	Any valid Vendor Specific IE, as defined in 7.4.17	The Vendor Specific IE, if present, in the Association Response command, as described in 7.5.1.2.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.

Table 3g—MLME-ASSOCIATE primitive parameters *(continued)*

Name	Type	Valid range	Description
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, PNC_NOT_FOUND, PNC_DENIED, PNC_BUSY, ALREADY_ASSOCIATED, NEIGHBOR_REFUSED, OTHER	Indicates the reason for a ResultCode of FAILURE.

6.3.5.1 MLME-ASSOCIATE.request

This primitive initiates the association procedure. The semantics of this primitive are:

```
MLME-ASSOCIATE.request      (
                            BSIDLength,
                            BSID,
                            PNID,
                            PNCAddress,
                            ChannelIndex,
                            NeighborPiconetRequest,
                            PiconetServicesInquiry,
                            Timeout
                            )
```

The primitive parameters are defined in Table 3g.

6.3.5.2 MLME-ASSOCIATE.confirm

This primitive reports the result of the association procedure. The semantics of this primitive are:

```
MLME-ASSOCIATE.confirm      (
                            DEVID,
                            VendorSpecificIE,
                            ResultCode,
                            ReasonCode
                            )
```

The primitive parameters are defined in Table 3g.

6.3.5.3 MLME-ASSOCIATE.indication

This primitive is used to indicate that a new DEV has associated with the same piconet as this DEV. The semantics of this primitive are:

```
MLME-ASSOCIATE.indication   (
                            DEVID,
                            DEVAddress
                            )
```

The primitive parameters are defined in Table 3g.

6.3.6 Disassociation from a piconet

The following primitives are used when a DEV disassociates from a PNC and when the PNC disassociates a DEV from the piconet, as described in 8.3.4. The parameters used for these primitives are defined in Table 3h.

Table 3h—MLME-DISASSOCIATE primitive parameters

Name	Type	Valid range	Description
DEVID	Integer	Any valid DEVID, as defined in 7.2.3	The DEVID of the disassociated DEV.
DEVAddress	MAC address	Any valid individual MAC address	The MAC address of the disassociated DEV
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, CURRENTLY_PNC, DEV_ATP_EXPIRED, PNC_ATP_EXPIRED, DEV_DISASSOCIATED, OTHER_PNC_ACTION, UNKNOWN	Indicates the reason for the disassociation of a DEV from a piconet.

6.3.6.1 MLME-DISASSOCIATE.request

This primitive initiates the procedure for a DEV to disassociate from a piconet. The semantics of this primitive are:

```
MLME-DISASSOCIATE.request       (
                                Timeout
                                )
```

The primitive parameter is defined in Table 3h.

6.3.6.2 MLME-DISASSOCIATE.confirm

This primitive is used to confirm the result of the disassociation procedure. The semantics of this primitive are:

```
MLME-DISASSOCIATE.confirm       (
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3h.

6.3.6.3 MLME-DISASSOCIATE.indication

This primitive is used to indicate that either this DEV or another DEV has been disassociated from the piconet. The semantics of this primitive are:

```
MLME-DISASSOCIATE.indication    (
                                DEVID,
                                DEVAddress,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3h.

6.3.7 Security management

These primitives are used to initialize, update or delete the security information as a result of a membership or key change process, as described in 9.3.4, or as the result of a security event, as described in 9.3.5 and 9.3.6. Primitives are also provided to transfer security messages. These primitives are suitable for use in an authentication process.

The parameters used for the MLME-MEMBERSHIP-UPDATE and MLME-SECURITY-ERROR primitives are defined in Table 3i.

Table 3i—MLME-MEMBERSHIP-UPDATE and MLME-SECURITY-ERROR primitive parameters

Name	Type	Valid range	Description
SECID	2 octets	As defined in 7.2.7.2	The identifier for the key.
OrigID	Integer	Any valid DEVID, as defined in 7.2.3, except for the BcstID, the McstID or the UnassocID	Either the PNCID, if this key is for the DEV's PNC personality, or the DEV's DEVID.
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3, except for the BcstID, the McstID or the UnassocID	The DEVID of the target DEV for this relationship.
MembershipStatus	Enumeration	MEMBER, NON-MEMBER	Indicates the membership status for the provided SECID. If NON-MEMBER, KeyInfoLength is 0.
KeyOriginator	Boolean	TRUE, FALSE	Indicates if the DEV is the key originator for this relationship. This is always true when the OrigID is the PNCID.
KeyInfoLength	Integer	0 or 16	Length of KeyInfo.
KeyInfo	Octet string	Any valid symmetric key for the symmetric key security operations, as defined in 10.3	The key used for protecting frames between this DEV and the TrgtID DEV.
SrcID	Integer	Any valid DEVID, as defined in 7.2.3, except for the BcstID, the McstID or the UnassocID	The DEVID of the DEV that is the source of a security error.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.

Table 3i—MLME-MEMBERSHIP-UPDATE and MLME-SECURITY-ERROR primitive parameters *(continued)*

Name	Type	Valid range	Description
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_ASSOCIATED, TARGET_UNAVAILABLE, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK, BAD_TIME_TOKEN, INVALID_SEC_VALUE, OTHER	The reason for a security error.

The parameters used for the MLME-SECURITY-MESSAGE primitive are defined in Table 3j.

Table 3j—MLME-SECURITY-MESSAGE primitive parameters

Name	Type	Valid range	Description
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the target of the MLME request.
OrigID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the originator of the MLME request.
OUI	Octet string	Any valid OUI, as defined in 7.4.7	A unique identifier for the security information, as described in 7.4.7.
SecurityInformationLength	Integer	0–252	The length of SecurityInformation in octets.
SecurityInformation	Octet string	Any valid octet string	Security information that will be passed from one DEV to another peer DEV in the piconet.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, TARGET_UNAVAILABLE, OTHER	The reason for a security error.

6.3.7.1 MLME-MEMBERSHIP-UPDATE.request

This primitive initiates the membership update procedure by either installing or removing a management key. Data keys are generated by the key originator of the relationship and are exchanged between peer MLMEs, as described in 9.3.4. The semantics of this primitive are:

```
MLME-MEMBERSHIP-UPDATE.request  (
                                OrigID,
                                TrgtID,
                                MembershipStatus,
                                SECID,
                                KeyOriginator,
                                KeyInfoLength,
                                KeyInfo,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3i.

6.3.7.2 MLME-MEMBERSHIP-UPDATE.confirm

This primitive indicates the result of a membership update request. The semantics of this primitive are:

```
MLME-MEMBERSHIP-UPDATE.confirm  (
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3i.

6.3.7.3 MLME-SECURITY-ERROR.indication

This primitive allows the MLME of any DEV to indicate a failed security processing operation. The semantics of this primitive are:

```
MLME-SECURITY-ERROR.indication  (
                                SrcID,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3i.

6.3.7.4 MLME-SECURITY-MESSAGE.request

This primitive initiates the sending of a Security Message command, as described in 7.5.9.1, to the target DEV in the piconet. The semantics of this primitive are:

```
MLME-SECURITY-MESSAGE.request   (
                                TrgtID,
                                OUI,
                                SecurityInformationLength,
                                SecurityInformation,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3j.

6.3.7.5 MLME-SECURITY-MESSAGE.confirm

This primitive indicates the result of a request to send a Security Message command. The semantics of this primitive are:

```
MLME-SECURITY-MESSAGE.confirm   (
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3j.

6.3.7.6 MLME-SECURITY-MESSAGE.indication

This primitive reports the reception of a Security Message command, as described in 7.5.9.1, from a DEV in the piconet. The semantics of this primitive are:

```
MLME-SECURITY-MESSAGE.indication  (
                                  OrigID,
                                  OUI,
                                  SecurityInformationLength,
                                  SecurityInformation
                                  )
```

The primitive parameters are defined in Table 3j.

6.3.8 PNC handover

These primitives are used as part of the handover process, as described in 8.2.3 and 8.2.4, where the current PNC's responsibilities are transferred to another DEV in the piconet. The parameters used for these primitives are defined in Table 3k.

Table 3k—MLME-PNC-HANDOVER and MLME-NEW-PNC primitive parameters

Name	Type	Valid range	Description
HandoverStatus	Enumeration	STARTED, IMPLICIT, PRELIMINARY, CANCELLED	Indicates if the PNC is beginning or cancelling a handover to the DEV.
NewPNCDEVAddress	MAC address	Any valid individual MAC address	The DEV address of the DEV assuming responsibilities as PNC.
SECMode	Enumeration	MODE_0, MODE_1	The security mode of the piconet, as defined in 7.3.1.

6.3.8.1 MLME-PNC-HANDOVER.indication

This primitive indicates the reception of a directed PNC Handover Request command, 7.5.3.1. The semantics of this primitive are:

```
MLME-PNC-HANDOVER.indication    (
                                HandoverStatus
                                )
```

The primitive parameter is defined in Table 3k.

6.3.8.2 MLME-PNC-HANDOVER.response

This primitive indicates that DME of the device targeted for PNC handover is ready to act as PNC. The semantics of this primitive are:

```
MLME-PNC-HANDOVER.response    ()
```

There are no primitive parameters.

6.3.8.3 MLME-NEW-PNC.indication

This primitive indicates that the role of the PNC has been assumed by a different DEV in the piconet. The semantics of this primitive are:

```
MLME-NEW-PNC.indication    (
                           NewPNCDEVAddress,
                           SECMode
                           )
```

6.3.9 Requesting DEV information from the PNC

This mechanism supports the ability for a DEV to request information from the PNC about either a specific DEV or all of the DEVs in the piconet, as described in 8.9.1. The parameters used for these primitives are defined in Table 3l.

Table 3l—MLME-DEV-INFO primitive parameters

Name	Type	Valid range	Description
QueriedDEVID	Integer	Any valid DEVID, as defined in 7.2.3, except for the McstID or the UnassocID	The DEVID of the DEV for which information is being requested from the PNC. If it is the BcstID, the request is for information for all of the DEVs in the piconet.
NumDEVInfo	Integer	1–mMaxNumValidDEVs	Number of entries in the DEVInfoSet.
DEVInfoSet	A set of DEV Info fields, as defined in 7.5.4.2.	A set containing 1 to mMaxNumValidDEVs instances of fixed length DEV Info fields	The DEVInfoSet is returned to indicate the results of a PNC Information Request command.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.

Table 3l—MLME-DEV-INFO primitive parameters *(continued)*

Name	Type	Valid range	Description
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, TARGET_NOT_ASSOCIATED, OTHER	Indicates the reason for a ResultCode of FAILURE.

6.3.9.1 MLME-DEV-INFO.request

This primitive initiates a request to the PNC for information regarding either a single DEV or all of the DEVs in the piconet. The semantics of this primitive are:

```
MLME-DEV-INFO.request       (
                            QueriedDEVID,
                            Timeout
                            )
```

The primitive parameters are defined in Table 3l.

6.3.9.2 MLME-DEV-INFO.confirm

This primitive provides the result of the request to the PNC for information regarding either a single DEV or all of the DEVs in the piconet. The semantics of this primitive are:

```
MLME-DEV-INFO.confirm       (
                            NumDEVInfo,
                            DEVInfoSet,
                            ResultCode,
                            ReasonCode
                            )
```

The primitive parameters are defined in Table 3l.

6.3.9.3 MLME-DEV-INFO.indication

This primitive indicates the reception of DEV information that was not requested using the MLME-DEV-INFO.request primitive. The semantics of this primitive are:

```
MLME-DEV-INFO.indication    (
                            NumDEVInfo,
                            DEVInfoSet
                            )
```

The primitive parameters are defined in Table 3l.

6.3.10 Security information retrieval

These primitives are used to request security information about other DEVs in the piconet, as described in 9.4.1. The parameters used for the MLME-SECURITY-INFO primitives are defined in Table 3m.

Table 3m—MLME-SECURITY-INFO primitive parameters

Name	Type	Valid range	Description
QueriedDEVID	Integer	Any valid DEVID, as defined in 7.2.3, except for the McstID or the UnassocID	The DEVID of the DEV for which information is being requested. If it is set to the BcstID, then the information is being requested for all DEVs.
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3	The DEVID of the DEV for which the security information request is intended.
OrigID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the DEV that initiated the MLME request.
NumSecurityRecords	Integer	0–65535	Number of entries in the SecurityRecordSet.
SecurityRecordSet	A set of Security Record fields, as defined in 7.5.4.4.	A set containing 0 or more instances of variable length Security Record field. The maximum number of instances depends on the size of the records, pMaxFrameBodySize and the length of the secure command security fields, as defined in 7.3.3.2	The SecurityRecordSet is returned to indicate the results of a Security Information Request command.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, TARGET_NOT_ASSOCIATED, OTHER	Indicates the reason for a ResultCode of FAILURE.

6.3.10.1 MLME-SECURITY-INFO.request

This primitive initiates a request to a DEV for security information regarding either a single DEV or all of the DEVs in the piconet. The semantics of the primitive are as follows:

```
MLME-SECURITY-INFO.request      (
                                TrgtID,
                                QueriedDEVID,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3m.

6.3.10.2 MLME-SECURITY-INFO.confirm

This primitive reports the result of the request to a DEV for security information regarding either a single DEV or all of the DEVs in the piconet. The semantics of the primitive are as follows:

```
MLME-SECURITY-INFO.confirm      (
                                TrgtID,
                                NumSecurityRecords,
                                SecurityRecordSet,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3m.

6.3.10.3 MLME-SECURITY-INFO.indication

This primitive indicates the reception of a request by a DEV for security information it manages regarding either a specific DEV or all of the DEVs in the piconet. The semantics of the primitive are as follows:

```
MLME-SECURITY-INFO.indication   (
                                OrigID,
                                QueriedDEVID
                                )
```

The primitive parameters are defined in Table 3m.

6.3.10.4 MLME-SECURITY-INFO.response

This primitive is used by a DEV to respond to an MLME-SECURITY-INFO.indication. The semantics of the primitive are as follows:

```
MLME-SECURITY-INFO.response     (
                                OrigID,
                                NumSecurityRecords,
                                SecurityRecordSet
                                )
```

The primitive parameters are defined in Table 3m.

6.3.11 Application specific data management

These primitives are used to request that the PNC add, modify or remove application specific data in the beacon and to report the reception of application specific data in a beacon, as described in 8.14. The parameters used for these primitives are defined in Table 3n. If the RequestType is “ADD,” then the MAC ignores the ASIEIndex.

Table 3n—MLME-APPLICATION-SPECIFIC primitive parameters

Name	Type	Valid range	Description
RequestType	Enumeration	ADD, MODIFY, REMOVE	If ADD, a request that a new ASIE be placed in the beacon. If MODIFY, a request to change the contents of an existing ASIE. If REMOVE, a request that a previously added ASIE be removed from the beacon.
ASIEIndex	Octet	0–255	An ID assigned by the PNC to each ASIE successfully added to the beacon. If RequestType is MODIFY or REMOVE, it specifies the ASIE to modify or remove.
RequestID	Octet	0–255	A unique number assigned by the requesting DEV to identify the request.
OUI	Octet string	Any valid OUI, as defined in 7.4.7	If RequestType is ADD or MODIFY, the OUI for the ApplicationData.
ApplicationDataLength	Integer	As defined in 7.4.7	If RequestType is ADD or MODIFY, the length of the ApplicationData to add to the beacon in octets.
ApplicationData	Octet string	Any valid octet string of length up to ApplicationDataLength	If the RequestType is ADD or MODIFY, the application specific data to add to the beacon, as described in 8.14.
NumApplicationSpecificData	Integer	0–255	Indicates the number of ApplicationSpecificData in the piconet beacon.
ApplicationSpecificDataSet	Set of application specific data, as defined in Table 3d.	A set containing zero or more instances of an ApplicationSpecificData	The ApplicationSpecificDataSet is returned to indicate the ApplicationSpecificData in the piconet beacon.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, UNKNOWN_ID, PNC_DENIED, OTHER	Indicates the reason for a ResultCode of FAILURE.

6.3.11.1 MLME-APPLICATION-SPECIFIC.request

This primitive is used to request to add, modify or remove application specific data in the piconet beacon. The semantics of this primitive are:

```
MLME-APPLICATION-SPECIFIC.request (
                                    RequestType,
                                    RequestID,
                                    ASIEIndex,
                                    OUI,
                                    ApplicationDataLength,
                                    ApplicationData,
                                    Timeout
                                    )
```

The primitive parameters are defined in Table 3n.

6.3.11.2 MLME-APPLICATION-SPECIFIC.confirm

This primitive is used to report the result of the request to add, modify or remove application specific data in the beacon. The semantics of this primitive are:

```
MLME-APPLICATION-SPECIFIC.confirm (
                                    ASIEIndex,
                                    ResultCode,
                                    ReasonCode
                                    )
```

The primitive parameters are defined in Table 3n.

6.3.11.3 MLME-APPLICATION-SPECIFIC.indication

This primitive is used to indicate that application specific data was in a successfully received beacon. The MLME passes all of the ASIEs found in every beacon to the DME. The semantics of this primitive are:

```
MLME-APPLICATION-SPECIFIC.indication(
                                    NumApplicationSpecificData,
                                    ApplicationSpecificDataSet
                                    )
```

The primitive parameters are defined in Table 3n.

6.3.12 Piconet services management

These primitives are used to transfer information regarding the services offered by DEVs in a piconet, as described in 8.3.2. The parameters used for these primitives are defined in Table 3o.

In Table 3o, a PiconetServicesSet is a set of PiconetServices. Each PiconetService consists of the elements shown in Table 3p.

Table 3o—MLME-ANNOUNCE-SERVICE and MLME-PICONET-SERVICES primitive parameters

Name	Type	Valid range	Description
OUI	Octet string	Any valid OUI, as defined in 7.4.7	The OUI for the ServiceData.
ServiceDataLength	Integer	0–124	The length of the ServiceData.
ServiceData	Octet string	Any valid octet string of length ServiceDataLength	The ServiceData, as described in 8.3.2.
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the DEV to request service information.
NumberOfServices	Integer	0–255	The number of PiconetServices in the PiconetServicesSet.
PiconetServicesSet	Set of piconet services, as defined in Table 3p.	A set containing zero or more instances of a PiconetService	The PiconetServicesSet is returned to indicate the PiconetServices offered.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, TARGET_NOT_ASSOCIATED, OTHER	Indicates the reason for a ResultCode of FAILURE.

Table 3p—Elements of PiconetService

Name	Type	Valid range	Description
DEVID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the DEV offering the service.
OUI	Octet string	Any valid OUI, as defined in 7.4.7	The OUI for the ServiceData.
ServiceDataLength	Integer	0–128	The length of the ServiceData.
ServiceData	Octet string	Any valid octet string of length up to ServiceDataLength	The ServiceData, as described in 8.3.2.

6.3.12.1 MLME-ANNOUNCE-SERVICE.request

This primitive is used to request the announcement of the availability of a service offered by this DEV. The semantics of this primitive are:

```
MLME-ANNOUNCE-SERVICE.request   (
                                OUI,
                                ServiceDataLength,
                                ServiceData,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3o.

6.3.12.2 MLME-ANNOUNCE-SERVICE.confirm

This primitive is used to report the result of the request to announce the availability of a service offered by this DEV. The semantics of this primitive are:

```
MLME-ANNOUNCE-SERVICE.confirm   (
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3o.

6.3.12.3 MLME-PICONET-SERVICES.request

This primitive is used to request piconet service information from a DEV in the piconet. If the target DEV is the PNC, then it is a request for all piconet service information previously announced by DEVs and stored in the PNC. The semantics of this primitive are:

```
MLME-PICONET-SERVICES.request   (
                                TrgtID,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3o.

6.3.12.4 MLME-PICONET-SERVICES.confirm

This primitive is used to report the result of the request for piconet service information from a DEV in the piconet. The semantics of this primitive are:

```
MLME-PICONET-SERVICES.confirm   (
                                NumberOfServices,
                                PiconetServicesSet,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3o.

6.3.12.5 MLME-PICONET-SERVICES.indication

This primitive indicates the reception of piconet services information that was not requested with the MLME-PICONET-SERVICES.request primitive. The semantics of this primitive are:

```
MLME-PICONET-SERVICES.indication (
                                 NumberOfServices,
                                 PiconetServicesSet
                                 )
```

The primitive parameters are defined in Table 3o.

6.3.13 Stream management

This mechanism supports the creation, modification, and termination of isochronous streams, as described in 8.5.1. The parameters used for the MLME-CREATE-STREAM, MLME-MODIFY-STREAM and MLME-TERMINATE-STREAM primitives are defined in Table 3q.

Table 3q—MLME-CREATE-STREAM, MLME-MODIFY-STREAM, and MLME-TERMINATE-STREAM primitive parameters

Name	Type	Valid range	Description
RequestID	Integer	0–255	A unique value created by the originating DME to match the request primitive with the response primitive it receives from the PNC MLME.
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the target DEV for an isochronous stream.
SourceDataRate	Integer	$1–(2^{64} – 1)$	The minimum required data rate of the stream source at the MAC SAP in bits per second, as described in D1.1.5.
DesiredDataRate	Integer	$1–(2^{64} – 1)$	The desired data rate of the stream source at the MAC SAP in bits per second, as described in D1.1.5.
MaxRetries	Integer	0–255	Specifies the maximum number of retries to attempt per transmitted frame.
ACKRequested	Boolean	TRUE, FALSE	Indicates if acknowledgments will be used for the stream.
MaxTransmitDelay	Duration	$1–(2^{64} – 1)$	Maximum allowed delay in microseconds for transmitting a MSDU once it is presented to MAC SAP.
UserPriority	Integer	As defined in Table A.1	User priority of the stream, as described in Table A.1.
TypicalDataFrameSize	Integer	0–pMaxFrameBodySize	The typical size in octets of an MSDU to be presented to the MAC SAP.
SECMode	Boolean	TRUE, FALSE	Indicates if security is to be applied to the stream.

Table 3q—MLME-CREATE-STREAM, MLME-MODIFY-STREAM, and MLME-TERMINATE-STREAM primitive parameters *(continued)*

Name	Type	Valid range	Description
StreamGrpID	Integer	0–255	A nonzero value specifies that channel time associated with this stream may be shared by other streams associated with the same stream group ID, as described in 8.5.1.
StreamIndex	Integer	Any valid stream index, as defined in 7.2.5	The index of a stream created or the index of a stream to modify or terminate.
OrigID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID of the source DEV for an isochronous stream.
AvailableDataRate	Integer	$1–(2^{64} – 1)$	If the request was successful, the data rate available for the stream. If the request was unsuccessful, the data rate that the PNC would have been able to allocate.
ReliabilityExponent	Integer	0–31	The negative power of 10 that is the maximum frame error ratio (FER) desired including retries and frames lost due to MaxTransmitDelay. For example, a value of 4 corresponds to an FER < 10^{-4}. If the value of the parameter is zero, then the parameter is ignored.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, TARGET_UNAVAILABLE, RESOURCES_ UNAVAILABLE, TERMINATED_BY_PNC, TERMINATED_BY_DEST, TRANSMIT_DELAY_UNS UPPORTED, PNC_BUSY DEV_IN_PS_MODE NOT_ASSOCIATED, UNKNOWN_STREAM, OTHER	The reason for a ResultCode of FAILURE.

6.3.13.1 MLME-CREATE-STREAM.request

This primitive is used to request the creation of an isochronous stream. The semantics of this primitive are:

```
MLME-CREATE-STREAM.request      (
                                RequestID,
                                TrgtID,
                                SourceDataRate,
                                DesiredDataRate,
                                MaxRetries,
                                ACKRequested,
                                MaxTransmitDelay,
                                UserPriority,
                                TypicalDataFrameSize,
                                SECMode,
                                StreamGrpID,
                                ReliabilityExponent
                                Timeout
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.2 MLME-CREATE-STREAM.confirm

This primitive is used to report the result of the request for the creation of an isochronous stream. The semantics of this primitive are:

```
MLME-CREATE-STREAM.confirm      (
                                RequestID,
                                StreamIndex,
                                AvailableDataRate,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.3 MLME-CREATE-STREAM.indication

This primitive is used to inform a DEV that it is the target of an isochronous stream sourced from another DEV in the piconet. The semantics of this primitive are:

```
MLME-CREATE-STREAM.indication   (
                                StreamIndex,
                                OrigID
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.4 MLME-MODIFY-STREAM.request

This primitive is used to request a modification to the parameters defining an existing isochronous stream. The semantics of this primitive are:

```
MLME-MODIFY-STREAM.request      (
                                StreamIndex,
                                SourceDataRate,
                                DesiredDataRate,
                                MaxRetries,
                                ACKRequested,
                                MaxTransmitDelay,
                                UserPriority,
                                TypicalDataFrameSize,
                                SECMode,
                                StreamGrpID,
                                ReliabilityExponent
                                Timeout
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.5 MLME-MODIFY-STREAM.confirm

This primitive is used to report the result of the request to modify the parameters defining an existing isochronous stream. The semantics of this primitive are:

```
MLME-MODIFY-STREAM.confirm      (
                                StreamIndex,
                                AvailableDataRate,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.6 MLME-TERMINATE-STREAM.request

This primitive is used to request the termination of an existing isochronous stream. The semantics of this primitive are:

```
MLME-TERMINATE-STREAM.request   (
                                StreamIndex,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.7 MLME-TERMINATE-STREAM.confirm

This primitive is used to report the result of the request to terminate an existing isochronous stream. The semantics of this primitive are:

```
MLME-TERMINATE-STREAM.confirm   (
                                StreamIndex,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3q.

6.3.13.8 MLME-TERMINATE-STREAM.indication

This primitive is used to report to a DEV other than the DEV terminating the stream that an isochronous stream has been terminated. The semantics of this primitive are:

```
MLME-TERMINATE-STREAM.indication   (
                                   StreamIndex,
                                   ReasonCode
                                   )
```

The primitive parameter is defined in Table 3q.

6.3.14 Piconet parameter management

These primitives allow a DEV acting as PNC to change the BSID of the piconet and allow all DEVs to be informed of changes to piconet characteristics, as defined in 8.10 and 8.11. The parameters used for these primitives are defined in Table 3r.

Table 3r—MLME-BSID-CHANGE and MLME-PICONET-PARM-CHANGE primitives parameters

Name	Type	Valid range	Description
IndicationChangeType	Enumeration	BSID, PNID, CHANNEL	Indicates the parameter of the piconet that was changed.
BSIDLength	Integer	As defined in 7.4.2	The number of octets in the BSID.
BSID	Octet string	As defined in 7.4.2	If a request or if the IndicationChangeType is BSID, a new BSID for the piconet, as described in 7.4.2.
PNID	Integer	0–65535	If the IndicationChangeType is PNID, the new ID for the piconet.
ChannelIndex	Integer	0–255	If the IndicationChangeType is CHANNEL, the new PHY-dependent channel number on which the piconet is operating.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_A_PNC, OTHER	The reason for a ResultCode of FAILURE.

6.3.14.1 MLME-BSID-CHANGE.request

This primitive initiates changing the BSID. The semantics of this primitive are:

```
MLME-BSID-CHANGE.request
                                    (
                                    BSIDLength,
                                    BSID
                                    )
```

The primitive parameter is defined in Table 3r.

6.3.14.2 MLME-BSID-CHANGE.confirm

This primitive reports the result of a request to change the BSID. The semantics of this primitive are:

```
MLME-BSID-CHANGE.confirm
                                    (
                                    ResultCode,
                                    ReasonCode
                                    )
```

The primitive parameters are defined in Table 3r.

6.3.14.3 MLME-PICONET-PARM-CHANGE.indication

This primitive reports to a DEV a change to a piconet characteristic that was not requested by the DME. The semantics of this primitive are:

```
MLME-PICONET-PARM-CHANGE.indication
                                    (
                                    IndicationChangeType,
                                    BSIDLength,
                                    BSID,
                                    PNID,
                                    ChannelIndex
                                    )
```

The primitive parameters are defined in Table 3r.

6.3.15 Power management

This mechanism supports the process of establishment and maintenance of power management (PM) modes of a DEV, as described in 8.13. The parameters used for these primitives are defined in Table 3s.

Table 3s—MLME-PS-SET-INFORMATION, MLME-SPS-CONFIGURE, MLME-PM-MODE-CHANGE, and MLME-MONITOR-PM-MODE primitive parameters

Name	Type	Valid range	Description
PMMode	Enumeration	ACTIVE, APS, SPS	The PM mode requested by the DEV, as described in 7.5.8.5.
MaxSupportedPSSets	Integer	As defined in 7.5.8.2 and 8.13	The total number of PS sets currently supported by the PNC of this piconet.
NumCurrentPSSets	Integer	As defined in 7.5.8.2 and 8.13	Indicates the number of currently active PS sets in the piconet.
PSSetStructureSet	Set of PSSetStructures	As defined in Table 3t	The PSSetStructureSet returns the information about the PS sets currently active in the PNC.
SetOperationType	Enumeration	CREATE, JOIN, LEAVE	The requested SPS set operation.
SPSSetIndex	Integer	As defined in 7.5.8.3	If the SetOperationType is JOIN or LEAVE, the SPS set index of the SPS set to join or leave. If the SetOperationType is CREATE, the SPS set index created by the PNC for the new SPS set.
DesiredWakeInterval	Integer	$0–(2^{32} – 1)$	If the SetOperationType is CREATE, the period in microseconds at which the requesting DEV would desire to transition from the SLEEP state to the AWAKE state.
WakeInterval	Integer	$0–(2^{32} – 1)$	The time period in microseconds at which DEVs in a power save set transition from the SLEEP state to the AWAKE state.
PMActiveEvent	Enumeration	DATA_PENDING, MAX_SLEEP	If PMMode is ACTIVE, an event that causes the MLME to change the PM mode of operation to ACTIVE.
MonitorOperationType	Enumeration	ENABLE, DISABLE	The PM monitor operation requested.
TrgtID	Integer	Any valid DEVID, 7.2.3	Specifies the DEVID of the target DEV for a PM monitor operation.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, TARGET_NOT_ ASSOCIATED, UNKNOWN_SET_INDEX, PNC_DENIED, PNC_BUSY, OTHER	The reason for a ResultCode of FAILURE.

In Table 3s, PSSetStructureSet is a set of PSSetStructures. Each PSSetStructure consists of the elements shown in Table 3t.

Table 3t—Elements of PSSetStructure

Name	Type	Valid range	Description
PSSetIndex	Integer	As defined in 7.5.8.2	The identifier for the PS set.
WakeInterval	Integer	$0–(2^{32} – 1)$	The period in microseconds at which DEVs in this power save set transition from the SLEEP state to the AWAKE state. The MAC divides this number by the superframe duration and rounds down to the nearest power of two.
NumMembers	Integer	0–255	The number of DEVs in this power save set.
MemberSet	Set of DEVIDs	Any valid DEVID, as defined in 7.2.3.	The DEVIDs of the DEVs in this power save set.

In Table 3t, MemberSet is a set of DEVIDs indicating the DEVIDs that are a member of the PS set.

6.3.15.1 MLME-PS-SET-INFORMATION.request

This primitive requests the current PS set information from the PNC. The semantics of this primitive are:

```
MLME-PS-SET-INFORMATION.request    (
                                   Timeout
                                   )
```

The primitive parameter is defined in Figure 3s.

6.3.15.2 MLME-PS-SET-INFORMATION.confirm

This primitive reports the result of the request to obtain the PS set information from the PNC. The semantics of this primitive are:

```
MLME-PS-SET-INFORMATION.confirm    (
                                   MaxSupportedPSSets,
                                   NumCurrentPSSets,
                                   PSSetStructureSet,
                                   ResultCode,
                                   ReasonCode
                                   )
```

The primitive parameters are defined in Table 3s.

6.3.15.3 MLME-SPS-CONFIGURE.request

This primitive requests a change to the current SPS set information maintained by the PNC. Possible requests include: create a new DSPS set and add the current DEVID, add the current DEVID to an existing SPS set, or remove the current DEVID from an existing set. The semantics of this primitive are:

```
MLME-SPS-CONFIGURE.request
                                (
                                SetOperationType,
                                SPSSetIndex,
                                DesiredWakeInterval,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.4 MLME-SPS-CONFIGURE.confirm

This primitive reports the result of a request to change the current SPS set information. The semantics of this primitive are:

```
MLME-SPS-CONFIGURE.confirm
                                (
                                SetOperationType,
                                SPSSetIndex,
                                WakeInterval,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.5 MLME-PM-MODE-CHANGE.request

This primitive requests a change to the DEV's PM mode of operation. The semantics of this primitive are:

```
MLME-PM-MODE-CHANGE.request     (
                                PMMode,
                                Timeout
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.6 MLME-PM-MODE-CHANGE.confirm

This primitive reports the result of the request to change the DEV's PM mode of operation. The semantics of this primitive are:

```
MLME-PM-MODE-CHANGE.confirm     (
                                PMMode,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.7 MLME-PM-MODE-CHANGE.indication

This primitive is used to report a change to the DEVs PM mode of operation that was not requested with an MLME-PM-MODE-CHANGE.request but rather was initiated by the MAC. One reason for the MAC to change PM modes is in response to a CTA allocated with the DEV as the destination while the DEV is in a power save mode, as described in 8.13.2.2. This primitive reports only changes in the DEV's PM mode. Changes in the PM mode of other DEVs in the piconet are reported with the MLME-MONITOR-PM-MODE.indication primitive. The semantics of this primitive are:

```
MLME-PM-MODE-CHANGE.indication
                                    (
                                    PMMode,
                                    PMActiveEvent
                                    )
```

The primitive parameter is defined in Table 3s.

6.3.15.8 MLME-MONITOR-PM-MODE.request

This primitive requests that the MLME enable or disable the monitoring of the PM mode for a DEV in the piconet. The semantics of this primitive are:

```
MLME-MONITOR-PM-MODE.request    (
                                MonitorOperationType,
                                TrgtID
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.9 MLME-MONITOR-PM-MODE.confirm

This primitive reports the result of a request that the MLME enable or disable the monitoring of the PM mode for a DEV in the piconet. The semantics of this primitive are:

```
MLME-MONITOR-PM-MODE.confirm    (
                                MonitorOperationType,
                                TrgtID,
                                PMMode,
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3s.

6.3.15.10 MLME-MONITOR-PM-MODE.indication

This primitive reports a change in the PM mode of a DEV in the piconet, for which PM monitoring was previously enabled using an MLME-MONITOR-PM-MODE.request. This primitive only reports changes to the PM modes of other DEVs in the piconet. Changes to the DEV's PM mode are indicated with the MLME-PM-MODE-CHANGE.indication primitive. The semantics of this primitive are:


```
MLME-MONITOR-PM-MODE.indication (
                                 TrgtID,
                                 PMMode
                                 )
```

The primitive parameters are defined in Table 3s.

6.3.16 Multicast operations

These primitives support multicast operations. The MLME-MULTICAST-CONFIGURATION primitives are used for multicast traffic that uses a Group Address and a GrpID assigned by the PNC, as described in 8.5.3. The parameters used for these primitives are defined in Table 3u.

Table 3u—MLME-MULTICAST-CONFIGURATION primitive parameters

Name	Type	Valid range	Description
RequestType	Enumeration	JOIN, LEAVE	Indicates if this is a request to join a multicast group or leave a multicast group.
GroupAddress	MAC address	Any valid group address, as defined in 7.4.18	A group address representing a specific multicast group.
GrpID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the DEVID assigned by the PNC to a multicast group associated with a specific GroupAddress.
Timeout	Integer	0–65535	The time in milliseconds allowed for the primitive to complete.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	REQUEST_TIMEOUT, NOT_ASSOCIATED, PNC_DENIED, PNC_BUSY, OTHER	The reason for a ResultCode of FAILURE.

The MLME-MULTICAST-RX-SETUP.request primitive controls the reception of multicast traffic that uses the McstID, as defined in 7.2.3. The parameters for this primitive are defined in Table 3v.

Table 3v—MLME-MULTICAST-RX-SETUP.request parameters

Name	Type	Valid range	Description
MulticastStatus	Enumeration	ENABLE, DISABLE, ALL, NONE	If ENABLE or DISABLE, indicates whether the MAC will pass multicast traffic defined by the stream index to the DME. If ALL, then all multicast traffic is passed to the DME. If NONE, then no multicast traffic will be passed to the DME. These restrictions apply only to frames received with the DestID set to the McstID.
SrcID	Integer	Any valid DEVID, as defined in 7.2.3	The DEVID of the source of a multicast stream.
StreamIndex	Integer	Any valid stream index, as defined in 7.2.5	The stream index of a multicast stream.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_ASSOCIATED, UNKNOWN_STREAM, SOURCE_UNAVAILABLE, RESOURCES_UNAVAILABLE	Indicates the reason for a ResultCode of FAILURE.

6.3.16.1 MLME-MULTICAST-CONFIGURATION.request

This primitive is used by a DEV to request to join or leave a multicast group defined by a particular group address. The semantics of this primitive are:

```
MLME-MULTICAST-CONFIGURATION.request(
                                    RequestType,
                                    GroupAddress,
                                    Timeout
                                    )
```

The primitive parameters are defined in Table 3u.

6.3.16.2 MLME-MULTICAST-CONFIGURATION.confirm

This primitive is used to report the result of a request by a DEV to join or leave a multicast group defined by a particular group address. The semantics of this primitive are:

```
MLME-MULTICAST-CONFIGURATION.confirm(
                                    GroupAddress,
                                    GrpID,
                                    ResultCode,
                                    ReasonCode
                                    )
```

The primitive parameters are defined in Table 3u.

6.3.16.3 MLME-MULTICAST-RX-SETUP.request

This primitive allows the DME to control multicast reception for frames with the DestID set to the McstID, as defined in 7.2.3, and to allow filtering for a particular stream index associated with the McstID. The semantics of this primitive are:

```
MLME-MULTICAST-RX-SETUP.request  (
                                 MulticastStatus,
                                 SrcID,
                                 StreamIndex
                                 )
```

The primitive parameters are defined in Table 3v.

6.3.16.4 MLME-MULTICAST-RX-SETUP.confirm

This primitive indicates the result of configuring the MAC for filtering multicast traffic. The semantics of this primitive are:

```
MLME-MULTICAST-RX-SETUP.confirm  (
                                 ResultCode,
                                 ReasonCode
                                 )
```

The primitive parameters are defined in Table 3v.

6.3.17 Timing synchronization

These primitives support the synchronization of upper layer functions with the beacon timing in the piconet. The beginning of the data preamble is observed on the air by all DEVs within a piconet while the delay between the observation and the delivery of the indication is known within a MAC by design (and communicated to the application by implementation-dependent means). The parameters used for these primitives are defined in Table 3w.

Table 3w—MLME-BEACON-EVENT primitive parameters

Name	Type	Valid range	Description
BeaconNumber	Integer	0–65535	The beacon number of the beacon that was received.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	NOT_SUPPORTED, OTHER	The reason for a ResultCode of FAILURE.

6.3.17.1 MLME-BEACON-EVENT.request

This primitive requests activation of the MAC synchronization support facility. The semantics of this primitive are:

```
MLME-BEACON-EVENT.request  ()
```

The primitive has no parameters.

6.3.17.2 MLME-BEACON-EVENT.confirm

This primitive confirms the activation of the MAC synchronization support facility. If the MAC does not support synchronization or if it encounters some other error, the response will have a ResultCode of FAILURE and the ReasonCode set to indicate the type of error. The semantics of this primitive are:

```
MLME-BEACON-EVENT.confirm       (
                                ResultCode,
                                ReasonCode
                                )
```

The primitive parameters are defined in Table 3w.

6.3.17.3 MLME-BEACON-EVENT.indication

This primitive indicates the beginning of a beacon's data preamble, whether the beacon is transmitted or correctly received by the MAC. The semantics of this primitive are:

```
MLME-BEACON-EVENT.indication    (
                                BeaconNumber
                                )
```

The primitive parameters are defined in Table 3w.

Delete 6.4 and all of its subclauses (including Table 31 and Table 32).

