IEEE P802.15
Wireless Personal Area Networks

Project	IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)	
Title	**WG Editor's instructions**	
Date Submitted	[January, 2013	
Source	[James P. K. Gilb] [Tensorcom] [Carlsbad, CA 92008]	Voice: [760-496-3264 E-mail: [last name at ieee dot org]
Re:	[]	
Abstract	[This document imparts a small portion of the wisdom and experience of the WG Technical Editor to provide some help the poor souls who have been cursed to become TG Technical Editors.]	
Purpose	[The purpose is to reduce the number of editorial and technical comments that the WG Technical Editor feels compelled to submit in an initial letter ballot.]	
Notice	This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.	
Release	The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15.	

1. Guidelines for the draft

This section describes the requirements for creating a good draft standard, from an editorial point of view. There does not seem to be any process or set of rules that will guarantee that the technical content is good.

The first thing to do is to read the most recent version of the **IEEE Standards Style Manual**. Then read it again. Put it under your pillow at night so that some of its wisdom will enter your mind via osmosis. You can find https://development.standards.ieee.org/myproject/Public/mytools/draft/styleman.pdf.[1]

1.1 The rules

The most important rule of all is that there shall be only one Technical Editor for the document. The Technical Editor may choose to appoint sub-editors, but all final decisions are made by the Technical Editor.

The 10 commandments of creating a proper draft are:

1) Thou shall not create new paragraph style. If you think you need a new paragraph style, you are wrong.
2) Thou shall not repeat normative information. Say it correctly in one place and one place only. In all other locations use a cross-reference.
3) Thou shall not import formatted text from another program. All text shall be included as raw, unformatted text. Pasting formatted text will invariably introduce new formats, which is forbidden by 1).
4) Thou shall not use the word “must.” While it is theoretically possible to use it correctly, mere mortals cannot be trusted with such a powerful term.
5) Use only one term per concept, (e.g., “PHY mode” and “Modulation and coding scheme”). Likewise, don’t re-use the same terminology for more than one concept.
6) If an acronym is defined for a term, use it exclusively throughout the document, don’t use the spelled out term other than in the Acronym clause and for its first use.
7) Thou shall not use more than five levels deep of subclause numbering. There is a reason the template only defines five levels. If you think you need more than five levels, you are wrong and need to re-think your document structure.
8) All units shall be tied to the associated number with a non-breaking space (esc-space-h).
9) Resist the urge to use the word “can.” When you think you want to use it, nine times out of ten the correct word is “may.”
10) There is no such thing as “may not” or “should not.” This is syntactically equivalent to “may” or “should” and just makes you look silly. What you probably mean to say is “shall not.”

1.2 Normative vs. informative

(The following is taken from the **IEEE Standards Style Manual**.)

Normative text means information that is required to implement the standard and is therefore officially part of the standard. Informative text is provided for information only and is therefore not officially part of the standard.

Normative text (information required to implement the standard) includes:

— The main clauses of the documents including figures and tables

[1]This location seems to move around a lot. Using the search tool on the IEEE-SA website is useless. Use a reasonable search page and look for IEEE Standards Style manual.

— Footnotes to tables
— Footnotes to figures
— Annexes marked "(normative)"

Informative text (text provided for information only) includes:

— Frontmatter
— Notes to text, tables, and figures
— Footnotes within text
— Annexes marked "(informative)" (e.g., Bibliography)

1.3 Normative terms

There are two types of text in a standard, normative and informative. Normative text describes required or optional behavior. Informative text is used to help the reader understand the standard or its use but does not place any restrictions on the implementation. It is extremely important that normative words are used for normative text and are not used for informative text.

