

Open source styles for standards and specifications (OS4): Microsoft Word styles and templates

Draft 0.58

Sponsor:

Friendly supporters

Abstract: These templates were developed by volunteers with extensive experience in the development of IEEE Standards. These templates are not mandated; each working group has the freedom to select the most applicable text-formatting tools. These templates may be freely used by other SDOs and/or private organizations.

Keywords: Style, formatting, template

Copyright (c) 2003 JGG. All rights reserved.

Redistribution and use in electronic and printed documents, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of visible template text must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Usage of these templates, with alternative visible text content, must reproduce the above copyright notice, this list of conditions and the following disclaimer as nonvisible text on the first or last page.
3. The name "JGG" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact dvj@alum.mit.edu.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JGG OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This Copyright notice can be removed and (as necessary) replaced with the appropriate company or Standard's Development Organization (SDO) Copyright notice after print-visible text has been replaced with alternative written material.

Background

This manual describes the use of document templates developed by for use when writing Cypress and IEEE drafts. This document is **preliminary** and **subject to change**.

Contacts

David V. James
3180 South Ct
Palo Alto, CA 94306
Home: +1.650.494.0926
+1.650.856.9801
Cell: +1.650.954.6906
Fax: +1.360.242.5508
Base: dvj@alum.mit.edu

Change history

The following table shows the change history for this user's manual.

Version	Date	Author	Comments
—	—	DVJ	Original content.
0.57	2004Oct19	DVJ	Text updated to track style changes and recent templates wording changes.
0.58	2005Nov25	DVJ	Editorial corrections of list and false-page-break styles.

Table of contents

1. Overview.....	7
1.1 Scope and purpose	7
1.1.1 Template contents.....	7
1.2 Standard clauses.....	8
2. References.....	9
3. Terms, definitions, and notation	10
3.1 Conformance levels	10
3.2 Glossary of terms	10
3.3 Unimplemented locations	11
3.4 Numerical values	11
3.5 Field notation.....	12
3.5.1 Field names.....	12
3.5.2 Fields within figures	13
3.5.3 Multibyte fields within frames.....	13
3.6 C-code notation.....	15
3.7 State machines	16
3.8 Field notations	17
3.8.1 Use of italics	17
3.8.2 Field conventions.....	17
3.8.3 Field value conventions	18
3.9 Informative notes	18
4. Abbreviations and acronyms.....	19
5. Using the templates.....	20
5.1 Installing the templates	20
5.2 Company document numbers	20
5.3 Paragraph styles	21
5.3.1 Baseline paragraph styles	21
5.3.2 Supplemental annex paragraph styles.....	23
5.3.3 Character styles (such as α).....	24
6. Clause and subclause headings	25
6.1 Subclause-2 heading	25
6.1.1 Subclause-3 heading	25
7. Definitions	27
7.1 Leading second-level indent	27
7.1.1 Leading third-level heading.....	27
7.1.2 Definition4 heading	27
7.2 Definition3 headings.....	27
8. Highest level definitions	28
9. Formatting styles.....	29
9.1 Notes.....	29
9.1.1 Editorial notes.....	29
9.1.2 Informative notes.....	29
9.2 Page breaks	30
9.3 Body styles.....	30

9.3.1 Body.....	30
9.3.2 BodyTight.....	30
9.3.3 BodyCenter.....	30
9.3.4 BodyBlock.....	30
9.4 Lists.....	31
9.4.1 Numerical lists.....	31
9.4.2 Dashed lists.....	32
9.5 Tables.....	32
9.5.1 Tables cells.....	32
9.5.2 Tables row labels.....	33
9.5.3 Table headings.....	33
9.5.4 Table footnotes.....	33
9.5.5 Common keyboard commands for working with tables.....	34
9.6 Equations.....	35
9.7 C-code listings.....	35
9.7.1 Shorter in-line C code.....	35
9.7.2 Lengthy C code.....	35
9.8 Footnotes.....	35
9.9 Auto numbering cross references.....	35
9.10 Autonumbered styles.....	36
10. Constructing Visio figures.....	37
10.1 Figures.....	37
10.1.1 Microsoft Visio drawings.....	37
10.1.2 Maintaining the graphics grid.....	37
10.1.3 Register layouts.....	38
10.1.4 Flow-chart styles.....	38
Annex A (informative) Bibliography.....	39
A.1 Bibliography references.....	39
A.2 Grammar references.....	40
Annex B (informative) Annex styles.....	41
B.1 Subannex-2 header.....	41
B.1.1 Subannex-3 header.....	41
B.2 Definitions.....	43
B.2.1 Leading third-level heading.....	43
B.2.2 Definition4 heading.....	43
B.3 Definition3 headings.....	43
Annex C (informative) Highest level definitions.....	45
Annex D (informative) Numbered annex styles.....	46
D.1 Tables and figures.....	46
D.1.1 Annex figures.....	46
D.1.2 Annex tables.....	46
D.1.3 Equation numbering difference.....	46
Annex E (informative) Common grammatical issues.....	47
E.1 Use of the word “the”.....	47
E.2 Use of the word “a” or “an”.....	47
E.3 Numbers: numerical or alphabetical?.....	48
E.4 Table cell justification.....	48
E.5 That and which.....	49
E.6 Gender-Neutral language.....	49

E.7 Use of the second-person form of address.....	49
Annex F (informative) General writing guidelines	50
F.1 Capitalization.....	50
F.1.1 When to capitalize	50
F.1.2 Heading capitalization	51
F.1.3 Noncapitalized definitions.....	51
F.1.4 Noncapitalized acronyms	51
F.1.5 Improper nouns.....	51
F.2 Definitions	52
F.2.1 Definition clauses	52
F.2.2 Definition text.....	52
F.3 Variable names	52
F.3.1 Spaceless variable names	52
F.3.2 Hyphenless variable names	52
F.3.3 Aliased variable names.....	53
F.4 Spelling checker	53
F.5 Cross references	53
F.6 Indentation	54
F.7 Table cell justification	54
F.8 Multi-field registers	55
F.9 Punctuation within lists	56
Annex G (informative) RPR writing examples.....	57
G.1 Field formats within figures.....	57
G.1.1 Frame format illustrations.....	57
G.1.2 Generic field illustrations.....	58
G.1.3 8-bit field format.....	58
G.1.4 16-bit field formats	59
G.1.5 32-bit field formats	59
G.2 Standard subclause formats.....	60
G.2.1 Common parameters	60
G.2.2 Variables defined in other clauses	61
G.3 State machine definitions.....	61
G.3.1 Single-queue transmit state machine.....	62
G.4 Service primitives	66
G.4.1 MA_DATA.request	66
Annex H (informative) Using word	68
H.1 Special characters.....	68
H.2 Inserting template into an existing document	68
Annex I (informative) IEEE specifics.....	69
I.1 IEEE references	69
I.2 IEEE cover pages	71
Annex J (informative) C code illustrations	75

List of figures

Figure 3.1—Expanded bit-field descriptions	13
Figure 3.2—Frame format illustrations	13
Figure 3.3—Flow-control sendA generation overview	16
Figure 10.1—Pointer illustration.....	37
Figure 10.2—Microsoft Visio expansion and contraction restrictions.....	37
Figure 10.3—Expanded bit-field descriptions ns	38
Figure 10.4—Flow-control <i>sendA</i> generation overview	38

List of tables

Table 3.1—Names of constants, registers and fields.....	12
Table 3.2—C code expressions	15
Table 3.3—Flow-control sendA generation details.....	16
Table 3.4—Names of fields and sub-fields	17
Table 3.5— <i>wrap</i> field values	18
Table 5.1—Paragraph styles	21
Table 5.7—Supplemental annex paragraph styles	23
Table 5.8—Supplemental annex paragraph styles	24
Table 9.1—Example table.....	32
Table 9.2—Row-numbered table	33
Table 9.3—Table footnotes.....	33
Table 9.4—Instructions for editing word tables.....	34

Open source styles for standards and specifications (OS4): Microsoft Word styles and templates

1. Overview

1.1 Scope and purpose

NOTE—Every document should start off the overview with a scope and purpose statement. Each should consist of a single paragraph outlining, as clearly as possible, the scope and purpose of the document. These should be viewed as executive summaries. The scope is intended to communicate the range of topics covered in the document; the purpose is intended to describe the reasons for generation of the document.

This document is intended to assist Company engineers in the development of standards, with the scope and purpose listed below:

Scope: This document describes the use of standard Microsoft Word¹ templates for creating ISO/IEC compatible standards. Company engineering documents may also use these style guidelines.

Purpose: To provided clarity and consistency of Company documentation developed for internal engineering uses, and to facilitate the transfer of such specifications to standards development organizations (SDOs) for the subset of specifications intended to be standardized.

The templates described by this document contain all the formatting necessary for the cover page, table of contents, list of tables, list of figures, main content, and annexes of your document. No index formatting has been provided, since the editors of this document do not ordinarily have the time to create an index.

1.1.1 Template contents

The templates described by this document contain all the formatting necessary for the cover page, table of contents, list of tables, list of figures, main content, and annexes of your document. These templates provide the editor with the following services:

- a) Boilerplate. Standard boilerplate as well as terms-and-definitions material is provided.
- b) Table of contents. Automatic generation of table of contents, optionally including:
 - 1) List of figures.
 - 2) List of tables.
- c) Autonumbering. All clauses, subclauses, annexes, and subannexes as well as tables, figures, equations, and table rows are automatically numbered.

¹ All product and company names mentioned in this document are trademarks of their respective holders.

- d) Commonality. The same look-and-feel templates are available in FrameMaker, Word, and OpenOffice.

The intent of this document is to improve productivity by improving the readability of architecture-specification documents. Perceived benefits to individual companies, that may choose to use these formats for internal specifications, include (but are not necessarily limited to) the following:

- a) Efficiency. Fewer hours will be consumed communicating necessary information.
- b) Quality. Fewer mistakes will occur due to misinterpreted specifications.
- c) Timeliness. Reviews and product design times will be reduced due to improved efficiency.
- d) Transferability. Company text can be readily copied into IEEE and ISO/IEC standards proposals.
- e) Sustainability. Following IEEE and ISO/IEC formats eliminates the need for document-style committees.

1.2 Standard clauses

Document contents are usually constrained by the type of document you are writing, or by documentation standards outlined by whatever agency or office requests or requires your document. However, a few clauses and annexes are expected to take the form described below.

- Clause 1. Overview shall be the first clause and shall start with scope and purpose subclauses.
- Clause 2. References shall be the second clause, edited as appropriate.
- Clause 3. Definitions and notation shall be the third clause, edited as appropriate.
- Annex A (informative). Bibliography shall appear in every document.

2. References

NOTE—References are listed here is their content is normative, in that the document would be incomplete without them. Other documents that provide background, but not specification material, should be included in Annex A.