6.5 MAC management

6.5.1 MAC PIB PNC group

Change Table 33 as shown:

Table 33—MAC PIB PNC group parameters

Managed object	Octets	Definition	Access
MACPIB_CAPEndTime	2	The end time of the CAP interval in the superframes, 8.6.	Read only
MACPIB_SuperframeDuration	2	Duration of the superframe.	Read only
MACPIB_PNCCapable	1 bit	1 if the DEV has the capability to become the PNC, 0 otherwise.	Read only
MACPIB_PNCDesMode	1 bit	1 if it is desired that the DEV be the PNC.	Read/write ~~only~~
MACPIB_MaxPSSets	1	The maximum number of PS sets supported by the PNC.	Read only
MACPIB_BSID	6–32	Identifies the piconet.	Read only
MACPIB_MaxAssociatedDEVs	~~2~~ 1	As defined in 7.4.11	Read only
MACPIB_MaxCTRqBs	~~2~~ 1	As defined in 7.4.11	Read only
MACPIB_SEC	1 bit	Indicates if the DEV is capable of operating a secure piconet as the PNC.	Read only

Table 33—MAC PIB PNC group parameters *(continued)*

Managed object	Octets	Definition	Access
MACPIB_PNCServicesBroadcast	1	0x00 = PNC sends information about its services 0x01 = PNC will not send information about its services	Read/write
MACPIB_AllowedChannelSet	Variable	A set of channel indices, one for each channel that the MAC is allowed to use for scanning and starting piconets.	Read/write
MACPIB_AssocVendorSpecificIE	Variable	A vendor specific IE, as defined in 7.4.17, that is sent in the Association Response command, as described in 7.5.1.2, when the DEV is acting as the PNC.	Read/write
MACPIB_DesiredATP	2	The ATP value to send in an Association Request command.	Read/write
MACPIB_PNID	2	If associated with a piconet, the PNID of that piconet.	Read only
MACPIB_CAPData	1	Indicates if the initial setting of the CAP Data Allowed field in the beacon, as described in 7.3.1.1.	Read/write
MACPIB_CAPCommand	1	Indicates if the initial setting of the CAP Commands Allowed field in the beacon, as described in 7.3.1.1.	Read/write
MACPIB_CAPAssociation	1	Indicates if the initial setting of the CAP Data Allowed field in the beacon, as described in 7.3.1.1.	Read/write
MACPIB_PiconetMaxTXPower	1	The maximum power allowed for transmission during certain times in the superframe as described in 7.3.1.1.	Read/write
MACPIB_MCTAUsed	1	The initial setting of the MCTA Used field as described in 7.3.1.1.	Read/write
MACPIB_NextPNCCapable	1 bit	Indicates if the DEV is capable of participating in the Next PNC procedure as described in 8.2.3b. 0 = DEV is not capable 1 = DEV is capable.	Read only

6.5.2 MAC PIB characteristics group

Change and insert the following rows in Table 34. (Insert the new rows at the end of the table.) The other rows are unchanged and are not shown.

Table 34—MAC PIB implementation group parameters

Managed object	Octets	Definition	Access
MACPIB_DEVAddress	~~6~~8	The MAC address of the DEV	Read only
MACPIB_AlwaysAwake	1 bit	As defined in 7.4.11	Read only
MACPIB_ListenToSource	1 bit	As defined in 7.4.11	Read only
MACPIB_ListenToMulticast	1 bit	As defined in 7.4.11	Read only

Table 34—MAC PIB implementation group parameters *(continued)*

Managed object	Octets	Definition	Access
MACPIB_CTARelinquishCapable	1 bit	Indicates if the DEV is capable of using time relinquished in a CTA by another DEV. 0x00 - DEV does not support. 0x01 - DEV does support.	Read only
MACPIB_DlyACKCapable	1 bit	Indicates if the DEV is capable of using Dly-ACK as the source. 0 - DEV does not support. 1 - DEV does support.	Read only
MACPIB_ImpACKCapable	1 bit	Indicates if the DEV is capable of using implied ACK (Imp-ACK) as the source. 0 - DEV does not support. 1 - DEV does support.	Read only
MACPIB_STPCapable	1 bit	Indicates if the DEV is capable of using the stream timeout period (STP). 0 - DEV does not support 1 - DEV does support	Read only

6.6 MAC SAP

In 6.6, delete the first three paragraphs, from "The MAC provides both stream ..." through "... due to the bursty nature of the service."

Delete Table 35 and insert the following table in its place:

Table 35—Summary of MAC SAP primitives

Name	Request	Confirm	Indication	Response
MAC-ASYNC-DATA	6.6.1	6.6.2	6.6.3	–
MAC-ISOCH-DATA	6.6.4	6.6.5	6.6.6	–

Delete Table 36 and insert the following table in its place:

Table 36—MAC-ISOCH-DATA and MAC-ASYNC-DATA primitive parameters

Name	Type	Valid range	Description
RequestID	Integer	0–511	An identifier, which is used to correlate a request to a response. It is unique per stream index for outstanding isochronous requests. It is unique among all outstanding asynchronous requests.
StreamIndex	Integer	Any valid stream index, as defined in 7.2.5	The stream with which the data is associated.

Table 36—MAC-ISOCH-DATA and MAC-ASYNC-DATA primitive parameters *(continued)*

Name	Type	Valid range	Description
TransmitTimeout	Duration	$1–(2^{32} – 1)$	Maximum allowed delay in microseconds from when the MSDU is been presented to the MAC SAP until the frame has finished transmission and the acknowledgment, if required is successfully received.
MaxRetries	Integer	0–255	Specifies the maximum number of retries to attempt per transmitted frame with a maximum value no greater than the MaxRetries value supplied in the original stream creation request.
SECMode	Boolean	TRUE, FALSE	Indicates if security is to be applied to the MSDU or if the MSDU was received securely.
UserPriority	Integer	As defined in Table A.1	User priority of the stream, as described in Table A.1.
ACKRequested	Boolean	TRUE, FALSE	Indicates if the request requires an acknowledgment, as described in 8.8, of the MSDU at the MAC layer.
ConfirmRequested	Enumeration	NEVER, ALWAYS, ON_ERROR	Indicates when a confirm primitive is required for the request.
SNAPHeaderPresent	Boolean	TRUE, FALSE	TRUE indicates that a logical link control/subnetwork access protocol (LLC/SNAP) header is present in the data frame.
Length	Integer	0–pMaxTransferUnitSize	The length of the MSDU in octets.
Data	Variable number of octets.	Any octet string the length of which is given by the Length parameter	MSDU portion of the primitive.
TransmitDelay	Duration	$1–(2^{32} – 1)$	Delay in microseconds from when the MSDU is presented to the MAC SAP until the frame has finished transmission and the acknowledgment, if required, has been successfully received. If the transmission fails due to timeout, this field shall be set to the TransmitTimeout value for this frame.
TrgtID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the target DEVID of an MSDU.
OrigID	Integer	Any valid DEVID, as defined in 7.2.3	Specifies the originator DEVID of an MSDU.
ResultCode	Enumeration	SUCCESS, FAILURE	Indicates the result of the MLME request.
ReasonCode	Enumeration	TRANSMIT_TIMEOUT, MAX_RETRIES, NOT_ASSOCIATED, OTHER	The reason for a ResultCode of FAILURE.

6.6.1 MAC-ASYNC-DATA.request

Change the first paragraph in 6.6.1 as shown:

This primitive is used to initiate the transfer of an ~~asynchronous data~~ MSDU from one MAC entity to another MAC entity or entities. If SNAPHeaderPresent is TRUE, then the MAC will use the SNAP data frame type for transmission. All asynchronous data should use LLC/SNAP headers to avoid interoperability problems in piconets where different FCSLs are being used. The semantics of this primitive are:

MAC-ASYNC-DATA.request	(RequestID, TrgtID, ~~OrigID~~ ~~Priority~~ ~~ACKPolicy~~ ~~Transmission~~TransmitTimeout, MaxRetries, SNAPHeaderPresent, UserPriority, ACKRequested, ConfirmRequested, Length, Data)

Delete 6.6.1.1 and 6.6.1.2.

6.6.2 MAC-ASYNC-DATA.confirm

Change the first paragraph in 6.6.2 as shown:

This primitive is used to report the result of a request to transfer an asynchronous MSDU from one MAC entity to another MAC entity or entities. This primitive is only generated if the ConfirmRequested parameter in the MAC-ASYNC-DATA.request with the same RequestID value is ALWAYS or is ON_ERROR and the ResultCode is FAILURE. ~~inform the FCSL of a successful delivery or a failed delivery.~~ The semantics of this primitive are:

MAC-ASYNC-DATA.confirm	(~~TrgtID,~~ ~~OrigID,~~ ~~Priority,~~ RequestID, TransmitDelay, ResultCode, ReasonCode)

Delete 6.6.2.1 and 6.6.2.2.

6.6.3 MAC-ASYNC-DATA.indication

Change the primitive definition in 6.6.3 as shown:

MAC-ASYNC-DATA.indication	(TrgtID, OrigID, SNAPHeaderPresent, Length, Data)

Delete 6.6.3.1 and 6.6.3.2.

6.6.4 MAC-ISOCH-DATA.request

Change the primitive definition in 6.6.4 as shown:

MAC-ISOCH-DATA.request	(RequestID, StreamIndex, ~~ACKPolicy,~~ ~~Transmission~~TransmitTimeout, MaxRetries, SNAPHeaderPresent, ACKRequested, ConfirmRequested, Length, Data)

Delete 6.6.4.1 and 6.6.4.2.

6.6.5 MAC-ISOCH-DATA.confirm

Change the first paragraph in 6.6.5 as shown:

This primitive is used to report the result of a request to transfer an isochronous MSDU from one MAC entity to another MAC entity or entities. This primitive is only generated if the ConfirmRequested parameter in the MAC-ISOCH-DATA.request with the same RequestID value is ALWAYS or is ON_ERROR and the ResultCode is FAILURE. ~~inform the FCSL of a successful delivery or a failed delivery~~. The semantics of this primitive are:

MAC-ISOCH-DATA.confirm	(RequestID, StreamIndex, TransmitDelay, ResultCode, ReasonCode)

Delete 6.6.5.1 and 6.6.5.2.

6.6.6 MAC-ISOCH-DATA.indication

Change the primitive definition in 6.6.6 as shown:

MAC-ISOCH-DATA.indication	(TrgtID, OrigID, StreamIndex, SNAPHeaderPresent, Length, Data)

Delete 6.6.6.1 and 6.6.6.2.

Delete 6.7 and all of its subclauses (including Table 37 and Table 38).

7. MAC frame formats

In Clause 7, delete the following:

- ***The first paragraph, which states "This clause specifies ...." This paragraph provides only introductory material and no technical information.***
- ***The second paragraph, which states "The MAC in all DEVs ...." Insert it as the first paragraph in 7.1.***
- ***The third paragraph, which states "For a frame to be correctly ...." An edited version of this paragraph is being inserted in 8.1.***

7.1 Frame format conventions

Insert the following paragraph (formerly the second paragraph in Clause 7) as the first paragraph in 7.1:

The MAC in all DEVs shall be able to validate the error free reception of every frame from the PHY using the frame check sequence (FCS). Note that the PHY only passes frames to the MAC that have passed the header check sequence (HCS) test. In addition, every DEV shall be able to construct a subset of the command frames for transmission and to decode another (potentially different) subset of the command frames upon reception. The particular subsets of these commands that a DEV shall construct and decode are determined by the functional capabilities supported by that particular DEV.

Change the last paragraph in 7.1 as shown:

All DEVs shall be assigned a DEV address, which is the 64-bit address as defined by IEEE Std 802®-2001. An individual MAC address is a MAC address with the group bit set to zero, as defined in IEEE Std 802-2001.

7.2 General frame format

7.2.1 Frame control

Replace Figure 9 with the following:

bits: b15–b14	b13	b12	b11	b10	b9	b8–b7	b6	b5–b3	b2–b0
Reserved	CTA relinquish	Imp-ACK NAK	Imp-ACK request	More data	Retry	ACK policy	SEC	Frame type	Protocol version

Figure 9—Frame control field format

7.2.1.2 Frame type

Change and insert the following rows in Table 39 as shown. The other rows are unchanged and are not shown.

Table 39—Valid frame type values
(numeric values in this table are shown in binary)

Type value b5 b4 b3	Frame type description	Subclause
101	LLC/SNAP data frame	7.3.5
~~101~~ 110–111	Reserved	

Change the title of 7.2.1.4 as shown:

7.2.1.4 ACK policy and implied ACK (Imp-ACK) request

Change first paragraph in 7.2.1.4 as shown:

The ACK Policy field ~~is~~ and Imp-ACK Request fields are used to indicate the type of acknowledgment procedure that the addressed recipient is required or allowed to perform. The use of the ~~ACK Policy field~~ ACK procedures is described in 8.8. The allowed values of the ACK Policy field and the Imp-ACK Request field are defined Table 40. The ACK policy of a frame is determined by the combination of the ACK Policy field and the Imp-ACK Request field.

Delete Table 40 and insert the following table in its place:

Table 40—Valid ACK Policy field type values (numeric values in this table are shown in binary)

Imp-ACK Request field b11	ACK Policy field b8 b7	ACK policy type	Description
0	00	No ACK	The recipient(s) does not acknowledge the transmission, and the sender treats the transmission as successful without regard for the result, as described 8.8.1.
0	01	Immediate ACK (Imm-ACK)	The addressed recipient returns an Imm-ACK frame after successful reception, as described in 8.8.2.
0	10	Delayed ACK (Dly-ACK)	The addressed recipient keeps track of the frames received with this policy until requested to respond with a Dly-ACK frame, as described in 8.8.3.
0	11	Dly-ACK request	The addressed recipient returns either an Imm-ACK or a Dly-ACK frame after successful reception, as described in 8.8.3.
1	01	Imp-ACK	The addressed recipient returns an Imm-ACK frame, a data frame, or a command frame after successful reception, as described in 8.8.4.

7.2.1.6 More data

After 7.2.1.6, insert the following subclauses as 7.2.1.7 and 7.2.1.8:

7.2.1.7 Imp-ACK negative acknowledgment (NAK)

The Imp-ACK NAK field shall be set to one by a DEV when all of the following conditions are true:

— The DEV is responding to a frame addressed to the DEV for which it has successfully received the MAC header with an ACK policy of Imp-ACK.

— The FCS check, 7.2.7.6, for the frame body failed.

The Imp-ACK NAK field shall be set to zero otherwise. The use of the Imp-ACK NAK bit is described in 8.8.4.

7.2.1.8 CTA relinquish

The CTA Relinquish field shall be set to one when the DEV relinquishes CTA ownership to another DEV. It shall be set to zero otherwise. The use of the CTA Relinquish field is described in 8.4.3.8.

7.2.3 SrcID and DestID

Change the second paragraph in 7.2.3 as shown:

The maximum number of valid DEVs, mMaxNumValidDEVs, is the maximum number of DEVIDs that the PNC is able to allocate in a piconet. This includes all of the regular DEVIDs, the PNCID, the GrpIDs, and the NbrIDs, but not the reserved IDs, the BcstID, McstID, or the UnassocID.

7.2.4 Fragmentation control

Insert the following paragraph after Figure 10 in 7.2.4:

The three octets that compose the Fragmentation Control field may be used for reporting PHY-dependent receive status information to the transmitting DEV in Imm-ACK and Dly-ACK frames. If the source DEV is not reporting PHY-dependent receive status information in an Imm-ACK or Dly-ACK frame, it shall set the fragmentation field of the frame to all zeros, i.e., 0x000000. All other values are PHY-dependent. The receive status for the 2.4 GHz PHY is defined in 11.7.

7.2.4.1 MSDU number

Change the third paragraph in 7.2.4.1 as shown:

Each MSDU number counter shall be set to zero when the DEV is initialized. The MSDU number counter for an isochronous stream shall be set to zero when the stream index for the isochronous stream is first assigned by the PNC, as described in 8.5.1.1.

7.2.5 Stream index

Change 7.2.5 as shown:

The Stream Index field reserved values are:

- 0x00 reserved for asynchronous data
- 0xFD reserved for MCTA traffic
- 0xFE reserved for unassigned streams
- 0xFF reserved for future use

DEVs use other values of the stream index as dynamically assigned by the PNC during the setup of the data stream, as described in 7.5.6.1. The PNC allocates a unique stream index value for each isochronous stream in the piconet. A DEV shall support each possible stream index value.

7.2.6 MAC header validation

Change the first sentence in 7.2.6 as shown:

When the PHY receives a frame, it validates the received frame's MAC header before passing the MAC header and ~~and~~ its associated MAC frame body to the MAC.

7.2.7 MAC frame body

7.2.7.3 Secure frame counter (SFC)

Change 7.2.7.3 as shown:

The Secure Frame Counter field shall be included in the frame body of all secure frames. The Secure Frame Counter field contains a 2-octet counter that is used to ensure the uniqueness of the nonce in a secure frame. A DEV shall not reuse a frame counter with the same time token, 7.3.1.1, and key, 9.3.5. The DEV shall initialize the SFC to zero for the first frame sent and increment it for each successive secure frame sent. When the time token, 7.3.1, is updated, the DEV shall reset the SFC to zero. ~~may reset the SFC to zero if desired or allow the counter to roll over.~~ In the case where the DEV receives a new key, the DEV shall set the SFC to zero.

7.3 Format of individual frame types

7.3.1 Beacon frame

7.3.1.1 Non-secure beacon frame

Replace Figure 11 with the following:

octets: 4	L_n	…	L_1	21
FCS	Information element-n	…	Information element-1	Piconet synchronization parameters
Beacon frame body				

10
MAC header

Figure 11—Non-secure beacon frame format

Replace Figure 13 with the following:

bits: b7	b6	b5	b4	b3	b2	b1	b0
CTA IEs unchanged	Other IEs unchanged	Reserved	SEC mode	MCTA used	CAP association	CAP commands	CAP data

Figure 13—Piconet mode field

Insert the following paragraphs after Figure 41 in 7.3.1.1:

The Other IEs Unchanged bit may be set to one if all the IEs (other than CTA IEs) in the beacon payload are identical to the IEs (other than CTA IEs) contained in the previous beacon, as described in 8.6.2.

The CTA IEs Unchanged bit may be set to one if the CTA IEs in the beacon are identical to the CTA IEs contained in the previous beacon, as described in 8.6.2.

7.3.2 Acknowledgment frames

7.3.2.1 Immediate ACK (Imm-ACK) frame

Change the row for the Fragmentation Control field in Table 44 as shown. The other rows are unchanged and are not shown.

Table 44—MAC header settings of an ~~immediate~~ Imm-ACK frame

Header field	Setting on transmission	Interpretation on reception
Fragmentation control	0x000000 or Receive Status value, 7.2.4	May be ignored

7.3.2.2 Delayed ACK (Dly-ACK) frame

Change the row for the Fragmentation Control field in Table 45 as shown. The other rows are unchanged and are not shown.

Table 45—MAC header settings of a Dly-ACK frame

Header field	Setting on transmission	Interpretation on reception
Fragmentation control	0x000000 or Receive Status value, 7.2.4	May be ignored

Change the fifth paragraph in 7.3.2.2 as shown:

The MPDUs ACKed field shall contain the number of MPDUs that are being ACKed with this frame. ~~This field shall be greater than or equal to 1.~~

7.3.4 Data frame

7.3.4.2 Secure data frame

Replace Figure 23 with the following:

octets: 4	8	L_n	2	2
FCS	Integrity code	Data payload	SFC	SECID

10
MAC header

Figure 23—Secure data frame format

Change the fifth paragraph in 7.3.4.2 as shown:

The Data Payload field is limited by the maximum allowed for the MAC frame body, as defined in 7.2. If the symmetric key security operations in use require~~s~~ data encryption, the Data Payload field shall be encrypted.

After 7.3.4.2, insert the following subclauses as 7.3.5, 7.3.5.1, and 7.3.5.2:

7.3.5 LLC/SNAP data frame

7.3.5.1 Non-secure LLC/SNAP data frame

The non-secure LLC/SNAP Data frame is identical to a Non-secure Data frame, as defined in 7.3.4.1, except that the Data Payload field includes an LLC/SNAP header as the first octets in the payload, as defined in A.1. The size of the combination of the data and LLC/SNAP header shall not exceed the limits for the Data Payload field, as defined in 7.3.4.1. A Non-secure LLC/SNAP Data frame shall be formatted as illustrated in Figure 22.

The frame type shall be set to the LLC/SNAP data frame value in Table 39 and the SEC bit shall be set to zero. The other fields in the MAC header take on values that are appropriate for that particular data frame. All fields in the MAC header of a Non-secure LLC/SNAP Data frame shall be decoded on reception.

7.3.5.2 Secure LLC/SNAP data frame

The Secure LLC/SNAP Data frame is identical to a Secure Data frame, as defined in 7.3.4.2, except that the Data Payload field includes an LLC/SNAP header as the first octets in the payload, as defined in A.1. The size of the combination of the data and LLC/SNAP header shall not exceed the limits for the Data Payload field, as defined in 7.3.4.1. A Secure LLC/SNAP Data frame shall be formatted as illustrated in Figure 23.

The frame type shall be set to the LLC/SNAP data frame value in Table 39 and the SEC bit shall be set to one. The other fields in the MAC header take on values that are appropriate for that particular data frame. All fields in the MAC header of a Secure LLC/SNAP Data frame shall be decoded on reception.

7.4 Information elements

Change and insert the following rows in Table 48 as shown. The other rows are unchanged and are not shown.

Table 48—Information elements

Element ID hex value	Element	Subclause	Present in beacon
0x10	Group ID	7.4.18	Non-beacon IE
0x11	Stream renew	7.4.19	Non-beacon IE
0x12	Next PNC	7.4.20	As needed
0x13	Piconet channel status	7.4.21	Non-beacon IE
0x14~~0x10~~–0x7F	Reserved		

7.4.2 BSID

Insert the following note after the second paragraph in 7.4.2:

NOTE—The BSID is not null terminated. The length of the field is determined from the length field of the IE.[1]

7.4.7 Application specific

Replace Figure 34 with the following:

octets: L_n	3	1	1
Application specific data	OUI	Length (=3+L_n)	Element ID

Figure 34—Application specific information element format

Change the second paragraph in 7.4.7 as shown:

The ~~Vendor~~ OUI field is the OUI assigned by the IEEE standards association registration authority committee (RAC), which shall be the sole registration authority. ~~A value of vendor OUI not understood by a receiving DEV causes the remainder of this IE to be ignored.~~ If a value of the OUI field is not understood by the receiving DEV, that DEV shall ignore the remainder of the associated ASIE.

[1]Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.

7.4.10 CTA status

Replace Figure 37 with the following:

octets: 2	2	1	1	1	1	1	1
Start beacon number	CTA sub-rate	CTRq info	Stream index	SrcID	DestID	Length (=8)	Element ID

Figure 37—CTA status information element format

In 7.4.10, delete the fifth paragraph, which states "The CTRq Control field is defined in 7.5.6.1." Insert the following paragraphs and figure in its place:

The CTRq info field shall be formatted as illustrated in Figure 37a.

bits: b7	b6	b5	b4	b3	b2–b0
Reserved	CTA rate type	CTA type	PM CTRq type	Terminate	Priority

Figure 37a—CTRq info field format

The Priority field is defined in A.1.

The Terminate bit shall be set to one if the stream has been terminated. Otherwise it shall be set to zero.

The PM CTRq Type bit is defined in 7.5.6.1.

The CTA Type bit is defined in 7.5.6.1.

The CTA Rate Type bit is defined in 7.5.6.1.

Change the sixth paragraph in 7.4.10 as shown:

The CTA Sub-Rate ~~Factor~~ field is set to the number of beacons between every CTA, as described in 7.5.6.1. If one or more CTAs are allocated per superframe, this value shall be set to zero.

7.4.11 Capability

Replace Figure 41 with the one shown:

bits: b7	b6	b5	b4	b3	b2–b0
PNC capable	PNC Des-mode	SEC	PSRC	Next PNC capable	Reserved

Figure 41—PNC rating field format

Insert the following paragraph after Figure 41 in 7.4.11:

The Next PNC Capable bit shall be set to one if the DEV is capable of performing the next PNC functionality as described in 8.2.3b. Otherwise the bit shall be set to zero.

Replace Figure 42 with the following:

bits: 7	b6	b5	b4	b3	b2	b1	b0
Preferred fragment size			Supported data rates				

bits: b15	b14	b13	b12	b11	b10	b9	b8
Reserved	STP	CTA relinquish	Imp-ACK	Dly-ACK	Listen to Multicast	Listen to Source	Always AWAKE

bits: b23	b22	b21	b20	b19	b18	b17	b16
Reserved							

Figure 42—DEV capabilities field format

Insert the following paragraphs after the seventeenth paragraph in 7.4.11:

The Dly-ACK bit shall be set to one if the DEV is capable of performing the Dly-ACK procedure as defined in 8.8.3. Otherwise the bit shall be set to zero.

The Imp-ACK bit shall be set to one indicate that the DEV is capable of performing the Imp-ACK procedure as defined in 8.8.4. Otherwise the bit shall be set to zero.

The CTA Relinquish bit shall be set to one if the DEV is capable of participating in the CTA relinquish procedure as defined in 8.4.3.8. Otherwise the bit shall be set to zero.

The STP bit shall be set to one to indicate that the DEV is capable of renewing its streams within the stream timeout period, as defined in 8.3.4. Otherwise the bit shall be set to zero.

7.4.16 Piconet services

Replace Figure 47 with the following:

octets: (0–127)	3	1	1	1
Piconet services	OUI	DEVID	Length (=0 to 131)	Element ID

Figure 47—Piconet services information element format

Change the third paragraph in 7.4.16 as shown:

The ~~Vendor~~ OUI field is defined in 7.4.7. If the value of the DEVID field is the PNCID, then the Piconet Services field is not present and the OUI field is 1 octet in length. In this case, the OUI field contains the following information:

0 - Reserved

1 - Broadcast of Piconet Services IE not allowed

2 - Piconet Services IE not supported

3-254 - Reserved

255 - Other failure

After 7.4.17, insert the following subclauses as 7.4.18 through 7.4.21:

7.4.18 Group ID

The Group ID IE is used to list the DEVs that are a member of a multicast group. The Group ID IE shall be formatted as illustrated in Figure 48a.

octets: 1–32	**1**	**1**	**8**	**1**	**1**
Group IDs	Start DEVID	GrpID	Group address	Length (=11 to 42)	Element ID

Figure 48a—Group ID information element format

The Group Address field is a 64-bit group address, as defined in IEEE Std 802-2001.

The GrpID field contains the DEVID that has been assigned by the PNC for the address in the Group Address field.

The Start DEVID field indicates the DEVID that corresponds to the first bit in the Group IDs field.

The Group IDs field contains a bitmap of 1 to 32 octets in length. Each bit of the Group IDs field when set to one indicates the DEV whose DEVID is equal to the start DEVID plus the bit position in the Group ID bit-map is a member of the multicast group identified by the Group Address and GrpID fields. The bits in the Group IDs field are set to zero otherwise. The bit position 0, i.e., the first bit or lsb of the bitmap corresponds to the start DEVID.

The bits corresponding to the PNCID, UnassocID, BcstID, McstID, NbrIDs and the reserved DEVIDs, 7.2.3, shall be set to zero upon transmission by the PNC and shall be ignored upon reception.

7.4.19 Stream renew

The Stream Renew IE is used by a DEV to renew the STP for streams for which it is the source. The Stream Renew IE shall be formatted as illustrated in Figure 48a.

octets: 0–252	**…**	**1**	**1**	**1**	**1**
Stream index n	…	Stream Index 2	Stream Index 1	Length (=0 to 252)	Element ID

Figure 48b—Stream renew information element format

The stream index fields contain the stream indices that the DEV wants to maintain.

7.4.20 Next PNC

The Next PNC IE is used to inform the members of the piconet which DEV will become the PNC after an implicit handover as described in 8.2.3b. The Next PNC IE shall be formatted as illustrated in Figure 48c.

octets: 2	**1**	**1**	**1**
Next PNID	Next PNC	Length (=3)	Element ID

Figure 48c—Next PNC information element format

The Next PNC field contains the DEVID of the DEV that the PNC has chosen to be the Next PNC.

The Next PNID field is the PNID that the next PNC shall use if it takes over as PNC as described in 8.2.3b.

7.4.21 Piconet channel status

The Piconet Channel Status IE is used to report those DEVs from which the source DEV has correctly received a MAC header. The Piconet Channel Status IE shall be formatted as illustrated in Figure 48d.

octets: 1	32	1	1
Channel status details	Bitmap of DEVs in range	Length (=3)	Element ID

Figure 48d—Piconet channel status information element format

The Bitmap of DEVs in Range field is a listing of all of the DEVs in the piconet that were heard by the DEV sending the command. The bit position zero, i.e., the first bit or lsb of the bitmap, corresponds to DEVID zero. The bit corresponding to a DEVID is set to one if a MAC header was correctly received from a DEV with that DEVID since the last Piconet Channel Status IE was sent. Otherwise the bit shall be set to zero.

The Channels Status Details field shall be set to one if the DEV scanned for power save DEVs. Otherwise this value shall be set to zero.

7.5 MAC command types

Change and insert the following rows in Table 50 as shown. The other rows are unchanged and are not shown.

Table 50—Command types

Command type hex value b15–b0	Command name	Subclause	Associated	Secure membership (if required)
0x001F	Announce response	7.5.5.3	X	
0x0020	PM mode change response	7.5.8.6	X	X
0x0021	ASIE request	7.5.9.3	X	X
0x0022	ASIE response	7.5.9.4	X	X
0x0023	Multicast configuration request	7.5.10.1	X	X
0x0024	Multicast configuration response	7.5.10.2	X	X
~~0x001D~~0x0025–0x00FF	Reserved			

7.5.1 Association and disassociation commands

7.5.1.2 Association response

Change the dashed list in the fifth paragraph in 7.5.1.2 as shown:

— 9–~~255~~254 –> Reserved

— 255 –> Other failure

7.5.1.3 Disassociation request

Change the dashed list in the second paragraph in 7.5.1.3 as shown:

— 5–~~255~~254 –> Reserved

— 255 –> Other failure

7.5.3 PNC handover commands

7.5.3.1 PNC handover request

Change the fifth paragraph in 7.5.3.1 as shown:

The allowed values of the Handover Status field are: ~~shall be set to zero when the PNC is starting the PNC handover process with the destination DEV. It shall be set to one if the PNC is cancelling the handover process with the destination DEV.~~

0 –> The PNC is starting the PNC handover process with the destination DEV.

1 –> The PNC is cancelling the handover process with the destination DEV.

2 –> The PNC is only transferring information and is not beginning the PNC handover process.

3 –> The PNC is starting the PNC handover process with information that was sent previously, as described in 8.2.3a.

4–255 –> Reserved.

7.5.4 Information request commands

7.5.4.5 Probe request

Change the seventh paragraph in 7.5.4.5 as shown:

If the IEs Requested field indicates that the CTA Status IE, as described in 7.4.10, is being requested from the destination DEV, the first octet of the Request Index field is set to the stream index of the stream for which CTA information is requested. If the Request Index field is set to zero, the DEV is requesting information about all isochronous streams directed to the requesting DEV and to the ~~BcstId~~ BcstID and ~~MestId~~ McstID. If the Information Requested field indicates that the CTA Status IE is not being requested from the destination DEV, the Request Index field has no meaning and shall be set to zero.

Change and insert the following rows in Table 51 as shown. The other rows are unchanged and are not shown.

Table 51—Rules for requesting IEs in a Probe Request command

Information element	Subclause	PNC allowed to request?	DEV allowed to request?
CTA status	7.4.10	Shall not request	~~Shall not~~ May request
Group ID	7.4.18	Shall not request	May request
Stream renew	7.4.19	Shall not request	Shall not request
Next PNC	7.4.20	Shall not request	Shall not request
Piconet channel status	7.4.21	May request	May request

7.5.4.6 Probe response

Change and insert the following rows in Table 52 as shown. The other rows are unchanged and are not shown.

Table 52—Rules for responding to requests in Probe commands

Information element	Subclause	DEV receives request from DEV	DEV receives request from PNC	PNC receives request from DEV
CTA status	7.4.10	Shall ignore	Shall ignore	Shall ~~ignore~~ respond
Group ID	7.4.18	Shall ignore	Shall ignore	Shall respond
Stream renew	7.4.19	Shall ignore	Shall ignore	Shall ignore
Next PNC	7.4.20	Shall ignore	Shall ignore	Shall ignore
Piconet channel status	7.4.21	May respond	May respond	May respond

7.5.5 Information announcement commands

7.5.5.2 Announce

Insert the following rows at the end of Table 53 as shown. The other rows are unchanged and are not shown.

Table 53—Rules for sending IEs in an Announce command

Information element	Subclause	PNC allowed to send?	DEV allowed to send?
Group ID	7.4.18	May send	Shall not send
Stream renew	7.4.19	Shall not send	May send
Next PNC	7.4.20	May send	Shall not send
Piconet channel status	7.4.21	May send	May send

After 7.5.5.2, insert the following subclause as 7.5.5.3:

7.5.5.3 Announce response

The Announce Response command is used to confirm the receipt of IEs from another DEV. The individual IEs used in this frame are described in 7.4. The Announce command shall be formatted as illustrated in Figure 71a.

octets: 2	**…**	**2**	**2**	**2**	**2**
Announce response block n	…	Announce response block 2	Announce response block 1	Length (=2*n)	Command type

Figure 71a—Announce response command format

The announce response blocks shall be formatted as illustrated in Figure 71b.

octets: 1	**1**
Reason code	IE element ID

Figure 71b—Announce response command format

The IE Element ID field contains the element ID of the IE for which the response is being sent.

The Reason Code field indicates the result of the Announce command for the IE specified in the IE Element ID field. The valid values of the Reason Code field are:

0 –> Success

1 –> Unsupported IE

2–254 –> Reserved

255 –> Other failure

7.5.6 Channel time allocation request, modification, and termination commands

7.5.6.1 Channel time request

Change the seventh paragraph in 7.5.6.1 as shown:

The Stream Index field is defined in 7.2.5. In the case where the DEV is requesting the creation of an isochronous stream, it is set to the unassigned stream value, as described in 7.2.5, by the originating DEV. In the case where the DEV is requesting the reservation or termination of an asynchronous channel time, it is set to the asynchronous stream value, as described in 7.2.5. When the stream index is other than the unassigned stream index or asynchronous stream index value, this CTRq is a request to modify or terminate an existing CTA. In the case where the DEV is requesting a specific MCTA interval, as described in 8.4.3.3, the stream index shall be set to the MCTA stream value, as described in 7.2.5, and the Target ID List field shall contain only the PNCID.

Replace Figure 74 with the following:

bits: b7	b6	b5	b4	b3	b2–b0
Target ID list type	CTA rate type	CTA type	PM CTRq type	Reserved	User priority

Figure 74—CTRq control format

Change the ninth paragraph in 7.5.6.1 as shown:

The User Priority field is defined in Table A.1.

Change the thirteenth and fourteenth paragraphs in 7.5.6.1 as shown:

The CTA Rate Factor field in conjunction with the CTA Rate Type field specifies the frequency at which the requesting DEV would like the PNC to allocate channel time. In the case of a super-rate request, the PNC will interpret the CTA Rate Factor as the maximum spacing allowed between CTAs allocated in a superframe, as described in D1.1.3.

For instance, in the case where the CTA Rate Type field is set to zero, a value indicating a super-rate CTA request, and the CTA Rate Factor field contains a value N greater than zero, the requesting DEV is requesting super-rate CTAs from the PNC. If these super-rate CTAs~~,~~ are allocated by the PNC, they will appear N times per superframe. A PNC shall support at least 8 CTAs per stream in the same superframe. The CTA Rate Type field set to zero and the CTA Rate Factor field set to zero shall be reserved.

Insert the following note after the fourteenth paragraph in 7.5.6.1:

NOTE—A subrate request always has a CTA Rate Factor greater than one. Thus a CTA Rate Type equal to one and a CTA Rate Factor equal to one is not allowed.

Change nineteenth, twentieth, and twenty-first paragraphs in 7.5.6.1 as shown:

For an isochronous request, the Minimum Number of TUs field indicates the minimum number of CTRq TUs ~~per CTA~~ required by the originating DEV to support the stream.

For an isochronous request, the Desired Number Of TUs field indicates the number of CTRq TUs ~~per CTA~~ that is desired by the requesting DEV. The Desired Number Of TUs field shall be greater than or equal to the Minimum Number Of TUs field.

~~For isochronous requests, the Minimum Number Of TUs and the Desired Number Of TUs are the number of TUs per CTA Rate Factor requested by the DEV.~~ In the case of an isochronous super-rate allocation, ~~it is~~ the Minimum Number of TUs and the Desired Number of TUs are the number of TUs requested in each superframe. In the case of an isochronous sub-rate allocation, ~~it is~~ the fields contain the number of TUs requested in each of the superframes containing the sub-rate CTA. For example, a request for a Minimum Number Of TUs of 4 with a sub-rate CTA Rate Factor of 4 indicates that the DEV is requesting 4 TUs every fourth superframe. Likewise, a request for a Minimum Number of TUs of 11 with a super-rate CTA Rate Factor of 4 indicates that the DEV is requesting at least 11 TUs per superframe, spread into 4 allocations that are evenly spaced in the superframe.

7.5.6.2 Channel time response

Change the enumerated list in the seventh paragraph in 7.5.6.2 as shown:

13–> PNC handover in progress

14–> STP expired

~~13–255~~15–254–> Reserved

255–> Other failure

7.5.7 Channel status commands

7.5.7.2 Channel status response

Change the fourth and fifth paragraphs in 7.5.7.2 as shown:

The RX Frame Count field contains the total number of frames, not including Imm-ACK frames, that were correctly received by the sender of this command, as described in 8.1. Only the directed frames transmitted by the destination of this command intended for the sender of this command are included.

The RX Frame Error Count field contains the total number of frames, not including Imm-ACK frames, that were received in error by the sender of this command from the destination of this command. A frame is considered to have been received in error if the ~~header passes the HCS validation but the frame body fails the FCS validation.~~ header is correctly received but the frame is not correctly received, as described in 8.1.

7.5.7.4 Remote scan response

Change the enumerated list in the second paragraph in 7.5.7.4 as shown:

3–~~255~~254–> Reserved

255–> Other failure

7.5.8 Power management commands

7.5.8.4 SPS configuration response

Change the enumerated list in the second paragraph in 7.5.8.4 as shown:

4–> Unique Wake Beacon ~~interval~~ Interval required

5–> DSPS set deleted by PNC

6–> PNC handover in progress

~~5–255~~7–254–> Reserved

255–> Other failure

After 7.5.8.5, insert the following subclause as 7.5.8.6:

7.5.8.6 PM mode change response

The PM Mode Change Response command is sent by the PNC as a response to a PM Mode Change Request command received from a DEV. The PM Mode Change Response command shall be formatted as illustrated in Figure 88a.

octets: 1	1	2	2
Reason code	PM mode	Length (=2)	Command type

Figure 88a—PM mode change response command format

The PM Mode field is defined in 7.5.8.5 and is set by the PNC to indicate current PM mode of the DEV.

The Reason Code field indicates the result of the attempt by the DEV to change PM modes. The valid reason codes are:

0–> Success

1–> Not a member of any existing SPS sets

2–254–> Reserved

255–> Other failure

7.5.9 Special commands

7.5.9.2 Vendor specific

After 7.5.9.2, insert the following subclauses as 7.5.9.3 through 7.5.10.2:

7.5.9.3 ASIE request

The ASIE request command is used to send an ASIE to the PNC to be put in the beacon. The DestID shall be set to the PNCID. The ASIE request command shall be formatted as illustrated in Figure 90a.

octets: n	1	1	1	2	2
ASIE	ASIE index	Request type	Request ID	Length (=3+n)	Command type

Figure 90a—ASIE request command format

The Request ID field is an identifier generated by the originating DEV that is unique among the DEV's ASIE requests.

The Request Type field indicates the type request. The valid values are:

0–> Add

1–> Modify

2–> Remove

3–255–> Reserved

The ASIE Index field is assigned by the PNC and is used to identify the ASIE for the request. If the Request Type field is "add," then the ASIE Index field shall be ignored by the PNC.

The ASIE field is defined in 7.4.7. If the Request Type field is "remove," then the length of Application Specific Data field in the ASIE shall be zero.