The normative words are:

— shall: Indicates a required behavior. Implementations are compliant if and only if they implement all of the required behaviors. The implication is that there is more than one possible option and the standard selects a subset of these as required.
— may: Indicates an optional behavior. Implementations are not required to implement this behavior, but are allowed to implement the behavior.
— should: Indicates an optional behavior whose implementations is recommended. Implementations are not required to implement the behavior, but the implementation is recommended.
— must: A behavior that follows as a natural consequence.
********* NEVER, EVER USE MUST IN A STANDARD **********
(The reason is that it is almost never used correctly and will most certainly confuse the reader).
— can: A behavior is possible as a natural consequence (could is syntactically equivalent to can).

NOTE: From a standards and compliance point of view, there is no difference between should and may. Should is generally used to placate voters who want a shall but are unable to convince the group.

Examples:

Shall: The PHY <u>shall</u> use a symbol rate of 100 MHz. (more than one symbol rate is physically possible, but only one is allowed by the standard).

Must: Night <u>must</u> follow morning. (There is no option here, condition A leads to condition B in all cases).

May: The MAC <u>may</u> implement sonar to determine the range of devices in the network. (The implementation of sonar is optional, compliant implementations are not required to do it.)

Should: The PHY <u>should</u> implement mode 3. (The PHY is allowed to implement mode 3 and the standard recommends that it be implemented. However, an implementation would still be compliant if mode 3 is not implemented.)

Can: When a large radio hits a user on the head, it <u>can</u> cause serious injury. (The standard is not making this possible, rather the injury is a potential consequence of the radio impacting the user's melon.)

1.4 Additional requirements

Drafts for 802.15 should be done in FrameMaker 7.1 or later. Copies of FrameMaker < 9 are hard to find, so using 9 or later is acceptable. The use of Microsoft Word is **strongly** discouraged for all but the most trivial of drafts. The IEEE provides a template file for Microsoft Word and shall be used if the Task Group Technical Editor decides to use the dark side of the force.

All figures not created in FrameMaker need to be included as separate image files and the original source file included for editing purposes. Acceptable formats for the image files are enhanced metafile (.emf) and encapsulated postscript (.eps). Visio (.vsd) is an acceptable source format.

If color is used in a figure, the figure shall be drawn in such a way that when it is printed in black and white the technical interpretation of the figure shall be the same.

Don't worry too much about the frontmatter, the IEEE editors will throw away your text and replace it with their own.

The ultimate source for understanding the format of an IEEE draft is the most recent version of the **IEEE Standards Style Manual**. You can get the latest version from http://standards.ieee.org.

1.5 Importing text from other programs

FrameMaker will open MS Word files and convert them to FrameMaker files. Unfortunately, it will also import the paragraph and page formats as well as any overrides in the document. Removing the imported formats is difficult and can't be guaranteed to work correctly as there are quite a few valid paragraph formats in the IEEE template.

When you receive a contribution in any format other than an untouched FrameMaker template, save the document as raw text. Open the text file and import all of the formats from the Clause template, removing all overrides. Only at this time can the text be copied and pasted into the draft. If you import directly from another program, it will bring in its character, paragraph and page formats, which is a violation of the commandments, as defined in 1.1.

1.6 Special considerations for ammendments

An ammendment is a special animal in the zoo known as IEEE standards. As a Technical Editor, you shall consider an ammendment to be a list of instructions given to another editor to create a new standard from an existing standard. As such, the ammendment consists of instructions that indicate how the existing standard, called the base document, is to be modified.

A few key points about an ammendment:

— The numbering of clauses, sub-clauses, figures, tables and equations in the base document is unchanged. Unlike the cross references to numbered items in the ammendment, cross references to numbered items in the based document shall be done as fixed numbers rather than as an updating cross reference.

— Because the numbering in the base document is fixed, when new numbered items are to be inserted in the ammended standard, special number needs to be used. For example, if a figure is to be added after Figure 20, the new figure would be numbered Figure 20a (which requires overriding the numbering for the paragraph). Refer to the most recent version of the **IEEE Standards Style Manual** for examples of the correct numbering.

— If you add a new figure, clause, subclause or table after the last numbered one in the document, you can simply begin numbering at that number.
— The title for the ammendment is the same as for the base document up until the end when it says "Ammendment N: <title specific to ammendment>". Refer to a published ammendment (most of the drafts get it wrong).