The following standards contain provisions that, through reference in this document, constitute provisions of this standard. All the standards listed are normative references. Informative references are given in Annex A. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

NOTE—The Reference paragraph style is normally applied to references, as illustrated below.

- [R1] IEEE Standards Style Manual, October 1996.²
- [R2] ANSI/ISO 9899-1990, Programming Language—C.^{3,4}
- [R3] *The Chicago Manual of Style*. Chicago: The University of Chicago Press
- [R4] *Words Into Type*. Englewood Cliffs, NJ: Prentice-Hall, Inc

All the standards listed are normative references. Informative references are given in Annex A. At the time of publication, the editions indicated were valid.

NOTE—Editors should be aware that approval votes on several draft standards have been delayed due incomplete reference lists. If other standards are referenced, your standard should clearly state which portions apply. Every reference should include a footnote, clearly specifying how this document can be obtained and/or purchased, so that your reviewers have access to these normative references.

NOTE—Standards a specific draft of a standard may be referenced, such as “ISO 646:1991.” This improves the stability of your standard, decoupling it from revisions or extensions in your referenced standard. Alternatively, you may reference the most recent version of a standard, such as “ISO 646.” This allows your standard to evolve over time and reduces dependencies on potentially out-of-print documents. Both approaches have their advantages and disadvantages that should be carefully considered by the working group.

² The IEEE Style Manual is available through the IEEE web site: <http://standards.ieee.org/guides/index.html>

³ Replaces ANSI X3.159-1989.

⁴ ISO documents are available from ISO Central Secretariat, 1 rue de Varembe, Case Postale 56, CH-1211, Genève 20, Switzer-land/Suisse; and from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036-8002, USA

3. Terms, definitions, and notation

These subclauses contain examples of specifications that may be included in an IEEE Standard.

3.1 Conformance levels

Several keywords are used to differentiate between different levels of requirements and optionality, as follows:

3.1.1 expected: Describe the behavior of the hardware or software in the design models assumed by this specification. Other hardware and software design models may also be implemented.

NOTE—The preceding “expected” conformance statement has been found to be useful in some standards. The following “may, shall, should” conformance definitions should be provided by IEEE standards; if provided, their definitions shall be as follows:

3.1.2 may: Indicates a course of action permissible within the limits of the standard with no implied preference (“may” means “is permitted to”).

3.1.3 shall: Indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (“shall” means “is required to”).

3.1.4 should: An indication that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (“should” means “is recommended to”).

3.2 Glossary of terms

NOTE—Other terms that have special meanings in the context of your document should also be included here. The numbering scheme is necessary for IEEE documents.

NOTE—The following terms illustrate the types of terms that could be included in your document. Recent standards avoid the use of *half-word*, *word*, or *double-word* to describe register or bus widths, as the meaning of *word* is highly context sensitive and therefore subject to misinterpretation.

3.2.1 byte: Eight bits of data, used as a synonym for octet.

3.2.2 doublet: Two bytes of data.

3.2.3 quadlet: Four bytes of data.

3.2.4 octlet: Eight bytes of data.

NOTE—Other terms that have special meanings in the context of your document should also be included here. The numbering scheme is necessary for IEEE documents.

3.3 Unimplemented locations

The capabilities of reserved, ignored, and unused values are carefully defined, to minimize conflicts between current implementations and future definitions.

3.3.1 reserved fields: A set of bits within a data structure that is defined in this specification as reserved, and is not otherwise used. Implementations of this specification shall zero these fields. Future revisions of this specification, however, may define their usage.

3.3.2 ignored location: Selected locations or portions of locations are partially implemented and are defined to be ignored (abbreviated as *ign* or *i*). An ignored value has an affiliated storage element, but the value in the storage elements has no side effect.

3.3.3 reserved location: Some locations or portions of locations are not implemented and are defined to be reserved (abbreviated as *res* or *r*). When a reserved value is written, a zero values shall be assumed; when read, the returned value shall be ignored.

3.3.4 unused location: Selected locations or portions of locations may be not implemented or partially implemented and are defined to be unused (abbreviated as *un* or *u*). For unused locations, the selection between reserved and ignored behaviors is implementation dependent.

3.4 Numerical values

NOTE—The following notation text, or its equivalent, are required to explain your notation. The subscript definition is preferred within English text, although other notations are used within code.
--

Decimal, hexadecimal, and binary numbers are used within this document. For clarity, decimal numbers are generally used to represent counts, hexadecimal numbers are used to represent addresses, and binary numbers are used to describe bit patterns within binary fields.

Decimal numbers are represented in their usual 0, 1, 2, ... format. Hexadecimal numbers are represented by a string of one or more hexadecimal (0-9,A-F) digits followed by the subscript 16. Binary numbers are represented by a string of one or more binary (0,1) digits, followed by the subscript 2. Thus the decimal number “26” may also be represented as “1A₁₆” or “11010₂”.

These notational conventions have one exception: MAC addresses and OUI/EUI values are represented as strings of 8-bit hexadecimal numbers separated by hyphens and without a subscript, as for example “01-80-C2-00-00-15” or “AA-55-11”.

3.5 Field notation

NOTE—All documents should describe their naming conventions, with text similar to the following.

3.5.1 Field names

This document describes values that are in memory-resident or control-and-status registers. For clarity, distinct capitalization conventions are used when naming different components, as illustrated in Table 3.1.

Table 3.1—Names of constants, registers and fields

Name	Row	Description
MAX_VALUE	1	A defined constant value
StateMachineName	2	A formal state machine name (if required)
parameter_value	3	A Service Primitive parameter
<i>RoutineName()</i>	4	A subroutine name, when referenced within text.
<i>runCommand</i>	5	A referenced control register.
<i>startCode</i>	6	The <i>startCode</i> field
<i>start</i>	7	The <i>start</i> bit
<i>runCommand.startCode</i>	8	The <i>startCode</i> field within the <i>runCommand</i> register
<i>runCommand.start</i>	9	The <i>start</i> bit within the <i>runCommand</i> register

NOTE—If ever applied to a variable name, the italics style should always be applied to that variable name, whether contained in tables, figures, or headings (but not C-code, where Courier is used).

NOTE—When lengthy descriptions are necessary, row numbers and following clarifications should supplement the descriptive column, as illustrated (for Table 3.1) in the text below.

Row 1: Constant values are spelled with capital letters; an underscore separates run-together words.

Row 2: Formal state machine names (when necessary) do not include blank spaces.

Row 3: Service primitive parameters include underscores, for consistency with the past and to differentiate these parameters from defined field values.

Row 4: Subroutine names (within normal text) are distinguished by italics and first-capital-letter. The italics formatting convention applies to the name, not associated special symbols, such as ‘(‘ and ‘)’.

Row 5, Row 6, Row 7: Register names, fields, and bit names start with a lower-case letter; each run-together word starts with a capital letter. Run-together names like *runCommand* are preferred because they are more compact than under-score-separated names (like “*run_command*”).

Row 8, Row 9: When their register location is unclear or ambiguous, the name of a fields includes the name of the register where that field is located.

3.5.2 Fields within figures

NOTE—Numbering of bits within registers causes problems, due to distinct bit numbering conventions adopted by little-endian and big-endian designers. Bit ordering arguments are not easily resolved by using the bit-transmission order, which can be PHY dependent or ambiguous. Arguments and confusions are best resolved by avoiding the numbering altogether, which also simplifies each illustration.

The location of fields within registers is specified by the cumulative widths of fields within the register, as illustrated in Figure 3.1. The width of each field (in bits) is implied by bottom-line tick marks; the field name is normally contained within its bounding rectangle. When the field name is smaller than its bounding rectangle, lines associate the field’s name with its location (as illustrated for *error*, *mode*, and *phase* bits).

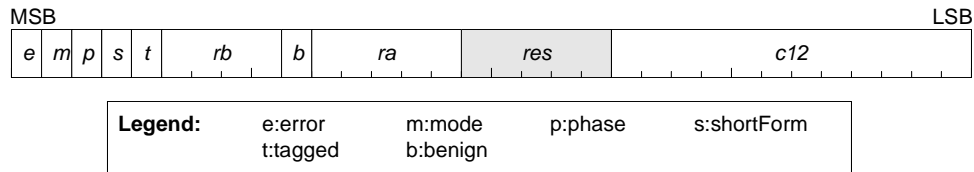


Figure 3.1—Expanded bit-field descriptions

NOTE—Abbreviations within figures are the only time that special names are spelled differently. Within such figures, the association with the abbreviated and standard name must be specified in the legend.

3.5.3 Multibyte fields within frames

Figure 3.2 provides an illustrative example of another possible byte-sequential data representation. These representations are drawn as fields (of arbitrary size) ordered along a vertical axis, with numbers along the left sides of the fields indicating the field sizes in bytes. Fields are drawn contiguously such that the transmission order across fields is from top to bottom. The example shows that *ttl*, *baseControl*, and *da* are 1-, 1- and 6-byte fields, respectively, transmitted in order starting with the *ttl* field first. .

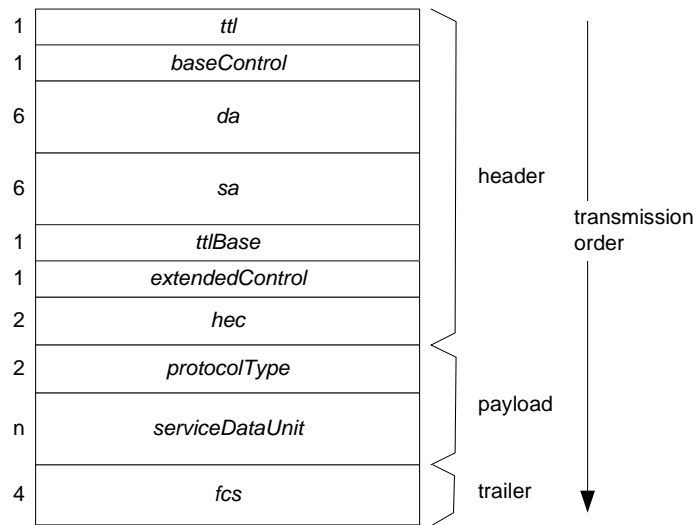


Figure 3.2—Frame format illustrations

NOTE—The height of the rectangles need not be proportional to the field size, but shall be monotonic with respect to the size of the contained field. Thus, if $\text{sizeof}(b) > \text{sizeof}(a)$, then $\text{height}(b) \geq \text{height}(a)$ where $\text{sizeof}(x)$ and $\text{height}(s)$ represent the byte-count and height associated with rectangle x .

NOTE—The right-side arrow and Transmission order text are useful when describing this notation. In other illustrations, this information would be redundant and distracting, and is therefore not recommended.