7.5.9.4 ASIE response

The ASIE request command is used to respond to a request to put an ASIE in the beacon. The SrcID shall be set to the PNCID. The ASIE request command shall be formatted as illustrated in Figure 90b.

octets: 1	1	1	2	2
Reason code	ASIE index	Request ID	Length (=3)	Command type

Figure 90b—ASIE response command format

The Request ID field is defined in 7.5.9.3.

The ASIE Index field is defined in 7.5.9.3.

The valid values of the Reason Code field are:

0–> Success

1–> Request rejected

2–> Unknown ASIE index

2–254–> Reserved

255–> Other failure

7.5.10 Multicast configuration commands

7.5.10.1 Multicast configuration request

The Multicast Configuration Request command is used to request a GrpID, 8.5.3. The DestID shall be set to the PNCID. The Multicast Configuration Request command shall be formatted as illustrated in Figure 90c.

octets: 8	1	2	2
Group address	Action	Length (=9)	Command type

Figure 90c—Multicast configuration request command format

The valid values of the Action field are:

0–> Join

1–> Leave

2–255–> Reserved

The Group Address field is defined in 7.4.17.

7.5.10.2 Multicast configuration response

The Multicast Configuration Response command is used by the PNC to respond to a request for a GrpID, as described in 8.5.3. The SrcID shall be set to the PNCID. The Multicast Configuration Response command shall be formatted as illustrated in Figure 90d.

octets: 1	1	8	2	2
Reason code	GrpID	Group address	Length (=10)	Command type

Figure 90d—Multicast configuration response command format

The Group Address field is defined in 7.4.17.

If the request for an GrpID was successful, the GrpID field is the DEVID, as defined in 7.2.3, that has been assigned by the PNC for the address in the Group Address field. Otherwise, the GrpID field shall be set to zero.

The valid values of the Reason Code are:

0–> Success

1–> Failure, lack of DEVIDs

2–> Failure, handover in progress

3–> Failure, resources unavailable

4–> Failure, not a valid group address

5–254–> Reserved

255–> Other failure

8. MAC functional description

8.1 Introduction

Change the seventh paragraph in 8.1 as shown:

An example MSC is shown in Figure 91 that illustrates three ~~two~~ MLME requests and the associated time-outs. In the first case, the request completes before the timeout expires and so the confirm returns with the ResultCode set equal to SUCCESS ~~COMPLETED~~. In the second case, the requested action completes unsuccessfully before the timer expires and so the confirm primitive is returned with the ResultCode set equal to FAILURE and the ReasonCode indicates the reason for the failure, if known. In the third case, the requested action does not complete before the timeout expires and so the confirm primitive is returned with the ResultCode set equal to FAILURE and the ReasonCode set equal to TIMEOUT.

Replace Figure 91 with the following:

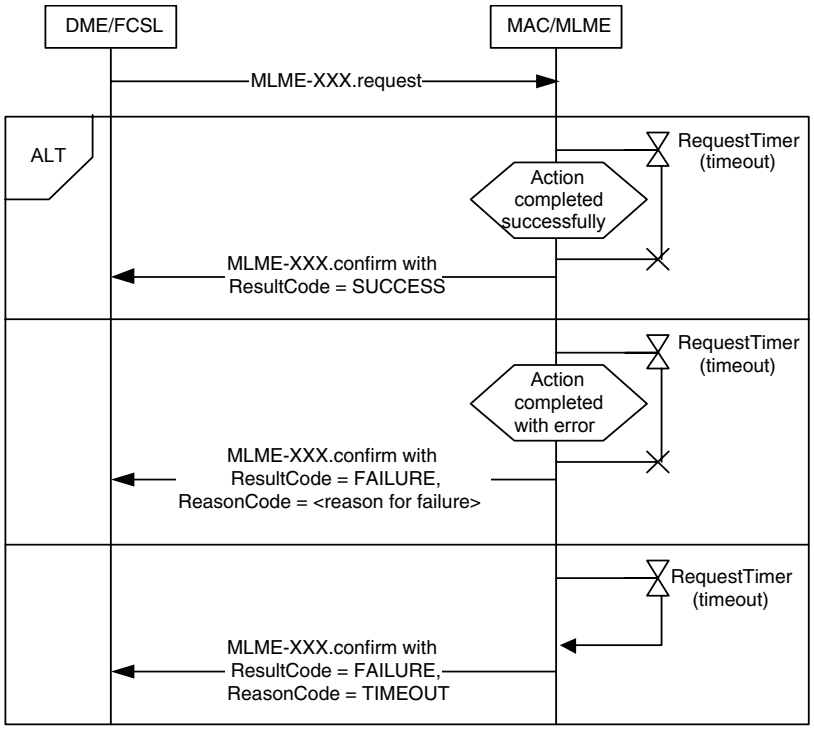

Figure 91—MSC showing examples of primitive timers

Insert the following paragraphs after Figure 91 in 8.1:

Throughout this clause, some of the procedures and MSCs are written as though the optional MLME SAP, 6.3, is exposed and is supported by the MLME. Unless stated otherwise, the procedures initiated by an MLME primitive may also initiated internally in the MAC. For procedures not shown to be initiated by an MLME primitive, the procedure is executed by the MAC without interaction with the DME.

For a MAC header to be correctly received by the MAC, the PHY first verifies the HCS. In addition, the MAC header shall have protocol revision supported by the MAC, have a DestID equal to a DEVID, BcstID, McstID, or when applicable the PNCID or UnassocID, and have a PNID equal to the PNID of the piconet with which the DEV is synchronized. Because the FCS validation is not required, it is possible for the MAC header to be correctly received even if the frame is not correctly received.

For a frame to be correctly received by the MAC, the MAC header shall be correctly received and the frame payload shall pass the FCS validation. If a DEV correctly receives a frame from an unassociated DEV, it may ignore the frame and may choose not to respond to the frame. If secure membership is required in the piconet and a DEV correctly receives a frame from a DEV that is not a member of the piconet, it shall ignore the frame and shall not respond to the frame, except for the ACK, if the ACK Policy field is set to Imm-ACK, Imp-ACK, or Dly-ACK Request.

8.2 Starting, maintaining, and stopping piconets

8.2.1 Scanning through channels

Change first paragraph in 8.2.1 as shown:

All DEVs shall use passive scanning to detect an active piconet. That is, DEVs shall be in receive mode for a period of time in a channel no less than mMinChannelScan~~, as specified in the MLME-SCAN.request,~~ to look for beacon frames from a PNC. If a particular BSID, PNID, or PNC address to scan for is not specified with an MLME-SCAN.request, ~~open scan is specified in the MLME-SCAN.request,~~ the DEV searches for any beacon frame. If a particular BSID, PNID, or PNC address to scan for is ~~open scan is not~~ specified, the DEV shall ignore all received frames not matching the ~~PNID and BSID~~ parameter or parameters contained in the request.

Change third and fourth paragraphs in 8.2.1 as shown:

DEVs search for piconets by traversing through all available PHY ~~the indexed~~ channels ~~indicated in the MLME-SCAN. request~~. A DEV may search the channels in any order as long as all valid channels are contained in the search pattern. The result of a scan shall include information on any parent, child, as described in 8.2.5, or 802.15.3 neighbor, as described in 8.2.6, piconets that were detected. This provides a complete inventory of each channel.

While searching, if any frame type other than a beacon frame is received, the searching DEV shall stay in the channel for a minimum of mMinChannelScan from the time of reception of first frame and look for a beacon from the PNC. ~~If the DEV finds only a frame and no beacon it shall report it as a part of the MLME-SCAN.confirm primitive.~~ The DEV shall scan all indicated channels to find piconets before returning the scan information via the MLME-SCAN.confirm primitive. The DEV shall only report piconets found due to the reception of a beacon frame as a part of the MLME-SCAN.confirm primitive.

8.2.2 Starting a piconet

Change the first three paragraphs in 8.2.2 as shown:

A DEV that is instructed to start a piconet through MLME-START.request, as described in 6.3.3.1, shall try only to start its own piconet and shall not attempt to associate with an existing piconet. ~~The DME shall have recently completed a scan procedure and will have chosen the channel in which to start the piconet.~~

The ~~DME~~ MAC should choose the channel with the least amount of interference to start the piconet based on the results of a recent scan (either DME initiated or MAC initiated) ~~the ChannelRatingList returned in the MLME-SCAN.confirm primitive, as described in 6.3.2.2~~.

Once the MAC has received ~~DME has chosen a channel, it shall issue~~ the MLME-START.request primitive ~~with the chosen channel. The DEV~~, it shall listen to the channel for mMinChannelScan duration to determine if the channel is still clear. If, at the end of this listening period, the ~~DEV~~ MAC determines that the channel is clear, the DEV, now the PNC, shall commence broadcasting its beacon once every superframe duration. ~~If, however, the DEV determines that the channel is no longer clear, it shall issue an MLME-START.confirm with a ResultCode indicating a failure to start the piconet. The DME then has options that include sending another MLME-START. request with a different ChannelIndex to start a piconet in a different channel, associating as a regular DEV and requesting the formation of a dependent piconet.~~ When the piconet starts, the PNC allocates an additional DEVID to itself for the purposes of exchanging data with other DEVs that become members of the established piconet.

Insert the following paragraph after the fifth paragraph in 8.2.2:

If the MAC determines that no channels are available, it will respond with an MLME-START.confirm with a ResultCode of FAILURE and ReasonCode of NO_CHANNELS_AVAILABLE.

8.2.3 PNC handover

Change the third paragraph in 8.2.3 as shown:

The PNC ~~shall~~ may allocate channel time with the chosen PNC capable DEV as the destination for the purpose of transferring information about the DEVs in the piconet and their current CTRqBs. ~~When the channel time has been allocated, the~~ The PNC shall first send a PNC Information command, as described in 7.5.4.2, to the chosen PNC capable DEV. In the PNC Information command, the PNC shall include all DEVs that are associated in the piconet, including any associated neighbor PNCs, the DEV personality of the PNC, and an entry for the PNCID. Once the PNC has successfully sent this command, it shall then begin sending all of the current channel time requests to the chosen PNC capable DEV using a PNC Handover Information command, as described in 7.5.3.3. Once the PNC has successfully sent the PNC Handover Information command, it shall send a PS Set Information Response command, as described in 7.5.8.2, to the new PNC. The PNC may fragment the PNC Information, PNC Handover Information, and PS Set Information Response commands using the process described in 8.7.

Insert the following paragraph after the fifth paragraph in 8.2.3:

The current PNC shall not accept any new commands that would change any of the information that will be transferred to the new PNC once the PNC handover process has been initiated. The PNC shall refuse these requests with the Reason Code field set to “Handover in progress” for commands that have this reason code defined.

Change the sixth and seventh paragraphs in 8.2.3 as shown:

Once the chosen PNC capable DEV has received the required information from the current PND, it shall respond to the current PNC with a PNC Handover Response command, as described in 7.5.3.2. This will

signal to the current PNC that the chosen PNC capable DEV is ready to take over as the new PNC. ~~After the PNC receives the PNC Handover Response command, it shall place a PNC Handover IE, as described in 7.4.9, in the beacon.~~

~~Meanwhile the chosen PNC capable DEV, after receiving an ACK to its PNC Handover Response command, will prepare to broadcast its first beacon as the new PNC. The current PNC shall place the PNC Handover IE in the beacon with the Handover Beacon Number field set to the beacon number of the superframe in which the new PNC will send its first beacon.~~ After the PNC receives the PNC Handover Response command with Reason Code field indicating success, it shall place a PNC Handover IE, as described in 7.4.9, in the beacon with the Handover Beacon Number field set to the beacon number of the superframe in which the new PNC will send its first beacon. Upon receiving the PNC Handover IE, the chosen PNC capable DEV will prepare to broadcast its first beacon as the new PNC. After sending the last beacon, the old PNC relinquishes control of the piconet~~, generates an MLME-PNC-HANDOVER.confirm to its DME,~~ and stops generating beacons. The new PNC shall broadcast its first beacon at the time the beacon would have been sent by the old PNC. This time may vary from the actual time due to clock inaccuracies of the old and new PNCs. The new PNC shall start sending beacons with the time token counter set to one more than the time token of the last beacon that will be sent by the old PNC. The new PNC shall begin using the PNCID as the SrcID for all beacon or command frames transmitted. The new PNC shall use the PNCID or its previously assigned DEVID as the SrcID for all data frames transmitted. When the PNC handover is successful, the association of the remaining DEVs with the piconet is unaffected and hence they are not required to ~~re-associate~~ reassociate with the new PNC.

Replace Figure 94 with the following:

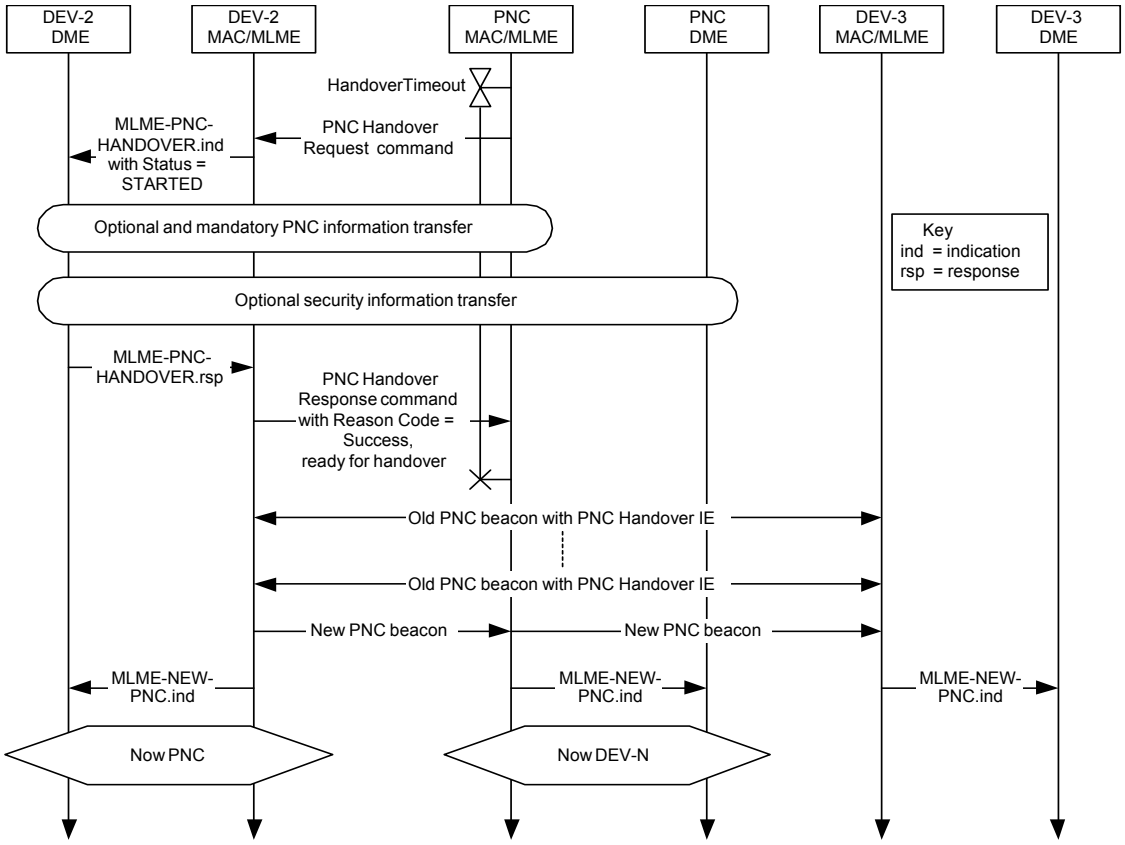

Figure 94—PNC handover MSC

Insert the following sentence and Figure 94a after Figure 94:

The mandatory and optional information transfer for PNC handover is illustrated in Figure 94a.

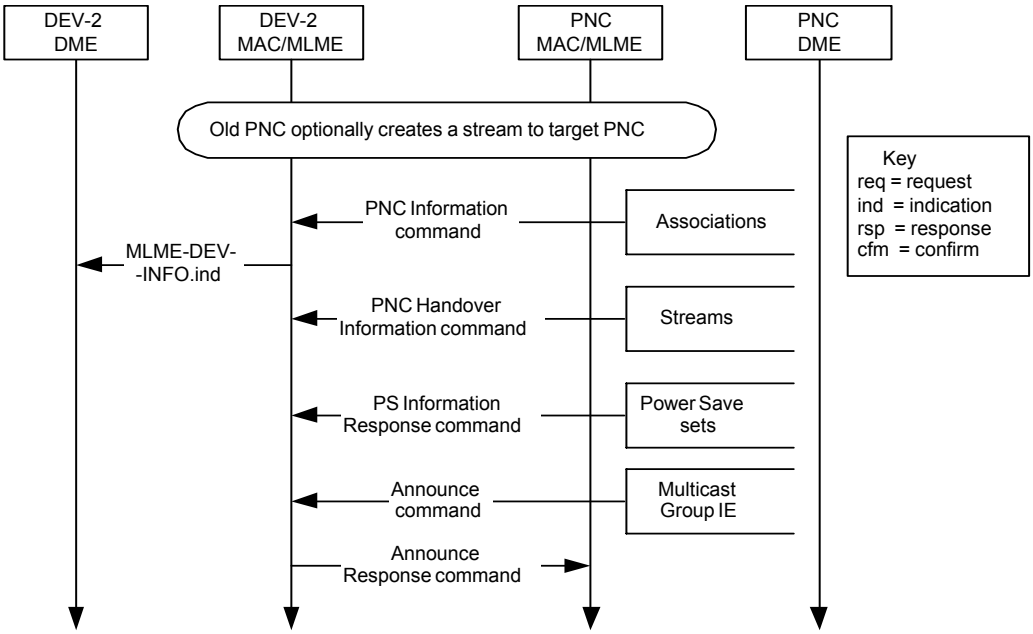

Figure 94a—Handover information transfer MSC

Change the eleventh paragraph in 8.2.3 as shown and insert Figure 94b after this paragraph:

~~In the MSC, the MLME-PNC-HANDOVER.response is sent when the DME is ready for the handover and is not tied to the arrival of the PNC Handover Information commands or PS Set Information Response commands.~~ The DME initiates the handover process using MLME-STOP.request with RequestType set to HANDOVER. This process is illustrated in Figure 94b. The DME may choose the target DEV or DEVs for the handover or allow the PNC to determine the target DEV. If the handover completes successfully, the MLME-STOP.confirm primitive is generated with a ResultCode set to SUCCESS. If the handover does not successfully complete within the time period specified by the DME, the PNC shall perform the PNC shutdown process defined by 8.2.7. After completion of the shutdown process, the MLME-STOP.confirm primitive is generated with the ResultCode set to FAILURE and the ReasonCode set to HANDOVER_FAILED.

Change the last paragraph in 8.2.3 as shown:

A dependent PNC receiving a parent beacon with a PNC Handover IE may immediately insert the Piconet Parameter Change IE into its beacons with the Change Type field set to MOVE, as described in 7.4.6, and the Superframe Timing field set to zero. A member of a ~~dependent~~ piconet that receives this Piconet Parameter Change IE in the beacon from the ~~dependent~~ PNC shall not transmit after the superframe which has a beacon number equal one less than the Change Beacon Number field in the Piconet Parameter Change IE until it has correctly received a beacon from its PNC, as described in 8.1. This requirement applies to DEVs in independent as well as dependent piconets.

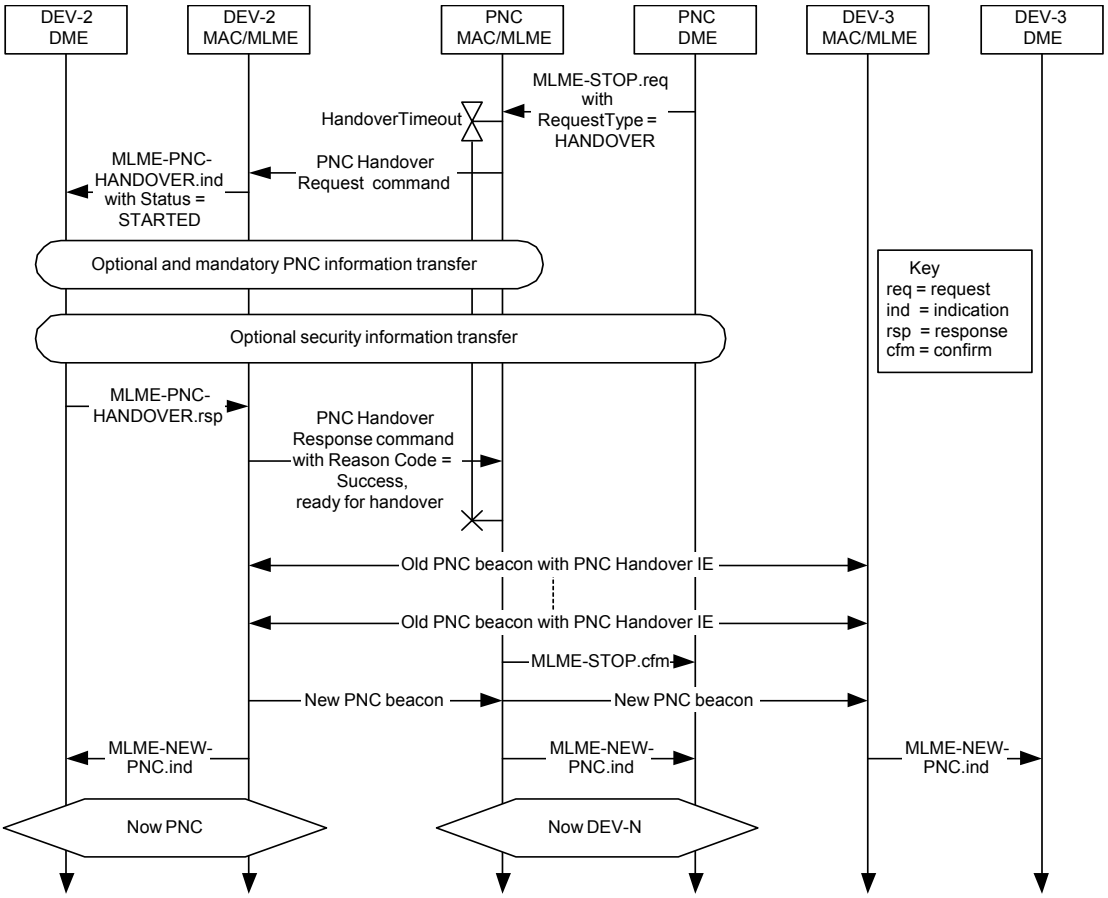

Figure 94b—DME initiated PNC handover

After 8.2.3, insert the following subclauses as 8.2.3a and 8.2.3b:

8.2.3a Preliminary handover

The Preliminary Handover procedure shall not be used by a dependent PNC.

If a DEV has been chosen as next PNC, the PNC shall periodically send handover information to that DEV so that it has the information it needs to take over as PNC. This sending of handover information is known as preliminary handover. In order to send the handover information, the PNC shall send the Handover Request command with the Handover Status field set to Preliminary Handover. The DEV receiving this frame shall only store the information for use at a later time, but not take over as PNC at this time.

Information transfer shall take place as shown in Figure 94a. In addition, any security information may be transferred as defined in 9.4.1.

When all the information has been received, the next PNC shall send the Handover Response command with Reason Code set to Success, Ready for Handover.

The current PNC may issue a handover based on previously transferred information by sending the Handover Request command with the Handover Status field set to Handover Based on Previous Information. Once the chosen PNC capable DEV is prepared to start operating as PNC, it shall respond to the current PNC with a PNC Handover Response command with Reason Code set to Success, Ready for Handover.

The handover process now proceeds as described in 8.2.3 from the point where the Handover Response command is sent.

8.2.3b Next PNC

The Next PNC procedure shall not be used for dependent piconets.

Although the PNC will normally attempt to perform a handover when it shuts down or leaves the piconet, this is not always possible. The PNC could be powered off in such a way that handover is not performed. The PNC could also move out of range of the rest of the DEVs in the piconet, or vice versa. In order to avoid the interruption in communication that would occur when the PNC disappears without handover, the PNC may choose a DEV in the piconet to be the next PNC.

The PNC should periodically evaluate the PNC capabilities of the members of the piconet and other factors to select a DEV to be the next PNC. In addition to the PNC capabilities field, the PNC should use information provided by the DEVs in the Piconet Channel Status IE, as described in 7.4.21, to determine which potential PNC could best be heard by the other DEVs in the piconet. In the context of the Piconet Channel Status IE, heard means correctly receiving a MAC header.

A DEV should continuously determine the DEVs that it can hear while it is in the AWAKE state, as defined in 8.13. If MCTAs are used in the piconet, DEVs that are not extremely power sensitive should listen to all of the uplink MCTAs.

The Next PNC IE, as described in 7.4.20, should be periodically announced in the beacon according to the beacon information announcement procedure as defined in 8.6.4.

The PNC also selects the Next PNID for the Next PNC IE. The PNC shall select a Next PNID such that it is different from any other PNID that has been detected as described in 8.10.3.

If no other DEV in the piconet is PNC capable, the PNC shall set the Next PNC field to zero and the Next PNID field to the current PNID in the Next PNC IE.

If a DEV has been chosen as next PNC, the PNC shall periodically perform preliminary handover as defined in 8.2.3a.

When the DEV identified as the next PNC fails to detect mMaxLostBeacons+1 consecutive beacons, it shall scan the channel for any frames. Since no frames are transmitted in dynamic CTAs or the CAP if the beacon was not correctly received, the next PNC shall use the presence of any frames from the current piconet as an indication that the beacon is still being transmitted but the next PNC has lost contact with the PNC. In this case the next PNC shall not take over the role of PNC for that piconet. If no MAC headers are correctly received during the 2 superframe duration scan, the next PNC shall begin sending out beacons using the same superframe duration and channel. The process of the next PNC taking over as PNC is known as implicit handover. There is a possibility that the next PNC has left the range of all of the other DEVs in the piconet as opposed to the original PNC going out of range or powering off.

The next PNC shall start its beacon after the time it would have expected the beacon from the previous PNC so that if they are both in range of some DEVs, their beacons will not collide. The next PNC should choose a position for the beacon that was unused based on the previous CTAs, to minimize the probability of collision. In order to reduce the possibility of two piconets in close range using the same PNID, the next PNC shall use the announced Next PNID as the PNID of the piconet.

DEVs that are not identified as the next PNC should store the Next PNID from the beacon. When the DEVs that are not identified as the next PNC fail to detect mMaxLostBeacons they should start scanning for beacons. If they do not see beacons with the current PNID, but see beacons with the PNID field set to the Next

PNID and the PNC Address field set to the DEV address that corresponds to the DEVID of the Next PNC, they are automatically associated with that piconet.

8.2.4 Dependent PNC handover

Replace Figure 95 with the following:

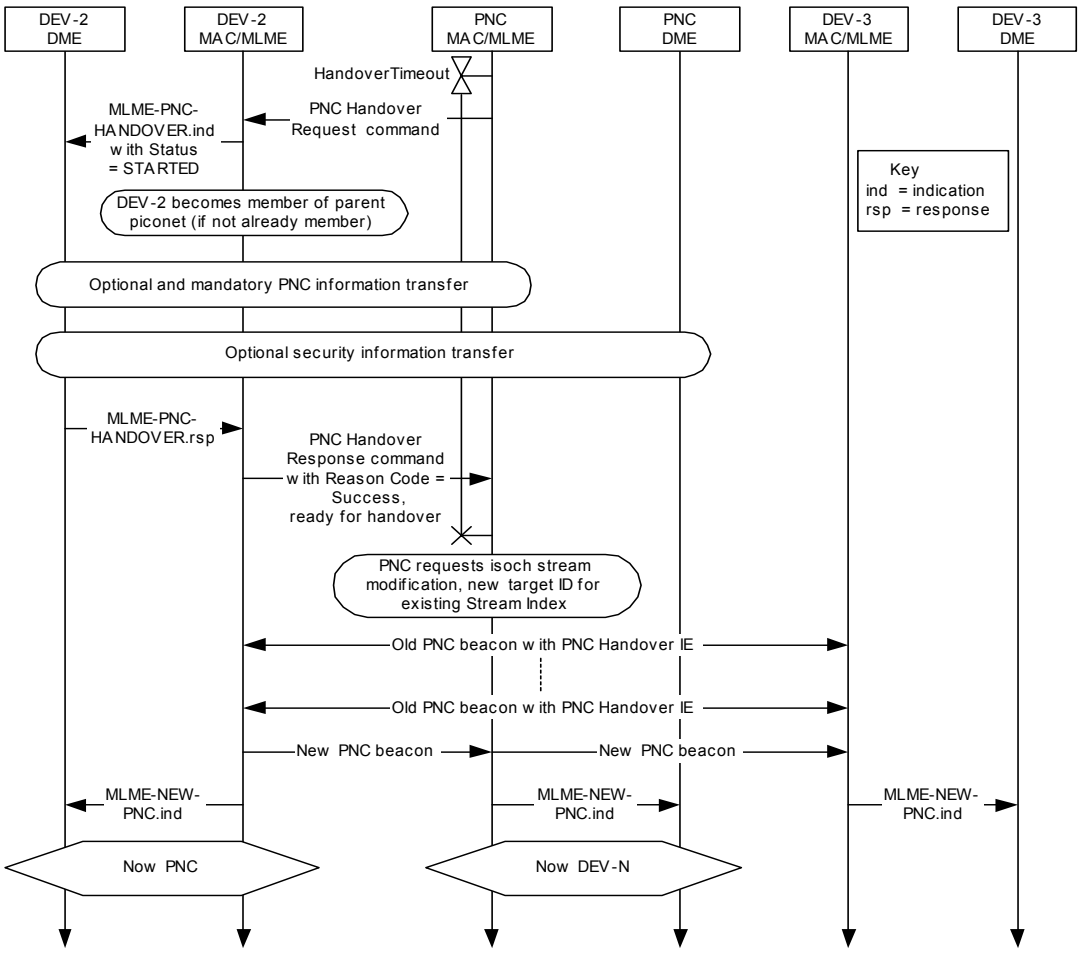

Figure 95—Successful PNC handover in a dependent piconet

Replace Figure 96 with the following:

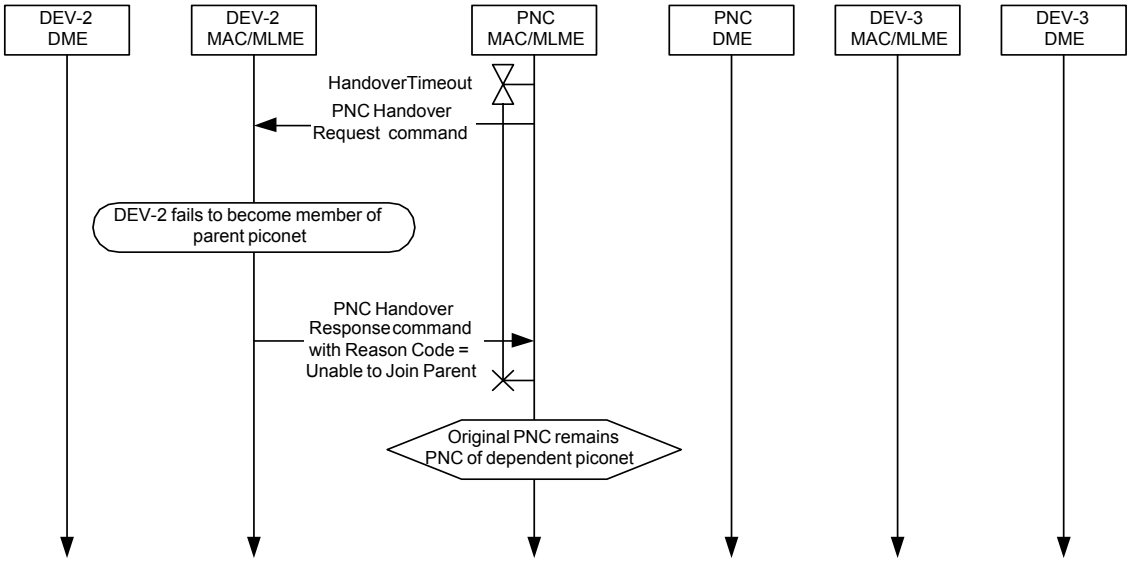

Figure 96—Failed dependent PNC handover when target DEV fails to join parent piconet

Replace Figure 97 with the following:

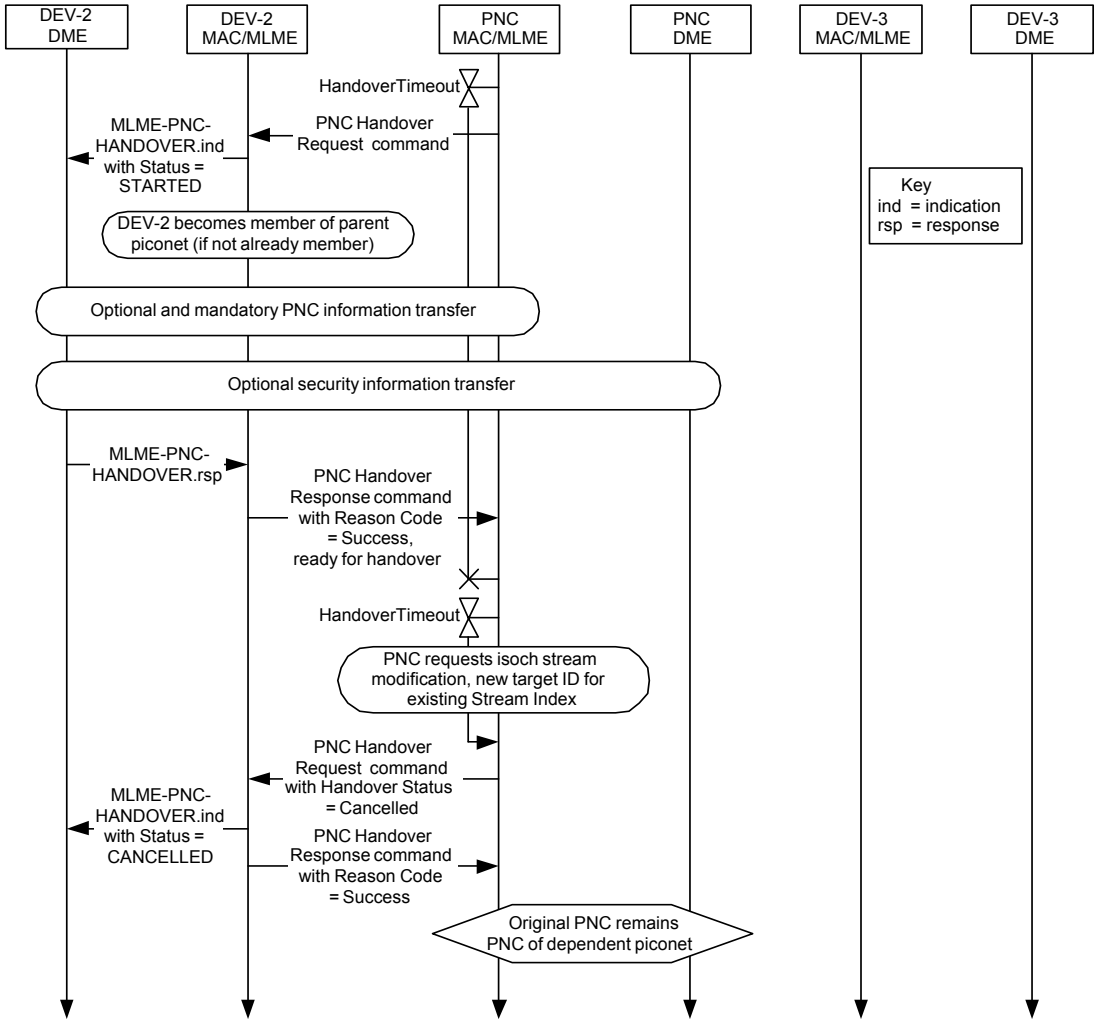

Figure 97—Failed dependent PNC handover when control for the dependent piconet CTA is handed over in the parent piconet

8.2.5 Child piconet

In 8.2.5, delete the third paragraph, which states "If the DEV receives ...."

Insert the following paragraph after the seventh paragraph in 8.2.5:

The FCSL initiates the formation of a child piconet using an MLME-START.request primitive while a DEV is currently a member of a piconet.

Replace Figure 99 with the following:

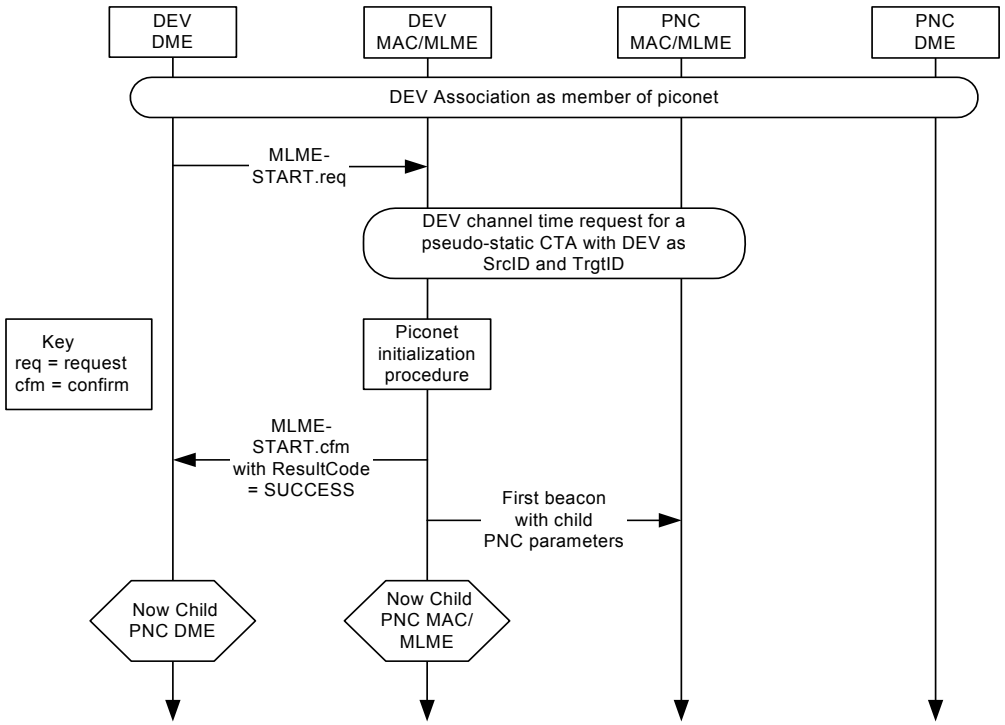

Figure 99—MSC for creating a child piconet

8.2.6 Neighbor piconet

Change the fourth paragraph in 8.2.6 as shown:

If the PNC permits the formation of a neighbor piconet and there is sufficient channel time available, the PNC shall allocate a private CTA using the NbrID as both the source and destination DEVID. ~~After receiving this channel time allocation in the beacon, the DEV DME configures the neighbor PNC parameters using the MLME-START-DEPENDENT.request and confirm primitives, as described in 6.3.3.3 and 6.3.3.4.~~

Insert the following paragraph after the eighth paragraph in 8.2.6:

The DME initiates the formation of a neighbor piconet by first using an MLME-ASSOCIATE.request with the NeighborPiconetRequest field set to TRUE. After successfully associating as a neighbor device, the DME uses an MLME-START.request to start neighbor PNC operations.