2. Style guide for 802.15

2.1 Capitalization

— Frame names (e.g., Beacon frame, MAC Command frame, etc.) are capitalized and include the word "frame"[2]
— Command names (e.g., Association Request command) are capitalized and include the word "command"[2]
— Field names are capitalized and include the word field, which is not capitalized, e.g., Destination PAN Identifier field. The only exception is in a figure that shows the field, in which the word "field" is not included.
— IE names are capitalized and include the acronym IE.
— PIB entries are always capitalized at word boundaries, but are preceeded by a lower case prefix, e.g., *mac* or *phy*.
— LME primitive names are ALL CAPS, but the .request, .indication, .response, .confirm are always lower case.
— Primitive parameters are capitalized at word boundaries.
— The 'status' primitive parameter is never capitalized and it shall always be the parameter that is used to return the status of the LME transaction.

2.2 Naming

— PIB entries are preceeded by *mac* or *phy* depending on the type of PIB entry.
— The words in an LME primitive name are separated by a dash.
— Primitive parameters contain no spaces, i.e., the words are joined together

2.3 Miscellaneous

— The name of a PIB parameter is italicized.
— Variables are italicized, even if they are not in an equation. The font face (upright, italics or bold) shall be the same in text as in an equation. For example, "$V_{\mathrm{NewDevice}}$" has an italics *V* because that is the variable, but is modified by Roman text that is not a variable, but rather is a description. Conversely, it would be "V_N" if both *V* and *N* are variables.
— Equations are not numbered unless there is a real need to cross reference the equation.
— Never define an acronym in a table or figure.

3. Process and time line

Congratulations on the dubious honor of being selected as a Technical Editor. You will soon learn why this is a (mostly) thankless job.

[2]This is not the case in IEEE Std 802.15.4-2011, but that will be corrected in the next revision.

3.1 Prior to downselection

Encourage the proposers to develop standards ready text. You can point to existing standards to provide an outline of what is needed in the proposal. This is also a good time to educate your group on the proper use of normative terms.

3.2 Getting ready for first working group letter ballot

Note: This section is still being developed.

3.3 Prior to sponsor ballot

When the draft is getting ready for Sponsor ballot, determine the person who be the IEEE project editor. Make friends with your project editor, they can make your life easy or hard, depending on their disposition towards you. The IEEE 802 liason can point you towards the project editor for your draft.

When the draft is looking like it is ready for Sponsor ballot, send a copy of the draft to the IEEE project editor for a review. Carefully read an understand the comments of your project editor and make the changes that are suggested. If you fail to follow their advice, you will regret it later at the end of the process.

3.4 After SA board approval.

So, you are done, right? The standard is approved? Not so fast. The technical editor needs to keep track of the draft as it moves through the final editing process to get it ready for publication. At this time, you will focus on making sure the frontmatter (you know, the stuff you ignored earlier), is correct. The rest of the standard can’t be touched (except for trivial editorial corrections, e.g., mis-spellings and puncutation errors).