3.6 C-code notation

The behavior of data-transfer command execution is frequently specified by C code, such as 3.1. To differentiate this code from textual descriptions, such C code listings are formatted using a fixed-width Courier font. Similar C-code segments are included within some figures.

```
// Return maximum of a and b values
Max(a,b) {
  if (a<b)
    return(LT);
  if (a>b)
    return(GT);
  return(EQ);
}
```

(3.1)

Since the meaning of many C code operators are not obvious to the casual reader, their meanings are summarized in Table 3.2.

Table 3.2—C code expressions

Expression	Description
<code>~i</code>	Bitwise complement of integer <i>i</i>
<code>i^j</code>	Bitwise EXOR of integers <i>i</i> and <i>j</i>
<code>i&j</code>	Bitwise AND of integers <i>i</i> and <i>j</i>
<code>i<<j</code>	Left shift of bits in <i>i</i> by value of <i>j</i>
<code>i*j</code>	Arithmetic multiplication of integers <i>i</i> and <i>j</i>
<code>!i</code>	Logical negation of Boolean value <i>i</i>
<code>i&& j</code>	Logical AND of Boolean <i>i</i> and <i>j</i> values
<code>i j</code>	Logical OR of Boolean <i>i</i> and <i>j</i> values
<code>i ^= j</code>	Equivalent to <code>i = i^j</code> .
<code>i == j</code>	Equality test, true if <i>i</i> equals <i>j</i>
<code>i != j</code>	Equality test, true if <i>i</i> does not equal <i>j</i>
<code>i < j</code>	Inequality test, true if <i>i</i> is less than <i>j</i>
<code>i > j</code>	Inequality test, true if <i>i</i> is greater than <i>j</i>

3.7 State machines

NOTE—The following illustrations have been found useful to some IEEE working groups and are therefore provided as optional content. This material should be deleted if such state machine specifications are not used within the document.

Flow charts are used throughout this document to illustrate high-level functionality, as illustrated in Figure 3.3. Flow charts are typically affiliated with an exact table-structured specification, as illustrated in Table 3.3.

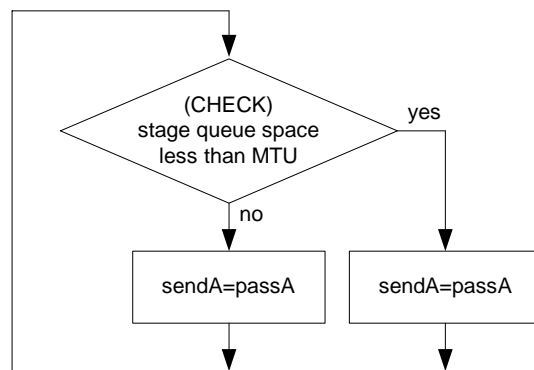


Figure 3.3—Flow-control sendA generation overview

Table 3.3—Flow-control sendA generation details

Current state		Row	Next state	
state	condition		action	state
START	$\text{DepthStageBuffer} < (\text{sizeStageBuffer} - \text{MTU})$	1	sendA= 0	START
	—	2	sendA= passA	

Table 3.3-1: At least one MTU of stage-queue storage is required to safely buffer client-supplied frames.

Table 3.3-2: Setting *sendA* to the *passA* value enables transmissions when stage-buffer space is available.

NOTE—The first state within all state machines should be distinctive and uniformly defined. The name START is recommended for this purpose.

3.8 Field notations

3.8.1 Use of italics

All field names or variable names (such as *level* or *myMacAddress*), and sub-fields within variables (such as *thisState.level*) are italicized within text, figures and tables, to avoid confusion between such names and similarly spelled words without special meanings. A variable or field name that is used in a subclause heading or a figure or table caption is also italicized. Variable or field names are not italicized within C code, however, since their special meaning is implied by their context. Names used as nouns (e.g., *subclassA0*) are also not italicized.

3.8.2 Field conventions

This document describes values that are packetized or MAC-resident, such as those illustrated in Table 3.4.

Table 3.4—Names of fields and sub-fields

Name	Description
<i>newCRC</i>	Field within a register or frame
<i>thisState.level</i>	Sub-field within field <i>thisState</i>
<i>thatState.rateC[n].c</i>	Sub-field within array element <i>rateC[n]</i>

Run-together names (e.g., *thisState*) are used for fields because of their compactness when compared to equivalent underscore-separated names (e.g., *this_state*). The use of multiword names with spaces (e.g., “This State”) is avoided, to avoid confusion between commonly used capitalized key words and the capitalized word used at the start of each sentence.

A sub-field of a field is referenced by suffixing the field name with the sub-field name, separated by a period. For example, *thisState.level* refers to the sub-field *level* of the field *thisState*. This notation can be continued in order to represent sub-fields of sub-fields (e.g., *thisState.level.next* is interpreted to mean the sub-field *next* of the sub-field *level* of the field *thisState*).

Unless specifically specified otherwise, reserved fields are reserved for the purpose of allowing extended features to be defined in future revisions of this standard. For devices conforming to this version of this standard, nonzero reserved fields are not generated; values within reserved fields (whether zero or nonzero) are to be ignored.

3.8.3 Field value conventions

This document describes values of fields. For clarity, names can be associated with each of these defined values, as illustrated in Table 3.5. A symbolic name, consisting of upper case letters with underscore separators, allows other portions of this document to reference the value by its symbolic name, rather than a numerical value.

Table 3.5—*wrap* field values

Value	Name	Description
0	WRAP_AVOID	Frame is discarded at the wrap point
1	WRAP_ALLOW	Frame passes through wrap points.
2-3	—	Reserved

Unless otherwise specified, reserved values are reserved for the purpose of allowing extended features to be defined in future revisions of this standard. Devices conforming to this version of this standard do not generate reserved values for fields, and process fields containing reserved values as though the field values were not supported. The intent is to ensure default behaviors for future-specified features.

A field value of TRUE shall always be interpreted as being equivalent to a numeric value of 1 (one), unless otherwise indicated. A field value of FALSE shall always be interpreted as being equivalent to a numeric value of 0 (zero), unless otherwise indicated.

3.9 Informative notes

Informative notes are used in this standard to provide guidance to implementers and also to supply useful background material. Such notes never contain normative information, and implementers are not required to adhere to any of their provisions. An example of such a note follows.

NOTE—This is an example of an informative note.

4. Abbreviations and acronyms

This document contains the following abbreviations and acronyms:

NOTE—Editors should update the following list of acronyms based on those used within their drafts.
--

CSR control and status register
IEEE Institute of Electrical and Electronics Engineers, Inc.
RAM random access memory
ROM read only memory
TCAM ternary content addressable memory.

5. Using the templates

The following instructions apply to the IEEE Standards template (*IEEE.doc*). If you have questions regarding any of the following instructions, or regarding installing templates in general, consult the Microsoft Word Help menu or contact Marketing Technical Publications.

5.1 Installing the templates

The templates are standard Microsoft Word documents with a *.doc* extension. These templates are entirely self-contained and will have no effect on your *Normal.dot* template file. You should place the affiliated templates into the default template directory for your system. To do this, double-click to open the template file and select **File/Save As**. Then, under **Save As Type** select the *Document Template (*.dot)* option. This automatically places the template into the default template directory.

To begin using the template, select File|New|General, and select *IeeeTemplate.dot*.

5.2 Company document numbers

All Company documents should have a distinctive identifier. If a number has not yet been obtained from document control, a default identifier should be used, as illustrated below:

CompanyDVJ2002Jul02

The parameters within this identifier include:

- Company—To identify this as a Company authored document.
- DVJ—A distinct identifier that identifies the author.
- 2002Jul02—The year, month, and day when the first draft was written.

5.3 Paragraph styles

5.3.1 Baseline paragraph styles

The specific paragraph styles that this template provides are listed in Table 5.1.

Table 5.1—Paragraph styles

Style name	Row	Description	Reference
Annexes	1	Places “Annexes” prior to the first annex header.	Annex A
Bibliography	2	For enumerating informative Bibliography references.	Annex A
Body	3	Basic text throughout.	9.1
BodyCenter	4	Centered text.	
Code	5	Courier 132-column code within the annex	Annex J
BodyTight	6	Acronym & abbreviation lists.	9.1
Copyright	7	The copyright notice on the front page footer.	page 1
Definition2	8	Second-level text for definition-inclusion purposes.	8.1
Definition2Like	9	Second-level text for definition-exclusion purposes.	8.2
Definition3	10	Third-level text for definition-inclusion purposes.	7.2.1
Definition3Like	11	Third-level text for definition-exclusion purposes.	7.2.2
Definition4	12	Fourth-level text for definition-inclusion purposes.	7.1.2.1
Definition4Like	13	Fourth-level text for definition-exclusion purposes.	7.1.2.2
Definition5	14	Fourth-level text for definition-inclusion purposes.	7.1.1.1.1
Definition5Like	15	Fifth-level text for definition-exclusion purposes.	7.1.1.1.2
EditorNote	16	Editorial note, work-in-progress comments	9.1
EquationMath	17	A numbered equation	9.6
EquationCode	18	A numbered C-code equation.	9.7.1
FigureTitle	19	Trailer placed after the figure	10
Heading 1	20	Clause heading	Clause 5.3.2
Heading 2	21	Second level subclause	6.1
Heading 3	22	Third level subclause	6.1.1
Heading 4	23	Fourth level subclause	6.1.1.1
Heading 5	24	Fifth level subclause (the last recommended level)	6.1.1.1.1
Heading 6	25	Sixth level subclause	6.1.1.1.1.1
Heading 7	26	Sixth level subclause (the last supported level)	6.1.1.1.1.1.1
List1	27	First-level list: 1) format	9.4.2
List1More	28	First-level list continued	
List1Dash	29	First-level dash list	

Style name	Row	Description	Reference
List2	30	Second-level list: a) format	9.4.2
List2More	31	Second-level list continued	
List2Dash	32	Second-level dash list	
List3	33	Third-level list: i) format	9.4.2
List3More	34	Third-level list continued	
List3Dash	35	Third-level dash list	
Note	36	Informational content	9.1.2
Reference	37	Normative references.	Clause 2
TableCellCenter	38	Center-justified table cell	9.5.1
TableCellCode	39	Centered C-code within a table cell	3.6
TableCellLeft	40	Left-justified table cell	9.5.1
TableCellRow	41	Center justified table-row number	9.5.1
Table Footnote	42	Use this style for footnotes appended to a table.	9.5.4
TableHeading	43	Used for the headings of tables.	9.5.1
TableTitle	44	Header line placed before the table	9.5.1
Title	45	24-point title	9.9
Title10	46	Abstract and keywords text in the title page.	

5.3.2 Supplemental annex paragraph styles

The specific paragraph styles that this template provides are listed in Table 5.7.