Replace Figure 101 with the following:

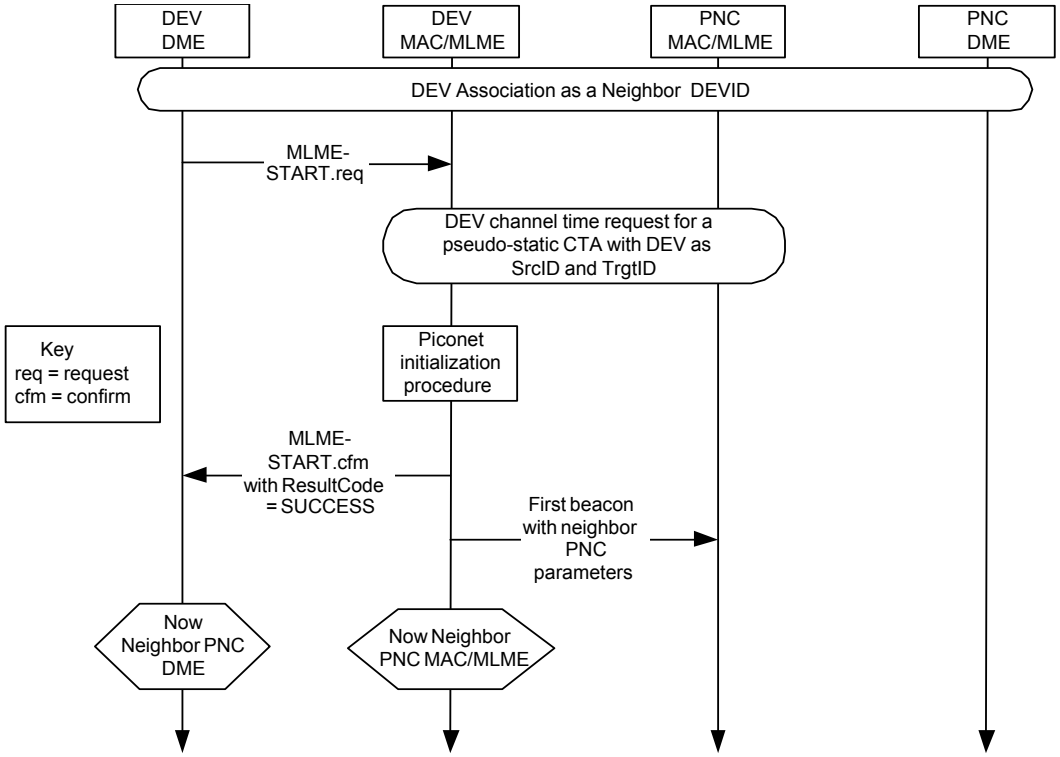

Figure 101—MSC for initiating a neighbor piconet

8.2.7 Stopping piconet operations

Change the paragraph in 8.2.7 as shown:

If the PNC is going to leave the piconet~~, perhaps due to a shutdown request from a higher layer~~ and there is no DEV capable of taking over as PNC or there is not sufficient time for a handover, the PNC will shut down the piconet operations. The DME may initiate the shutdown piconet operations using an MLME-STOP.request primitive with the RequestType set to SHUTDOWN.

8.2.7.1 Stopping an independent or parent piconet

Change the first paragraph in 8.2.7.1 as shown and insert Figure 101a after this paragraph:

If the PNC is going to remove itself from the piconet and no other DEVs are capable of taking over as the PNC, the PNC shall place ~~places~~ the PNC ~~shutdown~~ Shutdown IE, as described in 7.4.5, in the beacon. The PNC shall ensure that the shutdown announcement complies with the rules for beacon announcements in 8.6.4. The only exception to this requirement is if the PNC will be shutting down and does not have enough time to wait for the next system wake beacon to complete the handover process. The process of stopping a piconet without handing over is illustrated in Figure 101a.

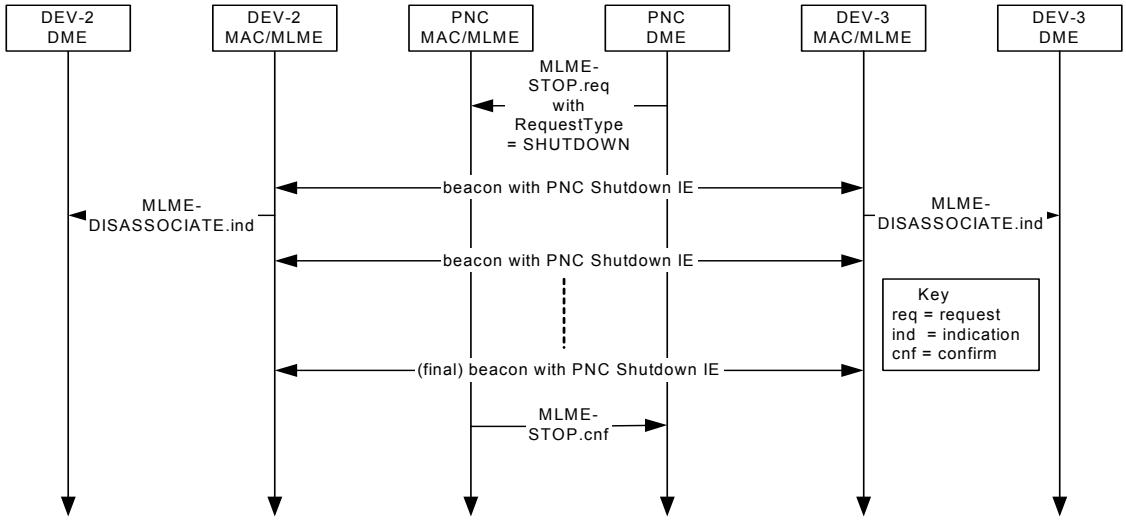

Figure 101a—MSC for stopping a piconet

8.2.7.2 Parent PNC stopping a dependent piconet

Change 8.2.7.2 as shown:

If the parent PNC wishes to stop a child piconet, it shall terminate the stream allocated to the child piconet using the isochronous stream termination procedure, as described in 8.5.1.3. If the parent PNC wishes to stop a neighbor piconet, it shall either send a Disassociation Request command, as described in 7.5.1.3, to the neighbor PNC or terminate the stream allocated to the neighbor piconet using the isochronous stream termination procedure, as described in 8.5.1.3. In either case, the dependent PNC shall ~~either~~ change channels, join another piconet as a dependent piconet, or immediately initiate its shutdown procedure, as described in 8.2.7.1. The parent PNC shall listen for the dependent PNC shutdown beacon sequence to determine when the dependent piconet CTA should be removed. The parent PNC may set a maximum time for the completion of the dependent shutdown sequence, after which the CTA will be removed regardless of the completion of the dependent shutdown procedure. ~~In the case of a child piconet, this timeout is set by the MLME while for a neighbor piconet, this time is set via the MLME-DISASSOCIATE. request primitive, as described in 6.3.6.1.~~ If the dependent PNC is a neighbor ~~that is~~ that is not operating ~~a piconet that is not~~ an 802.15.3 piconet, the parent PNC shall provide the same time as it would allow for its own shutdown sequence for the neighbor PNC to cease operations as a dependent piconet of the parent piconet before removing its private CTA.

8.3 Association and disassociation with a piconet

8.3.1 Association

In 8.3.1, delete the first paragraph, which states "Prior to the association ...."

Change the first sentence of the fourth paragraph in 8.3.1 as shown:

The PNC shall acknowledge all correctly received Association Request commands, as described in 8.1, by sending an Imm-ACK ~~frame with the DestID set to the UnassocID~~.

Replace Figure 102 with the following:

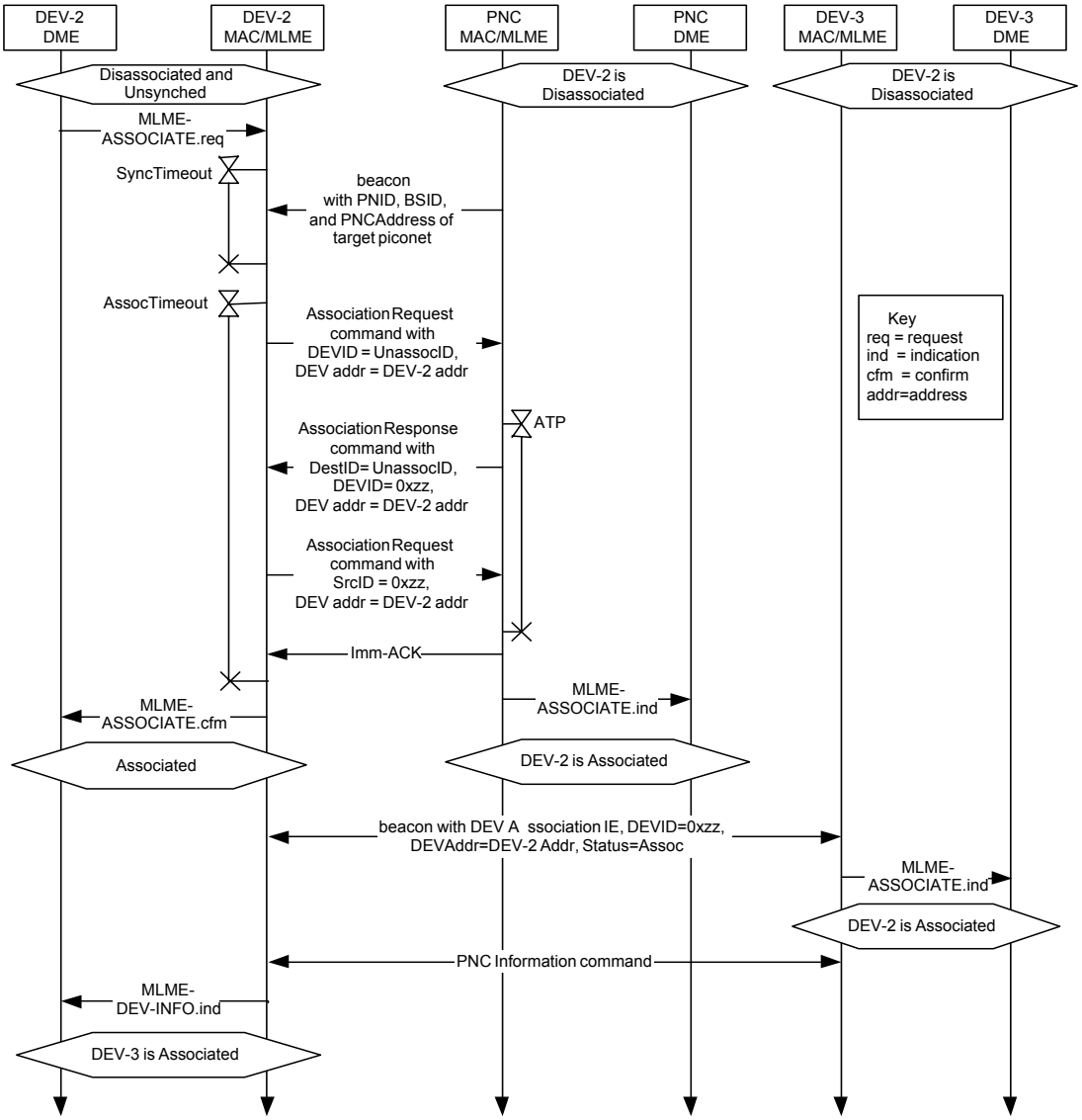

Figure 102—MSC of DEV-2 associating

Insert the following paragraph and note before the ninth paragraph in 8.3.1, change the ninth paragraph as shown, and insert the new paragraph as shown after the ninth paragraph:

When a PNC allows a DEV to associate, it shall assign a DEVID that is unique within the piconet. The first DEVID assigned should be randomly selected by the PNC. The PNC should assign DEVIDs so as to minimize the likelihood of collision with duplicate DEVID assignments within an interfering piconet and should assign the least recently used DEVIDs available.

NOTE—A simple method to accomplish these two goals is for the PNC to construct a list of available DEVIDs whenever it starts a new piconet. The first DEVID in the list is selected at random, followed by DEVIDs in monotonically increasing order that wrap around to one after DEVID 0xEC. This minimizes the chance that DEVIDs are duplicated if piconets with identical PNIDs interfere with each other. As DEVIDs become available for reassignment, they are added to the end of the list, which is thereby maintained in least recently used order.

~~The device IDs (DEVIDs) shall be assigned in sequence (increasing order) by the PNC except when PNC wishes to use a DEVID that was freed up after a DEV has left the piconet. After the PNC sends a Disassociation Request command, as described in 7.5.1.3, to a DEV, the PNC shall not reuse the same DEVID of that DEV until at least two times the ATP duration for that DEV has passed. The PNC shall ensure that there is only one associated DEV that has been allocated a given DEVID at any given time within the piconet. Similarly any associated DEV shall be allocated only one DEVID. The only exception to this is the PNC itself. The DEV serving as the PNC shall have two values of DEVID associated with it.~~ With the exception of the device that contains the PNC, there shall be a one-to-one correspondence between a DEV's device address and its assigned DEVID. The DEV that contains the PNC shall be assigned two DEVIDs: the ~~The~~ PNCID shall be assigned to the PNC function within the DEV and the other value of the DEVID shall be for use for all of the non-PNC traffic. When there is a coordination handover, as described in 8.2.3, the new PNC assumes the PNCID. The former PNC shall continue to use its non-PNCID DEVID for its non-PNC traffic. Hence the PNC is seen as two logical operational entities within the same DEV.

A DEVID released as a result of disassociation (see 8.3.4) shall not be reassigned until twice the disassociated DEV's association timeout period (ATP) has elapsed.

8.3.2 Piconet services

Replace Figure 103 with the following:

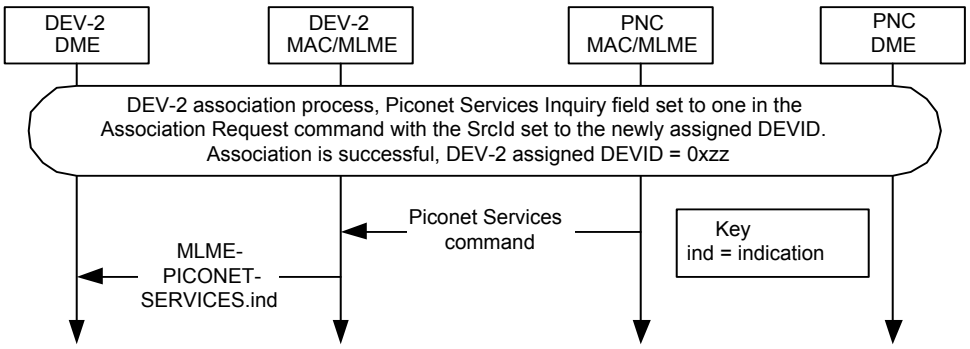

Figure 103—PNC sending the Piconet Services command to a newly associated DEV in response to a request in the association process

Change the third paragraph in 8.3.2 as shown and insert Figure 103a after this paragraph:

DEVs that are members of the piconet may place their own Piconet Services IE in the PNC's record of piconet services by sending the Piconet Services IE to the PNC using the Announce command. The PNC then sends an Announce command with DestID set to the BcstID containing the Piconet Services IE that it has added to its internal record of piconet services. If the PNC supports this capability, it retains the Piconet Services IEs of DEVs that have been sent to the PNC via the Announce command. The PNC will only save Piconet Services IEs for which it has space. Thus it is possible that the PNC would not retain a DEV's Piconet Services IE. After a DEV ~~disassoicates~~ disassociates from the piconet, the PNC shall delete the DEV's Piconet Services IE from its own record. The process of sending a Piconet Services IE to the PNC is illustrated in Figure 103a.

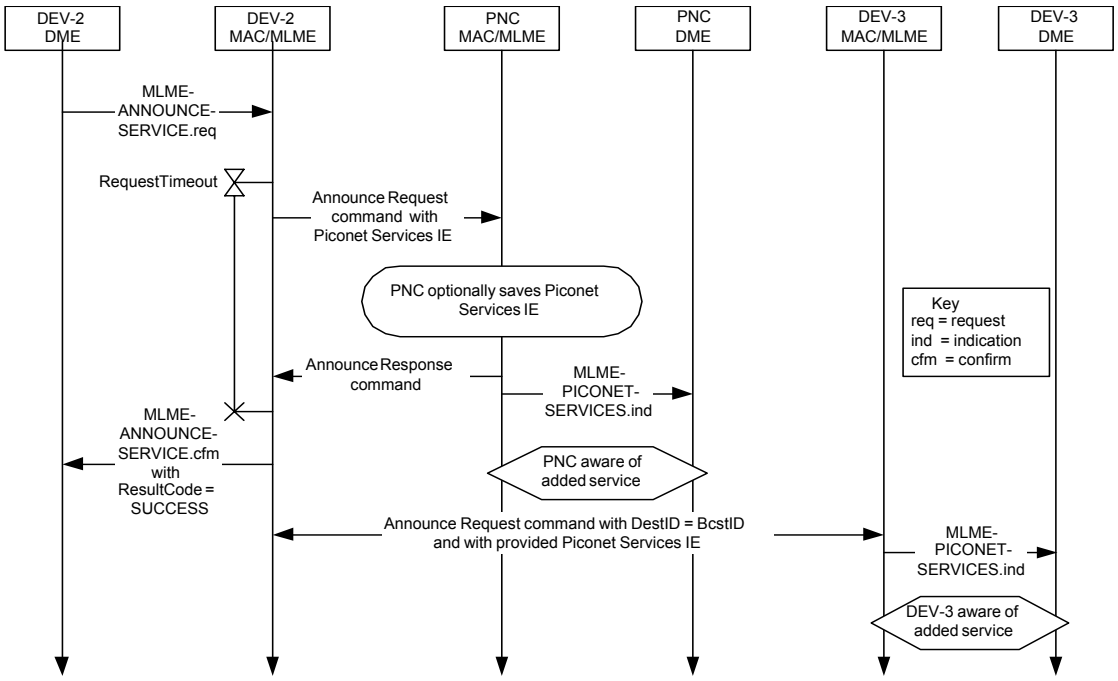

Figure 103a—DEV sending the Piconet Services IE to the PNC

Change the fourth paragraph in 8.3.2 as shown and insert Figure 103b after this paragraph:

If a DEV sends a Probe Request command to the PNC requesting the Piconet Services IE, the PNC responds with Probe Response commands that contain all of the Piconet Services IEs that it has in its internal record. If a DEV has not provided a Piconet Services IE to the PNC, the PNC sends the Piconet Services IE in the Probe Response command with the DEVID, a zero ~~Vendor~~ OUI and zero length Piconet Services field. If the PNC did not have enough space to save the Piconet Services IE that a DEV provided, it shall send in the Probe Response command a Piconet Services IE with length 1, i.e., it only contains the DEVID. If the PNC does not support the Piconet Services IE or if its policy is not to broadcast the Piconet Services IE, it shall respond to the request with a Piconet Services IE with length 2, the DEVID field set to the PNCID, and the OUI field set to the appropriate value, 7.4.16. The process of requesting the Piconet Services IEs from the PNC is illustrated Figure 103b.

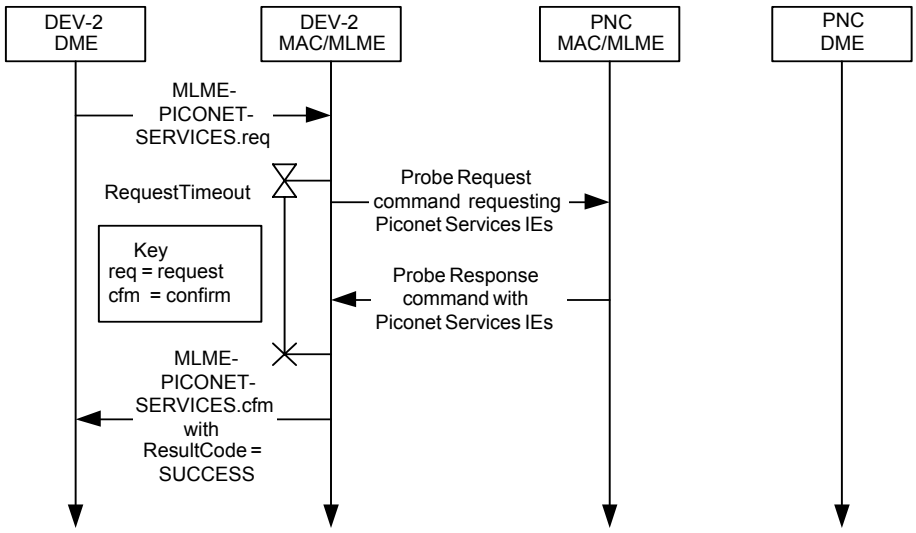

Figure 103b—DEV requesting Piconet Services IEs from the PNC

8.3.3 Broadcasting piconet information

Insert the following sentence at the end of the paragraph in 8.3.3:

The DEV Info field for the PNCID shall contain the same information as the DEV Info field for the PNC's DEV personality with the exception of the DEVID field, which shall be set to the PNCID.

8.3.4 Disassociation

Insert the following paragraphs after the third paragraph in 8.3.4:

If the DEV is STP capable, as indicated by the STP bit in its Capability IE, and is the originator of any allocated streams, it shall send an Announce command that includes the Stream Renew IE to reset the STP of all its streams. This command will also reset the ATP of the DEV.

If the DEV supports reporting the Piconet Channel Status, as defined in 8.2.3b, it should include the Piconet Channel Status IE in the Announce command sent to the PNC that is used to reset the ATP.

Change the fourth paragraph in 8.3.4 as shown:

If the beacons from the PNC are not received by the DEV for longer than the ATP, the DEV shall consider itself disassociated from the piconet and may try to associate again. The DEV notifies the DME that the ATP expired using the ~~MLME-ATP-EXPIRED.ind~~ MLME-DISASSOCIATE.ind primitive with the ReasonCode set to DEV_ATP_EXPIRED. In addition, if an associated DEV receives a broadcast PNC Information command from the PNC that is missing its DEV Info field, i.e., none of the DEV Info fields, 7.5.4.2, contains its DEV address, the DEV shall consider itself disassociated from the piconet.

Change the last sentence of the sixth paragraph in 8.3.4 as shown:

The PNC will also remove the disassociated DEV from any PS sets and multicast groups that it has joined and shall delete the Piconet Services IE, if any, for that DEV from its internal storage.

Insert the following paragraph after the sixth paragraph in 8.3.4:

If the DEV is disassociated, the PNC shall terminate all of the streams allocated to the disassociated DEV with its DEVID as either the SrcID or DestID. The PNC follows the process described in 8.5.1.3 except that it does not send any of the commands that would have had the disassociated DEV as the destination.

Replace Figure 104 with the following:

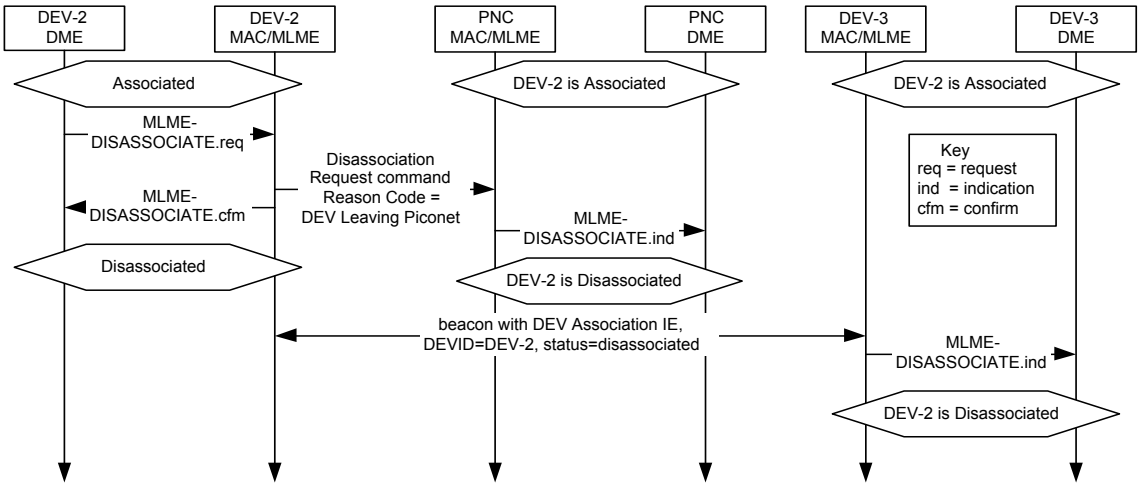

Figure 104—DEV initiated disassociation MSC

Replace Figure 105 with the following:

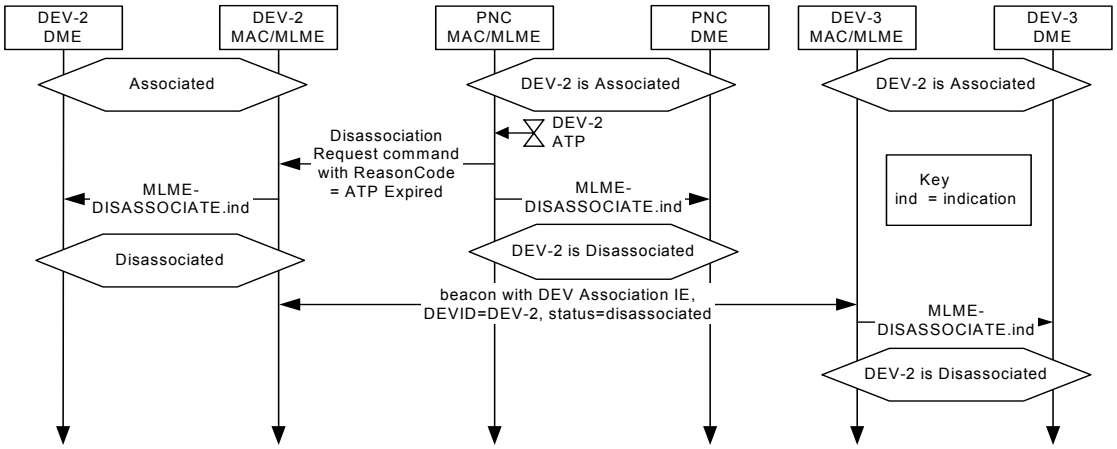

Figure 105—PNC initiated disassociation MSC

8.4 Channel access

Insert the following paragraph after Figure 106:

Contention access methods are used in contention periods (CPs). The CPs defined for this standard are the CAP and contention access CTAs, which are defined to be association MCTAs, association CTAs, open MCTAs and open CTAs.

8.4.1 Interframe space (IFS)

Change the second and third paragraphs in 8.4.1 as shown:

All Imm-ACK frames, frames sent as a response frame for Imp-ACK, and Dly-ACK frames shall start transmission over the medium a SIFS ~~duration~~ after the end of the transmission of the previous frame which requested the ACK. ~~A MIFS duration shall be allowed in the CTA between a frame and the next successive frame transmitted over the medium if the first frame either had the ACK Policy field set to either no-ACK or Dly-ACK.~~ The IFS between all received Imm-ACK frames and Dly-ACK frames and the next frame transmitted over the medium shall be no less than a SIFS. The IFS in a CTA between a frame and the next frame transmitted over the medium by the same DEV if the first frame had the ACK Policy field set to either no-ACK or Dly-ACK shall be no less than a MIFS.

During the CTAP, all DEVs shall use an IFS no less than a RIFS for retransmissions. During ~~the CAP~~ a CP, however, the retransmissions shall follow the CAP rules described in 8.4.2. The rules for acknowledgment and retransmissions are described in 8.8. The interframe space requirement for the beacon is ensured by the location of the CTAs, which is determined by the PNC, as described in 8.4.3.6.

Change the title of 8.4.2 as shown:

8.4.2 Carrier sense multiple access with collision avoidance (CSMA/CA) ~~Contention based channel access~~

Change the first paragraph in 8.4.2 as shown:

The basic medium access mechanism during the CAP is carrier sense multiple access with collision avoidance (CSMA/CA). The CSMA/CA contention method is also used for open CTAs and association CTAs. The PNC controls the type of data or commands that may be sent in the CAP via the CAP Control bits in the Piconet Mode field of the Piconet Synchronization Parameters field, as described in 7.3.1, in the beacon. A DEV shall only send frames of the type indicated by the Piconet Mode field in the beacon for the current superframe. The CAP Control bits in the Piconet Mode field may be changed by the PNC from superframe to superframe.

Insert the following paragraph after the first paragraph in 8.4.2:

Association CTAs are CTAs that have the SrcID set to the UnassocID and the DestID set to the PNCID. Association CTAs shall only be used to send Association Request commands to the PNC. Open CTAs are CTAs that have the SrcID set to the BcstID and the DestID set to a valid DEVID including GrpIDs, the McstID, and the BcstID. Open CTAs may be used to send either commands or data to the DEV or DEVs indicated by the DestID.

Change the third, fourth, and fifth paragraphs in 8.4.2 as shown:

During ~~the CAP~~ a CP, a DEV is allowed to transmit one frame at a time with backoff being applied to every frame attempted during the ~~CAP~~ CP, except for the Imm-ACK frame. The PNC may send a command a SIFS following the Imm-ACK of a frame in ~~the CAP~~ a CP or following a frame with ACK Policy field set to no-ACK in ~~the CAP~~ a CP. In this case, the PNC is not required to perform the backoff procedure before sending its frame.

In no case shall a DEV or the PNC extend its transmissions that started during ~~the CAP~~ a CP past the end of that CAP ~~into the CTAP~~. If an Imm-ACK is expected for that frame, the remaining time in the ~~CAP~~ CP needs to be long enough to accommodate the current frame, 2 SIFS times and the Imm-ACK frame at the same PHY rate as the transmitted frame. If there is insufficient time remaining in the ~~CAP~~ CP for the entire frame exchange sequence, then the DEV or the PNC shall not commence transmission of the frame.

The following backoff procedure shall be performed when sending frames (other than Imm-ACK) during ~~the CAP~~ a CP.

Change the seventh paragraph in 8.4.2 as shown:

The backoff time in ~~the CAP~~ a CP is measured at the air interface and indicates when a DEV may begin transmitting data. The DEV first waits a BIFS duration, as described in 8.4.1, from when the medium is determined to be idle before beginning the backoff algorithm. At the beginning of the CAP, the DEV may begin the backoff algorithm a SIFS after the end of the beacon transmission. If the PNC indicates that it is using an extended beacon, as described in 8.6.2, then the DEV shall wait until a SIFS after the last Announce command sent by the PNC as a part of the extended beacon before beginning the backoff procedure. At the beginning of a contention access CTA, the DEV may begin the backoff algorithm at the start time of the CTA.

Change the last two paragraphs in 8.4.2 as shown:

The backoff counter shall also be suspended outside of ~~the CAP~~ a CP duration. The backoff counter shall also be suspended if there is not enough time remaining in ~~the CAP~~ a CP for the DEV to send the frame. Note that the backoff counter is not maintained across superframes~~ and is not reset with each beacon~~. A DEV shall choose a new backoff_count at the start of every CP. If the total time elapsed since the frame was queued for transmission has exceeded the transmission timeout specified for the frame, the backoff counter shall be reset and the attempted transmission shall be cancelled.

When a directed frame is transmitted and the expected ACK is not correctly received by the DEV, as described in 8.1, the retry_count shall be incremented but shall not be set to more than 3. The backoff_count shall then be set to bw_random(retry_count). If the maximum number of retries for that frame has not been exceeded, the backoff procedure is again resumed.

8.4.3 Channel time allocation period channel access

8.4.3.1 Channel time allocations (CTA)

Change the first paragraph in 8.4.3.1 as shown:

The PNC divides the CTAP into channel time allocations (CTAs). A DEV that is ~~give~~given a directed CTA is guaranteed that no other DEVs will compete for the channel during the indicated time duration of the CTA. A DEV with a CTA may or may not make use of all the allocated time duration within the CTA. The selection of a stream, command, or asynchronous data for transmission during a CTA is determined locally by the DEV depending on the number of pending frames and the value of their User Priority ~~priorities~~fields. See A.1b.1 ~~A.1.2.1~~ for more information on priority management.

Insert the following sentence at the end of the fourth paragraph in 8.4.3.1:

If the PNC is unable to spread out the allocations, as described in D1.1.4, it should deny the channel time request.

Change the sixth paragraph in 8.4.3.1 as shown:

Pseudo-static CTAs shall be allocated only for isochronous streams and shall not be sub-rate allocations, as described in 7.5.6.1. If the PNC needs to change the duration or location of a pseudo-static CTA within the superframe, it shall change the corresponding CTA blocks in the beacon. The PNC shall not create any new CTAs for other stream ~~indicies~~ indices that overlap with the old time interval of the pseudo-static CTAs for mMaxLostBeacons number of superframes. However, the PNC may overlap the old and new time intervals of the same pseudo-static ~~psuedo-static~~ CTA within a superframe as it does not create the possibility of frame collisions. If the PNC sees the transmission of a PDU during the new allocation by the source of the old allocation before the expiration of mMaxLostBeacons number of superframes, the PNC may reuse the old allocation for another pair of DEVs. When the source DEV of a pseudo-static CTA receives a beacon with the new CTA, it shall cease using the old CTA and begin using the new CTA. When the destination DEV of a pseudo-static CTA receives a beacon with the new CTA, it shall begin receiving during the new CTA and may also receive during the old CTA.

Insert the following paragraph after the sixth paragraph in 8.4.3.1:

If the PNC needs to simultaneously change the positions of one or more of the CTAs in the superframe, including pseudo-static CTAs, the PNC may place the Piconet Parameter Change IE in the beacon with the Change Type field set to MOVE and the Superframe Duration field set to zero. After the superframe with the beacon number equal to the value in the Change Beacon Number field in the Piconet Parameter Change IE, a DEV shall not transmit until the DEV successfully receives a beacon with a beacon number greater than or equal to the value in the Change Beacon Number field in the Piconet Parameter Change IE.

Insert the following paragraph after the eighth paragraph in 8.4.3.1:

The PNC may allocate CTAs with the SrcID set to the BcstID. These CTAs are referred to as either open CTAs or open MCTAs, depending on the stream index, and use contention based channel access instead of TDMA. The type of contention access that is used is indicated by the Stream Index field in the CTA block. A stream index set to the asynchronous stream index indicates that CSMA/CA is used while the MCTA stream index indicates that slotted aloha is used. A DEV may request that the PNC modify the frequency and duration of the open CTAs or open MCTAs by sending a Channel Time Request command, as described in 7.5.6.1, to the PNC with the DestID set to the BcstID and the stream index set to the MCTA index. The CTA Rate Factor field, CTA Rate Type field, CTRq TU field, and Minimum Number of TUs field shall be set by the DEV to the desired duration and frequency requested for the open CTA allocation. The PNC is not required to allocate open CTAs or open MCTAs in the manner requested by a DEV; rather it may use this information to determine the frequency and duration of open CTAs or open MCTAs that it allocates. A DEV may modify its request for open CTAs or open MCTAs by sending another Channel Time Request command to the PNC with new parameters. A DEV is not allowed to terminate either an open CTA or an open MCTA. However, a DEV requests zero time for open CTAs or open MCTA by sending a Channel Time Request command to the PNC with the Minimum Number of TUs field set to zero.

Change the ninth paragraph in 8.4.3.1 as shown:

The More Data bit, as described in 7.2.1.6, in the Frame Control field is set to one to indicate that the source DEV could be sending more frames in the CTA. In order to save power at the destination DEV, a source DEV may indicate that it will not use the remaining time in the current CTA by setting the More Data bit to zero. The source DEV may retransmit a frame with More Data set to zero for which an ACK was expected but was not received. If the destination DEV receives a frame with the More Data bit set to zero with an ACK Policy other than no-ACK ~~field set to Imm-ACK, Dly-ACK or Dly-ACK Request~~, it should continue to listen for an implementation-dependent time after sending ~~the ACK frame~~ an acknowledgment to make sure that the source DEV is not going to retransmit the frame because it did not receive the ACK. The source DEV may choose to send a zero length frame with the More Data bit set to zero when it has no more frames to send in a CTA.

8.4.3.2 Channel time allocation (CTA) and channel time usage

Change the third paragraph in 8.4.3.2 as shown:

When a source DEV has a frame of any type for a destination DEV, the source DEV may send it during any CTA for that source DEV and destination DEV pair or to use the CAP to communicate that frame. The source DEV may also send a frame to a destination DEV in any CTA assigned to that source even if the destination DEV is different from ~~that~~ that indicated in the CTA block, provided the source DEV has determined that the destination DEV will be receiving in that CTA, as described in 7.4.11. The stream index in a transmitted frame shall be one of:

- The asynchronous stream index, as defined in 7.2.5
- The MCTA stream index, as defined in 7.2.5
- An established stream from the source DEV to the destination DEV

Change the fifth paragraph in 8.4.3.2 as shown:

In any superframe there may be one or more DEVs in the piconet that receives the beacon in error. This may not happen to the same DEV all the time but may happen to different DEVs at different times depending upon their location and type of interference to which they are subjected. If a DEV did not receive the beacon, it shall not transmit during the CAP or during any MCTA or dynamic CTA, except to ACK a directed frame sent to the DEV with the ACK Policy field set to ~~either~~ one of Imm-ACK, Imp-ACK, or Dly-ACK Request. DEVs with pseudo-static CTAs are allowed to transmit during these CTAs as long as the number of consecutive lost beacons is less than or equal to mMaxLostBeacons. A DEV shall stop transmitting in its pseudo-static CTA when the number of consecutive lost beacons exceeds mMaxLostBeacons. If a DEV that is the destination of a pseudo-static CTA misses a beacon, it should listen for the entire duration of the superframe in case the pseudo-static CTA is in the process of being moved. Any DEV that misses a beacon may also listen for the entire duration of the superframe to receive frames for which it is the destination.

8.4.3.3 Management CTAs

Change the first sentence of the first paragraph in 8.4.3.3 as shown:

Management CTAs (MCTAs) are ~~identical to~~ CTAs ~~except that the PNCID is either the SrcID or the DestID in the CTA and~~ that have the stream index ~~is~~ set to the MCTA stream index, as described in 7.2.5.

Change the second paragraph in 8.4.3.3 as shown:

An open MCTA is one where the SrcID is the BcstID, as described in 7.2.3, and the DestID is set to a valid DEVID including GrpIDs, the McstID, and the BcstID. Any DEV that is associated in the piconet may attempt to send a command frame to ~~the PNC~~ the DEV or DEVs that are indicated in the DestID in an open MCTA. An MCTA with the UnassocID as the SrcID is an association MCTA. Any DEV not currently associated in the piconet may attempt to send an Association Request command to the PNC in an association MCTA. Association Request commands shall not be sent in open MCTAs. Likewise, only Association Request commands shall be sent in association MCTAs. Open MCTAs with the DestID set to the PNCID enable the PNC to service a large number of DEVs with low MCTA requirements by using a minimum number of MCTAs. When there are few DEVs in a piconet, it might be more efficient to use MCTAs assigned to a DEV instead of using an open MCTA. Open MCTAs in which the DestID is not the PNCID are used by DEVs to send frames to other DEVs without having to request channel time.

Change the title of 8.4.3.4 as shown:

8.4.3.4 Slotted-aloha access ~~for open and association MCTAs~~

Change the first paragraph in 8.4.3.4 as shown:

Slotted aloha is ~~the access mechanism~~ used for contention access in an open MCTA or an association MCTA. The access to an open or association MCTA shall be controlled by a contention window CW_a maintained by each DEV. The contention window shall be derived from the number *a*, where *a* is the number of retransmission attempts made by the DEV. For the first access attempt, *a* shall be set to zero. The size of the contention window, CW_a, is defined as follows:

After 8.4.3.7, insert the following subclause as 8.4.3.8:

8.4.3.8 Relinquishing CTA time to another DEV

The PNC gives transmit control to the DEV that is the SrcID of a CTA for the duration of the CTA. The DEV that has transmit control in a CTA may, subject to the restrictions in this subclause, relinquish the remaining time in a CTA to another DEV. The DEV that relinquishes the channel time is referred to as the originating DEV while the DEV that is given the transmit control of the time in the CTA is referred to as the target DEV. The DEV that is the SrcID of the CTA begins the CTA with transmit control for the CTA.

The originating DEV relinquishes the remaining time in the CTA to a target DEV by setting the CTA Relinquish bit in the header of a frame which has the DestID set to the DEVID of the target DEV. Transmit control of the CTA can only be exchanged between the source DEV of the CTA and the destination DEV(s). If there are multiple destination DEVs for the CTA, transmit control may be given to any one of them.

The originating DEV should not initiate a CTA relinquish procedure unless it has determined that the receiving DEV has set the CTA Relinquish supported bit to one in the DEV Capabilities as defined in (7.4.11)

A DEV that receives transmit control of a CTA keeps control until the CTA end time or until it relinquishes the transmit control back to source DEV of the CTA.

The target DEV, after listening for an mCTARelinquishTimeout following reception of a frame with the CTA Relinquish bit set, may send a data frame to any potential DestID of the CTA. The target DEV listens to verify that the originating DEV is not retrying the frame that had the Relinquish CTA bit set.