4. Contributor’s guidelines

5. Miscellaneous

5.1 Message sequence charts

A message sequence chart (MSC) is illustrated in Figure 1

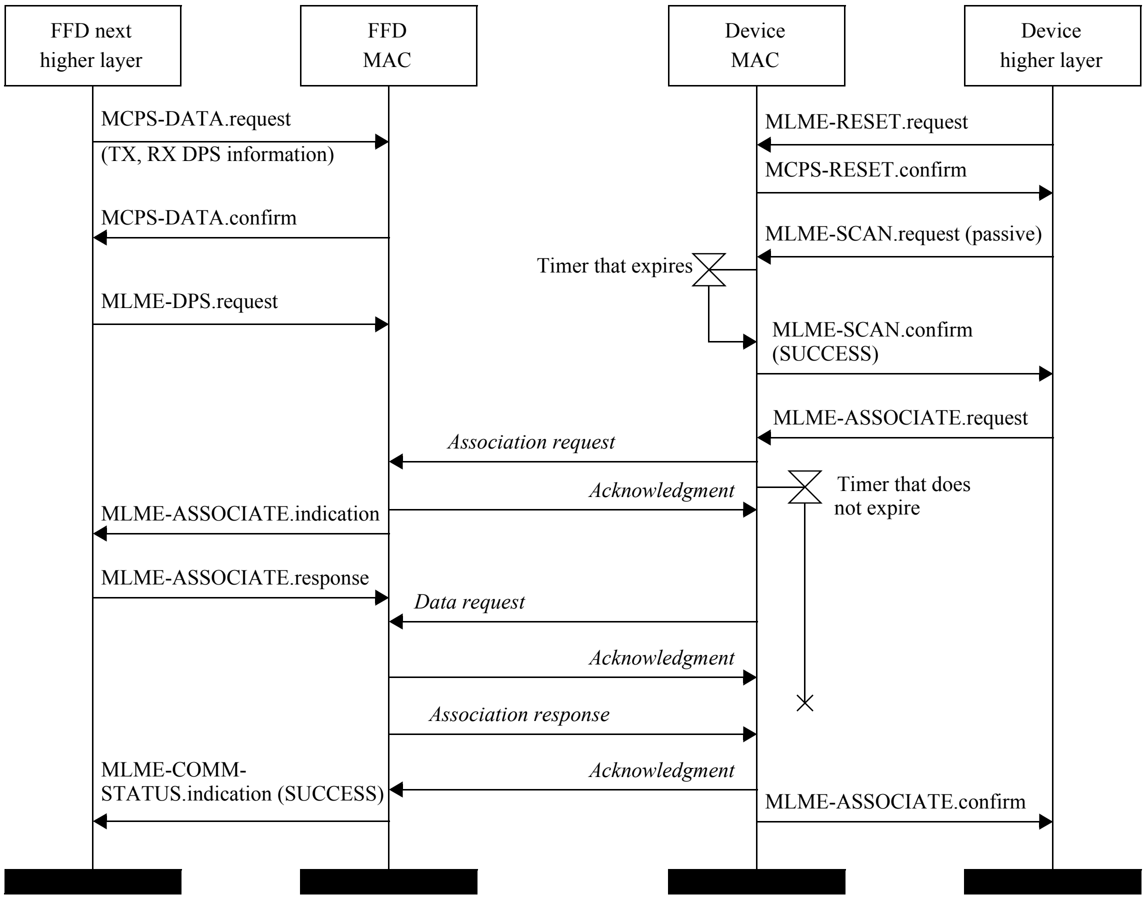

Figure 1—An example of a simple MSC

5.2 Using the graphics tools and anchored frames

5.3 Footnote tricks

In this case, we need a footnote that applies to multiple lines. The method to accomplish this is to insert the footnote on the first line to which it applies. For the other lines, simply add a cross-reference with a format that is “<Superscript><$paranum>”, ideally defining this a cross-reference format called “Footnote”. The appropriate cross-reference will be found via paragraph tags, in this case paragraph type TableFootnote.

Table 1—Test table

Test footnotes
This entry needs a footnote[a]
This entry needs a second footnote[b]
This one needs the same footnote as the first[a]
This one uses the second footnote[b]
So does this one.[b]

[a]This is the first footnote.
[b]This is the second footnote.

5.4 Inserting a text box in a Framemaker figure

An anchored frame, rather than just a text box, can be put into a Framemaker figure. One of the advantages of this is that the anchored frame has access to the paragraph formats defined in the document. Also, you can insert cross references into the anchored frame.

The method to do this is to use the Graphics -> Tools menu. This pops up a floating menu. You have two choices to insert text, one that looks like a page of text with a dashed line around it (for anchored frames) and the second which is just a large “A” (for inserting text boxes). Use the anchored frame button, which will allow you to drag and create an anchored frame in the Framemaker figure. Then set the format to the desired paragraph format and type away.