Table 5.7—Supplemental annex paragraph styles

Style name	Row	Description	Reference
~Definition2	1	Second-level text for definition-inclusion purposes.	8.1
~Definition2Like	2	Second-level text for definition-exclusion purposes.	8.2
~Definition3	3	Third-level text for definition-inclusion purposes.	7.2.1
~Definition3Like	4	Third-level text for definition-exclusion purposes.	7.2.2
~Definition4	5	Fourth-level text for definition-inclusion purposes.	7.1.2.1
~Definition4Like	6	Fourth-level text for definition-exclusion purposes.	7.1.2.2
~Definition5	7	Fourth-level text for definition-inclusion purposes.	7.1.1.1.1
~Definition5Like	8	Fifth-level text for definition-exclusion purposes.	7.1.1.1.2
~EquationMath	9	A numbered equation	9.6
~EquationCode	10	A numbered C-code equation.	9.7.1
~Heading1	11	Annex heading (new page)	Annex A
~Heading2	12	Second level subannex	B.1
~Heading3	13	Third level subannex	B.1.1
~Heading4	14	Fourth level subannex	B.1.1.1
~Heading5	15	Fifth level subannex	B.1.1.1.1
~Heading6	16	Sixth level subannex	B.1.1.1.1
~Heading7	17	Seventh level subannex	B.1.1.1.1

5.3.3 Character styles (such as α)

The specific paragraph styles that this template provides are listed in Table 5.7.

Table 5.8—Supplemental annex paragraph styles

Style name	Row	Description	Reference
Bold	1	Bold text modifier..	TBD
Emphasis	2	Italics text modifier.	TBD
NameItalic	3	Italic non-hyphenated (no-language) text	TBD
NamePlain	4	Plain non-hyphenated (no-language) text	TBD
Subscript	5	Subscripted text	TBD
Superscript	6	Superscripted text .	TBD
Symbol	7	Greek symbols, when included in titles	TBD

6. Clause and subclause headings

The previous *Clause and subclause headings* text was created using the *Heading 1* paragraph style and typing the text *Clause and subclause headings*. Each *Heading 1* heading is preconfigured to start on a new page; standards may modify this style to eliminate the page break or force an odd-page break, depending on the document length and editorial preferences.

6.1 Subclause-2 heading

The previous *Subclause-2 heading* text was created using the *Heading 2* paragraph style and typing the text *Subclause-2 heading*. The *Heading 2* heading is configured to force before and after spacing, but no page break.

6.1.1 Subclause-3 heading

The previous *Subclause-3 heading* text was created using the *Heading 3* paragraph style and typing the text *Subclause-3 heading*. The *Heading 3* heading is configured to force before and after spacing, but no page break.

6.1.1.1 Subclause-4 heading

The previous *Subclause-4 heading* text was created using the *Heading 4* paragraph style and typing the text *Subclause-4 heading*. The *Heading 4* heading is configured to force before and after spacing, but no page break.

6.1.1.1.1 Subclause-5 heading

The previous *Subclause-5 heading* text was created using the *Heading 5* paragraph style and typing the text *Subclause-5 heading*. The *Heading 5* heading is configured to force before and after spacing, but no page break.

This is the deepest level allowed by the IEEE! Thus, you should revise any IEEE document that requires to the use of any level-6 or lower levels. This formatting style is intended to force conformance to good document-style conventions, rather than providing the author with an unlimited range of ill-conceived nesting depths.

6.1.1.1.1.1 Subclause-6 heading

The previous *Subclause-6 heading* text was created using the *Heading 6* paragraph style and typing the text *Subclause-6 heading*. The *Heading 6* heading is configured to force before and after spacing, but no page break.

6.1.1.1.1.1.1 Subclause-7 heading

The previous *Subclause-7 heading* text was created using the *Heading 7* paragraph style and typing the text *Subclause-7 heading*. The *Heading 7* heading is configured to force before and after spacing, but no page break.

Note this is the deepest level that is supported by these formats! Thus, you should revise any document that requires to the use of any level-7 or lower levels. This formatting constraint results from having 9 number levels, with three of these reserved for figure-title, table-title, and equation-title purposes.

7. Definitions

Paragraphs of definitions may be distinctively numbered to facilitate their cross referencing by other parts of a document. This should only be done at the lowest level, to avoid discontinuities in the heading numbering, as has occurred with the numbering of 7.1.

7.1 Leading second-level indent

7.1.1 Leading third-level heading

7.1.1.1 Definition5 heading

Within fourth-level subclauses, a fifth-level field distinctively numbers paragraphs, as illustrated in 7.1.1.1.1 and 7.1.1.1.2.

7.1.1.1.1 definedValue5a: The *Definition5* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

7.1.1.1.2 definedValue5a: The *Definition5Like* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

7.1.2 Definition4 heading

Within third-level subclauses, a fourth-level field distinctively numbers paragraphs, as illustrated in 7.1.2.1 and 7.1.2.2.

7.1.2.1 definedValue4a: The *Definition4* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

7.1.2.2 definedValue4a: The *Definition4Like* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

7.2 Definition3 headings

Within second-level subclauses, a third-level field distinctively numbers paragraphs, as illustrated in 7.2.1 and 7.2.2

7.2.1 definedValue3a: The *Definition3* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

7.2.2 definedValue3a: The *Definition3Like* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

8. Highest level definitions

Within clauses, a second-level field distinctively numbers paragraphs, as illustrated in 8.1 and 8.2. These labels are numbers invoked through the use of LISTNUM variables, rather than distinct paragraph styles.

8.1 definedValue1a: The *Definition2* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

8.2 definedValue1b: The *DefinitionLike* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

9. Formatting styles

The following subclauses contain information for understanding the use of these templates. Though not comprehensive, they provide guidance for preparation of formal documents.

9.1 Notes

9.1.1 Editorial notes

<p>Editor's note: To be removed prior to publication. The spelling checker should be run before this specification is finalized.</p>

<p>NOTE—Comments within this document provide out-of-band explanations without inserting paragraph styles that would not normally be provided.</p>
--

The *EditorNote* style is associated with the editorial note above. These are intended to be included within drafts for the purpose of communicating information between the editor and readers. Information may clarify the intent of recent changes, or speculate on additional text needed to complete the document.

9.1.2 Informative notes

Informative notes can be inserted throughout the document, as illustrated by the following two-paragraph note. Notes are *not* official parts of the specification—they are merely informative. Contrast this with informative annexes, which are set off in separate sections of the document. This is the reason for setting notes in a different font size.

NOTE—The *Note* style (used in this paragraph) is used on note paragraphs that follow the note style, although notes are rarely longer than one paragraph.

The *Note* style can also be used without the leading “NOTE—“ typed characters, for continued notes, although notes are rarely longer than one paragraph.

9.2 Page breaks

Page breaks can be manually inserted by using the Insert/Break pulldown menu, as was done before the preceding subclause heading. This convention has been found to be convenient and avoids the need to override default paragraph styles.

9.3 Body styles

9.3.1 Body

The *Body* style is associated with the basic text blocks in this document, including this example. The *Body* style is 10-point Times New Roman, black, with right and left justification. Times New Roman is the serif font that is used primarily throughout this document. Exceptions are Arial and Courier fonts: Arial is used for headings and figure/table/equation titles; Courier is used for C code.

9.3.2 BodyTight

The *BodyTight* style is similar, but less paragraph-to-paragraph spacing is specified. Thus, such styles are appropriate for acronym listings, as listed below although normally utilized within 0.

NSE Network search engine
RPR Resilient packet ring
SRAM Static random access memory

9.3.3 BodyCenter

The *BodyCenter* style is used for centered text, such as written below.

This style is provided for use on IEEE front-cover pages.

Note that centering should be done using a distinct style instead of by using a style override, so that the centering property will survive automatic reapplication or importation of the styles by subsequent editors.

9.3.4 BodyBlock

The *BodyBlock* style forces text to remain on one page, rather than split across pages, as used below.

```
MA_DATA.request  
(  
    parameter1,  
    parameter2,  
    parameter3  
)
```

9.4 Lists

In general, lists are used to display information that does not require explanation or that is offered by way of explanation. Lists that are written in phrases should not allow each item to end in a period or other closing punctuation; rather, only the last item of such a list should allow a period at the end. If however each item in the list is a sentence or a series of sentences, use closing punctuation for each item. Also, it is not necessary to repeat subject information in each list item, nor is it necessary to precede every list with a paragraph ending in a semicolon. It is best to end the paragraph above a list with a period (especially with sentence lists), or with no punctuation at all (with phrase lists).

9.4.1 Numerical lists

For generating numerical lists, start with the *Body* style. The *Body* style is three levels deeper, starting with Arabic numerals at the first level, lowercase letters at the second and finally lower-case roman numerals at the third. To increase or decrease the indent, click:



(the **Increase Indent** button) or Ctrl+M to increase the indent level.



(the **Decrease Indent** button) or Ctrl+Shift+M to decrease the indent level.

Any list is limited to three levels; lists with more than three nested levels must be rewritten to reduce the number of levels. Furthermore:

- a) The *Body* paragraph style with a first-level indent also generated this auto numbered *List1* list.
- b) The *Body* paragraph style with a first-level indent also generated this auto numbered *List1* list.
- c) The *Body* paragraph style with a first-level indent also generated this auto numbered *List1* list.
 - 1) The *Body* paragraph style with two manual indents generated this 2nd-level *List2* list.
 - i) The *Body* paragraph style with three manual indents generated this 3rd-level list.
 - ii) The *Body* paragraph style with three manual indents also generated this 3rd-level list.

You can insert an additional non-numbered paragraphs in a 3rd-level list. To do this, choose the *List3More* style for the non-numbered paragraph, as was done here.

- 2) The *Body* paragraph style with two manual indents generated this 2nd-level *List2* list.

You can insert an additional non-numbered paragraphs in a 2nd-level list. To do this, choose the *List2More* style for the non-numbered paragraph, as was done here.

- d) The *Body* paragraph style with a first-level indent also generated this auto numbered *List1* list.

You can insert an additional non-numbered paragraphs in a 1st-level list. To do this, choose the *List1More* style for the non-numbered paragraph, as was done here.

9.4.2 Dashed lists

When listing a small number of items, you may prefer a dashed list. For example, a simple set of objectives could be placed in a dashed list, illustrated below:

- The *List1Dash* paragraph style was used to generate this dashed first-level list.
- The *List1Dash* paragraph style was also used to generate this dashed first-level list.
 - A *List2Dash* paragraph style indented once generated this dashed second-level list.
 - A *List2Dash* paragraph style indented once generated this dashed second-level list.
 - A *List3Dash* paragraph style indented twice generated this dashed third-level list.
 - A *List3Dash* paragraph style indented twice also generated this dashed third-level list.

Note that the same indentation spacing is used for *List1&List1Dash*, *List2&List2Dash*, *List3&List3Dash* paragraph styles, but the dash versions use the *em* dash, instead of numerated values. The ISO/IEC specification doesn't allow the use of bulleted lists, so dashes are used at all levels.