If the DEV relinquishing time in a CTA does not correctly receive a header, as described in 8.1, from the DEV to which it relinquished the CTA within mCTARelinquishTimeout time, it shall maintain transmission control of the channel.

The CTA Relinquish procedure shall not be used in the CAP or in any contention CTA.

Any DEV that is responding to frames in a CTA may set the more-data bit in the header of an Imm-ACK frame or data frame to indicate to the current CTA transmit control owner that they would like to get a transmission opportunity. The current CTA transmit control owner may choose to relinquish the CTA, send a frame with an Implied ACK request, or to ignore the request.

If the DEV that has transmit control of the CTA wishes to relinquish the CTA and has no data to send, it may set the CTA Relinquish bit in an Imm-ACK frame. The same rules apply as when the CTA Relinquish bit is used in a data frame.

The target of a CTA relinquish shall relinquish the CTA only to DEV that is SrcID of the CTA. The current CTA transmit control owner shall always initiate the relinquish frame exchange.

Figure 113a illustrates the process of relinquishing transmit control to another DEV in the piconet.

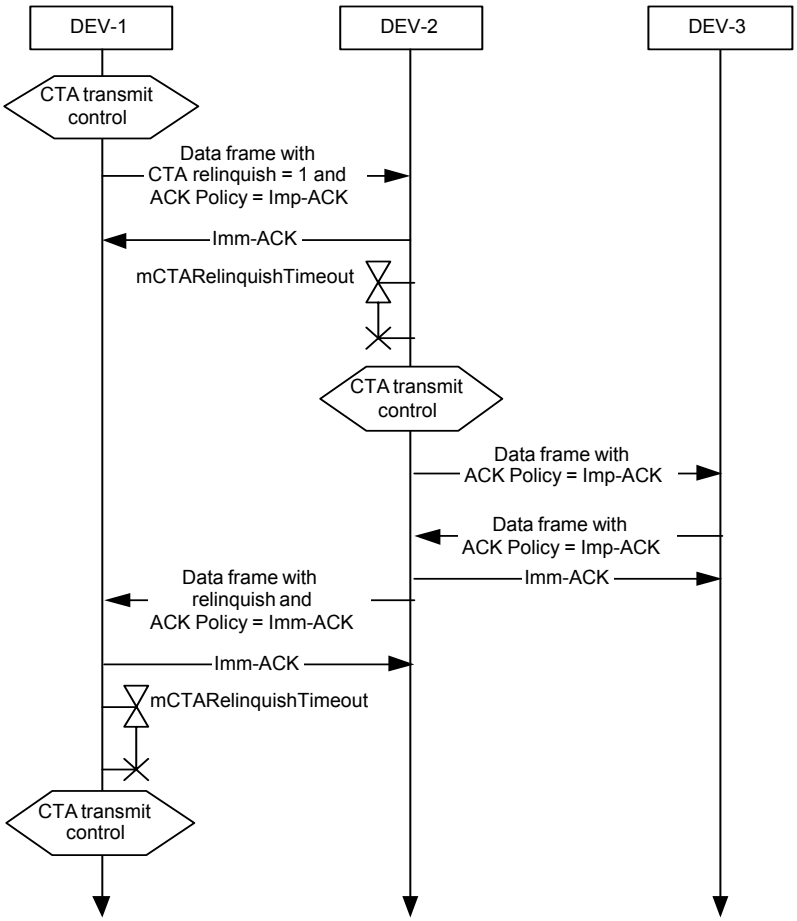

Figure 113a—Message sequence chart for relinquishing CTA time when the target DEV has data to send

8.5 Channel time management

8.5.1 Isochronous stream management

Insert the following paragraphs after the third paragraph in 8.5.1:

The StreamGrpID parameter of the MLME-CREATE-STREAM.request and MLME-MODIFY-STREAM.request primitives allows a DME to define associations between streams for the purpose of sharing channel time. When a StreamGrpID other than zero is assigned, a CTA should be shared fairly amongst all streams within the group associated with the StreamGrpID. The DME is responsible for assigning unique StreamGrpIDs for stream groups.

The PNC may split a super-rate allocation into more CTAs than required to satisfy the CTA Rate Factor and may also split a sub-rate allocation into more than one CTA in the same superframe.

A DEV may request the creation (and subsequent modification and termination) of allocations, isochronous and asynchronous, in which the DestID is the PNCID. Likewise, the PNC may create allocations, both

isochronous and asynchronous, in which the SrcID is the PNCID. However, while these allocations are transferred in the PNC handover process, the purpose for the allocations might be lost when the DEV that is the PNC changes. The PNC shall only allocate multiple CTAs per superframe if the time allocated for each CTA is no more than one CTRq TU different from any of the other CTAs allocated for that stream.

8.5.1.1 Isochronous stream creation

Change the first item in the dashed list of the second paragraph in 8.5.1.1 as shown:

— The Stream Index field is set to an unused value other than the asynchronous stream index. This indicates ~~to indicate~~ that the isochronous stream has been allocated channel time. The PNC should assign the least recently used stream index available.

Change the fourth paragraph in 8.5.1.1 as shown:

The PNC shall announce the creation of all ~~pseudo-static~~ streams ~~and of all sub-rate streams. It shall also announce creation of a streams for which the destination DEV, or any intended destination DEV in the case of broadcast and multicast streams, is in power save mode. The PNC shall make the announcement~~ with the CTA Status IE, as described in 7.4.10, using the beacon information announcement mechanism, as described in 8.6.4. The PNC shall issue the initial CTA for the stream in the superframe indicated in the CTA Status IE.

Change the fifth paragraph in 8.5.1.1 as shown below:

The CTA Status IE shall have the stream index of the new allocation, ~~and~~ the CTRq Control field ~~and CTA Rate Factor field~~ from the corresponding Channel Time Request command, and the CTA Sub-Rate field set appropriately, 7.4.10. In addition, the PNC shall allocate the first CTA of the stream in the superframe with the beacon number, as described in 7.3.1.1, indicated by the Start Beacon Number field of the CTA Status IE.

Change the tenth paragraph in 8.5.1.1 as shown below:

DEVs perform multicast negotiations at a higher layer. A DEV may set up a multicast stream either by using a GrpID assigned by the PNC, as described in 8.5.3, or by using the McstID. ~~A DEV sets up a multicast stream at the request of the upper layer by sending a request to the PNC for a stream with the multicast ID as the destination.~~ A DEV enables reception of a multicast stream that has the DestID set to the McstID ~~by using~~ with the MLME-MULTICAST-RX-SETUP.request. This primitive tells the MAC to receive frames from a particular source DEV with the ~~DestD~~ DestID set to the McstID and with the stream index specified in the primitive ~~MLME~~. A DEV enables the reception of multicast traffic addressed to the GrpID by joining a multicast group, as described in 8.5.3.

Change the eleventh paragraph in 8.5.1.1 as shown below (splitting into two paragraphs):

If the target DEV is in ~~either~~ DSPS ~~or APS mode~~ and is part of the same SPS Set as the channel time request, and the PNC grants the allocation, the PNC shall respond with a Channel Time Response command with a Reason Code of SUCCESS. Neither the originator nor the target DEV need to change PM modes for the stream to be allocated.

However, if the target DEV is either in DSPS mode and is not part of the same SPS Set as the channel time request or in APS mode, and the PNC grants the channel time request, the PNC shall set the Reason Code in the Channel Time Response command to “Success, DEV in PS mode.” The PNC shall place the PCTM IE in the beacon with a bit set for the target DEV, as described in 7.4.8.

Insert the following paragraph after the fourteenth paragraph (i.e., "If the PNC does not receive...") in 8.5.1.1:

An STP capable DEV shall renew its allocated streams at least once every ATP by sending an Announce command with the Stream Renew IE to the PNC containing the stream index of every stream that the originator wants to keep. Streams that have been modified within the current ATP may be omitted from the Stream Renew IE sent during the same ATP.

Replace Figure 114 with the following:

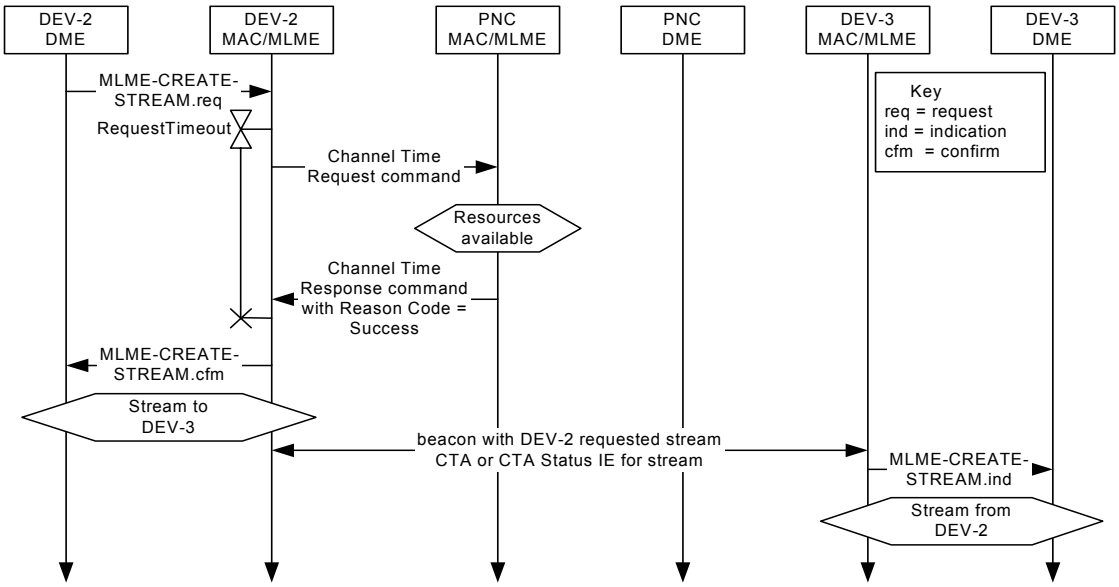

Figure 114—MSC for creating a DEV-2 to DEV-3 stream

Replace Figure 115 with the following:

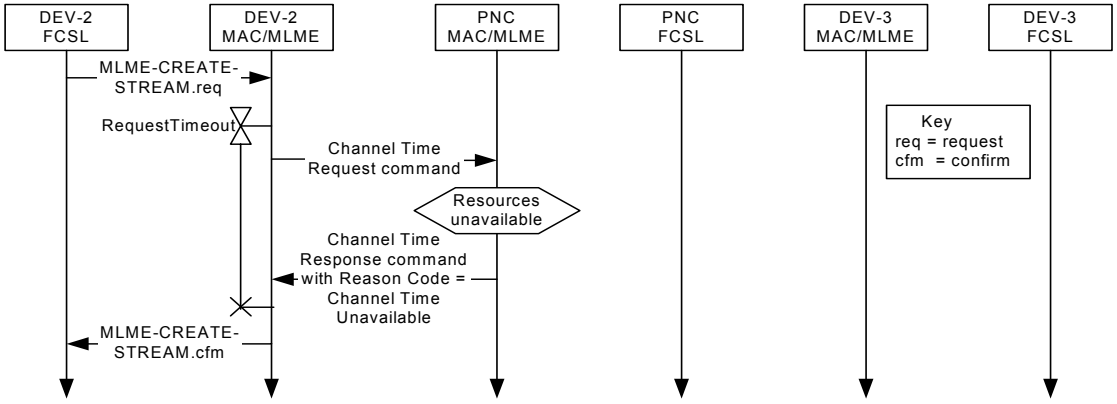

Figure 115—MSC for a denied DEV-2 to DEV-3 stream

In 8.5.1.1, delete Figure 116 and the associated paragraph, which states "Figure 116 illustrates ...."

8.5.1.2 Isochronous stream modification

Insert the following note after the third paragraph in 8.5.1.2:

NOTE—If a request is rejected, the Available Number of TUs is calculated using the CTRq TU associated with the last accepted request. Only if the request is accepted will the CTRq TU size for an exiting stream be modified.

Change the fourth paragraph in 8.5.1.2 as shown:

The PNC shall announce the modification of ~~those~~ all streams ~~for which the destination DEV, or any intended destination DEV in the case of broadcast and multicast streams, is in power save mode. The PNC shall announce the modification of those streams for which one or more of the PM CTRq Type, CTA Rate Type, and the CTA Rate Factor fields are modified. The PNC shall make the announcement~~ with the CTA Status IE, as described in 7.4.10, using the beacon information announcement mechanism, as described in 8.6.4. The PNC shall issue the first modified CTA for the stream in the superframe indicated in that IE. If the target DEV is in DSPS mode, the PNC shall also allocate an uplink MCTA in the same superframe as when the CTA is first allocated with the DSPS DEV as the source and the PNC as the destination that is long enough to handle a PM Mode Change command, a Channel Time Request command with 4 isochronous CTRqBs, and the associated Imm-ACKs and SIFSs.

Insert the following paragraph after the eighth paragraph in 8.5.1.2:

If the DEV is STP capable, a request to modify a stream will reset the STP of that stream.

Replace Figure 117 with the following:

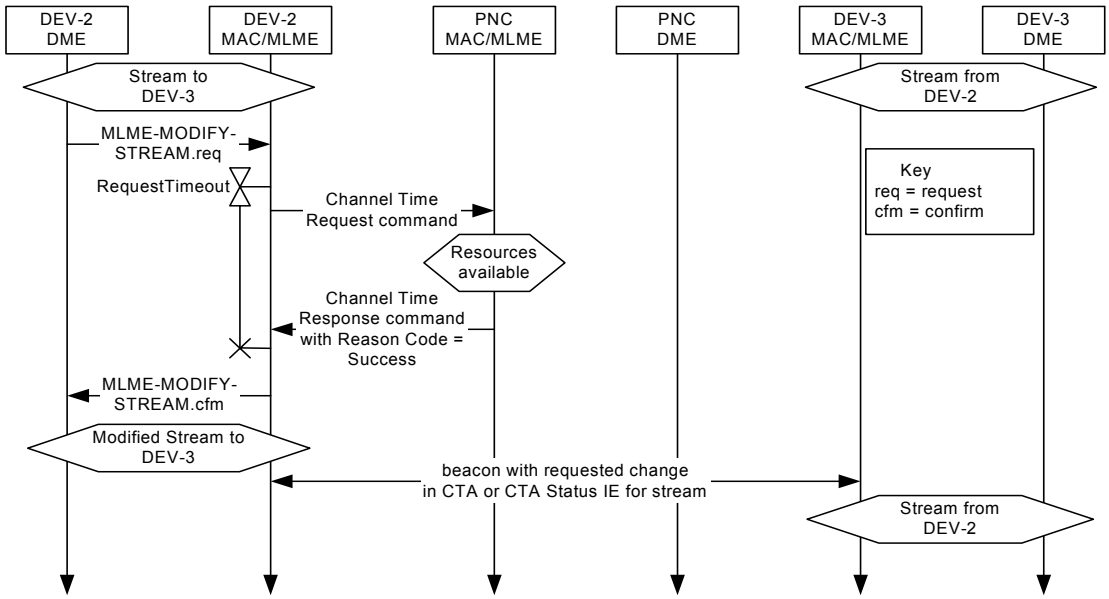

Figure 117—MSC for modifying a stream

Replace Figure 118 with the following:

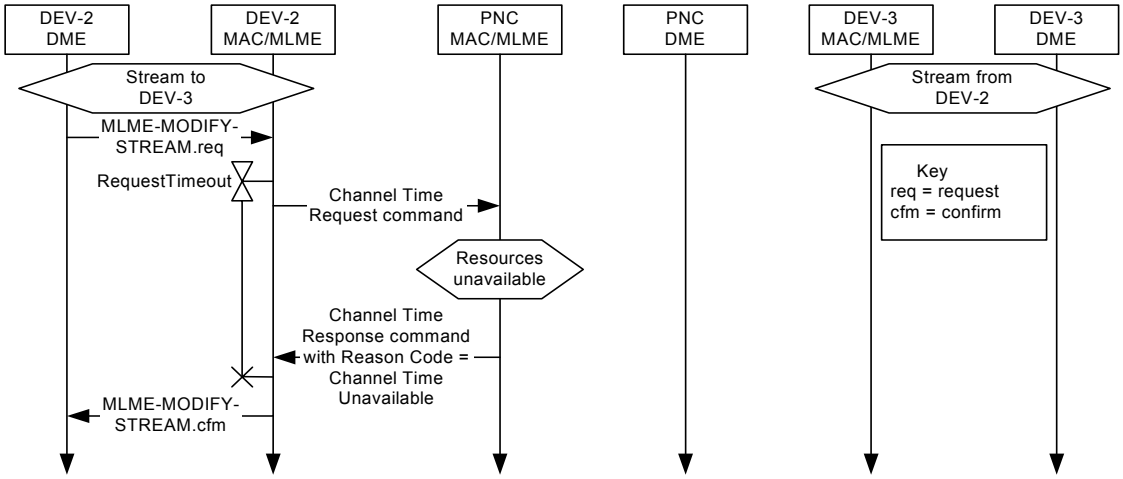

Figure 118—MSC for a denied stream modification

Replace Figure 119 with the following:

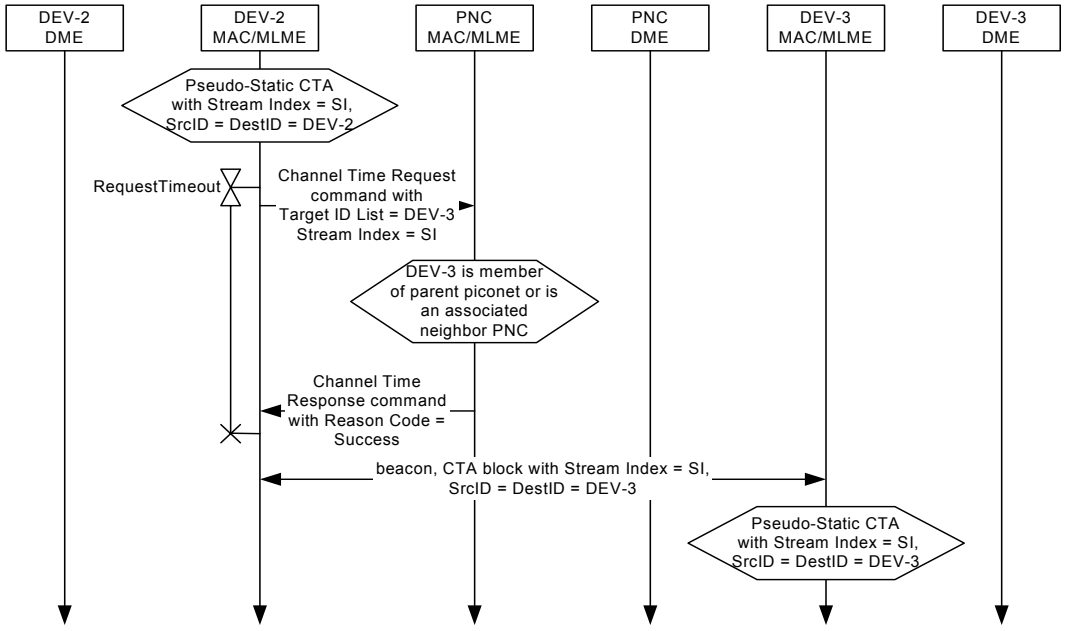

Figure 119—MSC for the handing over control of a private, pseudo-static CTA

8.5.1.3 Isochronous stream termination

Change the second paragraph in 8.5.1.3 as shown:

The PNC, upon receiving a Channel Time Request command from a DEV requesting stream termination, shall respond with an Imm-ACK. In the case where the originating DEV is requesting a stream termination, the PNC shall then notify the target DEV of the termination via ~~a null CTA block~~ CTA Status IE in the

beacon. The CTA Status IE shall have the stream index set to the value of the terminated stream and the Terminate bit set to indicate that the stream was terminated, as described in 7.4.10. The PNC shall ensure that the CTA Status IE announcements comply with the rules for beacon announcements in 8.6.4. ~~The null CTA block shall appear in at least mMinBeaconInfoRepeat consecutive beacons. For CTAs that were not allocated every beacon, i.e. sub-rate CTAs, the first null CTA block shall be placed starting in the beacon when the next CTA would have been allocated. A null CTA block has the stream index, SrcID and DestID set to the appropriate values with zero values for the CTA location and CTA duration, 7.4.1.~~ Figure 120 illustrates the MSC for termination of a stream by a source DEV

Replace Figure 120 with the following:

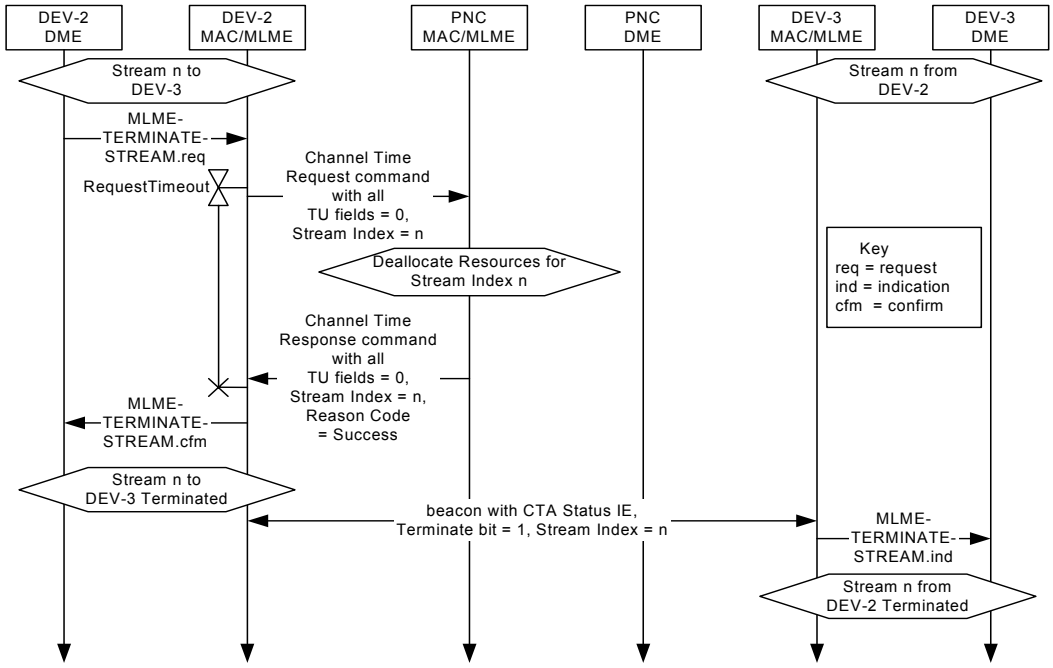

Figure 120—MSC of source DEV-2 requesting termination of its stream

Replace Figure 121 with the following:

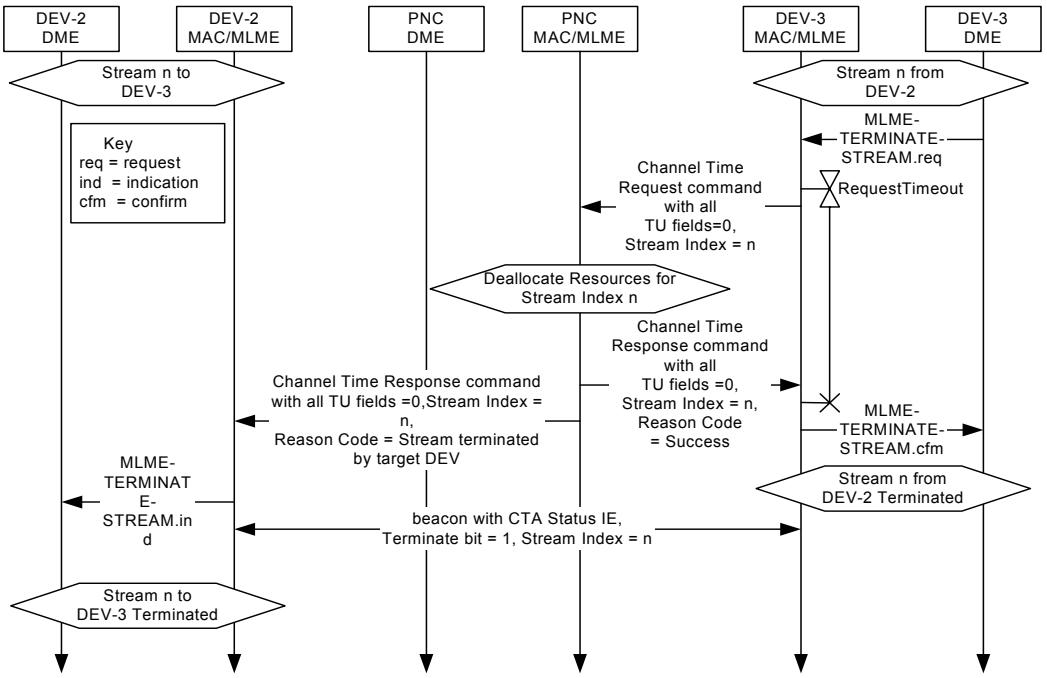

Figure 121—MSC of target DEV-3 requesting termination of source DEV-2's stream

Change the fourth paragraph in 8.5.1.3 as shown:

In the case where the PNC decides to terminate an originating DEV's stream, the PNC shall notify the source DEV via a Channel Time Response command and the target DEV via a CTA Status IE in the beacon. The CTA Status IE shall have the stream index set to the value of the terminated stream and the Terminate bit set to indicate that the stream was terminated, as described in 7.4.10. The PNC shall ensure that the CTA Status IE announcements comply with the rules for beacon announcements in 8.6.4. ~~null CTA in at least mMinBeaconInfoRepeat consecutive beacons. For CTAs that were not allocated every beacon, e.g. sub-rate CTAs, the first null CTA block shall be placed starting in the beacon where the next CTA would have occured.~~ Figure 122 illustrates the termination of a source DEV's stream by the PNC.

Replace Figure 122 with the following:

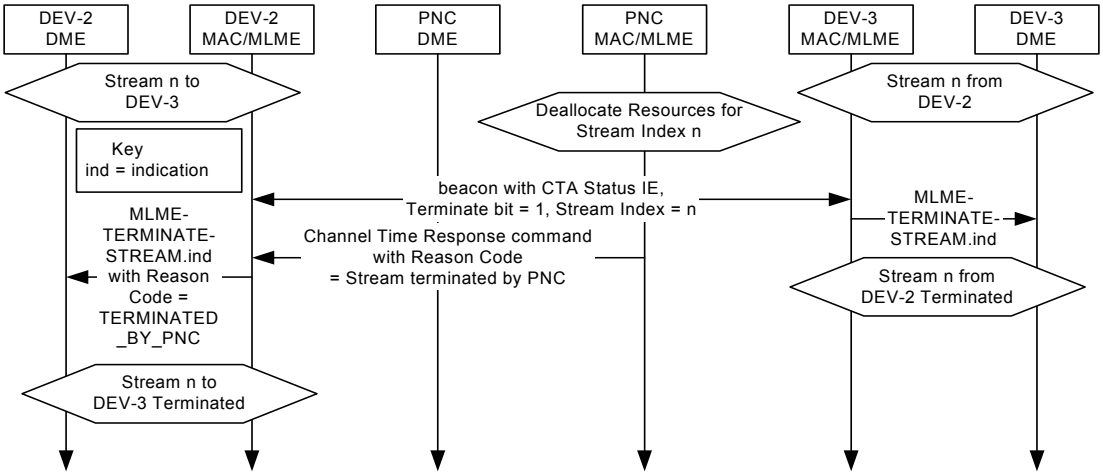

Figure 122—MSC of PNC terminating a stream

Insert the following paragraph after the fourth paragraph in 8.5.1.3:

If the originator of a stream is STP capable, the PNC may terminate any stream that has not been renewed within the ATP of the originator of the stream. If the PNC terminates the stream, it shall use the stream termination procedure described in this subclause for the PNC terminating a stream with the Reason Code in the Channel Time Response command set to "STP expired."

8.5.2 Asynchronous channel time reservation and termination

8.5.2.1 Asynchronous channel time reservation

Replace Figure 123 with the following:

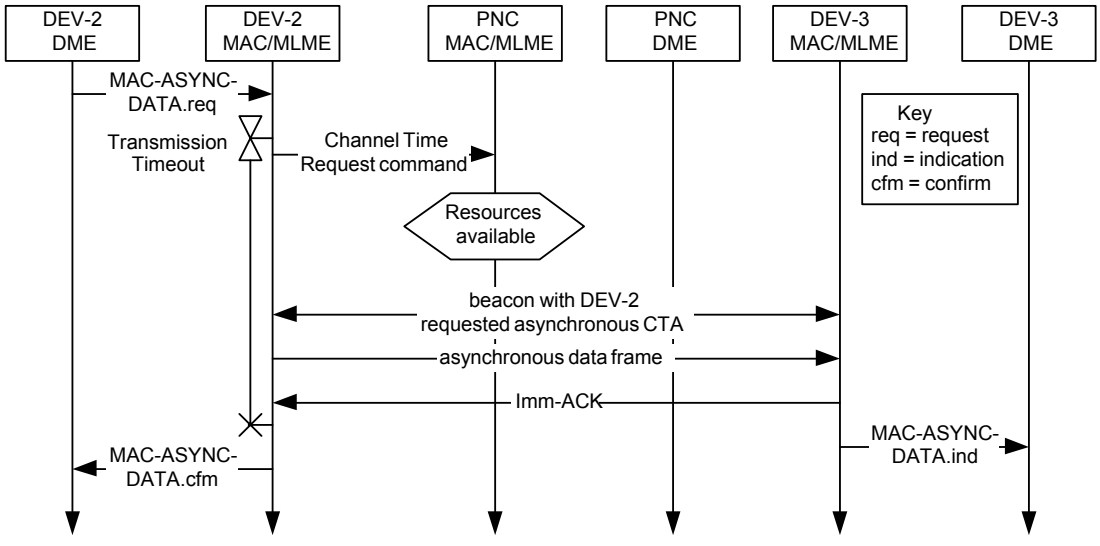

Figure 123— MSC for reserving asynchronous data channel time

8.5.2.2 Asynchronous channel time termination

After 8.5.2.2, insert the following subclause as 8.5.3:

8.5.3 Multicast group configuration

Group addresses are defined in IEEE Std 802-2001. Because IEEE Std 802.15.3 uses DEVIDs for addressing, the PNC may assign a DEVID to be used for a group address. The PNC also keeps track of all of the DEVs that request the use of a particular group address by maintaining a list of their DEVIDs and the associated group address. A group of DEVs that have been registered with the PNC using a particular group address is called a multicast group.

A DEV requests a DEVID for a group address, called a GrpID, from the PNC using the Multicast Configuration Request command, 7.5.10.1, with the Group Address field set to the desired group address and the Action field set to "Join." If a GrpID is not currently assigned as a DEVID for that Group Address and the PNC has the resources available, the PNC should assign an GrpID for the Group Address and respond to the originating DEV with the Multicast Configuration Response command, 7.5.10.2. If the request was successful, the PNC adds the originating DEV to the multicast group associated with the GrpID.

If the PNC has already assigned a GrpID for the address in the Group Address field and the PNC has the resources available, it shall add the originating DEV's DEVID to the multicast group.

If the originating DEV's request is granted, the PNC shall send the Multicast Configuration Response command to the originating DEV with the GrpID field set to the value assigned to that group address and the Reason Code set to "Success."

If the address in the Group Address field does not correspond to a valid group address, IEEE Std 802-2001, the PNC shall not assign a GrpID and shall send the Multicast Configuration Response command to the originating DEV with the GrpID set to zero and the Reason Code field set to "Failure, not a valid group address."

If the PNC is unable to fulfill the originating DEV's request for a GrpID, the PNC shall send the Multicast Configuration Response command to the originating DEV with the GrpID set to zero and the Reason Code field set to the appropriate value.

When a DEV no longer needs to use the group address, it shall send the Multicast Configuration Request command to the PNC with the Group Address field set to the address and the Action field set to "Leave." When the PNC receives this command, it shall remove the DEV from the multicast group and respond with the Multicast Configuration Response command with the GrpID field set to zero, the Group Address field set to the same value as in the request command and the Reason Code field set to "Success." The PNC shall always respond to a properly formatted Multicast Configuration Request command with the Action field set to "Leave" with a Multicast Configuration Response command with the Reason Code set to "Success." If the address in the Group Address field corresponds to a multicast group that has the originating DEV as a member, the PNC shall remove the DEV from the multicast group.

If the PNC is unable to support an existing multicast group, it shall terminate all CTAs with the GrpID as either the SrcID or DestID using the stream termination procedure in 8.5.1.3. The PNC shall also place a DEV Association IE in the beacon, following the repetition rules described in 8.6.4, with the DEVID set to the GrpID and the Association Status field set to disassociated. When a DEV receives such an IE it will know that the multicast group no longer exists.

If a multicast group no longer has any members, either due to disassociation or requests from the DEVs to leave the group, the PNC shall free the GrpID. A GrpID shall be allocated and reused according to the rules for assigning DEVIDs described in 8.3.1. A GrpID shall not be reported in the PNC Information command.

During PNC handover, the old PNC shall send one or more Announce commands, as defined in 7.5.5.2, to the new PNC with the Group ID IEs, as defined in 7.4.18, that correspond to the GrpIDs that are currently in use.

The MSC for a DEV successfully joining a multicast group is illustrated in Figure 123a.

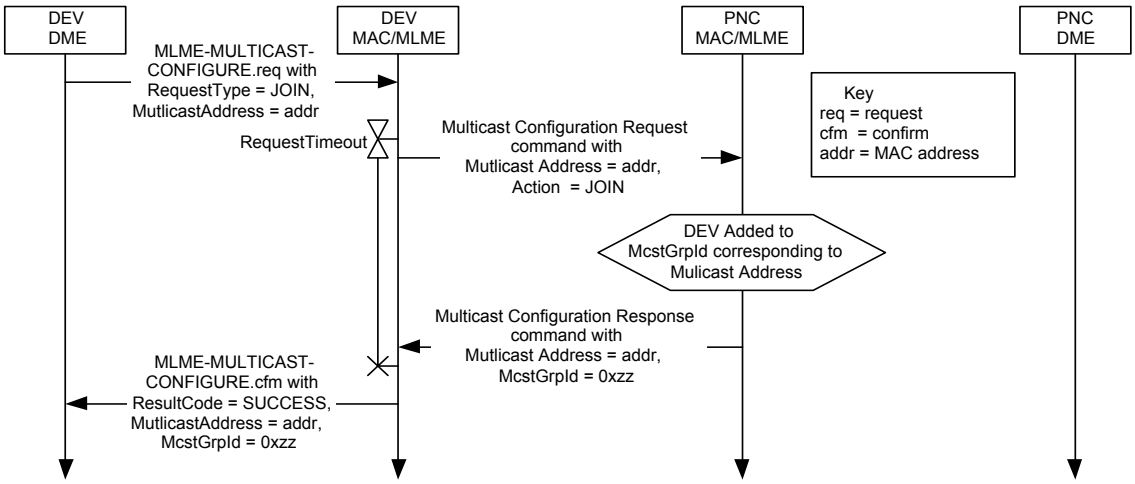

Figure 123a—Message sequence chart for a DEV successfully joining a multicast group

The MSC for leaving a multicast group is illustrated in Figure 123b

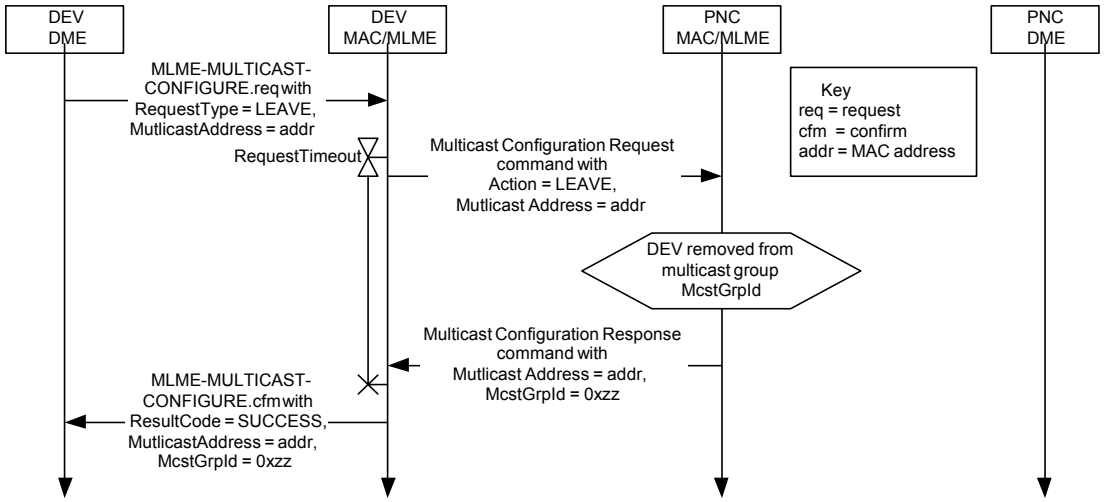

Figure 123b—Message sequence chart for a DEV leaving a multicast group

8.6 Synchronization

8.6.2 Beacon generation

Change the second paragraph in 8.6.2 as shown:

If the PNC determines that the beacon frame is too large or if it is going to split the information in the beacon frame, it may send one or more Announce commands with the SrcID set to the PNCID and the DestID set to the ~~BcstId~~ BcstID following the beacon. This is called an extended beacon. Unless it is specified otherwise, the term beacon applies to both the beacon frame and the Announce commands that make up the extended beacon. The IFS between the beacon frame, the first Announce command, and any additional Announce commands shall be less than a SIFS and greater than a MIFS. ~~The first Announce command shall be sent one MIFS following the beacon with any additional Announce commands following one MIFS after the the prior Announce command.~~ If the PNC sends some of the beacon information in the broadcast Announce commands, it shall set the More Data bit to indicate more data in the Frame Control field of the beacon frame and in all but the last Announce command frame used to communicate the IEs. The CAP or the CTAP, if the CAP ~~isn't~~ is not present, begins after the last Announce command that is part of the extended beacon. The PNC shall send CTA IEs, BSID IE, and the Parent Piconet IE, if present, only in the beacon frame and not in any of the broadcast Announce commands. The Announce commands are sent to the BcstID and so the ACK Policy field shall be set to no-ACK in these frames.

Insert the following paragraphs after the second paragraph in 8.6.2:

If all of the CTA IEs contained in the beacon are identical to the CTA IEs transmitted in the previous beacon, the PNC may set the CTA IEs Unchanged bit appropriately in the Piconet Mode field of the Piconet Synchronization Parameters field, 7.3.1.1.

If all of the IEs (other than CTA IEs) contained in the beacon are identical to the IEs (other than CTA IEs) transmitted in the previous beacon, the PNC may set the Other IEs Unchanged bit appropriately in the Piconet Mode field of the Piconet Synchronization Parameters field, 7.3.1.1.

8.6.3 Beacon reception

Insert the following paragraphs at the end of 8.6.3:

If a DEV receives a beacon frame with the CTA IEs Unchanged bit set in the Piconet Mode field of the Piconet Synchronization Parameters field, and if the DEV has correctly received the previously transmitted beacon, the DEV may assume that the CTA IEs contained in the beacon frame are identical to the CTA IEs received in the previous beacon. If the CTA IEs Unchanged bit is not set, or if the previous beacon frame was not correctly received, then the DEV shall consider that the CTA IEs in the current beacon contain new information.

If a DEV receives a beacon frame with the Other IEs Unchanged bit set in the Piconet Mode field of the Piconet Synchronization Parameters field, and if the DEV has correctly received the previously transmitted beacon, the DEV may assume that the IEs (other than CTA IEs) contained in the beacon frame and any Announce commands that make up the beacon are identical to the IEs (other than CTA IEs) received in the previous beacon. If the Other IEs Unchanged bit is not set, or if the previous beacon (beacon frame and any Announce commands) was not correctly received, then the DEV shall consider that the IEs (other than CTA IEs) in the current beacon contain new information.

8.6.4 Beacon information announcement

Insert the following row at the end of Table 57. The other rows are unchanged and are not shown.