Note that the anchored frame is also used for the figure titles, using the FigTitle paragraph format (or the AFigTitle and A1FigTitle in the Annexes).

6. Specifying layer management entities (LMEs)

A service access point (SAP) describes the services that a layer provides to another layer, usually a higher layer. Because of this, the SAP can only specify the behavior of its layer and not the layer that uses the SAP.

Unfortunately, there are some bad habits that developed regarding the specification of an LME SAP. An incorrect (editorially) LME SAP specification is shown in 6.1

6.1 Incorrect LME SAP formatting example

6.1.1 MLME-ASSOCIATE.request

The MLME-ASSOCIATE.request primitive allows a device to request an association with a coordinator.

6.1.1.1 Semantics of the service primitive

The semantics of the MLME-ASSOCIATE.request primitive are as follows:

```
MLME-ASSOCIATME.request     (
                            CapabilityInformation
                            )
```

Table 2 specifies the parameters for the MLME-ASSOCIATE.request primitive.

Table 2—MLME-ASSOCIATE.request parameters

Name	Type	Valid range	Description
CapabilityInformation	Bitmap	See 7.3.1.1.2	Specifies the operational capabilities of the associating device.

6.1.1.2 Appropriate usage

The MLME-ASSOCIATE.request primitive is generated by the next higher layer of an unassociated device and issued to its MLME to request an association with a coordinator. If the device wishes to associate with a coordinator on a beacon-enabled PAN, the MLME may optionally track the beacon of that coordinator prior to issuing this primitive.

6.1.1.3 Effect on receipt

On receipt of the MLME-ASSOCIATE.request primitive, the MLME of an unassociated device first updates the appropriate PHY and MAC PIB attributes and then generates an association request command (see 7.3.1.1), as dictated by the association procedure described in 7.5.3.1.

If the association request command cannot be sent to the coordinator due to the CSMA-CA algorithm indicating a busy channel, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of CHANNEL_ACCESS_FAILURE.

If the MLME successfully transmits an association request command, the MLME will expect an acknowledgment in return. If an acknowledgment is not received, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of NO_ACK (see 7.5.6.4).

If the MLME of an unassociated device successfully receives an acknowledgment to its association request command, the MLME will wait for a response to the request (see 7.5.3.1). If the MLME of the device does not receive a response, it will issue the MLME-ASSOCIATE.confirm primitive with a status of NO_DATA.

If the MLME of the device extracts an association response command frame from the coordinator, it will then issue the MLME-ASSOCIATE.confirm primitive with a status equal to the contents of the association status field in the association response command (see 7.3.1.2.3).

On receipt of the association request command, the MLME of the coordinator issues the MLME-ASSOCIATE.indication primitive.

If any parameter in the MLME-ASSOCIATE.request primitive is either not supported or out of range, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of INVALID_PARAMETER.

6.1.2 MLME-ASSOCIATE.indication

The MLME-ASSOCIATE.indication primitive is used to indicate the reception of an association request command.

6.1.2.1 Semantics of the service primitive

The semantics of the MLME-ASSOCIATE.indication primitive are as follows:

```
MLME-ASSOCIATE.indication      (
                               DeviceAddress,
                               CapabilityInformation
                               )
```

Table 3 specifies the parameters for the MLME-ASSOCIATE.indication primitive.

Table 3—MLME-ASSOCIATE.indication parameters

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64-bit IEEE address.	The address of the device requesting association.
CapabilityInformation	Bitmap	See 7.3.1.1.2	The operational capabilities of the device requesting association.

6.1.2.2 When generated

The MLME-ASSOCIATE.indication primitive is generated by the MLME of the coordinator and issued to its next higher layer to indicate the reception of an association request command (see 7.3.1.1).