9.5 Tables

Tables are intended to summarize the use of multiple items, not to describe them in detail. A row-reference column can be added to elaborate specific rows of the table. The following subclauses describe the methods used to create tables.

9.5.1 Tables cells

You should insert tables by cut-and-pasting a previously created table, since the author does not yet understand how to specify table formats. For example, you could start with a three-columns and three-rows table of Table 9.1.

Table 9.1—Example table

Heading A	Heading B	Description
ValueA	NameA	DescriptionA
ValueB	NameB	DescriptionB
ValueC	NameC	DescriptionC

Use the **Table** menu to modify the number of rows and columns, or to straddle rows/columns as desired.

The *TableTitle* style was applied to the caption above the table, although the caption itself is called *Table*.

The *TableHeading* style is used on the top row, which sets the style as {bold and centered}. The *TableCellCenter* style is used to center-justify table entries, as illustrated in the left columns. The *TableCellLeft* style is used to left-justify table entries, as illustrated in the right column.

9.5.2 Tables row labels

Tables should contain small items, not paragraphs of explanatory text. A row-number cross-reference can be used to cross-reference a row with detailed following comments, as illustrated in Table 9.2. The *TableCellRow* style is used to establish the row count within the Row headed column. **Be sure to include a space in each of these cells**, or the cross-referencing will not function correctly.

Table 9.2—Row-numbered table

Heading A	Heading B	Row	Description
ValueA	NameA	1	DescriptionA
ValueB	NameB	2	DescriptionB
ValueC	NameC	3	DescriptionC

Row 1: A detailed description of a row can be placed after the table. This has the benefit of maintaining a concise table, while allowing each row to be described in detail.

Row 2, Row 3: Multiple rows can be described together, by placing both cross-referenced before the combined description, as was done here.

The first part of the row cross reference is attached to a *TableTitle* style; the second portion is attached to the *TableCellRow* entries within that table. In both cases, the reference component is created using the **Insert|Cross-reference** command with the *Numbered item* reference type, then searching for the desired number-paragraph value.

9.5.3 Table headings

If the table exceeds one page, you can have the heading repeat on all pages. To do this,

- a) Select the heading you want to repeat on all pages.
- b) Select the **Headings** menu item from the **Table** menu.

9.5.4 Table footnotes

Do *not* use word's automatic footnote feature for creating table footnotes. This feature is for creating document footnotes and endnotes only. Footnotes are manually placed within a figure, as illustrated in Table 9.3.

Table 9.3—Table footnotes

Heading A	Heading B	Description
ValueA	NameA	DescriptionA ¹
ValueB	NameB	DescriptionB
ValueC	NameC	DescriptionC

¹This is an example of a table footnote.

To set a table footnote, do the following:

- a) Type the footnote letter next to the text where you want to refer to the footnote. Then adjust by: Select the letter; go to **Format|Font**; select the **Superscript** option; click **OK**.

- b) With the letter still selected, bookmark it with a unique bookmark name by selecting **Insert|Bookmark**, click **Add**, and name it something like *TableFootnote1*.
- c) Go to where the table footnote should be located and add a cross-reference to the location of the footnote by selecting **Insert|Cross-reference**, selecting **Bookmark**, and choosing the bookmark you just created. Type in the table footnote.
- d) If the table is centered, you will need to move the footnote indent (the small square below the triangle in the ruler when the footnote text is selected) so that the beginning of the paragraph aligns with the left line of the table. Also, the right indent will have to move so that it is lined up with the right of the table.

Note that the style in the last row of a table (and all rows for that matter) defaults with the **Keep with next** setting. This ensures that any table footnotes will remain with the table. If you don't have table footnotes, and you do not want the table to keep with the next paragraph, be sure to remove this setting. To do this,

- a) Select the last row.
- b) Choose **Format|Paragraph|Line** and page breaks tab.
- c) Clear Keep with next.
- d) Click **OK**.

9.5.5 Common keyboard commands for working with tables

Useful techniques for editing tables are summarized in Table 9.4.

Table 9.4—Instructions for editing word tables

Intended action	Keyboard action
To insert a tab within a cell	Press Ctrl+Tab
To add a new row	On the last row and last column, press Tab.
To select a row	Move the cursor to the left of the row, and click.
To delete a row	Select the row, and press Shift+Delete.
To center an entire table	Alt+A+W+tab+tab+tab+T+Enter

9.6 Equations

An equation involves the use of the *EquationMath* format, as illustrated in Equation 9.1. The equation may be left-justified, indented, or centered based on the number of left-side tabs. A rightside autonumber specifier provides the right-side equation number, not the paragraph style itself.

$$v = 2^7 \left(-b_0 + \sum_{i=1}^l b_i 2^{-i} \right) \quad (9.1)$$

An unnumbered equation involves the use of the same *EquationMathMore* format, but without the rightside autonumber specifier, as illustrated below.

$$v = 2^7 \left(-b_0 + \sum_{i=1}^l b_i 2^{-i} \right)$$

9.7 C-code listings

9.7.1 Shorter in-line C code

A few lines of C code can be expressed as an equation, as illustrated by Equation 9.2. The *EquationCode* style is used on the first line, to specify the proper C-Code font and provide the numerical label. The *EquationCodeMore*

```

crcCheck= CrcStep32(crcValue);
if (crcCheck!=crcValid)
    error|= 1;
crcCheck= CrcStep32(crcValue);

```

(9.2)

9.7.2 Lengthy C code

Lengthy C code (which is typically defined as code listings covering more than ½ page) should be placed in an annex, such as Annex J. An annex can support a wider 132-character width while allowing the code to be more easily extracted by the reader.

9.8 Footnotes

Footnotes should be numbered sequentially using Word's automatic footnote feature⁵. Footnotes can also be applied to tables (see 9.5.4).

9.9 Auto numbering cross references

To enter cross-references, use the **Insert|Cross-references** command. This command allows you to have dynamic cross-references that change if you move source text. Usually, cross-references use the heading number, and sometimes the clause text.

⁵ This is an example of a footnote.

9.10 Autonumbered styles

Several autonumbered styles are utilized by these formats, although their usage is expected to be transparent to the user. For the convenience of the reader and future style editors, these autonumbered formats are summarized below.

- *BodyParts*: Assigns numbers to each 1st, 2nd, and 3rd level list.
The 1st level list is cleared by a previous *Body* style.
The 2nd and 3rd level list numbers are cleared at the start of each 1st level list.
- *Headings*: Assigns numbers to each clause and subclause, as well as each figure, table, and equation within a clause.
The figures, tables, and equations are cleared at the start of each clause.
The subclause numbers are cleared at the start of each higher-level clause/subclause.
- *HeadingsA*: Assigns numbers to each annex; also each subannex, figure, table, and equation within an annex.
The figures, tables, and equations are cleared at the start of each annex.
The subannex numbers are cleared at the start of each higher-level annex/subannex.
- *ListDash*: Prepends a dash to each 1st, 2nd, and 3rd level list.
- *TableRowCount*: Assigns distinct sequential numbers to each *TableCellRow* paragraph.
The *TableHeading* format within the header of each table clears this number count.

10. Constructing Visio figures

10.1 Figures

The following subclauses describe the creation and/or development of figures using Microsoft Visio. This is the preferred drawing package, since conversions to FrameMaker are expected in the near future.

10.1.1 Microsoft Visio drawings

We mandate the use of Microsoft Visio Professional 5.0 for the creation of figures. Microsoft Visio has a smoother and more elegant interface for creating graphics than is supplied by Microsoft Word’s graphics sub-package.

The simplest way to create a new figure is to cut-and-paste an existing figure, picking a figure that has similar graphics elements. If the original figure is not grouped, you will need to use the arrow selection tool to select all drawing elements before copying. Figures with useful register templates are provided in Figure 10.1 and Figure 10.3.



Figure 10.1—Pointer illustration

The *FigureTitle* style was applied to the line below the figure. The autonumbering is provided by utilizing the caption capability, rather than autonumbered paragraph styles. Thus, selection of the header, right click, and *Toggle Field Codes* selection would yield:

Figure { STYLEREF 1 \s } {SEQ Figure * ARABIC \s 1 }—Pointer illustration

Since copies often lose the shading details of the original, shaded colors should be limited to three: white, light gray, and black. When black shading is used, the text color should be white. As examples, in Figure 10.1, the *res* field is shaded grey.

10.1.2 Maintaining the graphics grid

The reference point within a Microsoft Visio drawing, is the bottom-left corner. To maintain the snap-grid, this reference point should never be changed. Thus, expansion (or contraction, not shown) of a Microsoft Visio drawing should be made by using a click-and-slide motion on the top handle, the right handle, or the top-right handle only, as illustrated in Figure 10.2.

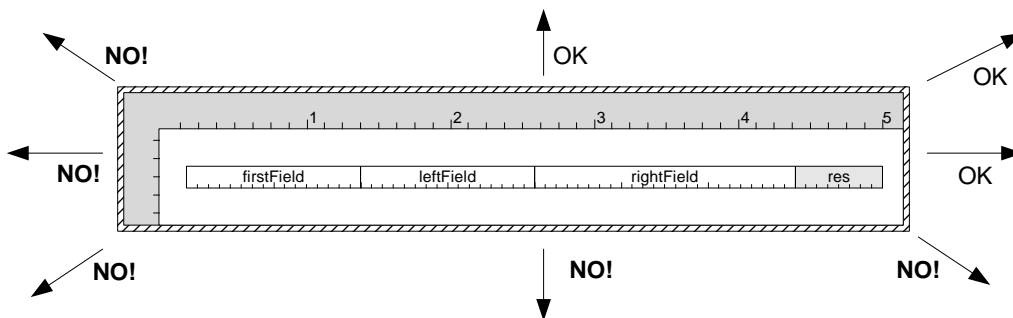


Figure 10.2—Microsoft Visio expansion and contraction restrictions

10.1.3 Register layouts

The location of fields within registers is specified by the cumulative widths of fields within the register, as illustrated in Figure 10.3. The width of each field (in bits) is implied by bottom-line tick marks; the field name is normally contained within its bounding rectangle.

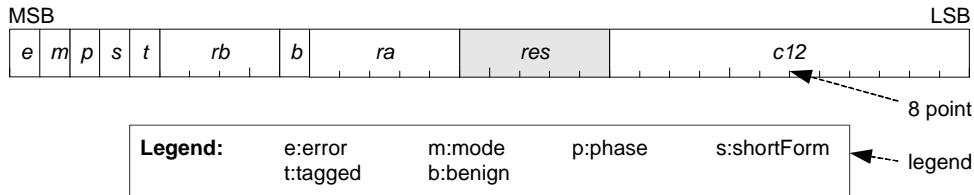


Figure 10.3—Expanded bit-field descriptions ns

Since copies often lose the shading details of the original, shaded colors should be limited to three: white, grey, and black. When black shading is used, the text color should be white.