Table 57—Repeated beacon announcements

Information element	Clause	Announced in	Intended for	Clause
Next PNC	7.4.20	mMinBeaconInfoRepeat	All DEVs	8.2.3b

Delete 8.6.5 and Figure 125.

8.8 Acknowledgment and retransmission

Change the paragraph in 8.8 as shown:

There are ~~three~~ four acknowledgment types defined for this standard~~;~~: no acknowledgment (no-ACK), immediate acknowledgment (Imm-ACK), ~~and~~ delayed acknowledgment (Dly-ACK), and implied acknowledgment (Imp-ACK).

8.8.3 Delayed acknowledgment

Change the text in the second, third, and fourth paragraphs in 8.8.3 as shown:

If the MAC header was correctly received, as described in 8.1, and the destination DEV accepts the use of Dly-ACK, it shall respond with a Dly-ACK frame, acknowledging the received data frame, if it was correctly received, and setting the Max Burst field to a value representing the maximum number of pMaxFrameBodySize MPDUs the source DEV may send in one burst. Because the receiver buffer requirement is equal to Max Burst field times pMaxFrameBodySize, the source may send as many smaller frames as will fit in the receive buffer window, up to a maximum of Max Frames, as provided in the Dly-ACK frame, as described in 7.3.2.2. The destination DEV may change the value of the Max Burst field and the Max Frames field ~~value~~ in each Dly-ACK frame. ~~The MPDUs ACKed field shall be set to one and the MPDU ID field shall contain the information for the frame that was sent to negotiate the Dly-ACK.~~

If the destination DEV wants to decline the use of the Dly-ACK mechanism, it shall reply with an Imm-ACK frame. ~~The source upon reception of the Imm-ACK shall send a MAC_ISOCH_DATA.confirm with the ResultCode set to DLY_ACK_FAILED to the FCSL. This implies acknowledgment of the data frame and additionally indicates that the use of the Dly-ACK policy has been refused by the destination. The FCSL would then notify the DME that the Dly-ACK negotiation failed, which might require a modification of the channel time allocation.~~

If the ~~max burst~~ value of the Max Burst field is set to zero, the source DEV shall stop transmitting in the current CTA and reopen the Dly-ACK mechanism by sending a single frame with the ACK Policy field set to Dly-ACK Request in the next CTA for this stream. If the value is not zero, the source DEV may continue transmission in the current CTA, if the time is available.

Insert the following paragraphs after the fourth paragraph in 8.8.3:

A DEV shall send a Dly-ACK frame in response to a correctly received MAC header, as described in 8.1, that has the ACK Policy set to Dly-ACK request and the DestID set to the DEVID of this DEV or, if applicable, the PNCID. Because the Dly-ACK frame is sent even if the FCS failed on the data frame with the

ACK Policy set to Dly-ACK request, the DEV shall check the contents of the Dly-ACK frame to determine which, if any, MSDUs were correctly received.

A DEV should not initiate a Dly-ACK procedure unless it has determined that the receiving DEV supports Dly-ACK by checking the DEV Capabilities field in the Capability IE, as defined in 7.4.11.

Change the eighth paragraph in 8.8.3 as shown:

The destination MAC shall deliver MSDUs for each isochronous stream in ascending MSDU number order to its FCSL. If necessary to accomplish this, a destination MAC may discard correctly received, as described in 8.1, (and potentially acknowledged) frames.

After 8.8.3, insert the following subclause as 8.8.3a:

8.8.3a Implied acknowledgment (Imp-ACK)

Imp-ACK is one method that allows a CTA to be used for bi-directional data transfer. With Imp-ACK, the ACK is implied when the target DEV sends any frame, called the response frame, in response to a frame that has an ACK policy of Imp-ACK.

A DEV should not initiate an Imp-ACK procedure unless it has determined that the receiving DEV supports Imp-ACK by checking the DEV Capabilities field in the Capability IE, as defined in 7.4.11.

The originating DEV sets the ACK Policy to Imp-ACK in a frame to begin the Imp-ACK procedure. Only the DEV that is the source of a CTA shall initiate the Imp-ACK procedure. When the target DEV successfully receives a frame with the ACK policy set to Imp-ACK, the DEV shall respond with either an Imm-ACK frame or with a command or data frame. The target DEV shall only send a frame to the originating DEV, unless transmit control of the CTA has been relinquished as described in 8.4.3.8.

If the target DEV sends a frame to the originating DEV in response to a frame with ACK policy of Imp-ACK, the target DEV shall set the ACK policy of the response frame to one of the following; no-ACK, Imm-ACK or Imp-ACK.

If there is not sufficient time in the CTA for the response frame and any required acknowledgments before the end of the CTA, the target DEV shall respond with an Imm-ACK. If the target DEV does not know the end time of the current CTA, it shall only send an Imm-ACK frame as the response frame. The target DEV shall only send one frame in response to a frame with the ACK policy set to Imp-ACK. The response frame may be of any length as long as the frame and any required ACKs do not exceed the end of the current CTA.

If the target DEV successfully receives a MAC header for a frame with the ACK policy set to Imp-ACK but does not successfully receive the frame body, i.e., the FCS check fails, it may still send a data or command frame in response. In this case, the target DEV shall set the Imp-ACK NAK field, 7.2.1.7, to indicate that it did not successfully receive the frame from the originating DEV in the response frame.

As in any CTA, a DEV should only transmit to a DEV that it knows is listening to the CTA. Imp-ACK shall not be used for broadcast or multicast frames, i.e., frames with the DestID set to BcstID, McstID or a GrpID. Imp-ACK shall not be used in the CAP or a contention CTA.

The use of Imp-ACK to enable bi-directional communication in a CTA does not imply that the stream is bi-directional as well. All streams in IEEE Std 802.15.3 are unidirectional. Only the originating DEV is allowed to send frames using the stream index assigned by the PNC. Frames transmitted in the opposite direction in the CTA shall use either the asynchronous stream index or a stream index that has been assigned to the DEV that is sending the frame.

If the destination DEV in a CTA has data to send to the source DEV of that CTA, it should set the more data bit in Imm-ACK frames that it sends to the source. This will indicate to the source that the destination is requesting that the source either use Imp-ACK or relinquish the CTA to the destination, as described in 8.4.3.8.

Figure 126a shows an example of the use of Imp-ACK where successive frames have an ACK policy of Imp-ACK.

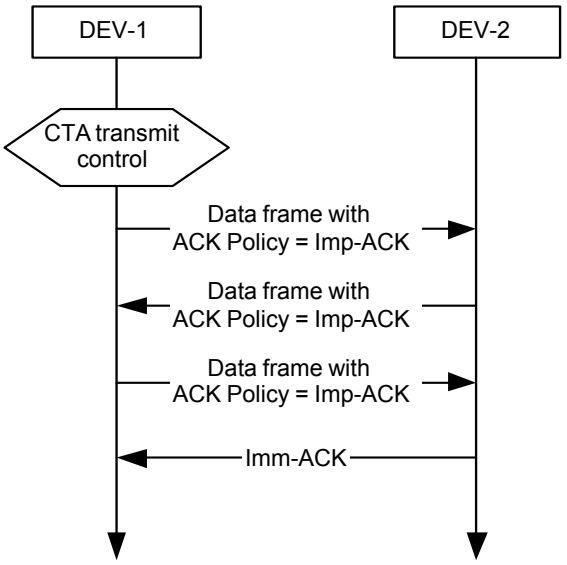

Figure 126a—Example of the use of Imp-ACK where the responding frame is sent with an ACK policy of Imp-ACK

Figure 126b shows an example of the use of Imp-ACK where one of the frames fails the FCS verification and the destination responds with the Imp-ACK NAK bit set.

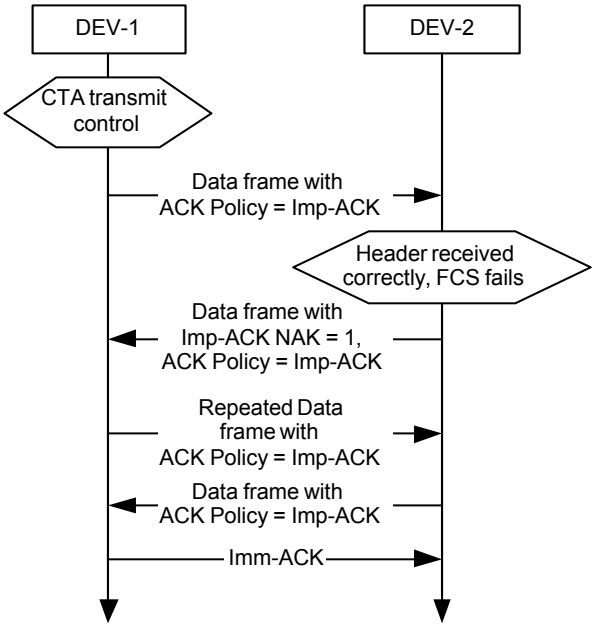

Figure 126b—Example of the use of Imp-ACK where one of the frames fails FCS verification

8.8.4 Retransmissions

Change the second paragraph in 8.8.4 as shown:

During CTAs within the CTAP when an Imm-ACK or Dly-ACK is expected~~,~~ but is not received ~~during a RIFS~~, the source DEV ~~shall~~ may start the retransmission of the frame (or the transmission of a new frame if the failed frame's retransmission limit has been met) ~~after the end of RIFS~~ a RIFS after the end of the last frame transmitted as long as there is enough channel time remaining in the CTA for the entire frame exchange.

8.9 Peer discovery

8.9.1 PNC information request

Replace Figure 127 with the following:

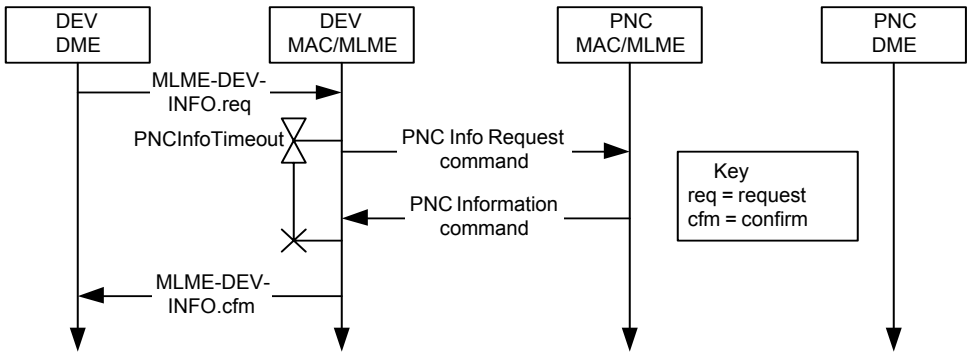

Figure 127—MSC for acquiring information regarding a specific DEV or all of the DEVs from the PNC using the PNC Information Request and PNC Information commands

8.9.2 Probe request and response

Change the second paragraph in 8.9.2 as shown:

A DEV may request information about an isochronous stream by sending a Probe Request command requesting the CTA Status IE, as described in 7.4.10, with the Request Index field set to the stream index of the stream for which CTA information is requested. If the Request Index field is set to zero, the DEV is requesting information about all isochronous streams directed to the requesting DEV and to the ~~BcstId~~ BcstID and ~~McstId~~ McstID. The PNC shall respond to a Probe Request command containing a request for the CTA Status IE by sending Probe Response command(s) containing the appropriate CTA Status IE(s).

Insert the following paragraph after the second paragraph in 8.9.2:

If a DEV requests the CTA Status IE for a stream index that is not currently allocated by the PNC, the PNC shall respond with a Probe Response command containing a CTA Status IE with the following information:

— The SrcID and DestID fields both set to the UnassocID.
— The Stream Index field set to the requested stream index.
— All other fields set to zero.

In 8.9.2, delete Figure 128 and the associated paragraph, which states "Figure 128 illustrates ...."

8.9.3 Announce command

In 8.9.3, delete Figure 129 and the associated paragraph, which states "Figure 129 illustrates ...."

8.9.4 Channel status request

In 8.9.4, delete Figure 130 and the associated paragraph, which states "Figure 130 illustrates ...."

8.9.5 Remote scan

In 8.9.5, delete Figure 131 and the associated paragraph, which states "Figure 131 illustrates ...."

8.9.6 PNC channel scanning

Change the third paragraph in 8.9.6 as shown:

If the PNC initiates a scan of one or more alternate channels, the PNC shall not transmit a beacon for one or more beacon intervals. The PNC shall not suspend beacon transmissions for more than twice mMinChannelScan ~~aMinChannelScan~~. The PNC, upon returning to its current channel and resuming the transmission of its beacons, shall increment the Time Token field by the number of beacons not sent during the time the PNC was scanning one or more alternate channels.

8.10 Changing piconet parameters

Change the second paragraph in 8.10 as shown:

A PNC shall not change either the pseudo-static CTAs or the pseudo-static CTA blocks during a piconet parameter change. If the parent needs to move a pseudo-static CTA because the superframe duration is being reduced, it shall do so prior to using the superframe duration change process, as described in 8.10.2. ~~If a child or 802.15.3 neighbor piconet has the same superframe duration as the parent, then it shall use the value of the Change Beacon Number field in the Piconet Parameter Change IE from the parent's beacon in the Piconet Parameter Change IE in its own beacon.~~ A child or 802.15.3 neighbor PNC shall use the value of the Change Beacon Number field in the Piconet Parameter Change IE from the parent's beacon to calculate the Change Beacon Number field in the Piconet Parameter Change IE in its own beacon. The exceptions to this are:

— When the parent is changing its PNID or BSID.

— A child or neighbor PNC decides not to change channels with the parent PNC and is shutting down, as described in 8.11.1.

8.10.3 Setting the PNID or BSID

Change the second paragraph in 8.10.3 as shown:

~~The PNID is chosen by the PNC when it starts the piconet and shall only be changed if the PNC detects another piconet with the same PNID on any channel.~~ The PNC shall choose a PNID when it starts a piconet; the PNID should be selected randomly. An existing piconet's PNID shall be changed only if the PNC detects another piconet with the same PNID on any channel. The same PNID may be persistent when the PNC restarts a piconet that ended without handing over control to a PNC capable DEV.

Change the seventh paragraph in 8.10.3 as shown:

The MSC in Figure 134 describes the ~~PNID and~~ BSID change process~~es~~.

Replace Figure 134 with the following:

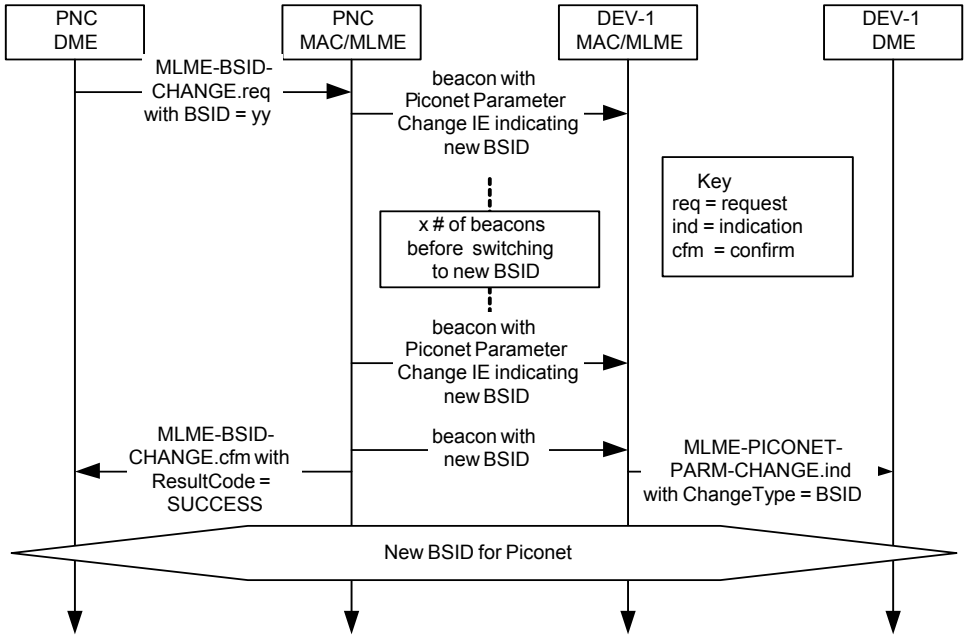

Figure 134—MSC for changing the BSID for a piconet

8.11 Interference mitigation

8.11.1 Dynamic channel selection

Change the fourth paragraph in 8.11.1 as shown:

No DEV ~~All DEVs~~ shall ~~not~~ transmit on the new channel until a beacon has been correctly received, as described in 8.1, on the new channel.

Insert the following paragraph and note after the sixth paragraph in 8.11.1:

In the last beacon transmitted prior to a channel change, the PNC should provide time at the end of the superframe for the DEVs in the piconet to change channels. The time required for a DEV to change channels is pPHYChannelSwitchTime.

NOTE—One method to allow time for the channel change is for the PNC not to place CTAs just prior to the first beacon that will be sent on the new channel.

Replace Figure 135 with the following:

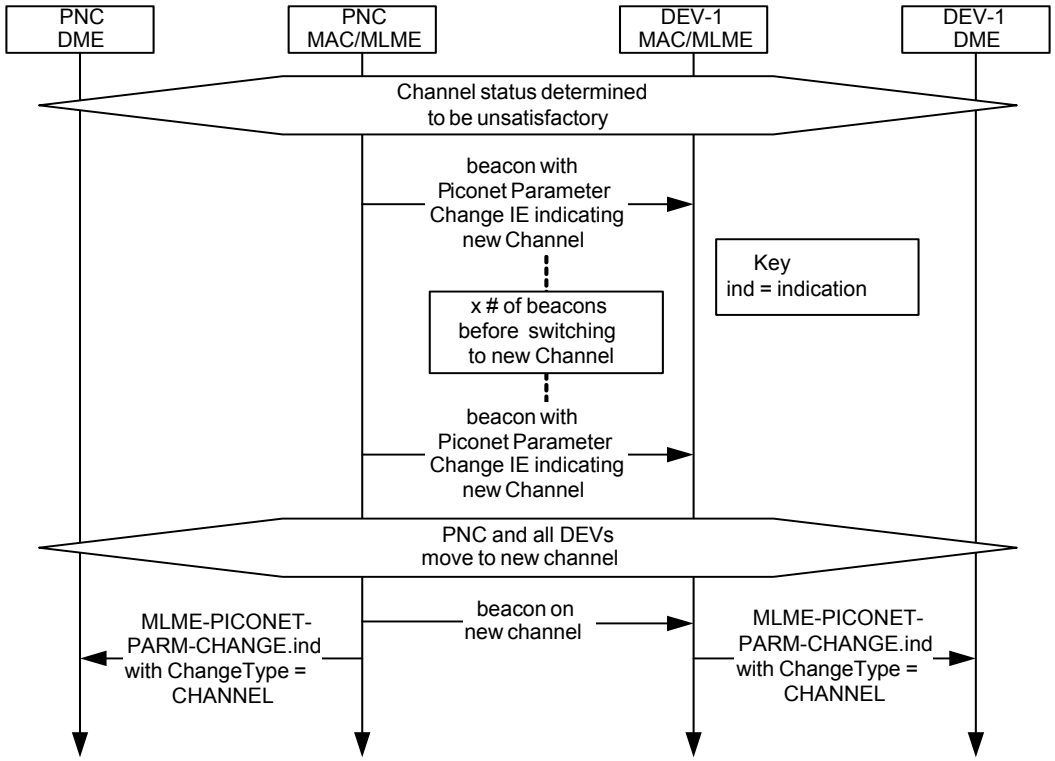

Figure 135—MSC for changing the piconet's channel

8.11.2 Transmit power control

Change the paragraph in 8.11.2 as shown:

Two independent forms of transmit power control (TPC) are available for 802.15.3 systems~~,~~: a maximum power for the CAP, the beacon, and directed MCTAs~~,~~ and an adjustable power in a CTA. The goal of TPC in the CAP is to prevent one DEV from having better access to the medium in the CAP due to a higher transmit power level. Adjustable transmitter power in the CTA is intended to support reduced power usage as well as reducing the overall interference levels generated by the piconet. The transmit power used in a CTA is not limited by the transmit power used for the CAP, beacon, or directed MCTAs.

8.11.2.2 Adjustable transmitter power in a CTA

In 8.11.2.2, delete Figure 136 and the associated paragraph, which states "Figure 136 illustrates ...."

8.13 Power management

Change the third paragraph in 8.13 as shown:

A DEV shall always establish membership with the piconet in ACTIVE mode. ~~If the DEV MLME changes its PM mode to ACTIVE without the prompting of the DME, it notifies the DME with the MLME-PMMODE- ACTIVE.indication primitive as described in 6.3.22.7.~~

8.13.1 Piconet synchronized power save (PSPS) mode

Change the first paragraph in 8.13.1 as shown:

A DEV in PSPS mode shall listen to all system wake beacons, as announced by PNC, and is required to be in the AWAKE state during system wake superframes as indicated in Table 59. The wake beacon for PSPS DEVs is determined by the PNC. PSPS mode is identified by a PS Set Index equal to one. If a DEV in PSPS mode does not correctly receive the system wake beacon, it shall be in the AWAKE state during the expected beacon ~~tranmsision~~ transmission times to receive the following beacons until a beacon is correctly received, as described in 8.1.

8.13.2 Device synchronized power save (DSPS) mode

8.13.2.1 Creation, use, and management of DSPS sets

Change the seventh paragraph in 8.13.2.1 as shown:

When the last member of a DSPS set has left, the DSPS set shall be terminated by the PNC. The PNC may also delete a DSPS set. The PNC deletes a DSPS set by placing the PCTM IE, 7.4.8, in the beacon and setting the bit for each of the DEVs that are members of that DSPS set and are in a PS mode. The PNC also removes from the beacon, if present, the PS Status IE of the DSPS set that the PNC is deleting. When the PNC determines that a DEV in that DSPS set is in AWAKE state, it shall send an SPS Configuration Response command to the DEV with SPS Set Index field set to the value of the DSPS set index that is being deleted and the Reason Code field set to "DSPS set deleted by PNC." If a DEV is in DSPS mode and it receives a beacon that does not have the PS Status IE for its DSPS set, the DEV should send a PS Set Configuration Request command to the PNC to determine if the DSPS set has been deleted.

8.13.2.2 CTA timing in DSPS mode

Delete "subrate" and insert "sub-rate" in its place in Figure 138 and Figure 139.

8.13.4 Message sequence charts for power save modes

Replace Figure 140 with the following:

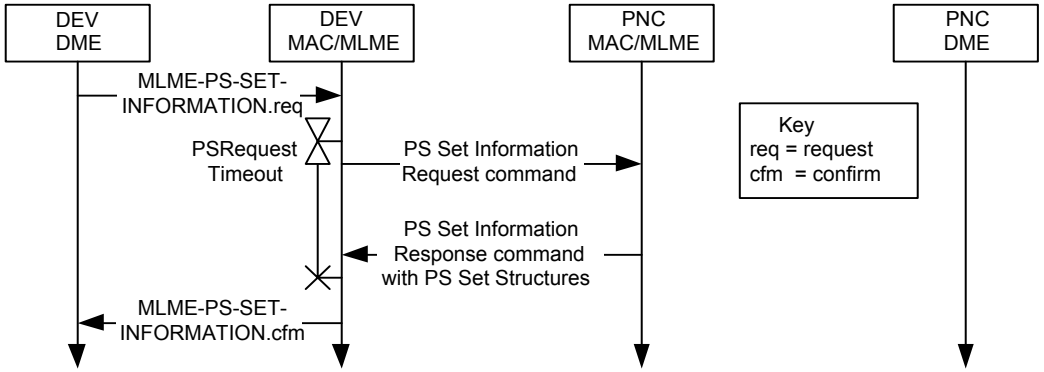

Figure 140—MSC for PS set information exchange

Replace Figure 141 with the following:

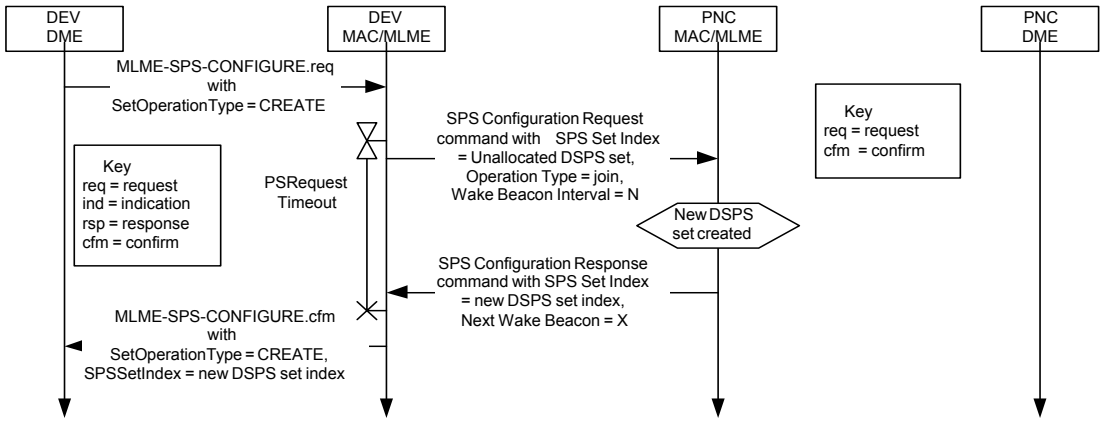

Figure 141—MSC for DSPS set creation

Replace Figure 142 with the following:

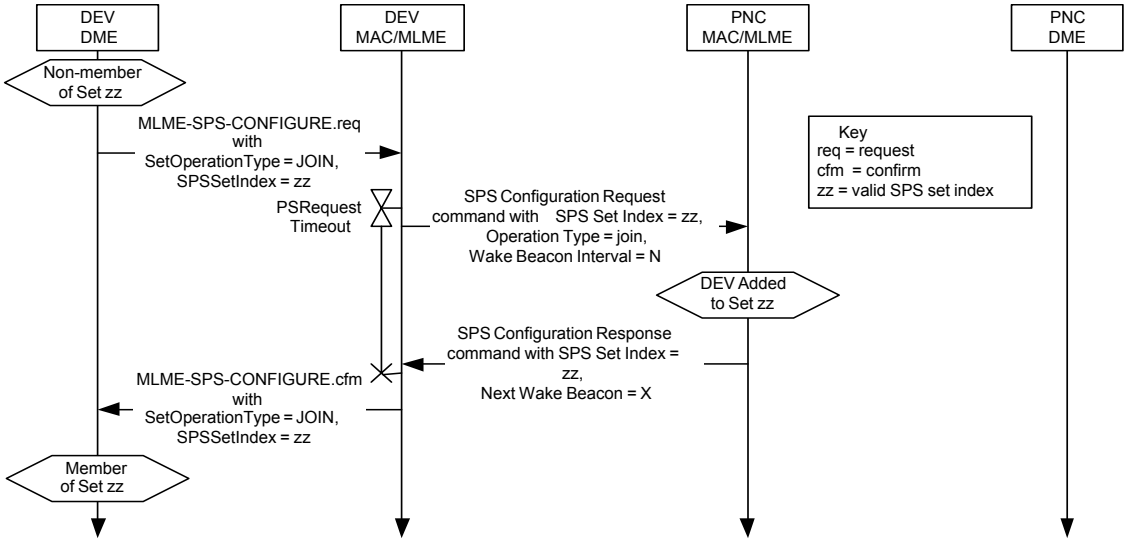

Figure 142—MSC showing a DEV joining an existing SPS set

Replace Figure 143 with the following:

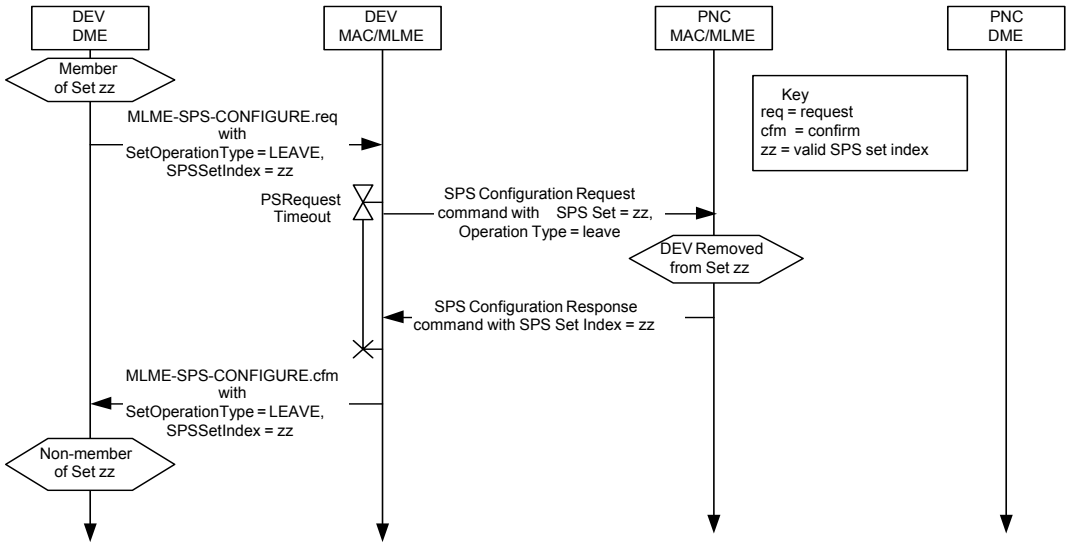

Figure 143—MSC showing a DEV leaving an SPS set

Replace Figure 144 with the following:

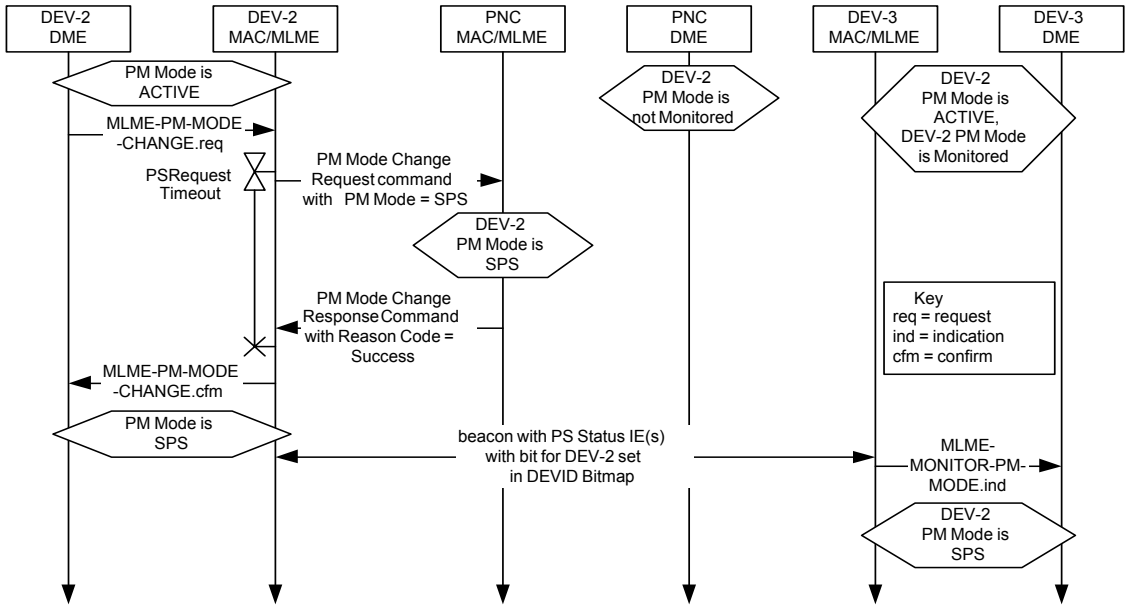

Figure 144—MSC showing DME initiated PM mode change from ACTIVE to an SPS mode

Replace Figure 145 with the following:

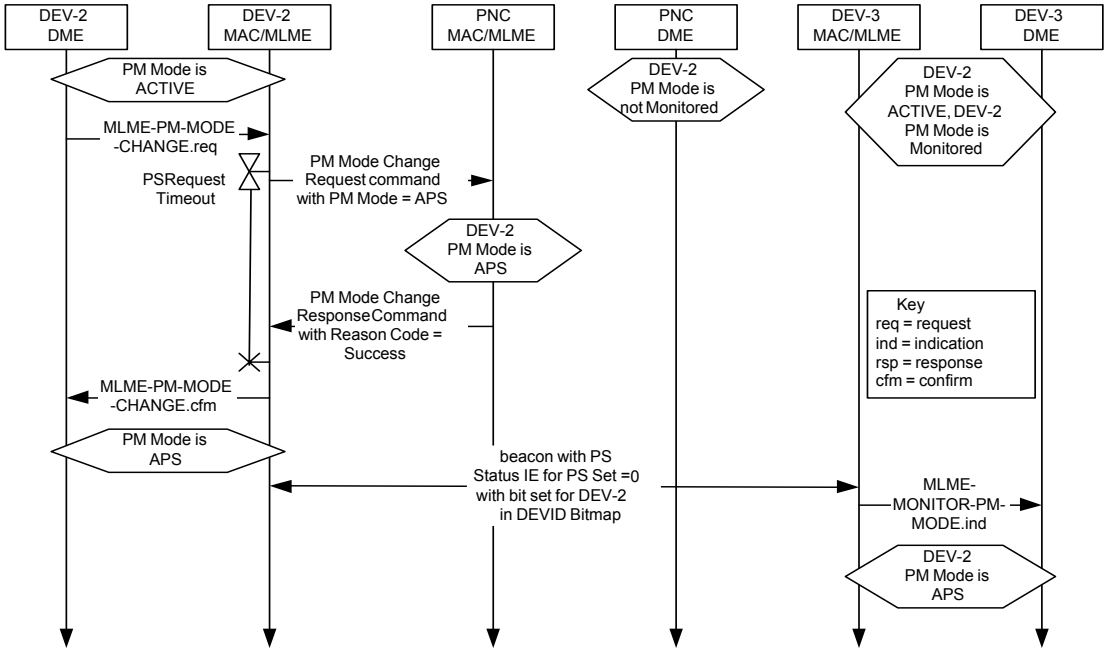

Figure 145—MSC showing DME initiated PM mode change from ACTIVE to APS

Replace Figure 146 with the following:

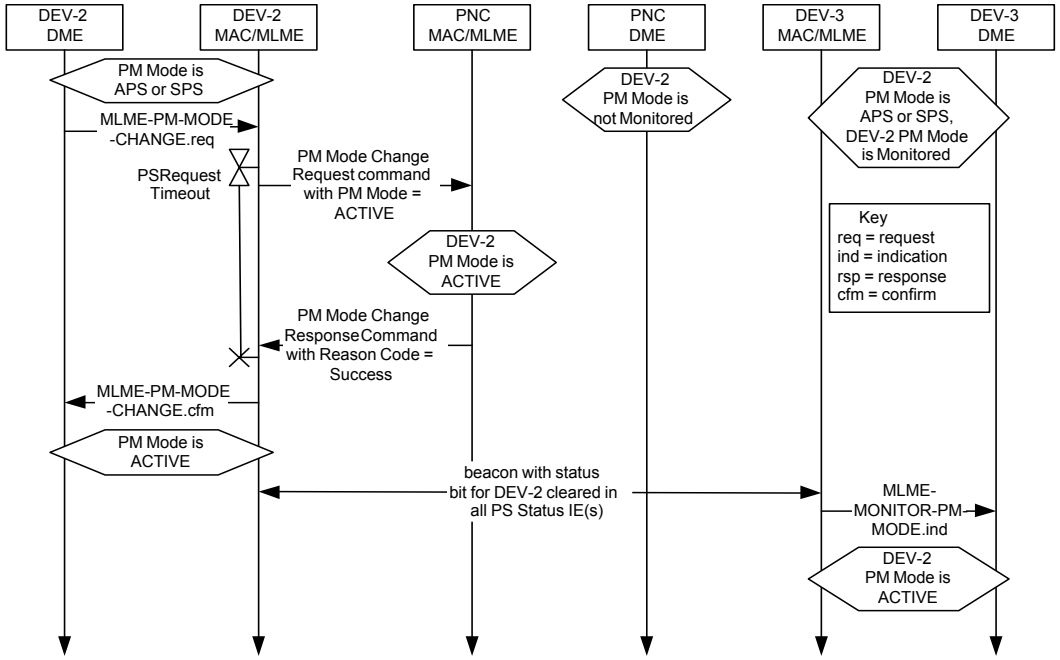

Figure 146—MSC showing DME initiated PM mode change from any PS mode to ACTIVE

Replace Figure 147 with the following:

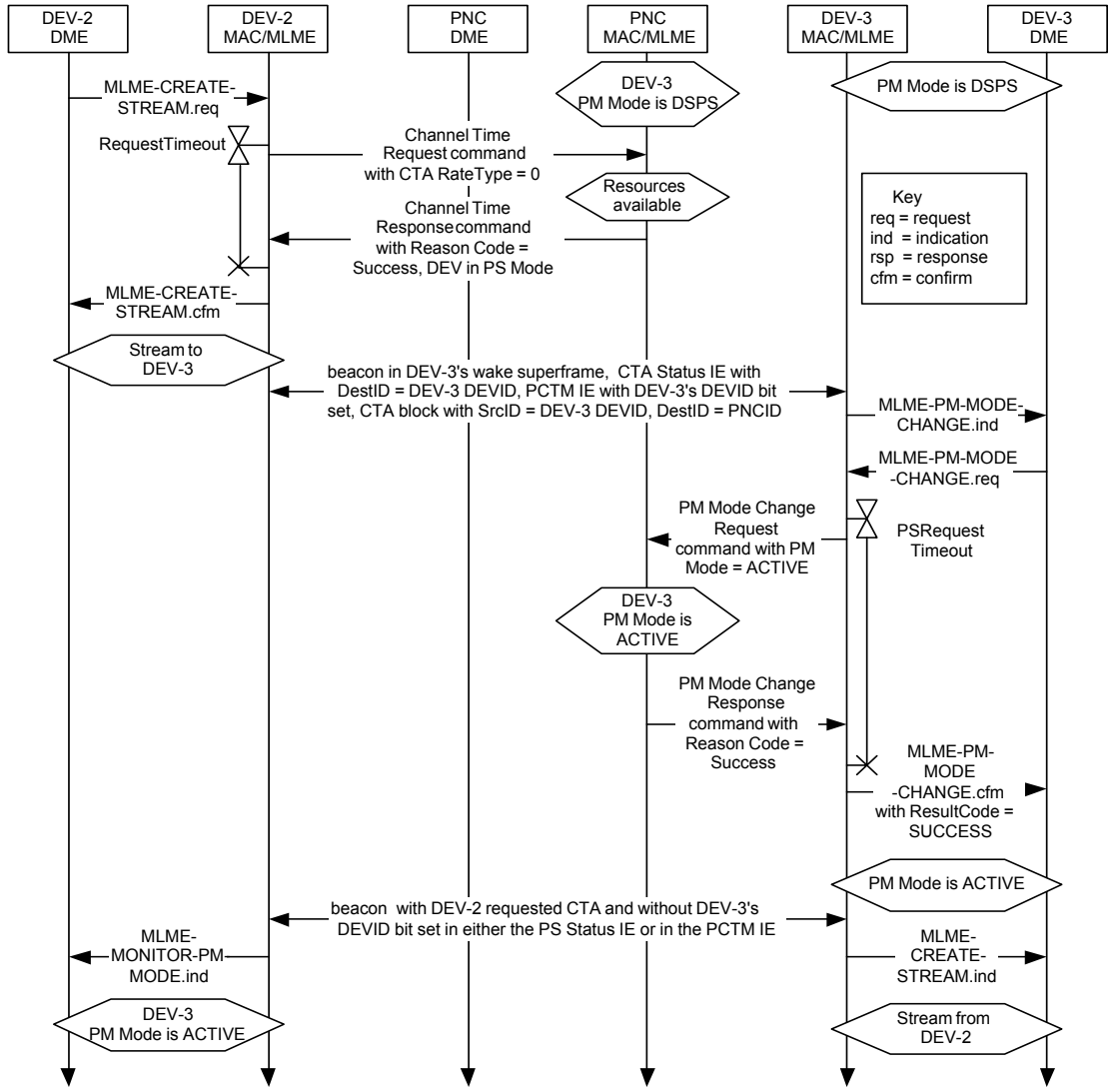

Figure 147—Message sequence chart showing MLME initiated PM mode change from DSPS to ACTIVE in response to a new channel time allocation

Delete 8.14 and insert the following subclause as 8.14:

8.14 ASIE operation

The ASIE is used to implement out of scope features that require additional functionality by both the PNC and one or more of its piconet member DEVs. The additional functionality is defined as an enhancement that does not violate the standard and allows DEVs that do not have the functionality to operate normally. The Application Specific Data field in this IE provides the messages that are only interpreted by the targeted DEV.