6.1.2.3 Appropriate usage

When the next higher layer of a coordinator receives the MLME-ASSOCIATE.indication primitive, the coordinator determines whether to accept or reject the unassociated device using an algorithm outside the scope of this standard. The next higher layer of the coordinator then issues the MLME-ASSOCIATE.response primitive to its MLME.

The association decision and the response should become available at the coordinator within a time of macResponseWaitTime (see 7.5.3.1). After this time, the device requesting association attempts to extract the association response command frame from the coordinator, using the method described in 7.5.6.3, in order to determine whether the association was successful.

6.1.3 MLME-ASSOCIATE.response

The MLME-ASSOCIATE.response primitive is used to initiate a response to an MLME-ASSOCIATE.indication primitive.

6.1.3.1 Semantics of the service primitive

The semantics of the MLME-ASSOCIATE.response primitive are as follows:

```
MLME-ASSOCIATE.response         (
                                DeviceAddress,
                                AssocShortAddress,
                                status
                                )
```

Table 4 specifies the parameters for the MLME-ASSOCIATE.response primitive.

Table 4—MLME-ASSOCIATE.response parameters

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address	The address of the device requesting association.
AssocShortAddress	Integer	0x0000–0xffff	The 16-bit short device address allocated by the coordinator on successful association. This parameter is set to 0xffff if the association was unsuccessful.
status	Enumeration	See 7.3.1.2.3	The status of the association attempt.

6.1.3.2 Appropriate usage

The MLME-ASSOCIATE.response primitive is generated by the next higher layer of a coordinator and issued to its MLME in order to respond to the MLME-ASSOCIATE.indication primitive.

6.1.3.3 Effect on receipt

When the MLME of a coordinator receives the MLME-ASSOCIATE.response primitive, it generates an association response command (see 7.3.1.2). The command is sent to the device requesting association using indirect transmission, i.e., the command frame is added to the list of pending transactions stored on the coordinator and extracted at the discretion of the device concerned using the method described in 7.5.6.3.

Upon receipt of the MLME-ASSOCIATE.response primitive, the coordinator attempts to add the information contained in the primitive to its list of pending transactions. If there is no capacity to store the transaction, the MAC sublayer will discard the frame and issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_OVERFLOW. If there is capacity to store the transaction, the coordinator will add the information to the list. If the transaction is not handled within macTransactionPersistenceTime, the transaction information will be discarded and the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_EXPIRED. The transaction handling procedure is described in 7.5.5.

The MAC sublayer enables its receiver immediately following the transmission of the frame and waits for an acknowledgment from the recipient (see 7.5.6.4). If the MAC sublayer does not receive an acknowledgment, the frame will remain in the transaction queue until either another request for the frame is received and correctly acknowledged or until macTransactionPersistenceTime is reached. If macTransactionPersistenceTime is reached, the transaction information will be discarded and the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_EXPIRED.

If the frame was successfully transmitted and an acknowledgment was received, if requested, the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of SUCCESS.

If any parameter in the MLME-ASSOCIATE.response primitive is not supported or is out of range, the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of INVALID_PARAMETER.

6.1.4 MLME-ASSOCIATE.confirm

The MLME-ASSOCIATE.confirm primitive is used to inform the next higher layer of the initiating device whether its request to associate was successful or unsuccessful.

6.1.4.1 Semantics of the service primitive

The semantics of the MLME-ASSOCIATE.confirm primitive are as follows:

```
MLME-ASSOCIATE.confirm        (
                              AssocShortAddress,
                              status
                              )
```

Table 5 specifies the parameters for the MLME-ASSOCIATE.confirm primitive.

Table 5—MLME-ASSOCIATE.confirm parameters

Name	Type	Valid range	Description
AssocShortAddress	Integer	0x0000–0xffff	The short device address allocated by the coordinator on successful association. This parameter will be equal to 0xffff if the association attempt was unsuccessful.
status	Enumeration	SUCCESS, CHANNEL_ACCESS_FAILURE, NO_ACK, NO_DATA, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK, or INVALID_PARAMETER.	The status of the association attempt.