NOTE—The fields of Figure 10.3 were planned to illustrate several of the larger byte-boundary tick marks. Depending on the field alignments, these are oftentimes hidden by the field's bounding box.

Each field name that would extend beyond its bounding box is abbreviated; a legend associates the local abbreviation with the formal field name. The legend entries are ordered, in that a left-to-right scan of the fields corresponds to a left-to-right-then-top-to-bottom scan of the legend entries. A space should be included after the colon. The dotted line around the legend is a distinct graphics rectangle, which can be sized appropriately.

10.1.4 Flow-chart styles

A distinct *FigureTextCenter* style is used within flow chart decision components, as illustrated in Figure 10.4. This Arial 8-point centered style is typically applied to text within conditional-branch components.

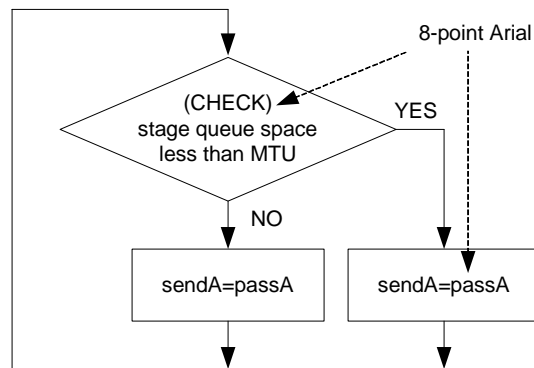


Figure 10.4—Flow-control *sendA* generation overview

Annexes

Annex A (informative) Bibliography

The annex always starts with the Annex style followed by the ~Headings1 style. The first annex is always the bibliography, which lists informational references that provide useful background information for understanding the specification.

The most recent editions of the following texts are recommended by the IEEE as guides on points of editorial style and usage:

- [B1] *IEC Multilingual Dictionary of Electricity, Electronics, and Telecommunications*, Amsterdam: Elsevier Science Publishers..
- [B2] *IEEE-SA Standards Board Bylaws*, New York: Institute of Electrical and Electronics Engineers, Inc.
- [B3] *IEEE-SA Standards Companion*, New York: Institute of Electrical and Electronics Engineers, Inc.
- [B4] *IEEE-SA Standards Operations Manual*, New York: Institute of Electrical and Electronics Engineers, Inc
- [B5] *IEEE100, The Authoritative Dictionary of IEEE Standards Terms*, Seventh Edition, New York, Institute of Electrical and Electronics Engineers, Inc.
- [B6] Miller, C., and Swift, K. *The Handbook of Nonsexist Writing*. New York: Harper Collins.
- [B7] *Webster's New Collegiate Dictionary*. Springfield, MA: Merriam-Webster, Inc.

A.1 Bibliography references

The following is a list of on-line locations that are useful resources for creating a bibliography that contains multiple types of references. This list is by no means comprehensive, but it will serve to answer most questions about the creation and development of bibliographies, and will in some cases provide pointers to other sites of interest.

- [B8] <http://www.spaceless.com/WWWVL/>⁶
- [B9] <http://www.lmu.ac.uk/lss/lss/docs/Harvard/bib.htm>⁷
- [B10] <http://www.lmu.ac.uk/lss/lss/docs/Harvard/types.htm>⁸
- [B11] <http://standards.ieee.org/catalog/olis/index.html>⁹

⁶ The Electronic References & Scholarly Citations of Internet Sources

⁷ The Harvard Style of Referencing, "Cite References in a Bibliography"

⁸ The Harvard Style of Referencing, "Source Types and Examples"

⁹ The Institute of Electrical and Electronics Engineers Standards Online.

A.2 Grammar references

The following is a list of resources that provide information and pointers for further research into the intricacies of the English language. Most common questions may be answered in the FAQ sections of these online sites; the more esoteric points of interest may be resolved with only a little looking around.

[B12] <http://ccc.commnet.edu/grammar/>¹⁰

[B13] <http://www.ohiou.edu/esl/english/index.html>¹¹

[B14] <http://englishplus.com/grammar/>¹²

[B15] <http://www.edunet.com/english/grammar/index.cfm>¹³

¹⁰Guide to Grammar and Writing

¹¹Ohio State University English as a Second Language, "Resources for English Language and Culture"

¹²English Grammar Slammer (n.b.: although this site is for a particular product, it provides a great deal of information and help in the area of English language and grammar)

¹³Education Net: Language, "The Online English Grammar."

Annex B (informative) Annex styles

An annex begins with the *Annex 1* paragraph style, which serves as the header for the annex.

This annex heading format differs from the traditional annex format by concatenating the traditional three headings into one line. This is done to make table-of-contents (TOC) generation simple. The Annex 1 heading will appear in the table-of-contents page as it appears above. It should be structured as follows:

ANNEX *annex number: Annex title* (informative|normative)

As is true for clauses, five heading styles are supported in annexes – *Annex 1*, *Annex 2*, *Annex 3*, *Annex 4*, and *Annex 5*. Only *Annex 1*, *Annex 2*, and *Annex 3* appear in the table of contents.

B.1 Subannex-2 header

The previous heading applied the *~Headings2* paragraph style to the *Subannex-2 header* text. The *~Headings2* heading forces before and after spacing, but no page break.

B.1.1 Subannex-3 header

The previous heading applied the *~Headings3* paragraph style to the *Subannex-3 header* text. The *~Headings3* heading forces before and after spacing, but no page break.

B.1.1.1 Subannex-4 header

The previous heading applied the *~Headings4* paragraph style to the *Subannex-4 header* text. The *~Headings4* heading forces before and after spacing, but no page break.

B.1.1.1.1 Subannex-5 header

The previous heading applied the *~Headings5* paragraph style to the *Subannex-5 header* text. The *~Headings5* heading forces before and after spacing, but no page break.

This is the deepest level allowed by the IEEE! Thus, you should revise any IEEE document that requires to the use of any level-6 or lower levels. This formatting style is intended to force conformance to good document-style conventions, rather than providing the author with an unlimited range of ill-conceived nesting depths.

B.1.1.1.1.1 Subannex-6 heading

The previous *Subannex-6 heading* text was created using the *~Headings6* paragraph style and typing the text *Subannex-6 heading*. The *~Headings6* heading is configured to force before and after spacing, but no page break.

Note this is the deepest level that is supported by these formats! Thus, you should revise any document that requires to the use of any level-7 or lower levels. This formatting constraint results from having 9 number levels, with three of these reserved for figure-title, table-title, and equation-title purposes.

B.1.1.1.1.1 Subannex-7 heading

The previous *Subannex-7 heading* text was created using the *~Headings7* paragraph style and typing the text *Subannex-7 heading*. The *~Headings7* heading is configured to force before and after spacing, but no page break.

Note this is the deepest level that is supported by these formats! Thus, you should revise any document that requires the use of any level-7 or lower levels. This formatting constraint results from having 9 number levels, with three of these reserved for figure-title, table-title, and equation-title purposes.

B.2 Definitions

Paragraphs of definitions may be distinctively numbered to facilitate their cross referencing by other parts of a document. This should only be done at the lowest level, to avoid discontinuities in the heading numbering, as has occurred with the numbering of 7.1.

B.2.1 Leading third-level heading

B.2.1.1 Definition5 heading

Within fourth-level subclauses, a fifth-level field distinctively numbers paragraphs, as illustrated in B.2.1.1.1 and B.2.1.1.2.

B.2.1.1.1 definedValue5a: The *Definition5* formats were applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

B.2.1.1.2 definedValue5b: The *Definition5Like* formats were applied to this paragraph. This definition would not be included in the IEEE dictionary.

B.2.2 Definition4 heading

Within third-level subclauses, a fourth-level field distinctively numbers paragraphs, as illustrated in B.2.2.1 and B.2.2.2.

B.2.2.1 definedValue4a: The *Definition4* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

B.2.2.2 definedValue4b: The *Definition4Like* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

B.3 Definition3 headings

Within second-level subclauses, a third-level field distinctively numbers paragraphs, as illustrated in B.3.1 and B.3.2.

B.3.1 definedValue3a: The *~Definition3* formats were applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

B.3.2 definedValue3b: The *~Definition3Like* formats were applied to this paragraph. This definition would not be included in the IEEE dictionary.

Annex C

(informative)

Highest level definitions

Within annexes, a second-level field distinctively numbers paragraphs, as illustrated in C.1 and C.2.

C.1 definedValue1a: The *~Definition2* paragraph style was applied to this paragraph. This definition would be automatically included in the IEEE dictionary.

C.2 definedValue1b: The *~Definition2Like* paragraph style was applied to this paragraph. This definition would not be included in the IEEE dictionary.

Annex D (informative) Numbered annex styles

D.1 Tables and figures

D.1.1 Annex figures

Within the annex, a distinct figure autonumber coding is used, as illustrated in Figure D.1. The distinct style allows the coupling of figure and annex heading styles. Although the caption coding is different, cross-referencing is done in a similar fashion.

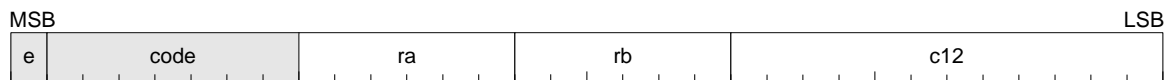


Figure D.1—Instruction illustration

D.1.2 Annex tables

Within the annex, a distinct table autonumber coding is used, as illustrated in Table 3.1. Although the caption coding is different, cross-referencing is done in a similar fashion.

Table D.1—Names of command, status, and CSR values

Name	Row	Description
<i>MoverCsr.control</i>	1	The mover's control register.
<i>Command.code</i>	2	The code field within a command entry.
<i>Status.count</i>	3	The count field within a status entry

The caption numbering rules also apply to code tables and equations (not illustrated).

Row 1: An example of how to describe the first row.

Row 2: An example of how to describe the second row.

Row 3: An example of how to describe the final row.

D.1.3 Equation numbering difference

Within the annex, a distinct equation autonumber coding is used, as illustrated in Equation D.1. The distinct style allows the coupling of equation and annex heading styles. Although the caption name is different, cross referencing is done in a similar fashion.

$$v = 2^7 \left(-b_0 + \sum_{i=1}^l b_i 2^{-i} \right) \quad (\text{D.1})$$

Annex E

(informative)

Common grammatical issues

This annex is designed to point out and clarify common grammatical issues that affect many writers of English that have English as a secondary language. This section is presently small, but will grow over time to support the needs of specification writers.

E.1 Use of the word “the”

The word “the” is often misplaced in English sentences. Though there about 11 rules for using this word, only seven of them are directly applicable to writing our specifications.