Multiple ASIEs may be placed in the beacon by the PNC. The designer should minimize the size of each ASIE used to support the custom application.

To place a new ASIE in the beacon, the DEV sends the ASIE Request command to the PNC with the Request Type field set to "add" and the ASIE Index field set to zero, as shown in Figure 147a. The requesting DEV selects a unique value of the Request ID field in the ASIE Request command, 7.5.9.3, to be able to determine which ASIE Response command is associated with this request. The ASIE Index field shall be ignored by the PNC in an ASIE Request command when the value of the Request Type field is set to "add."

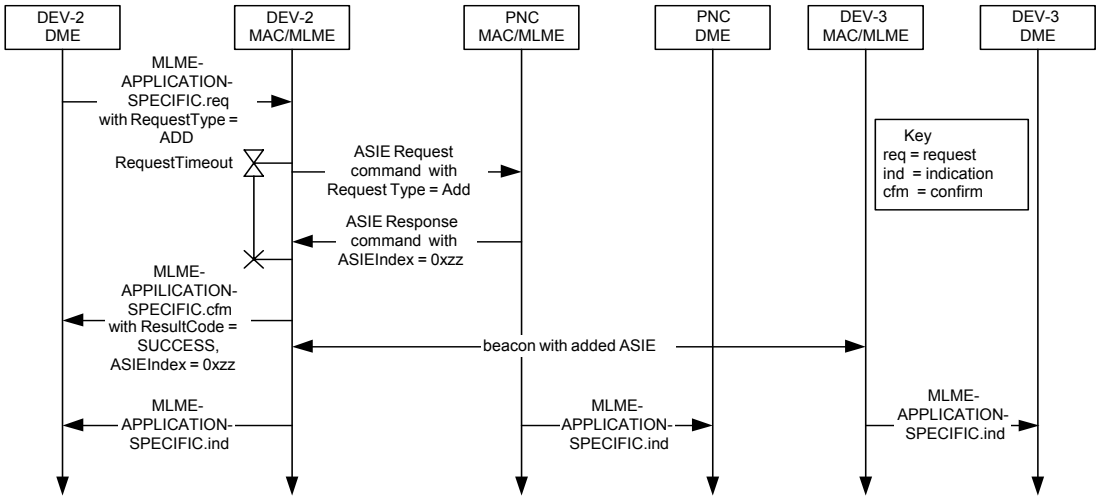

Figure 147a—Message sequence chart for a DEV adding an ASIE to the beacon

If the PNC grants the request to place the ASIE in the beacon, it shall respond with an ASIE Response command to the originating DEV with the Request ID set to the same value as in ASIE Request command, the ASIE Index set to an unused value and the Reason Code set to "success."

If the PNC refuses the request to place the ASIE in the beacon, it shall send an ASIE Response command to the originating DEV with the Request ID set to the same value as in the ASIE Request command and the Reason Code set to the appropriate value. The ASIE Index shall be set to zero by the PNC and shall be ignored by the originating DEV upon reception when the Reason Code is other than "success."

A DEV may modify an existing ASIE by sending the ASIE Request command to the PNC with the Request Type field set to "modify," the ASIE Index field set to the value assigned by the PNC for that ASIE and the Request ID field to a value unique for that DEV. This process is illustrated in Figure 147b.

If the PNC allows the request to modify an existing ASIE, it shall respond with an ASIE Response command to the originating DEV with the Request ID and ASIE Index set to the same values as in ASIE Request command and the Reason Code set to "success."

If the PNC refuses the request to modify the ASIE in the beacon, it shall send an ASIE Response command to the originating DEV with the Request ID and the ASIE Index set to the same values as in the ASIE Request command and the Reason Code set appropriately. If the PNC refuses a request to modify an ASIE, it does not delete the ASIE, but it shall continue placing the existing ASIE in the beacon.

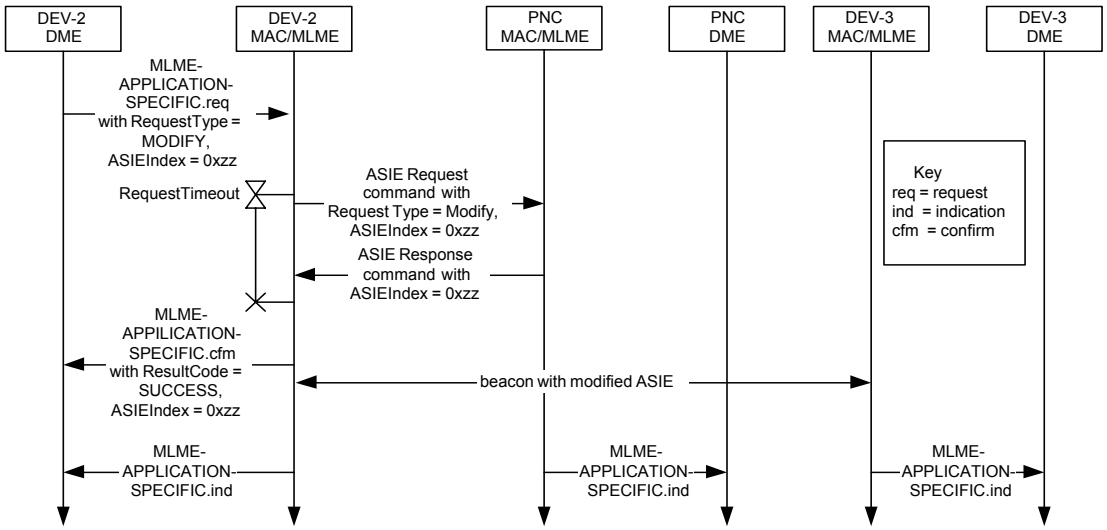

Figure 147b—Message sequence chart for a DEV modifying an ASIE in the beacons

A DEV may also request that the PNC remove the ASIE from the beacon by sending the ASIE Request command to the PNC with the Request Type field set to "remove" and the ASIE Index field set to value for the ASIE that is to be removed. If the PNC successfully receives this command with the ASIE Index value that corresponds to an ASIE that is being currently managed by the PNC, it shall no longer place that ASIE in the beacon and shall send an ASIE Response command to the DEV with the Reason Code set to "success," and the Request ID and ASIE Index set to the same values as in ASIE Request command. This procedure is illustrated in Figure 147c.

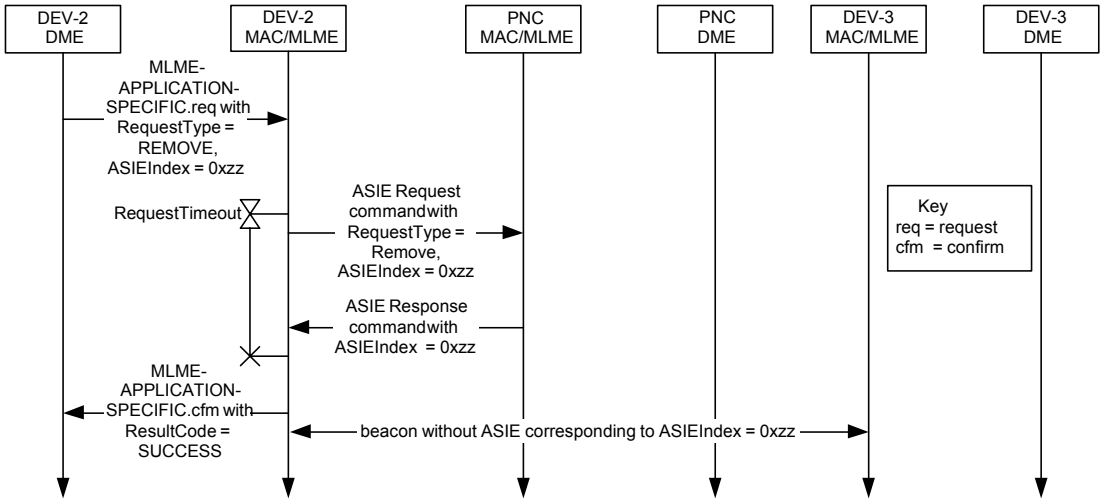

Figure 147c—Message sequence chart for a DEV removing an ASIE from a beacons

If the value of the ASIE Index in an ASIE Request command with Request Type set to "modify" or "remove" does not correspond to an ASIE that is currently being managed by the PNC, the PNC shall refuse the request with Reason Code set to "Unassigned ASIE Index."

Only the DEV that requested an ASIE be added to the beacon shall be allowed to modify or remove that ASIE.

Once the PNC has allowed a request to add or modify an ASIE, it shall place the ASIE in every beacon until the originating DEV either has been disassociated or has requested that the PNC remove the ASIE from the beacon. The PNC shall remove all ASIEs that were sent by a DEV when that DEV disassociates from the piconet. ASIEs are not transferred during handover and so DEVs that need ASIEs in the beacon will need to send ASIE Request commands to the new PNC once it has taken over the piconet.

8.15 MAC parameters

Insert the following row at the end of Table 60. The other rows are unchanged and are not shown.

Table 60—MAC sublayer parameters

Parameter	Value
mCTARelinquishTimeout	3×SIFS + 2×(duration of Imm-ACK frame)

Insert the following row in Table 61 after the pBackoffSlot row. The other rows are unchanged are not shown.

Table 61—MAC sublayer parameters - 2.4 GHz PHY dependent

Parameter	Subclause
pPHYChannelSwitchTime	11.2.7.5

9. Security

9.1 Security mechanisms

9.1.1 Security membership and key establishment

Change 9.1.1 as shown:

The method by which a DEV becomes a member of a security relationship and obtains the appropriate key is outside of the scope of this standard. The Security Message command has been included as a special command to assist in the implementation of vendor specific protocols for establishing security relationships and any related data. It can be achieved with higher layer protocols that are not specified in this standard. The MAC/MLME is informed of changes to the membership of a security relationship and the key for that relationship with the MLME-MEMBERSHIP-UPDATE primitive, as described in ~~6.3.9.1~~ 6.3.7.1.

9.1.4 Data integrity

Change 9.1.4 as shown:

Data integrity uses an integrity code to protect data from being modified by parties without the cryptographic key. It further provides assurance that data came from a party with the cryptographic key. Integrity may be provided using a key shared by all piconet DEVs or using a key shared between only two DEVs. All

secure data frames that fail integrity checks are passed to the DME using MLME-SECURITY-ERROR.indication~~indicate~~ and no other action is taken on the frame by the MLME.

9.1.6 Command integrity protection

Change 9.1.6 as shown:

The integrity of commands may be protected just like any other data. Integrity protected commands sent between the PNC and a DEV shall be protected using the PNC-DEV management key. All secure commands that fail integrity checks are passed to the DME using MLME-SECURITY-ERROR.indication~~indicate~~ and no other action is taken on the frame by the MLME.

9.3 Security support

9.3.4 Membership update

Change the first and second paragraphs in 9.3.4 as shown:

When the DME determines that there has been a change of membership status with a particular DEV or when a management ~~or data~~ key is changed, the DME shall issue an MLME-MEMBERSHIP-UPDATE.request to its MLME. This membership status change or key change may be the result of a successful establishment of a security relationship, key update process, termination of a security relationship, or some other event. The process by which this change occurs is outside the scope of this standard.

When the MLME receives the MLME-MEMBERSHIP-UPDATE.request, it shall first examine the TrgtID to determine the membership relationship to modify. If the TrgtID is the PNCID, ~~the data key corresponds to the piconet group data key,~~ the management key corresponds to the management key for the relationship with the PNC and the MembershipStatus indicates whether the DEV is a secure member of the piconet. Otherwise, the management key ~~and data key correspond to keys~~ is for a peer-to-peer relationship with the DEV indicated by the TrgtID and the MembershipStatus indicates whether the DEV shares a secure relationship with that peer DEV. If the OrigID is the PNCID, then the management key corresponds to the management key for a secure relationship with a DEV in the piconet and will be used for PNC related frames.

Change the fifth and sixth paragraphs in 9.3.4 as shown:

If the MembershipStatus is set to MEMBER, the MLME shall examine the KeyInfoLength field to determine if a new key is being added or a key is being deleted. If the KeyInfoLength field is set to zero, the MLME shall securely delete the key and SECID corresponding to the management key ~~or data key~~ for that relationship ~~depending on whether the KeyType is set to MANAGEMENT or DATA~~. When the key is deleted, ~~If the deleted key is the management key stored for the relationship,~~ the DEV is unable to transmit or successfully receive frames to any DEV that require protection with the management key, as described in 9.3.8, but the DEV may continue to use the data key corresponding to that relationship and the piconet group data key. This may occur, for instance, during PNC handover, in which the management key with the PNC is no longer valid (since the PNC has changed), but the piconet group data key is still valid. ~~If the deleted key is a data key stored for a peer-to-peer relationship, the DEV is unable to transmit or successfully receive frames that require protection with the data key, as described in 9.3.8, but the DEV may continue to use the management key and piconet group data key. If the deleted key is the piconet group data key, the DEV is unable to transmit or successfully receive frames that require protection with that key, as described in 9.3.8, with the exception of secure beacons, as described in 9.3.6~~.

If the MembershipStatus is set to MEMBER and the KeyInfoLength field is not 0, ~~the MLME shall examine the KeyType field to determine which key is to be updated. If the KeyType is set to MANAGEMENT,~~ the MLME shall set the SECID, key originator field, and key for the management key of this relationship to the

values in the SECID, KeyOoriginator, and KeyInfo fields, respectively, from the MLME-MEMBERSHIP-UPDATE.request. ~~If the KeyType is set to DATA, the MLME shall ignore the key originator field and set the SECID and key to the values in the SECID and KeyInfo fields respectively from the MLME-MEMBERSHIP-UPDATE.request.~~

9.3.5 Secure frame generation

Change the first paragraph in 9.3.5 as shown:

When a DEV wishes to send a secure frame, it shall use the keying material required for the type of frame and by the relationship between the sending DEV and the receiving DEV. For each security relationship, there are two keys used to protect secure frames: a management key and a data key. Table 62 provides a listing of which of the keys shall be used to protect secure frames and which frames shall be sent without security. A DEV shall not send a secure frame if the only key selection in Table 62 is "none." ~~'none'.~~ A DEV shall not send an unprotected frame or a frame with an incorrect SECID when security is required for that frame. If the DEV is unable to find the corresponding key that is to be used, the MLME shall return an MLME-SECURITY-ERROR.indication to the DME with the ReasonCode set to UNAVAILABLE-KEY and shall not transmit the requested frame.

Insert the following paragraph after the fourth paragraph in 9.3.5:

A DEV shall only send frames that have increasing SFCs in a superframe, except for frames that are retransmitted with the same SFC without any intervening frames having been sent.

9.3.6 Secure frame reception

Change the first paragraph in 9.3.6 as shown:

Before any security operations have been performed on a received frame, the DEV shall check the FCS. Table 62 provides a listing of the keys that shall be used to protect secure frames and the frames that shall be sent without security. A DEV may ignore any secure frame if the only key selection in Table 62 is "none." ~~'none'.~~ A DEV ~~may~~ shall ignore any non-secure frame or a secure frame with an incorrect SECID when security is required.

Change the third, fourth, and fifth paragraphs in 9.3.6 as shown:

When a DEV receives a secure beacon frame, as defined in 7.3.1.2, ~~(a beacon with the SEC field in the Frame Control field set to one,~~ the DEV shall determine if the received time token is greater than the CurrentTimeToken and less than the LastValidTimeToken + mMaxTimeTokenChange ~~aMaxTimeTokenChange~~. If not, the MLME shall return an MLME-SECURITY-ERROR.indication to the DME with the ReasonCode set to BAD-TIME-TOKEN and shall not perform any additional operations on the received beacon. The DEV shall also determine if the SECID matches the SECID of the piconet group data key stored in the MAC/MLME, or the SECID of a valid old piconet group data key, as described in 9.2.5. If the SECID does not match, the DEV may set the CurrentTimeToken to the value in the beacon and request a new piconet group data key, as described in 9.3.2. If both of these checks succeed, the DEV shall check the integrity code on the beacon using the piconet group data key. If this succeeds, the DEV shall accept the beacon and set the LastValidTimeToken and CurrentTimeToken to be the time token in the beacon. If the DEV is able to determine that it missed a beacon or that the beacon was corrupted and if CurrentTimeToken is less than LastValidTimeToken + mMaxTimeTokenChange ~~aMaxTimeTokenChange~~ – 1, the DEV should increment the CurrentTimeToken to maintain synchronization with other DEVs in the piconet.

When a DEV receives a secure non-beacon frame, it shall use the appropriate keying material depending on the type of frame, SECID, and ~~source address~~ SrcID found in the frame. If the SECID in the frame does not correspond to known keying material in the receiving DEV, the MLME shall return an

MLME-SECURITYERROR. indication to the DME with the ReasonCode set to UNAVAILABLE-KEY and shall not perform any additional operations on the received frame. A DEV shall reject all frames that do not have an SFC that is strictly greater than the last SFC received from a DEV in that superframe.

If there are no previous security errors in the processing of the frame, the DEV shall apply the operations defined by the symmetric key security operations to the frame, as defined in 10.3.2~~ see Table 10.3.2~~. If any of the security operations fail, the MLME shall return an MLME-SECURITY-ERROR.indication to the DME with the ReasonCode set to FAILED-SECURITY-CHECK and shall not perform any additional operations on the received frame. If the security operations have been successfully performed and the frame has been modified appropriately, the DEV may then continue to process the frame.

9.3.8 Key selection

Change and insert the following rows in Table 62 as indicated. The other rows are unchanged and are not shown.

Table 62—Key selection for secure frames

Frame type or command	None	PNC-DEV mgmt. key	Piconet group data key	Peer-to-peer mgmt. key	Peer-to-peer data key	Comment
Data frame	~~X~~		X		X	~~Data frames may be sent at any time with or without security.~~ Only secure data frame shall be exchanged between DEVs that have a secure relationship. Secure data frames between DEVs that share a peer-to-peer relationship shall use the peer-to-peer data key, otherwise they shall use the piconet group data key.
LLC/SNAP data frame			X		X	Only secure LLC/SNAP data frames shall be exchanged between DEVs that have a secure relationship. Secure LLC/SNAP data frames between DEVs that share a peer-to-peer relationship shall use the peer-to-peer data key, otherwise they shall use the piconet group data key.
Announce response	X	X	X	X		If the Announce response command is sent to or from the PNC before the DEV becomes a secure member of the piconet, the command shall not be secured by any key. If the DEVs do not share an individual relationship, the piconet group data key shall be used. Otherwise, the management key (peer-to-peer or PNC-DEV) for the relationship shall be used.
PM mode change response		X				
ASIE request		X				

Table 62—Key selection for secure frames *(continued)*

Frame type or command	None	PNC-DEV mgmt. key	Piconet group data key	Peer-to-peer mgmt. key	Peer-to-peer data key	Comment
ASIE response		X				
Multicast configuration request		X				
Multicast configuration response		X				

9.4 Protocol details

9.4.1 Security information request and distribution

Change the first paragraph in 9.4.1 as shown:

A DEV establishing membership in a security relationship~~, a DEV~~ may request or send security information to another DEV. This most often is done directly before or during the PNC handover process, but may be done at any time.

Replace Figure 148 with the following:

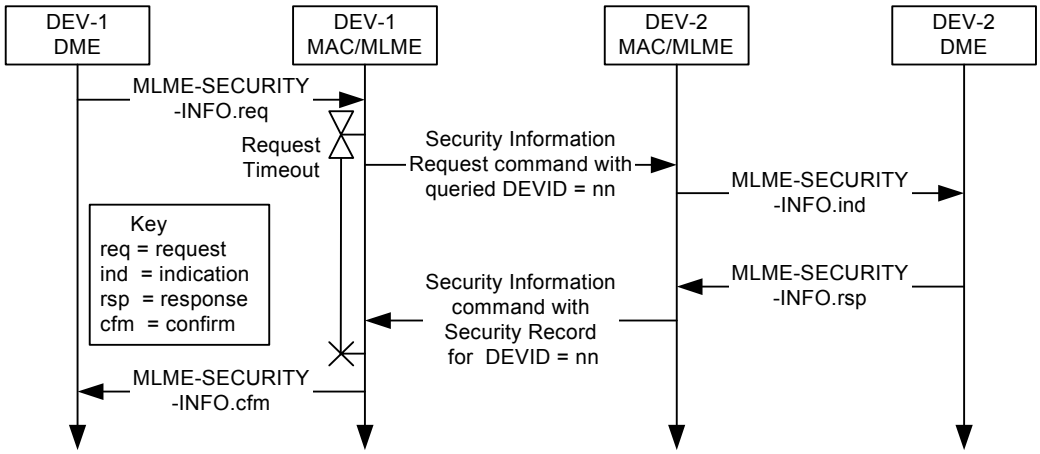

Figure 148—Message sequence chart for DEV-DEV Security Information request

Replace Figure 149 with the following:

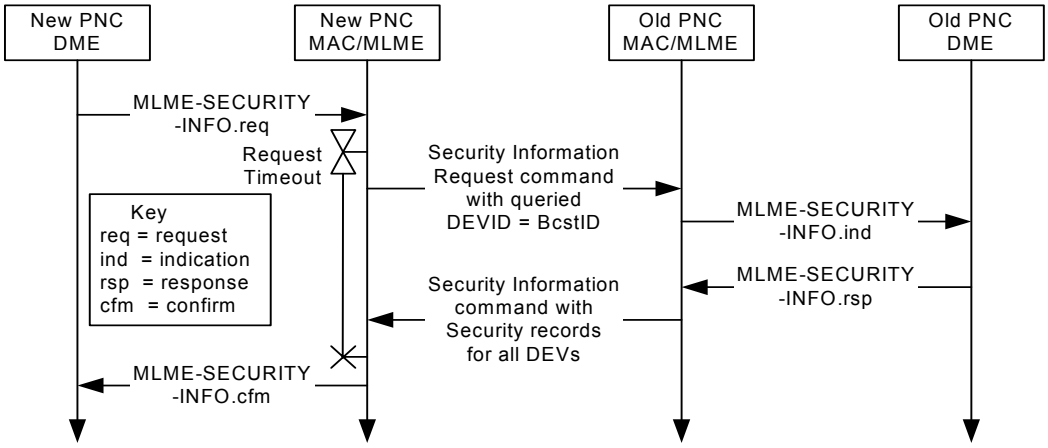

Figure 149—Message sequence chart for New PNC-Old PNC security information transfer

Replace Figure 150 with the following:

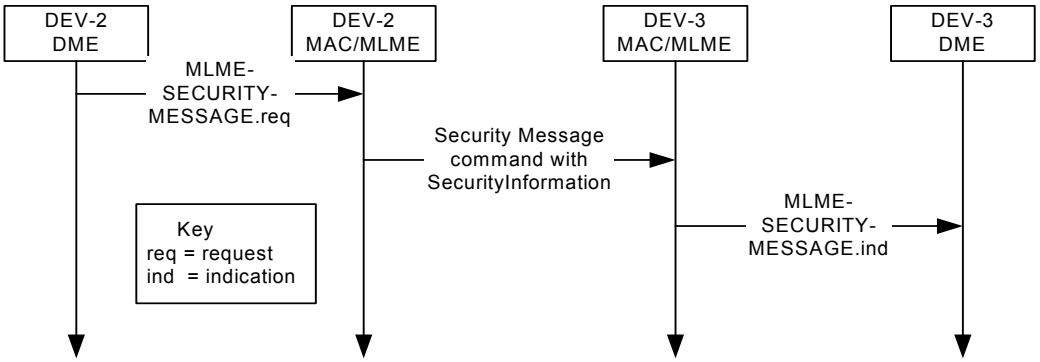

Figure 150—Message sequence chart for sending security information with the Security Message command

9.4.2 Key distribution protocol

Change the first paragraph in 9.4.2 as shown:

In a secure piconet or in a secure peer-to-peer relationship, the key originator may wish to update the current data protection key by initiating the ~~distribute~~ key distribution protocol described here. For a change in the piconet group data key, the PNC sends the new piconet group data key to each member of the piconet using the Distribute Key Request command. DEVs do not respond to a Distribute Key Request command sent by the PNC, other than with an Imm-ACK if the frame FCS is valid. Note that the Imm-ACK does not indicate that the Distribute Key Request command has passed cryptographic verification, only that the FCS was valid. For a change in a peer data key, the key originator in the relationship initiates the ~~distribute~~ key distribution protocol. The key originator sends the Distribute Key Request command to the DEV with which it is updating the key. A DEV that successfully receives a Distribute Key Request command that also passes the data authentication, 10.4.2, shall respond to the key originator with the Distribute Key Response command. The key originator should initiate this protocol with each DEV with their respective shared key whenever the key is updated.

In 9.4.2, delete Figure 151 and the associated text, which states "Figure 151 illustrates...."

In 9.4.2, delete Figure 152 and the associated text, which states "Figure 152 illustrates...."

9.4.3 Key request protocol

Change the first paragraph in 9.4.3 as shown:

In a secure piconet, if a DEV receives a frame or beacon with an unknown SECID, it may initiate the request key protocol ~~described here~~ in order to obtain the unknown key from the key originator of the relationship. The DEV initiates the protocol by sending the Request Key command to the key originator. When the key originator receives a Request Key command that has a valid Integrity Code, it checks to see whether it has a secure relationship with the requesting DEV. If there is a secure relationship, the key originator sends the Request Key Response command to the requesting DEV using the management key for that secure relationship. ~~Figure 153 illustrates the message flows for the key request protocol between a DEV and the key originatory.~~

Delete Figure 153.

10. Security specifications

10.2 Symmetric cryptography building blocks

10.2.4 Nonce value

Change the first paragraph in 10.2.4 as shown:

The nonce used for CCM encryption and authentication shall be a 13-octet field, dependent on the frame in which it is used, consisting of the 8-bit SrcID followed by the 8-bit DestID followed by the current 6-octet time token followed by the 2-octet secure frame counter followed by the 3-octet Fragmentation Control field from the MAC header. In order to preserve the security of the symmetric algorithms, this nonce shall be unique. As a result, the DEV shall not reuse any 2-octet sequence number within a single superframe that is ~~intented~~ intended for a particular DEVID (as this would cause a repeated nonce). This uniqueness is guaranteed by the use of the SrcID, which guarantees that different DEVs sharing the same key will use a different nonce, by the time token, which is different for every superframe with a given key and by the DestID and secure frame counter, which guarantee uniqueness within a superframe as long as a DEV does not send more than 65536 frames to a particular DestID within that superframe. If a frame is retransmitted and a single bit in the header or frame body has changed, a new nonce shall be used. To ensure this, each time a frame is retransmitted the secure frame counter shall be incremented.

11. PHY specification for the 2.4 GHz band

11.2 General requirements

11.2.7 PHY layer timing

Change Table 71 as shown:

Table 71—PHY layer timing parameters

PHY parameter	Value	Subclause
pPHYMIFSTime	2 μs	11.2.7.4
pPHYSIFSTime	10 μs ± 0.5 μs	11.2.7.2
pCCADetectTime	5* × 16/11 μs	11.6.5
pPHYChannelSwitchTime	500 μs	11.2.7.5

11.2.7.4 Time between successive transmissions

Change the first paragraph in 10.2.7.4 as shown:

The minimum time between successive transmissions shall be pPHYMIFSTime, including the power-up ramp specified in 11.5.7.

11.7 PHY management

Delete the fourth paragraph, which states "The encoding of the DataRate parameter …." Delete Table 91. Insert the following paragraphs and figure in place of the fourth paragraph in 11.7:

The Fragmentation Control field, 7.2.4, in an Imm-ACK or Dly-ACK frame may be used by the DEV to report the status of the frame that is being ACKed. In the case of a Dly-ACK frame, the status is valid only for the frame that prompted the Dly-ACK frame and not for other frames indicated in the Dly-ACK frame, 7.3.2.2. DEVs are neither required to send receive status information in Imm-ACK and Dly-ACK frames nor required to decode the field in Imm-ACK or Dly-ACK frames. If the Fragmentation Control field is used to send receive status, for the 2.4 GHz PHY it shall be formatted as illustrated in Figure 185a.

bits: b23–b8	b7–b3	b2–b0
Reserved	LQI	RSSI

Figure 185a—Format of fragmentation control field when used for receive status information

The RSSI field and LQI field are set to the values determined for the frame that is being ACKed. The values for RSSI and LQI are defined in 11.6.6 and 11.6.7, respectively.

Change Table 92 as shown:

Table 92—PHY PIB implementation group parameters

Managed object	Octets	Definition	Access
PHYPIB_Type	1	0x00=2.4 GHz	~~Read only~~ Read/write
PHYPIB_RegDomainsSupported	Variable	One octet for each regulatory domain supported, as defined for PHYPIB_CurrentRegDomain	~~Read only~~ Read/write
PHYPIB_CurrentRegDomain	1	0x00 = European Telecommunications Standards Institute (ETSI) 0x01 = Federal Communications Commission (FCC) 0x02 = Industry Canada (IC) 0x03 = Association of Radio Industries and Businesses (ARIB) 0x04–0xFF = Reserved	~~Read only~~ Read/write
PHYPIB_DataRateVector	1	Encodes the data rates, defined in Table 89 and 11.7.	~~Read only~~ Read/write
PHYPIB_NumChannelsSupported	1	Value = 0x05, see 11.2.3.	~~Read only~~ Read/write
PHYPIB_CurrentChannel	1	Indicates the channel that is currently being used, see 11.2.3.	~~Read only~~ Read/write
PHYPIB_CCAThreshold	1	The CCA threshold in dBm, encoded in ~~2's~~ twos complement format. The value is implementation-dependent but no larger than the value listed in 11.6.5.	~~Read only~~ Read/write
~~PHYPIB_FrameLengthMax~~ PHYPIB_MaxFrameLength	2	pMaxFrameBodySize, 11.2.8.1.	~~Read only~~ Read/write
PHYPIB_PreferredFragmentSize	1	The preferred fragment size of a DEV. The encoding is defined in Table 90.	Read/write

Change Table 93 as shown:

Table 93—PHY PIB implementation group parameters

Managed object	Octets	Definition	Access
PHYPIB_DiversitySupported	1	Numeric entry that indicates the number of antennas that are available.	~~Read only~~ Read/write
PHYPIB_MaxTXPower	1	The maximum TX power that the DEV is capable of using, 7.4.11, implementation-dependent.	~~Read only~~ Read/write
PHYPIB_TXPowerStepSize	1	The step size for power control supported by the DEV, 7.4.12, implementation-dependent.	~~Read only~~ Read/write
PHYPIB_NumPMLevels	1	Number of power management levels supported. The range is 1 to ~~255~~ 8 and the value is implementation-dependent.	~~Read only~~ Read/write
PHYPIB_PMLevelReturn	Variable	Table of vectors with number of entries given by PHYPIB_NumPMLevels. Each vector is the time required to change between power saving states of the PHY. Vector number 0 is the time required to change the PHY from the off state to a state where it is ready to receive commands. Other values are implementation-dependent.	~~Read only~~ Read/write

Annex A

(normative)

Frame convergence sublayer

Change the title of A.1 as indicated below:

A.1 Generic convergence sublayer

Change the third paragraph in A.1 as shown and delete Figure A.2:

~~Figure A.2 illustrates the~~ The relationship between an FCSL and the rest of the 802.15.3 ~~protrocol~~ protocol entities is illustrated in Figure 3 (in 6.1).

Insert the following paragraph after the third paragraph in A.1:

Because IEEE Std 802.15.3 supports more than one FCSL, a mechanism is required to allow the MAC to deliver each received data frame to the correct FCSL. For instance, a 1394 FCSL would not correctly process a frame from an 802.2 FCSL. An LLC/SNAP header in the MSDU is used to determine the correct FCSL for delivery. Once communication has been established between FCSLs, streams are identified by the stream index and so the LLC/SNAP header is not necessary. However, all asynchronous traffic should use the LLC/SNAP data frame format to avoid problems when multiple FCSLs are present in the same piconet. If the MAC fragments the MSDU when it is sent over the air, there is only one LLC/SNAP header in the fragments because there is only one header prepended to an MSDU.

Change subclause numbering from A.1.1 to A.1a as shown:

A.1a ~~A.1.1~~ FCSL PDU classification

Change the second and third paragraphs in A.1.1 (now A.1a) as shown:

The classification process uses one or more classification parameter sets to analyze each frame entering the FCSL. In the case of an 802.2 classifier, the classification set includes ~~a classification priority,~~ the stream index~~,~~ and protocol specific parameters such as destination MAC address, source MAC address, and optionally a UserPriority ~~priority~~ parameter. If an FCSL PDU, received from an upper layer protocol, matches the specified protocol specific parameters, it is then sent to the MAC-SAP for delivery using the stream indicated by the stream index. If the FCSL PDU does not match the specified protocol parameters, either the frame may be delivered using ~~either~~ a default stream index (i.e., asynchronous stream index) or the frame may be discarded. The policy for deciding the method used to handle a frame in this instance is outside of the scope of this standard. Figure A.3 provides a graphical representation of the entities involved.

In the case where more than one classification parameters set is available, the classification process first shall use the classification parameters set containing the highest valued UserPriority ~~classification priority~~ parameter. If no match is found with the first classification parameters set, the next highest UserPriority ~~priority classification~~ parameters set will be applied. This process will repeat itself until either the incoming frame is properly matched and assigned to a specific stream index for subsequent delivery, or there are no more classification parameters sets available and the incoming frame is either discarded or delivered with a default delivery QoS (i.e., Best Effort).

Change subclause numbering on A.1.2 to be A.1b as shown:

A.1b ~~A.1.2~~ IEEE 802.2 FCSL

Change the third attribute of the second item in the dashed list in A.1.2 (now A.1b) as shown:

3) User~~~~Priority, an 802.1D™ hierarchical QoS scheme

Change the fourth item in the dashed list in A.1.2 (now A.1b) as shown:

— Map each received PDU source and destination address to a corresponding 802.15.3 SrcID and DestID~~, by communicating in an unspecified manner with the DME which maintains this information~~. Deliver each valid frame SDU to the MAC-SAP.

Change subclause numbering from A.1.2.1 to A.1b.1 as shown:

A.1b.1 ~~A.1.2.1~~ IEEE 802.2 FCSL QoS support

Change the lettered item in A.1.2.1 (now A.1b.1) as shown:

b) Hierarchical IEEE 802.1D QoS UserPriority ~~priority~~ scheme.

Change the second paragraph in A.1.2.1 (now A.1b.1) as shown:

The 802.1D UserPriority ~~priority~~ scheme, which describes up to eight (0–7) ~~differenct~~ different QoS levels, may be included in the rule set for the 802.2 FCSL classifier. Each of the 8 different QoS priorities is ~~are~~ described in Table A.1.

Change the header row of Table A.1 as shown. The other rows are unchanged and are not shown.

Table A.1—IEEE 802.1p traffic types

~~User priority~~ UserPriority	Traffic type	Used for ~~for:~~	Comments

Change subclause numbering from A.1.2.2 to A.1b.2 as shown:

A.1b.2 ~~A.1.2.2~~ Data entity inter-relationships

Replace Figure A.4 with the following:

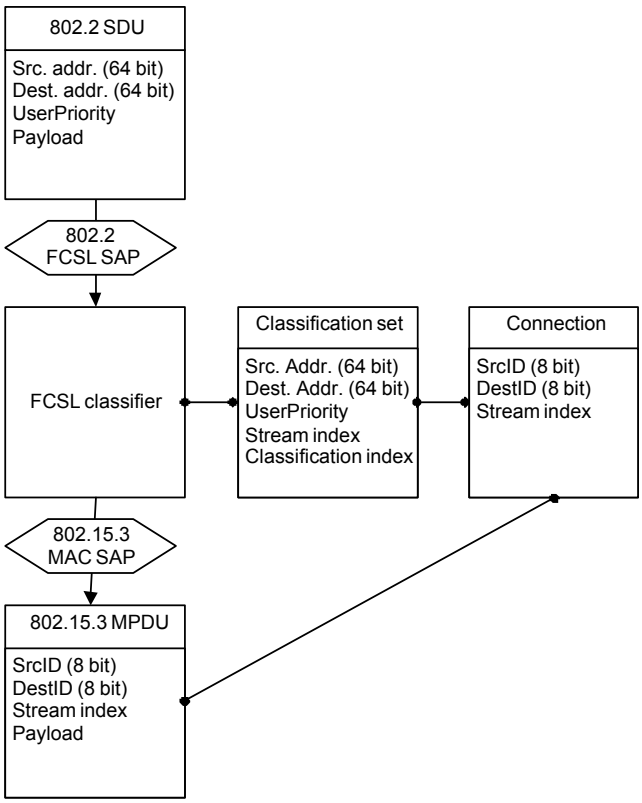

Figure A.4—Data entity inter-relationship model

A.2 802.2 FCSL SAP

A.2.1 MA-UNITDATA.request

Change the primitive definition in A.2.1 as shown:

MA-UNITDATA.request	(SourceAddress, DestinationAddress, RoutingInformation, Data, UserPriority, ServiceClass)

Change the sixth paragraph in A.2.1 as shown:

The UserPriority parameter specifies a prioritized QoS request for the MSDU transfer. IEEE 802.15.3 allows 8 integer values between 0 and 7 inclusive for directly indicating prioritized QoS.

A.2.3 MA-UNITDATA-STATUS.indication

Change the primitive definition in A.2.3 as shown:

```
MA-UNITDATA-STATUS.indication   (
                                SourceAddress,
                                DestinationAddress,
                                TransmissionStatus,
                                ProvidedUserPriority,
                                ProvidedServiceClass
                                )
```

Change the enumerated item in the fourth paragraph in A.2.3 as shown:

3–Unsupported UserPriority ~~priority~~ (for UserPriority values ~~priorities~~ other than an integer value between 0 and 7 inclusive)

Change the fifth paragraph in A.2.3 as shown:

The ProvidedUserPriority parameter specifies the user priority ~~priority~~ that was used for the associated data unit transfer, as defined in A.2.1.

Annex B

(informative)

Security considerations

B.1 Background assumptions

B.1.4 Security key lifecycle issues

B.1.4.1 Key lifecycle

Change B.1.4.1 as shown:

In order to maintain security, care needs to be taken to protect keys from exposure for their entire lifetime. This standard provides ~~the DEV-host~~ the necessary methods for good key life cycle management. The requirements for key life cycle management depend on the type of application.

After B.2.4, insert the following subclauses as B.3 through B.3.2:

B.3 Properties of the 802.15.3 security suite

B.3.1 Key usage

In general, a 128-bit AES key should not be used more than 2^{64} times to produce an IC or to encrypt a frame. If a DEV sends a frame encrypted by a key once every microsecond, it would send approximately 2^{45} frames every year. Thus, to avoid security problems, an implementation should change its management keys at least once every 2^{19} = 524 288 years. More conservative implementations that are concerned with security should change management keys at least once every millennium. Of course, after that many years computation power will have increased dramatically and 128-bit AES keys likely will no longer be considered to be secure.

B.3.2 Replay prevention

This standard uses a Time Token, 7.3.1.1, and Secure Frame Counter (SFC), 7.2.7.3, to provide a method to detect and defeat potential replay attacks. The SFC allows up to 65 535 frames to be sent in a single superframe or one every microsecond for the largest possible superframe. The Time Token is 6 octets and so it will repeat only once every $2^{48}/2^{35} = 2^{13}$ years ~ 8192 years if the PNC uses a 1 ms superframe duration.

Because the nonce includes the Time Token, a replay of one of Distribute Key Request, Distribute Key Response, Request Key or Request Key Response commands would fail for anything other than the current superframe. A replay of one of these commands would not fail IC check if either

— The piconet restarts with a lower time token and so eventually the same time token will be used; or

— The time token rolls over in the current piconet (once every 8192 years for a 1 ms superframe duration) and the same SECID is being used by that DEV (which may be true for the management key in shared key operation).