6.1.4.2 When generated

The MLME-ASSOCIATE.confirm primitive is generated by the initiating MLME and issued to its next higher layer in response to an MLME-ASSOCIATE.request primitive. If the request was successful, the status parameter will indicate a successful association, as contained in the status field of the association response command. Otherwise, the status parameter indicates either an error code from the received association response command or the appropriate error code from Table 5. The status values are fully described in 7.1.3.1.3 and subclauses referenced by 7.1.3.1.3.

6.1.4.3 Appropriate usage

On receipt of the MLME-ASSOCIATE.confirm primitive, the next higher layer of the initiating device is notified of the result of its request to associate with a coordinator. If the association attempt was successful, the status parameter will indicate a successful association, as contained in the status field of the association response command, and the device will be provided with a 16-bit short address (see Table 87). If the association attempt was unsuccessful, the address will be equal to 0xffff, and the status parameter will indicate the error.

6.2 Correct format for LME SAP

The correct form is that the .request and .response primitives do not have “When generated” subclauses. The .indication and .confirm do not have “Effect of receipt” or “Appropriate usage”

In addition, the description of status codes is defined in the .confirm primitive.

6.2.1 MLME-ASSOCIATE.request

The MLME-ASSOCIATE.request primitive allows a device to request an association with a coordinator.

The semantics of the MLME-ASSOCIATE.request primitive are as follows:

```
MLME-ASSOCIATME.request     (
                            CapabilityInformation
                            )
```

Table 2 specifies the parameters for the MLME-ASSOCIATE.request primitive.

Table 6—MLME-ASSOCIATE.request parameters

Name	Type	Valid range	Description
CapabilityInformation	Bitmap	See 7.3.1.1.2	Specifies the operational capabilities of the associating device.

On receipt of the MLME-ASSOCIATE.request primitive, the MLME of an unassociated device first updates the appropriate PHY and MAC PIB attributes and then generates an association request command (see 7.3.1.1), as dictated by the association procedure described in 7.5.3.1.

If the association request command cannot be sent to the coordinator due to the CSMA-CA algorithm indicating a busy channel, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of CHANNEL_ACCESS_FAILURE.

If the MLME successfully transmits an association request command, the MLME will expect an acknowledgment in return. If an acknowledgment is not received, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of NO_ACK (see 7.5.6.4).

If the MLME of an unassociated device successfully receives an acknowledgment to its association request command, the MLME will wait for a response to the request (see 7.5.3.1). If the MLME of the device does not receive a response, it will issue the MLME-ASSOCIATE.confirm primitive with a status of NO_DATA.

If any parameter in the MLME-ASSOCIATE.request primitive is either not supported or out of range, the MLME will issue the MLME-ASSOCIATE.confirm primitive with a status of INVALID_PARAMETER.

6.2.2 MLME-ASSOCIATE.indication

The MLME-ASSOCIATE.indication primitive is used to indicate the reception of an association request command.

The semantics of the MLME-ASSOCIATE.indication primitive are as follows:

```
MLME-ASSOCIATE.indication      (
                               DeviceAddress,
                               CapabilityInformation
                               )
```

Table 3 specifies the parameters for the MLME-ASSOCIATE.indication primitive.

Table 7—MLME-ASSOCIATE.indication parameters

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64-bit IEEE address.	The address of the device requesting association.
CapabilityInformation	Bitmap	See 7.3.1.1.2	The operational capabilities of the device requesting association.

The MLME-ASSOCIATE.indication primitive is generated by the MLME of the coordinator and issued to its next higher layer to indicate the reception of an association request command (see 7.3.1.1).

The next higher layer of a coordinator determines whether to accept or reject the unassociated device using an algorithm outside the scope of this standard. The next higher layer of the coordinator then issues the MLME-ASSOCIATE.response primitive to its MLME.

The association decision and the response should become available at the coordinator within a time of macResponseWaitTime (see 7.5.3.1).