- a) Showing that something or someone has already been spoken or written about, or is already known to the reader.
“The function block shown above shows...”
not “Function block above shows...”
- b) Used with things because they are well known.
“The 61883-1 document states...”
not “61883-1 document states...”
- c) Showing that something is the only one of its kind.
“The READ DESCRIPTOR command reads data from the Subunit Identifier Descriptor”
not “READ DESCRIPTOR command reads the data from Subunit Identifier Descriptor”
- d) Showing the reader that the writer means one particular thing.
“The printer shall respond to the command by...”
not “Printer shall respond to command by...”
- e) Used for particular sets of things.
“All of the subunit’s descriptor length fields shall be...”
not “All of subunit’s descriptor length fields shall be...”
- f) with comparison
“The larger the filesize, the longer it takes to download.”
not “With larger of filesize, longer it takes to download.”
- g) with superlative
“This shall be the largest data structure.”
not “This shall be largest data structure.”

E.2 Use of the word “a” or “an”

The word *a* or *an* when preceding a word that starts with a vowel) is also often misplaced and misused in English sentences. Though there about 6 rules for using this word, only several of them are really applicable to writing our specifications.

- a) One among many
“When the button is pressed, a new picture is taken.”
not “When button is pressed, the new picture is taken.”
- b) One in particular
“The output is a color picture”
not “The output is color picture”

- c) General one
“The target sends an FTP packet”
not “Target sends FTP packet”

E.3 Numbers: numerical or alphabetical?

In general, any number lower than ten shall be written alphabetically, unless it is used in an equation or in a sequence. As examples:

- Command A can process ten pictures. Command B can process 100.
- The range of values shall be between 5 – 15.

E.4 Table cell justification

Table cells are sometimes centered and sometimes left justified, as illustrated in Table E.1.

Table E.1—Example table

Value	Name	Description
0	TEST_START	Start of the built-in self test
1	TEST_STOP	Stops the command-invoked self test at the next subtest completion
2-15	TEST_ABORT	Aborts the command-invoked self test immediately

Text alignment in table cells shall have the following rules:

- a) If the text is a sentence, or resembles a sentence, then the text is left-justified.
- b) If the text is not a sentence (e.g. it is a phrase of one or two words), then the text is centered.

E.5 That and which

(Extracted from [R1] IEEE Standards Style Manual)

The words *that* and *which* are commonly misused; they are not interchangeable. *That* is best reserved in essential (or restrictive) clauses, *which* is appropriate in nonessential (or nonrestrictive), parenthetical clauses. Simply stated, if a comma can be inserted before the word *that* or *which*, the word should be *which*. If a comma would not be used, the word to use is *that*.

Examples:

- a) Defining the inputs and outputs provides a better understanding of the steps *that* are necessary to complete the process.
- b) Defining the inputs and outputs provides a better understanding of these steps, *which* are explained in 5.1 through 5.9.

E.6 Gender-Neutral language

(Extracted from [R1] IEEE Standards Style Manual)

In order to reflect the changing practices in language usage, the IEEE Standards Department uses, in as many cases as possible, generic titles (such as chair rather than chairman) in the body of the standard. The following practices shall apply:

- a) When writing in the third person, the phrase *he or she* shall be used. The male or female pronoun alone or the variations *he/she* or *s/he* shall not be used. Also, the pronoun *they* shall not be used as a singular pronoun.
- b) If a particular sentence becomes cumbersome when *he or she* is used, the sentence should be rewritten in the plural or completely rewritten to avoid using pronouns. The indefinite pronoun *one* should be avoided. In references to a company, the pronoun *it*, not *we* or *they*, should be used.

E.7 Use of the second-person form of address

(Extracted from [R1] IEEE Standards Style Manual)

The second-person form of address (*you*) should not be used or implied in standards, e.g., *You should avoid working on lines from which a shock or slip will tend to bring your body toward exposed wires.* This should be rewritten to identify the addressee, as follows: *Employees should avoid working on lines from which a shock or slip will tend to bring their bodies toward exposed wires.*

Annex F (informative) General writing guidelines

F.1 Capitalization

Left by themselves, engineers seem to Capitalize Anything that could possibly be important, including Acronyms, Application-Specific Terms, and Variable names, as illustrated in this sentence. The intent is oftentimes to distinguish between normal English and words with specialized meaning. The effect, however, is that excessive (and oftentimes inconsistent) use of capitalization complicates parsing of the sentence, leading to loss of clarity and increased numbers of technical ambiguities.

F.1.1 When to capitalize

In general, capitalization is only necessary for proper nouns and the first word of headings: other uses are strongly discouraged. Alternative techniques are available to delineate words that have standard-specific requirements, as follows:

- a) Variables can be easily distinguished from normal English words by any of the following naming conventions:
 - 1) A command register or *startTest* field can be distinguished from English words by using an italics style, with a capital at the start of each following run-together word.
 - 2) An *ExecutionRoutine*() or group of *CommonControl* registers can be distinguished by capitalizing the first run-together word.
 - 3) A `source_address` parameter can be distinguished by the underscore that separates run together words.

Most importantly, a variable or field name should not be split into separate words separated by spaces and tied together by the common use of Capital Letters.

- b) Constants can be easily distinguished from normal English words by using all capitals within the name. Such conventions normally apply to a RESET constant or an enumerated START_FAST value.
- c) Names with special meaning, such as ringlet or port, can be defined in an early *Terms and definitions* clause, and therefore need not be capitalized when used within the text.

Many standards have incorrectly used capitals in an attempt to delineate the boundaries of a special variable name, such as a Negotiation Control Register. However, the following questions must be addressed for the sentence to be correctly parsed, or the document correctly searched for alternate descriptions:

- a) Does this refer to a specific *negotiationControl* instance of a register?
- b) Does this refer to a specific *negotiationControlRegister* location?
- c) Does this refer to generic negotiation-control registers?
- d) Does this refer to a negotiation register called *control*?
- e) Does this refer to a control register called *negotiation*?

While these distinctions may be moot to the editor or familiar readers, they severely restrict document comprehension by unfamiliar engineers and/or non-native English speakers. Do not do this!

F.1.2 Heading capitalization

The first word of a clause, subclause, table, table-heading, figure, or figure insert is nominally capitalized, such as:

P.1.2 Shared control and status registers (CSRs)

An exception is when the heading starts with a special variable name, such as:

P.1.2 *command* register

Only proper nouns, such as California, are capitalized within the other words of a clause, subclause, table, table-heading, figure, or figure insert. Adjacent words are not necessarily capitalized, as in:

P.1.2 Habits of California wildlife

F.1.3 Noncapitalized definitions

Terms within a glossary or definitions of terms are not capitalized, unless these are proper nouns. As examples:

3.2.29 bridge: A functional unit interconnecting two or more networks on ...

3.2.30 congestion domain: The set of contiguous links associated ...

3.2.31 cyclic redundancy check (CRC): A specific type ...

3.2.32 loop round trip time (LRTT): The time that it takes for a control frame...

3.2.33 medium access control (MAC) sublayer: The portion of the data link layer...

F.1.4 Noncapitalized acronyms

Words within acronyms are not capitalized, unless these are proper nouns. As examples:

CRC	cyclic redundancy check
FEC	forward error correction
MSB	most significant bit
LSB	least significant bit
SF	signal fail
SDH	synchronous digital hierarchy
WAN	wide area network
WTR	wait to restore
USA	United States of America

F.1.5 Improper nouns

The term improper noun describes a noun that is not a proper noun, but is (typically for historical or confusion reasons) capitalized like a proper noun. To reduce improper noun usage, consider the following when classifying your proper nouns.

- Is this a particular person, place, or thing? The United States of America and California are particular places or names; the distinct counties of California are not.
- Does this avoid confusion that cannot be solved by the appropriate standard-specific definition?
- Is this an acronym? By itself, this is insufficient justification for capitalization (see F.1.4). The capitalization of a cyclic redundance check is unaffected by its CRC acronym abbreviation.
- Is this a title? Only the first word in a title is normally capitalized, so this is insufficient (see F.1.2).

- e) Is this a definition? A defined term is not necessarily a proper noun (see F.1.2).

Note that many standards have generated improper nouns due to improper interpretations of (c,d,e) usage. You may have to sometimes cope with such conventions, but don't propagate them.

F.2 Definitions

F.2.1 Definition clauses

From the style manual: Acronyms shall not be defined in the definitions clause. Instead, a following clause should be used. However, standards commonly place related content within a definitions clause, expanding the header text as follows:

3. Terms, definitions, and notation

F.2.2 Definition text

From the style manual: The terms should not be used in its own definition. Correct usage is as follows:

4.1.2 doublet: Two bytes.

Incorrect usage is as follows:

4.1.2 doublet: A doublet is two bytes of data.

F.3 Variable names

F.3.1 Spaceless variable names

The preceding (see F.1) capitalization conventions are consistent with C programming conventions. Descriptive code can be easily used to specify formal definitions, as illustrated by Equation F.1.

```
errorCount += (protocolErrors + incomingCount.crcBad + clientCount.typeBad); (F.1)
```

This is clear and concise, when compared to the pseudo-defined pseudo code of Equation F.2. The lack of extra spaces also reduces the equation lengths, so more of the equations will be able to fit on one line. The use of C-type conventions, to name subfields by their function (as opposed to position) adds to clarity, while simplifying text maintenance by defining bit-field positions in only one place.

```
(Error Count) ← (F.2)  
((Protocol Errors) + (Incoming Count)[15:14] + (Client Count)[12:11])
```

F.3.2 Hyphenless variable names

Using run-together hyphenless variable names also has the advantage that descriptive code can be easily used to specify formal definitions, as illustrated by Equation F.3.

```
errorCount += (protocolErrors + incomingCount.crcBad + clientCount.typeBad); (F.3)
```

For example, a hyphenated name could have the subtraction-operator ambiguity associated with Equation F.4.

```
errorCount += (protocol-errors + incomingCount.crcBad + clientCount.typeBad); (F.4)
```

F.3.3 Aliased variable names

Never reference the same variable using two separate names, except within the context of a figure legend (see F.8). As an example, the interchangeable use of *DA* and *Destination Address* terms raises the following concerns:

- a) The term *DA* could refer to either *destination address* (a generic term) or *Destination Address* (a specific field within frames).
- b) If the *Destination Address* is placed in the transmitted frame, and the *DA* is checked in the received frame, the reader will be confused.
- c) An acronym-based abbreviation could easily be confused with a constant, such as PASS or FAIL.

F.4 Spelling checker

Always run a spelling checker on your text **before** submitting to Sponsor Ballot. To simplify spelling-checker and/or character-pattern searches, the following guidelines have been found to be useful:

- a) Ellipsis. The special ellipsis character ‘...’ should be used, not three consecutive periods. This eliminates spurious duplicate-period spell-checker reports.
- b) Spaces. Only one space should be used between words and sentences. This allows a search to readily find duplicate-space errors.
- c) Styles. Special character styles are appropriate for special names, variable names, and field names. For example, styles used within this paper have the following properties:
 - 1) The *NamePlain* style is applied to special names (e.g., StateMachineName) with the effect of inhibiting end-of-line hyphenation and spurious spelling-checker reports.
 - 2) The *NameItalic* style is applied to field and variable names (e.g., *thisValue.thatField*) to force an italic style, while inhibiting end-of-line hyphenation and spurious spelling-checker reports.