In the case where the command is replayed in the same superframe, the duplicate detection algorithm will discard the second occurrence sent by the attacker.

In the case of a piconet starting with a lower time token, the duplicate detection will fail and the IC will pass in the case of shared keys if the same management key and SECID are used. If higher layer mutual authentication is used, then the management keys and their SECIDs will change each time the piconet is restarted and the DEVs reauthenticate.

Suppose that the attacker waits until the piconet starts again and the desired time token occurs and replays one of the key distribution commands and it passes IC check and duplicate detection.

If the Distribute Key Request command is replayed the DEV will update its current key to the distributed (old) key. It will then use this data key for communications with the attacking DEV (provided it is using the same DEVID). However, the attacking DEV will not be able to provide correct ICs for any new frames or read any encrypted data because it still does not have either of the two keys (data or management). If it previously had access to the data (i.e., it was part of the piconet as a trusted DEV), it would be able to communicate with the DEV using this key until the DEV found out that the key was "faked." If the key in the replayed Distribute Key Request command has a different SECID than the current one in the piconet, which is likely, the DEV will find out when the next beacon is received. If somehow the current PNC is using the same SECID as the replayed frame, the DEV will get IC errors on the beacon and will send a Request Key command to the PNC to get a new key.

If the Distribute Key Response command is replayed, the receiving DEV (the KO, but not the PNC, because the PNC does not use this command) will think that the remote DEV has had its key updated with the indicated SECID. However, if the KO does not have its Distribute Key state machine operating with a pending key distribution, the KO should drop this indication. If the KO is currently using a different SECID for the relationship, it should probably interpret this as some sort of a failure and send a Distribute Key Request command to the DEV with the current key. If the KO has a pending Distribute Key operation and receives this command with a different SECID, it would interpret it as a failure. Finally, if the KO is distributing a key to that DEV with the same SECID, it will finish the key distribution process. It is possible that the receiving DEV will not have yet received the key and so it will then receive frames with a new SECID. This will cause that DEV to issue a Key Request command to resync with the KO. This case does not represent a security breach.

If the Request Key command is replayed the KO will see this as a valid request for a key and sends the encrypted key to the attacking DEV. The attacking DEV, lacking the management key, will be unable to decrypt the resulting Request Key Response command. Thus it will not be able to get the data key. Furthermore, the attacker will not be able to repeat the request because the Time Token will have incremented in the next superframe and the replayed command will fail integrity check.

If the Request Key Response command is replayed it will have the same effect as the Distribute Key Request command with one exception. If the receiving DEV did not send a Request Key command, it should ignore this command and register it as a security error.

If, for some reason, the DEV accepts the new (old) key, it would begin to use it for data encryption, which will prompt the KO to register a security error and update the data keys.

The attacker, by these methods alone, has not gained access to the shared secret, but he is able to convince the DEV to take some predictable action and to use the same key, possibly with the same nonce (this is extremely unlikely due to all of the items in the nonce that change on a frame by frame basis).

Annex D

(normative)

Protocol implementation conformance statement (PICS) proforma

D.7 PICS proforma—IEEE Std. 802.15.3-2003

D.7.1 Major roles for IEEE 802.15.3 DEVs

Insert the following row at the end of Table D.1. The other rows are unchanged and are not shown.

Table D.1—Functional DEV types

Item Number	Item Description	Reference	Status	Support		
				N/A	Yes	No
FD3	Supports 2.4 GHz PHY	11	O			

D.7.2 PHY functions

Change Table D.2 as shown.

Table D.2—PHY functions

Item Number	Item Description	Reference	Status	Support		
				N/A	Yes	No
PLF1	Conforms to general requirements (i.e., timing, frequency ~~freqency~~, etc.)	11.2	FD3: M			
PLF1.1	Able to detect ~~dectect~~ 802.11b networks	11.2.4	FD3: O			
PLF2	Supports 22 Mb/s DQPSK modulation	11.3.1, 11.3.2, 11.3.3	FD3: M			
PLF2.1	Supports 33, 22, and 11 Mb/s modulations	11.3.2, 11.3.4	FD3: O			
PLF2.2	Supports 44, 33, 22, and 11 Mb/s modulations	11.3.2, 11.3.4	FD3: O			

Table D.2—PHY functions *(continued)*

Item Number	Item Description	Reference	Status	Support		
				N/A	Yes	No
PLF2.3	Supports 55, 44, 33, 22, and 11 Mb/s modulations	11.3.2, 11.3.4	FD3: O			
PLF3	Encodes and decodes PHY frame format	11.4	FD3: M			
PLF4	Conforms to transmitter requirements ~~requirments~~	11.5	FD3: M			
PLF5	Conforms to receiver requirements	11.6	FD3: M			
PLF5.1	Supports link quality assessment	11.6.7	PLF2.1: M, PLF2.2: M, PLF2.3: M,			
PLF6	PHY PIB values supported	11.7	FD3: M			
PLF7	CAP mandatory	11.2.10	FD3: M			
PLF8	Provide receive status in Imm-ACK or Dly-ACK frames	11.7	FD3: O			

D.7.3 Major capabilities for the MAC sublayer

D.7.3.1 MAC frames

Change and insert the following rows in Table D.3 as shown. The other rows are unchanged and are not shown.

Table D.3—MAC frames

Item Number	Item Description	Reference	Transmitter		Receiver	
			Status	Support N/A Yes No	Status	Support N/A Yes No
MF2.8	Non-secure data	7.3.4.1	O ~~M~~		O ~~M~~	
MF2.9	Secure data	7.3.4.2	O ~~S2: M~~		O ~~S2: M~~	
MF2.10	Non-secure LLC/SNAP data frame	7.3.5.1	O		O	
MF2.11	Secure LLC/SNAP data frame	7.3.5.2	O S2: M		O S2: M	
MF3.15	Overlapping PNID	7.4.15	~~S2:~~ M		~~S2:~~ FD2: M	

Table D.3—MAC frames *(continued)*

Item Number	Item Description	Reference	Transmitter		Receiver	
			Status	Support N/A Yes No	Status	Support N/A Yes No
MF3.18	Group ID	7.4.18	O		O	
MF3.19	Stream renew	7.4.19	O		O	
MF3.20	Next PNC	7.4.20	O		O	
MF3.21	Piconet channel status	7.4.21	O		O	
MF4.33	Announce response	7.5.5.3	M		M	
MF4.34	PM mode change response	7.5.8.6	FD2: M		O	
MF4.35	ASIE request	7.5.9.3	O		O	
MF4.36	ASIE response	7.5.9.4	O		O	
MF4.37	Multicast configuration request	7.5.10.1	O		O	
MF4.38	Multicast configuration response	7.5.10.2	O		O	

D.7.3.2 MAC sublayer functions

Insert the following rows in Table D.4 at the location indicated by the item number, e.g., insert MLF16.6 after MLF16.5 and insert MLF23 after MLF22.3. The other rows are unchanged and are not shown.

Table D.4—MAC sublayer functions

Item Number	Item Description	Reference	Status	Support		
				N/A	Yes	No
MLF16.6	Imp-ACK	8.8.3a	O			
MLF23	**ASIE**					
MLF23.1	Capable of requesting ASIEs	8.14	O			
MLF23.2	Capable of putting ASIEs in a beacon	8.14	FD2: O			
MLF24	**Relinquish CTA**					
MLF24.1	Capable of relinquishing CTA time	8.4.3.8	O			
MLF24.2	Capable of using CTA time released by other DEV	8.4.3.8	O			

Table D.4—MAC sublayer functions *(continued)*

Item Number	Item Description	Reference	Status	Support		
				N/A	Yes	No
MLF25	**Multicast configuration**					
MLF25.1	Request to join or leave a multicast group	8.5.3	FD1: O			
MLF25.2	Maintain list of multicast groups	8.5.3	FD2: O			
MF26	**Handover extensions**					
MF26.1	Preliminary handover capable	8.2.3a	FD2: O			
MF26.2	Next PNC capable	8.2.3b	FD2: O			
MF26.3	Report status with Piconet Channel Status IE	7.4.21	O			

After Annex D, insert the following new informative annex as Annex D1:

Annex D1

(informative)

Implementation considerations

D1.1 Channel time requests

D1.1.1 Types of CTAs

Various types of CTAs are defined in the standard depending on the type of access method, destination and the ability to be changed by the PNC. The types of CTAs used in the standard are listed in Table D1.1.

Table D1.1—Types of CTAs in the standard

CTA type	SrcID	DestID	Stream index	Access method(s)
CAP	N/A	N/A	N/A	Uses CSMA/CA, not a real CTA, but it is assigned time in the superframe.
Regular CTA	Any valid single DEVID[a]	Any valid single DEVID	A regular stream index	TDMA with transmit control transfer.
Regular MCTA	Any valid single DEVID	Any valid DEVID[b]	MCTA stream index	TDMA with transmit control transfer. This is the same functionality as a regular CTA.
Association CTA	UnassocID	PNCID	Asynchronous stream index	CSMA/CA
Association MCTA	UnassocID	PNCID	MCTA stream index	Slotted aloha
Private CTA	Any valid single DEVID	Any valid single DEVID that is the same as the SrcID	A regular stream index	Not defined by PNC, handled by DEV that has control of the CTA.
Open CTA	BcstID	Any valid DEVID	Asynchronous stream index	CSMA/CA
Open MCTA	BcstID	Any valid DEVID	MCTA stream index	Slotted aloha

[a]A single DEVID is a DEVID that corresponds to a single physical DEV.
[b]Any nonreserved DEVID, as defined in 7.2.5. This includes DEVIDs that refer to multiple DEVs, e.g., GrpIDs, McstID and BcstID

In addition, CTAs are either dynamic or pseudo-static, as described in 8.4.3.1. All CTAs, with the exception of regular CTAs and private CTAs, are dynamic CTAs. All private CTAs are pseudo-static CTAs. Regular

CTAs can be either dynamic or pseudo-static, depending on the CTA Type field in the Channel Time Request command when the CTA was originally requested.

A DEV is able to request the creation or a change in a CTA using the Channel Time Request command, as described in 7.5.6.1. The PNC interprets the command based on the target ID and stream index. The various interpretations of these requests are listed in Table D1.2.

Table D1.2—Interpretation of parameters in a Channel Time Request command

DestID	Stream index	CTA type	Description
Any DEVID	Unassigned	Regular CTA	New CTA
Any DEVID	Assigned stream index	Regular CTA	Modify or terminate existing CTA
Any DEVID	Asynchronous stream index	Regular CTA	Reserve or terminate asynchronous CTA
Same as SrcID	Unassigned	Private CTA	New private CTA
Same as SrcID	Assigned stream index	Private CTA	Modify or terminate existing private CTA
DEVID different than SrcID	Stream index previously assigned to private CTA	Private CTA	Handover control of the private CTA to the DEV indicated in the DestID field
UnassocID, Reserved DEVID	Any	N/A	Not allowed in a request, only the PNC assigns association CTAs
BcstID	MCTA stream index	Open MCTA	Modify request for an open MCTA, PNC takes this as a suggestion
BcstID	Asynchronous stream index	Open CTA	Modify request for open CTAs, PNC takes this as a suggestion
PNCID	MCTA stream index	Regular MCTA	Modify request for DEV to PNC CTAs, PNC takes this as a suggestion

D1.1.2 Interpretation of channel time requests

The channel time request is based on the CTRq TU, as defined in 7.5.6.1, which indicates the smallest unit of time that the DEV needs in the allocation. The CTRq TU is specified because it allows the PNC to allocate time in useful amounts and also to split up an allocation so that the MaxTransmitDelay requirements, as defined in 6.3.13, for other allocations can be met. Consider the following example:

— CTRq 1: Channel time is required one-half of the superframe duration, MaxTransmitDelay required is twice the superframe duration.

— CTRq 2: Channel time required is one-tenth of the superframe duration, MaxTransmitDelay required is one quarter of the superframe duration.

To meet the needs of both requests, the PNC will have to split CTRq 1 into multiple allocations so that CTRq 2 will have at least four separate allocations spread throughout the superframe with the required MaxTransmitDelay.

The PNC interprets the channel time request based on the CTRq TU, Minimum Number of TUs, Desired Number of TUs, CTA Rate Factor and the CTA Rate Type fields. The CTRq TU is used both to change the other numbers into time and to specify the smallest usable time for the request. Examples of how the PNC interprets these CTRq parameters are shown in Table D1.3.

Table D1.3—Possible allocations based on CTRq parameters

Minimum number of TUs	Desired number of TUs	CTA rate factor	CTA rate type	Allocation by the PNC
11	14	4	0 (super-rate)	11 to 14 CTRq TUs per superframe in every superframe, in at least 4 allocations spread out evenly. One possible allocation is 4 allocations, two with 4 CTRq TUs and two with 3 CTRq TUs.
11	14	4	1 (sub-rate)	11 to 14 CTRq TUs in a superframe, but the allocation occurs every 4th superframe and does not occur in the three intervening superframes. This is an average allocation of 2.75 to 3.5 CTRq TUs per superframe.
5	5	0	0 (super-rate)	Not allowed; the CTA Rate Factor and CTA Rate Type fields cannot simultaneously be zero.
10	10	1	0 (super-rate)	10 CTRq TUs per superframe in every superframe, possibly in one allocation. However, the PNC is free to split this allocation into multiple CTAs to support MaxTransmitDelay requirements of other CTRqs.

If a DEV sets the Desired Number of TUs field equal to the Minimum Number of TUs field, then it is indicating that its minimum required bit rate and desired bit rate are the same value. The DEV is also allowed to set the Desired Number of TUs field to a value greater than the Minimum Number TUs field to indicate a desired bit rate greater than the minimum required bit rate. Asynchronous traffic is also able to use the isochronous method for channel time allocation by setting the Minimum Number of TUs to zero and the Desired Number of TUs such that it specifies the entire superframe. In this last case the PNC will understand that the DEV needs as much time as possible, but not at the expense of other time-critical streams.

The PNC will indicate the number of CTRq TUs allocated per superframe with the Available Number of TUs field in the Channel Time Response command. If the PNC is unable to grant the request due to unavailable channel time, the PNC uses the Available Number of TUs field in the Channel Time Response command to indicate the number of CTRq TUs that it would have been able to grant.

D1.1.3 Determining CTA Rate Factor from stream requirements

In order to provide timely delivery of data, applications need periodic communication opportunities in such a way that the time between these opportunities is bounded. In some cases, a single channel time allocation in each superframe is sufficient to provide this level of service. However, some applications have latency requirements that require more than one channel time allocation in each superframe. For example, if an application needs to keep its latency below 10 ms and the superframe duration is 65 ms, then more than one channel time allocation per superframe, a super-rate allocation, is required. This subclause discusses a method that determines the correct CTA Rate Factor to request for a channel time allocation based on an upper bound on the maximum amount of time required between transmit opportunities.

The steps involved are:

a) Determine the maximum time allowed between transmit opportunities (MaxTransmitDelay).

b) Determine the channel time required per superframe to support both the minimum bit rate (TimeRequiredPerSuperframe) and the desired bit rate (TimeDesiredPerSuperframe).

c) Calculate AllocationCriteria using:
AllocationCriteria = 2 × (SuperframeDuration – TimeRequiredPerSuperframe).

d) If the AllocationCriteria is greater than the MaxTransmitDelay, then a super-rate allocation is required (proceed to step e). Otherwise, a sub-rate allocation will suffice (skip to step f).

e) Make the following super-rate calculations:
   1) CTA Rate Factor = (SuperframeDuration – TimeRequiredPerSuperframe)/MaxTransmitDelay.
   2) Round this value up to the next highest integer.
   3) MinTimeToRequest = TimeRequiredPerSuperframe.
   4) DesiredTimeToRequest = TimeDesiredPerSuperframe.
   5) Skip to step g.

f) Make the following sub-rate calculations:
   1) CTA Rate Factor = MaxTransmitDelay/(SuperframeDuration – TimeRequiredPerSuperframe).
   2) Round this value down to the next power of 2 (because sub-rates are required to be powers of 2).
   3) MinTimeToRequest = (CTA Rate Factor × TimeRequestedPerSuperframe).
   4) DesiredTimeToRequest = (CTA Rate Factor × TimeDesiredPerSuperframe).
   5) If the amount of channel time requested is a significant fraction of the superframe, the request could be denied by the PNC and so a lower CTA Rate Factor may be selected by the DEV. In this case, different MinTimeToRequest and DesiredTimeToRequest values would be calculated based on the CTA Rate Factor that was selected. If the amount of channel time is a still a significant fraction of the superframe even with a sub-rate CTA Rate Factor of 2, the DEV may request a super-rate allocation and use the calculations of step e.

g) Use the CTRq TU to convert MinTimeToRequest and DesiredTimeToRequest into Minimum Number of TUs and Desired Number of TUs for the Channel Time Request command.

D1.1.3.1 Example 1

The following is an example assuming a SuperframeDuration of 10 ms:

a) The stream requires a MaxTransmitDelay of less than 2 ms.

b) The stream requires 2 ms of channel time per superframe (TimeRequiredPerSuperframe = 2 ms) and desires 3 ms of channel time per superframe (TimeDesiredPerSuperframe = 3 ms).

c) AllocationCriteria = 2 × (10 ms – 2 ms) = 16 ms.

d) AllocationCriteria is greater than MaxTransmitDelay, so a super-rate allocation is required (proceed to step e).

e) Make the following super-rate calculations:
   1) CTA Rate Factor = (10 ms – 2 ms)/(2 ms) = 8/2 = 4.
   2) No rounding required.
   3) MinTimeToRequest = TimeRequiredPerSuperframe = 2 ms.
   4) DesiredTimeToRequest = TimeDesiredPerSuperframe = 3 ms.

f) (Skipped)

g) If the CTRq TU is 1 ms, then:
Minimum Number of TUs = MinTimeToRequest / 1 ms = 2,
Desired Number of TUs = DesiredTimeToRequest / 1 ms = 3.

The parameters for a Channel Time Request command for this example are:

— CTA Rate Type = 0 (super-rate)
— CTA Rate Factor = 4

— CTRq TU = 1 ms
— Minimum Number of TUs = 2
— Desired Number of TUs = 3

D1.1.3.2 Example 2

The following in another example assuming a SuperframeDuration of 10 ms:

a) The stream requires a MaxTransmitDelay of less than 50 ms.
b) The stream requires 1 ms of channel time per superframe (TimeRequiredPerSuperframe = 1 ms) and desires 1.25 ms of channel time per superframe (TimeDesiredPerSuperframe = 1.25 ms).
c) AllocationCriteria = 2 × (10 ms – 1ms) = 18 ms.
d) AllocationCriteria is less than MaxTransmitDelay, so a sub-rate allocation will suffice (proceed to step f).
e) (Skipped)
f) Make the following sub-rate calculations:
    1) CTA Rate Factor = 50 ms / (10 ms – 1ms) = 51/9 = 5.6.
    2) 5.6 is rounded down to 4 (next lower power of 2).
    3) MinTimeToRequest = (CTA Rate Factor × TimeRequiredPerSuperframe) = 4 ms.
    4) DesiredTimeToRequest = (CTA Rate Factor × TimeDesiredPerSuperframe) = 5 ms.
    5) At this point, if MinTimeToRequest and DesiredTimeToRequest are considered a significant portion of the superframe, then a lower sub-rate factor could be selected and the MinTimeToRequest and DesiredTimeToRequest recalculated (not done in this example).
g) If the CTRq TU is 1 ms, then:
Minimum Number of TUs = MinTimeToRequest / 1 ms = 4,
Desired Number of TUs = DesiredTimeToRequest / 1 ms = 5.

The parameters for a Channel Time Request command for this example are:

— CTA Rate Type = 1 (sub-rate)
— CTA Rate Factor = 4
— CTRq TU = 1 ms
— Minimum Number of TUs = 4
— Desired Number of TUs = 5

D1.1.4 PNC interpretation of CTA rate factor

The PNC interprets the CTA Rate Factor field as follows:

a) If the CTA Rate Type = 0 (super-rate), the PNC calculates the MaxTransmitDelay required using:

    MaxTransmitDelay = (SuperframeDuration – TimeRequiredPerSupeframe)/(CTA Rate Factor).

b) If the CTA Rate Type = 1 (sub-rate), the PNC uses the CTA Rate Factor to create a sub-rate allocation. The maximum MaxTransmitDelay that will be provided by the PNC using the provided CTA Rate Factor is calculated using:

    MaxTransmitDelay=(CTA Rate Factor×SuperframeDuration) – TimeRequiredPerSuperframe.

The MaxTransmitDelay calculated by the PNC will not always be equal to the MaxTransmitDelay desired by the DEV. However, the MaxTransmitDelay will always be less than or equal the MaxTransmitDelay desired by the DEV.

Regularly occurring opportunities to transmit (MaxTransmitDelay) does not guarantee that a specific latency will be achieved. Due to the nature of wireless communications there is no guarantee that all transmission attempts will be successful. Therefore, specifying a maximum of 10 ms between transmit opportunities does not guarantee that a latency of 10 ms will necessarily be achieved.

D1.1.5 Creating channel time requests from MLME requests

This subclause describes one possible method of converting MLME-CREATE-STREAM.request and MLME-MODIFY-STREAM.request parameters into a Channel Time Request command. However, the implementer is free to choose another method.

As an example, consider the following requirements for a stream:

- A SourceDataRate of 8 Mb/s throughput.
- A DesiredDataRate of 10 Mb/s throughput.
- The MaxTransmitDelay is 5 ms.
- High priority stream.
- MaxRetries is four.
- ReliabilityExponent is 4 (i.e., the FER needs to be less than 10^{-4}).
- AggregateDataFrameSize of 1000 octets.

The MAC of the source DEV has the current parameters of the piconet as well as an estimate of the channel quality between it and the destination DEV. Assume the following parameters for the channel and piconet:

- The FER is less than 5%.
- Data rates available on the link between the two DEVs are 22, 33, 44.
- Superframe duration = 10 ms.
- ACK policy is Imm-ACK.
- Overhead is approximately 30 μs per frame, an ACK duration is 30 μs and the SIFS is 10 μs.

Based on the preceding information, the source MAC will perform the following calculations:

- Need 8 Mb/s / 8000 bits/frame = 1000 frames/second.
- Desire 10 Mb/s / 8000 bits/frame = 1200 frames/second.
- NumberofRetries = ceiling($\log_{10}$(FER)/ReliabilityExponent) = ceiling($\log_{10}$(0.05)/4) = 4.
- With up to four retries of a frame, and additional 5.27% of frames are required, which translates to a minimum 1053 frames/second and a desired 1264 frames/second.
- Time per frame is = 8000 bits/44 Mb/s + 90 μs overhead = 271 μs/frame, so CTRq TU = 271 μs.
- In each superframe, the DEV:

  Needs to send minimum of 1053 frames/second × 10 ms/superframe = 11 frames/superframe.

  Desires to send 1263 frames/second × 10 ms superframe = 13 frames/superframe.
- Time per frame is = 8000 bits/44 Mb/s + 90 μs overhead = 272 μs/frame.
- The TimeRequiredPerSuperframe = 11 frames/superframe × 0.272 ms = 3 ms and the TimeDesiredPerSuperframe = 13 frames/superframe × 0.272 ms = 3.6 ms.
- Using D1.1.3, calculate AllocationCriteria = 2 × (10 ms - 3 ms) = 14 ms.
- The AllocationCriteria is greater than the MaxTransmitDelay (5 ms), so a super-rate is required.
- Using D1.1.3, CTA Rate Factor = (10 ms – 3 ms) / 5 ms = 1.4, round up to 2.
- Using D1.1.3, MinTimeToRequest = TimeRequiredPerSuperframe = 3 ms/superframe.
- Using D1.1.3, DesiredTimeToRequest = TimeDesiredPerSuperframe = 3.6 ms/superframe.

— Using D1.1.3, Minimum Number of TUs to request is 3 ms / 0.272 ms = 11 and the Desired Number of TUs to request is 3.8 ms / 0.272 ms = 13.

The DEV needs to perform two more checks. The first is to check to see if the number of CTRq TUs in a CTA are sufficient to allow timely retries. In order to meet the latency requirement for successfully transmitting a frame, there needs to be time for the retransmissions within a single CTA. Otherwise, the retransmissions will occur too late. The NumberofRetries for this example is 4, thus a minimum of 5 CTRq TUs are required in each CTA (1 for the frame plus 4 for retries). In this case, the minimum allocation will be 5 CTRq TUs in one CTA and 6 in another, which will meet this requirement. If the NumberofRetries had been higher, 6 for example, the DEV needs increase the Minimum Number of TUs to 14 so that there is at least 7 CTRq TUs in each CTA. Alternately, the DEV could request a higher CTA rate factor.

NOTE—Using this method will improve the latency guarantee so that it includes the maximum retries required within one CTA. However, this will also greatly increase the over-allocation of time in the superframe.

The second check is to verify that using the maximum number of retries does not delay frames in the following CTA past the end of that CTA. For this example, the number of frames required per superframe, without retries, is ten. If one frame requires 4 retries, then the request needs to allow for up to 14 CTRq TUs per superframe. Likewise, to support the desired data rate, 12 frames plus 4 retries are required for a total of 16 CTRq TUs per superframe. To meet this, the DEV will need to increase the numbers in the request.

Thus the DEV should use the following parameters in its Channel Time Request command:

— CTA Rate Type = 0 (super-rate)
— CTA Rate Factor = 2
— CTRq TU = 272 μs
— Minimum Number of TUs = 14
— Desired Number of TUs = 16

The PNC upon receiving the above parameters in the DEVs request would calculate the maximum space allowed between allocated CTAs:

— Minimum amount of time requested = 11 × 0.271 ms = 3 ms.
— Using D1.1.3, calculate MaxTransmitDelay = (10 ms – 3 ms) / 2 = 3.5 ms.

Based on this calculation the PNC is able to satisfy the request by providing a minimum 11 CTRq TUs and a maximum 13 CTRq TUs per superframe, allocated as 2 CTAs per superframe, with the maximum space between CTAs of 3.5 ms and a minimum CTA size of 1 CTRq TU (272 μs). It is not possible for the PNC to know that the desired MaxTransmitDelay required was 5 ms. The PNC will indicate the number of CTRq TUs allocated per superframe in the Channel Time Response command. If the request can not be granted by the PNC because the minimum number of TUs per superframe is not available, the PNC will indicate the number of CTRq TUs that are available in the Channel Time Response command.

As a second example, consider the example above modified such that the MaxTransmitDelay requirement was 20 ms instead of 5 ms. Because 20 ms is greater than the calculated AllocationCriteria of 14 ms, a sub-rate allocation would suffice.

— Using D1.1.3, sub-rate CTA Rate Factor = 20 ms / (10 ms - 3 ms) = 20 / 7 = 2.86, round down to 2.
— Using D1.1.3, MinTimeToRequest = (CTA Rate Factor × TimeRequiredPerSuperframe)
= (3 ms × 2) = 6 ms.
— Using D1.1.3, DesiredTimeToRequest = (CTA Rate Factor × TimeDesiredPerSuperframe)
= (3.6 ms × 2) = 7.2 ms.
— Using D1.1.3, the DEV may continue with the sub-rate request with a sub-rate factor of 2, or the DEV may decide to make a super-rate request instead. (For this example, assume that the DEV continues with the subrate request.)

— Using D1.1.3, Minimum Number of TUs to request is 6 ms / 0.272 ms = 22 and the Desired Number of TUs to request is 7.2 ms / 0.272 ms = 26.

Again, for first reliability, the minimum CTA should be no less than 5 CTRq TUs based on the Numberof-Retries. The minimum number of CTRq TUs requested is 22, which meets this requirement. For the second criteria, 20 frames are required every 2 superframes and 24 frames are desired every 2 superframes. Allowing for an additional 4 retries increases the request to use 24 and 28.

Thus the DEV could use the following parameters in its request:

— CTA Rate Type = 1 (sub-rate)
— CTA Rate Factor = 2
— CTRq TU = 272 μs
— Minimum Number of TUs = 24
— Desired Number of TUs = 28

The PNC is able to satisfy the request by providing a minimum 22 CTRq TUs and a maximum 26 CTRq TUs every other superframe, with a minimum CTA size of 1 CTRq TU (272 μs). The PNC does not need to worry about MaxTransmitDelay requirements because the requesting DEV has already determined that the sub-rate factor selected will meet the MaxTransmitDelay requirements. The MaxTransmitDelay provided by the PNC using a sub-rate factor of 2 can be calculated as follows:

— TimeRequiredPerSuperframe = 22 × 0.271 ms = 6 ms.
— Using D1.1.3, MaxTransmitDelay = (2 × 10 ms) – 6 ms = 14 ms.

D1.1.6 Interpreting channel time requests

When the PNC grants channel time in response to a Channel Time Request command, the value of the Available Number of TUs field returned in the Channel Time Response command will be greater than or equal to Minimum Number of TUs from the corresponding request. If the request was a super-rate request, the Available Number of TUs field indicates the number of CTRq TUs allocated per superframe. If the request was a sub-rate request, the Available Number of TUs field indicates the number of CTRq TUs allocated per superframe at the requested sub-rate.

When the PNC denies a Channel Time Request command, the value of the Available Number of TUs returned in the Channel Time Response command will be less than Minimum Number of TUs from the corresponding request. If the request was a super-rate request, the Available Number of TUs field indicates the number of CTRq TUs that the PNC could have allocated per superframe. If the request was a sub-rate request, the Available Number of TUs field indicates the number of CTRq TUs the PNC could have allocated per superframe at the requested sub-rate.

Regardless of the result of the channel time request, the Available Number of TUs field can be converted to an AvailableDataRate and returned in the MLME-CREATE-STREAM.confirm primitive. If the request was granted, the AvailableDataRate is the data rate received. If the request was denied, the AvailableDataRate is the data rate the PNC could have provided.

If a super-rate was requested, the steps involved in determining an AvailableDataRate from Available Number of TUs are:

— Frames Per Superframe = Available Number of TUs
— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe
— Frames Per Second = Total Frame Per Second – FER
— AvailableDataRate = Frames Per Second × Bits Per Frame

If a sub-rate was requested, the steps involved in determining an AvailableDataRate from Available Number of TUs are:

— Frames Per Superframe = Available Number of TUs / CTA Rate Factor

— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe

— Frames Per Second = Total Frame Per Second × (1 – FER)

— AvailableDateRate = Frames Per Second × Bits Per Frame

As an example, consider the stream requirements, channel conditions, and calculated channel time request parameters for the first example shown in D1.1.5.

If the super-rate request is granted and the Available Number of TUs field of the corresponding Channel Time Response command contains the value 12, the AvailableDataRate received can be calculated as follows:

— Frames Per Superframe = Available Number of TUs = 12

— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe
= 12 frames/superframe / 0.010 s/superframe = 1200 frames/s

— Frames Per Second = Total Frame Per Second × (1 - FER) = 1200 × (1 – 5.26%) = 1140 frames/s

— AvailableDataRate = Frames Per Second × Bits Per Frame = 1140 frames/s × 8000 bits/frame = 9.12 Mb/s

If the request is denied and the Available Number of TUs field of the corresponding Channel Time Response command contains the value 8, the same steps can be followed to determine the data rate that the PNC could have provided:

— Frames Per Superframe = Available Number of TUs = 8 frames/superframe

— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe
= 8 frames/superframe / 0.010 s/superframe = 800 frames/s

— Frames Per Second = Total Frame Per Second × (1 – FER) = 800 × (1 – 5.26%) = 760 frames/s

— AvailableDataRate = Frames Per Second × Bits Per Frame = 760 frames/s × 8000 bits/frame = 6.08 Mb/s

As a second example, consider the stream requirements, channel conditions, and calculated channel time request parameters for the second example shown in D1.1.5.

If the sub-rate request is granted and the Available Number of TUs field of the corresponding Channel Time Response command contains the value 24, the AvailableDataRate can be calculated as follows:

— Frames Per Superframe = Available Number of TUs / CTA Rate Factor
= 26 / 2 = 13 frames/superframe

— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe
= 13 frames/superframe / 0.010 s/superframe = 1300 frames/s

— Frames Per Second = Total Frames Per Second × (1 – FER) = 1300 × (1 – 5.26%) = 1235 frames/s

— AvailableDataRate = Frames Per Second × Bits Per Frame = 1237 frames/s × 8000 bits/frame = 9.88 Mb/s

If the request is denied and the Available Number of TUs field of the corresponding Channel Time Response command contains the value 20, the same steps can be followed to determine the AvailableDataRate that the PNC could have provided at the requested sub-rate:

— Frames Per Superframe = Available Number of TUs / CTA Rate Factor
= 20 / 2 = 10 frames/superframe

— Total Frames Per Second = Frames Per Superframe / Seconds Per Superframe
= 10 frames/superframe / 0.010 s/superframe = 1000 frames/s

— Frames Per Second = Total Frame Per Second × (1 – FER) = 1000 × (1 – 5.26%) = 950 frames/s

— AvailableDataRate = Frames Per Second × Bits Per Frame = 950 frames/s × 8000 bits/frame = 7.60 Mb/s

D1.2 Sample frames

The subclause presents sample frames that provide examples of the HCS and FCS calculations as well as the scrambler from the 2.4 GHz PHY. Two data frames are presented with the following characteristics:

— Scrambler seed = 0b01

— Data rate = 55 Mb/s

— Payload length = 20 octets

— Protocol version = 0

— Frame type = data frame

— Security off (SEC = 0)

— ACK policy = Imm-ACK

— Retry = 0

— More data = 1

— PNID = 100

— DestID = 5

— SrcID = 3

— MSDU number = 320 (0x140)

— Fragment number = 3

— Last fragment = 4

— Stream index = 13

Two frame payloads are provided. The first has pseudo-random data in the frame payload. The second frame payload has octets that increase in value by one, i.e., octet 0 has value 0, octet 10 has value 10, etc.

In the figures that follow, the bits are listed lsb on the right, msb on the left with four octets per line. The lowest numbered octets are the first line and higher number octets on subsequent lines, i.e., octets 3, 2, 1 and 0 are on line 1 (octet 3 on the left, octet 0 on the right) while octets 7, 6, 5 and 4 would be on line 2 (octet 7 on the left, octet 4 on the right).

The PHY header, MAC header and HCS are common to both frames and are shown in Figure D1.1.

The frame body and FCS for pseudo-random data are shown in Figure D1.2.

The MAC frame body length is 160 + 32 = 192 bits. Because the frame is encoded with 5 bits per symbol, three stuff bits, 11.4.6, are added to make it an integer number of symbols, bringing the length to 195 bits. After the stuff bits are added, the scrambler is applied, 11.4.4, to the MAC header, HCS and MAC frame body, a total of 291 bits, resulting in the bit stream, shown in Figure D1.3. This is the bit stream will be modulated and sent with the PHY preamble and header over the air.

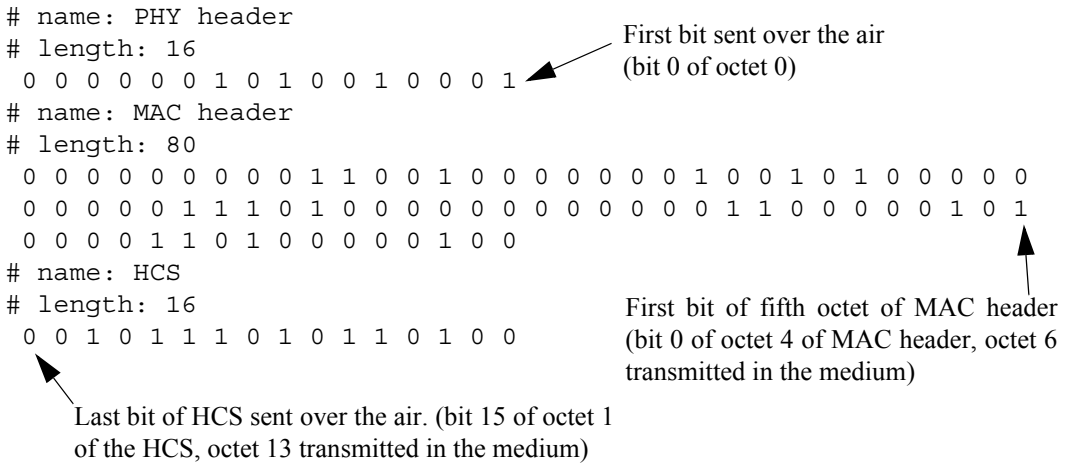

Figure D1.1—PHY header, MAC header and HCS for sample frame

```
# name: Frame payload
# length: 160
 0 0 0 1 1 1 1 0 1 1 0 0 0 0 1 0 0 1 1 1 1 0 0 0 0 1 0 0 1 1 0 1
 0 1 0 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 1 0 1 1 1 0 1 0 0 1
 0 0 1 1 1 0 1 1 0 1 0 1 1 1 0 0 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1 1 0
 1 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 1 0 1 0 1 0 0 1 0 1 0 0 0
# name: FCS
# length: 32
 0 1 0 1 0 1 0 0 1 0 0 0 0 1 1 1 1 1 0 1 1 0 0 0 1 1 1 0 0 0 1 0
```

Figure D1.2—Frame payload and FCS for sample frame with pseudo-random data

```
# name: Scrambled MAC header, HCS, frame payload and FCS
# length: 291
 0 1 1 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 0 1
 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 1 0 1
 0 0 1 0 1 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0
 1 0 0 1 1 1 1 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 0 0 1 0 1
 0 1 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 0 1
 1 0 1 1 0 0 1 1 1 1 0 0 0 0 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 1 1 0 0 1 1 1 1 0
 0 1 1 1 0 0 0 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 1 0 1 1 0 1 0 1 0 0
 0 1 1 0 0 0 1 0 1 0 0 0 0 1 1 1 1 0 0 1 0 0 0 0 1 1 1 0 0 0 1 0
 0 0 0
```

Figure D1.3—MAC header, HCS and MAC frame body for sample frame with pseudo-random data after scrambler has been applied

The frame body and FCS for incremented data are shown in Figure D1.4.

```
# name: Frame payload
# length: 160
 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0
 0 0 0 0 1 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0
 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 0
 0 0 0 1 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0
# name: FCS
# length: 32
 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 0
```

Figure D1.4—Frame payload and FCS for sample frame with incremented data

The scrambled MAC header, HCS, and MAC frame body for the incremented data frame are illustrated in Figure D1.5.

```
# name: Scrambled MAC header, HCS, frame payload and FCS
# length: 291
 0 1 1 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 0 1
 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 1 0 1
 0 0 1 0 1 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0
 1 0 0 0 0 0 1 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 0 0
 0 0 1 0 1 1 1 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0
 1 0 0 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 0 0 0 0 0
 1 0 1 0 0 1 0 1 1 0 1 0 0 0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 0 0
 1 1 1 1 0 0 1 1 0 0 0 1 0 0 1 1 0 1 1 0 1 1 1 0 1 1 1 0 1 1 0 0
 0 0 0 0 1 1 0 1 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 0 1 0 0 1 0 0
 0 0 0
```

Figure D1.5—MAC header, HCS and MAC frame body for sample frame with incremented data after scrambler has been applied

D1.3 64-bit DEV address mapping

IEEE Std 802.15.3 DEV addresses are 64 bits. This allows for efficient use of OUI addressing space, but it causes problems for bridging between 802.15.3 and other IEEE 802 protocols that use 48-bit MAC addresses. It is straightforward to calculate a 64-bit address from a 48-bit address, [B16].

A 48-bit MAC addresses consist of a 24-bit OUI and a 24-bit extension identifier. The process for generating a 64-bit address from a 48-bit address is put the two octets "0xFF 0xFF" between the OUI and the 24-bit extension identifier. Normally 64-bit MAC addresses should not be assigned with these octets set to "0xFF 0xFF" to avoid ambiguity between 64-bit MAC addresses and 48-bit MAC addresses encapsulated in 64-bit addresses. However, because the owner of an OUI is responsible for assigning 48-bit and 64-bit MAC addresses, the owner of the OUI can ensure that it does not assign a 48-bit MAC address to any device with the same 24-bit extension as a 64-bit address.

Annex E

(informative)

Bibliography

Insert the following entry to the bibliography:

[B16] "IEEE Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority," http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.