6.2.3 MLME-ASSOCIATE.response

The MLME-ASSOCIATE.response primitive is used to initiate a response to an MLME-ASSOCIATE.indication primitive.

The semantics of the MLME-ASSOCIATE.response primitive are as follows:

```
MLME-ASSOCIATE.response        (
                               DeviceAddress,
                               AssocShortAddress,
                               status
                               )
```

Table 4 specifies the parameters for the MLME-ASSOCIATE.response primitive.

Table 8—MLME-ASSOCIATE.response parameters

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address	The address of the device requesting association.
AssocShortAddress	Integer	0x0000–0xffff	The 16-bit short device address allocated by the coordinator on successful association. This parameter is set to 0xffff if the association was unsuccessful.
status	Enumeration	See 7.3.1.2.3	The status of the association attempt.

When the MLME of a coordinator receives the MLME-ASSOCIATE.response primitive, it generates an association response command (see 7.3.1.2). The command is sent to the device requesting association using indirect transmission, i.e., the command frame is added to the list of pending transactions stored on the coordinator and extracted at the discretion of the device concerned using the method described in 7.5.6.3.

Upon receipt of the MLME-ASSOCIATE.response primitive, the coordinator attempts to add the information contained in the primitive to its list of pending transactions. If there is no capacity to store the transaction, the MAC sublayer will discard the frame and issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_OVERFLOW. If there is capacity to store the transaction, the coordinator will add the information to the list. If the transaction is not handled within macTransactionPersistenceTime, the transaction information will be discarded and the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_EXPIRED. The transaction handling procedure is described in 7.5.5.

The MAC sublayer enables its receiver immediately following the transmission of the frame and waits for an acknowledgment from the recipient (see 7.5.6.4). If the MAC sublayer does not receive an acknowledgment, the frame will remain in the transaction queue until either another request for the frame is received and correctly acknowledged or until macTransactionPersistenceTime is reached. If macTransactionPersistenceTime is reached, the transaction information will be discarded and the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of TRANSACTION_EXPIRED.

If the frame was successfully transmitted and an acknowledgment was received, if requested, the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of SUCCESS.

If any parameter in the MLME-ASSOCIATE.response primitive is not supported or is out of range, the MAC sublayer will issue the MLME-COMM-STATUS.indication primitive with a status of INVALID_PARAMETER.

6.2.4 MLME-ASSOCIATE.confirm

The MLME-ASSOCIATE.confirm primitive is used to inform the next higher layer of the initiating device whether its request to associate was successful or unsuccessful.

The semantics of the MLME-ASSOCIATE.confirm primitive are as follows:

```
MLME-ASSOCIATE.confirm          (
                                AssocShortAddress,
                                status
                                )
```

Table 9 specifies the parameters for the MLME-ASSOCIATE.confirm primitive.

Table 9—MLME-ASSOCIATE.confirm parameters

Name	Type	Valid range	Description
AssocShortAddress	Integer	0x0000–0xffff	The short device address allocated by the coordinator on successful association. This parameter will be equal to 0xffff if the association attempt was unsuccessful.
status	Enumeration	SUCCESS, CHANNEL_ACCESS_FAILURE, NO_ACK, NO_DATA, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK, or INVALID_PARAMETER.	The status of the association attempt.

The MLME-ASSOCIATE.confirm primitive is generated by the initiating MLME and issued to its next higher layer in response to an MLME-ASSOCIATE.request primitive. If the request was successful, the status parameter will indicate a successful association, as contained in the status field of the association response command. Otherwise, the status parameter indicates either an error code from the received association response command or the appropriate error code from Table 5. The status values are fully described in 7.1.3.1.3 and subclauses referenced by 7.1.3.1.3.

If the association attempt was successful, the status parameter will indicate a successful association, as contained in the status field of the association response command, and the device will be provided with a 16-bit short address (see Table 87). If the association attempt was unsuccessful, the address will be equal to 0xffff, and the status parameter will indicate the error.