F.5 Cross references

Cross references should be clear and concise, with the following guidelines:

- a) The term subclause, not section, is used to describe distinctively numbered portions of a clause.
- b) A proper cross reference, such as “(see F.4)”, does not include the word “subclause” or “section”.
- c) The simple “(see F.4)” text is usually preferred to using multiple or complex alternatives.
 - “, as considered in F.4.”
 - “, as constrained by F.4.”
 - “, as described in F.4.”
 - “, as discussed in F.4.”
 - “, as illustrated in F.4.”
 - “, as shown in F.4.”
 - “, as specified in F.4.”
 - “, as standardized in F.4.”
 - “, as mandated by F.4.”
- d) Never type the numbers within “(see 4.4)”. Explicit cross-reference markers are preferred, because:
 - 1) Numbers are automatically updated as subclauses are inserted, deleted, or removed.
 - 2) FrameMaker, Word, and OpenOffice can generate valuable hyperlinks from marked values.

F.6 Indentation

Most textual styles have tabs set at regular 18-point intervals, allowing standard styles to be used for special purposes, such as the service primitive listing below:

```
MA_DATA.request
(
    destination_address,
    service_class,
    mac_protection          // optional
)
```

Older standards have attempted to space the arguments further to the right, as illustrated below. The original intent was to mimic a distinctive hierarchical C-coding style, with a first-heading dependent indentation. Unfortunately, the correct tab distance is difficult to maintain (without per-instance customization of tab settings), and the inconsistency of tab'd data is distracting to the reader. This usage should be deprecated.

```
MA_DATA.request (
    destination_address,
    service_class,
    mac_protection          // optional
)
```

F.7 Table cell justification

Table cells are sometimes centered and sometimes left justified, as illustrated in Table F.1.

Table F.1—Example table

Value	Name	Description
0	TEST_START	Start of the built-in self test.
1	TEST_STOP	Stops the command-invoked self test after the next subtest completion.
2-3	TEST_ABORT	Aborts the command-invoked self test immediately.

Text alignment in table cells is based on the following rules. There is flexibility in the application of these rules, in that long variable names could be classified as either (a) or (b). However, the same justification rules should be applied to all entries within the same column.

- a) If the text is a sentence, or resembles a sentence, then the text is left-justified.
- b) If the text is not a sentence (e.g. it is a phrase of one or two words), then the text is centered.

F.8 Multi-field registers

Numbering of bits within registers causes numerous problems, due to distinct bit numbering conventions adopted by little-endian and big-endian designers. Bit ordering arguments are not easily resolved by using the bit-transmission order, which can be PHY dependent or ambiguous. Arguments and confusions are best resolved by avoiding the numbering altogether, which also eliminates the clutter of numerical values within each illustration.

The location of fields within registers is specified by the cumulative widths of fields within the register, as illustrated in Figure F.1. The width of each field (in bits) is implied by bottom-line tick marks; the field name is normally contained within its bounding rectangle.

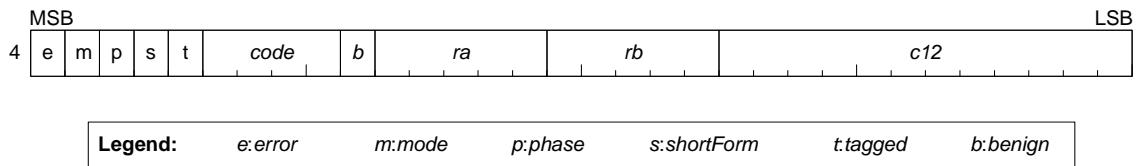


Figure F.1—Expanded bit-field descriptions

The legend facilitates the use of abbreviated names within figures, so that the length of names can be reduced to meet bounding-box size requirements. The legend applies only to the figure; full names (not their abbreviations) should be used outside of the context of the figure. Two forms of variable names (the full name and the abbreviation) would simply confuse the reader with minimal benefits.

If this were to be a defined *control* register, its field contents can still be clearly specified by referring to *control.error* bit or *control.code* field values.

F.9 Punctuation within lists

A variety of punctuation conventions have been used within lists. Rather than propagate such diversity, the following recommendations are provided:

- a) Lists described with a preceding *as follows* or *following* text end with a colon (see above).
- b) Lists or sublists of sentences or sentence-like text are punctuated as sentences.
 - 1) These start with a capital letter.
 - 2) These end with a period.
- c) Lists or sublists of items or values are capitalized as within normal sentences, as listed below.
 - 1) error counts
 - 2) consistency checks
 - 3) CRC
 - 4) books
 - 5) *errorCount* fields
 - 6) United States of America

Variable lists have similar properties, as illustrated below. The text after the em-dash is either capitalized and punctuated as a sentence (*automatic* and *sendReady* items), or listed as isolated words (*rxFilter* items). No spaces should be included on either side of the em dash.

automatic

Indicates how the frame should be transmitted.

TRUE—Transmit in the MAC-specified direction(s).

FALSE—Transmit in the client-specified direction.

sendReady

Indicates whether the sender is ready to accept another data frame for transmission.

TRUE—The sender is ready.

FALSE—(Otherwise.)

rxFilter

The type of filtering done when receiving frames destined for the client.

BASIC—host clients

FLOOD—bridge clients

For the *automatic* control signal, the actions triggered by TRUE and FALSE conditions are both described. For the *sendReady* status signals, the TRUE condition suffices; there is no need (or value) in defining the complementary FALSE condition, so consistently abbreviated (*Otherwise.*) text is provided.

Annex G (informative) RPR writing examples

The following guidelines have been found useful within the IEEE P802.17 Resilient packet ring (RPR) Working Group. Other groups are welcome to utilize and or all of these documentation conventions, based on which ones are found to meet their requirements.

G.1 Field formats within figures

G.1.1 Frame format illustrations

Figure G.1 is an example of how frame formats should be illustrated (in this context, frame connotes the media-independent portion of a transmitted packet). Names and numbers use an 8-point Arial font (as is true for all figures). Each name is horizontally centered in its box, byte numbers are right justified, and header/payload/trailer names are left justified. The byte-size and field names are vertically centered in their box. Similarly, the header/payload/trailer names are vertically centered with their spanning line.

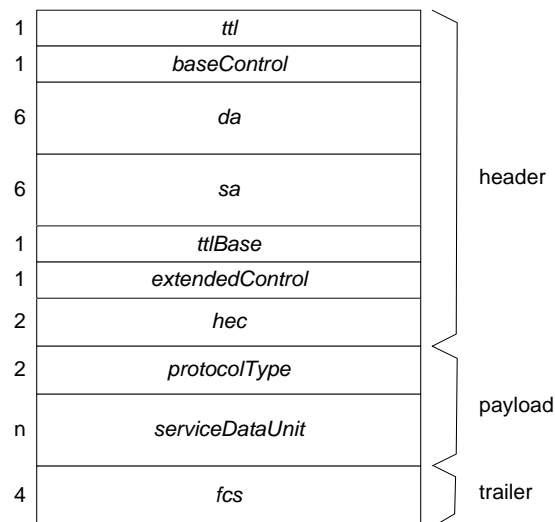


Figure G.1—Frame format illustrations

The fields are stacked vertically and have adjacent byte-size labels. The size of boxes should vary monotonically with its byte size: if $\text{Sizeof}(\text{fieldA}) > \text{Sizeof}(\text{fieldB})$ then $\text{Height}(\text{fieldA}) \geq \text{Height}(\text{fieldB})$.

G.1.2 Generic field illustrations

Figure G.2 illustrates how subfield components of multibyte fields should be produced. For clarity, short and longer tick marks are used to delineate bit and byte positions respectively.

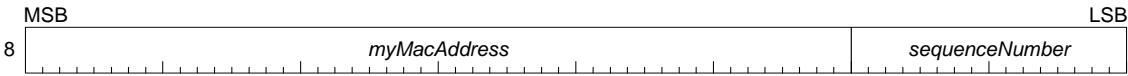


Figure G.2—Example of a bit-field illustration

The field names, byte-size numbers, and MSB/LSB labels use an 8-point Arial font. Each field name is horizontally centered in its box; the byte number and LSB label are right justified; the MSB label is left justified. The byte-size and field names are vertically centered in their box, then moved upward one point.

Depending on the width of field illustrations, the spacing between tick marks may be changed. Such tick-mark-spacing changes are most conveniently performed by grouping tick marks and box from an existing example (such as Figure G.2) and then stretching the grouped image. To simplify current and future editing, care should be taken to ensure that the final box is snapped to the 3-point grid and the tick-mark spacing is an exact multiple of the 3-point grid size.

G.1.3 8-bit field format

Figure G.3 illustrates how subfield components of a 1-byte field should be illustrated. Only short tick marks are used, since there is no need to delineate byte locations. The font and alignment associated with field names, byte-size numbers, and MSB/LSB labels is specified in xx. Abbreviations are used when the field name would extend beyond its bounding box; see yy for details.

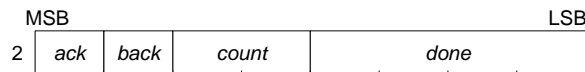


Figure G.3—8-bit field format

G.1.3.1 *ack*: A bit that has been illustrated on the left.

G.1.3.2 *back*: A bit that has been illustrated near the left.

G.1.3.3 *count*: A 3-bit field that has been illustrated near the center.

G.1.3.4 *done*: A 3-bit field that has been illustrated near the right.

NOTE—The preceding text illustrates how the Definition4Like style is used within figure-field definitions.

G.1.4 16-bit field formats

Figure G.4 illustrates how subfield components of a 2-byte field should be illustrated. Short tick marks and one longer tick mark are used to delineate bit and byte locations. The font and alignment associated with field names, byte-size numbers, and MSB/LSB labels is specified in xx.

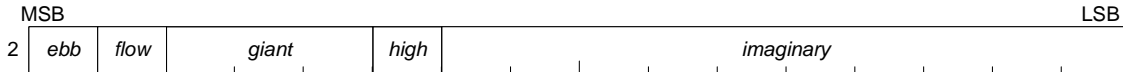


Figure G.4—16-bit field format

Abbreviations are used when the field name would extend beyond its bounding box; see G.1.5 for details.

G.1.5 32-bit field formats

Figure G.5 illustrates how subfield components of a 2-byte field should be illustrated. Short tick marks and three longer tick marks are used to delineate bit and byte locations. The font and alignment associated with field names, byte-size number, and MSB/LSB labels is specified in xx. Each field name that would extend beyond its bounding box is abbreviated; a legend associates the local abbreviation with the formal field name.

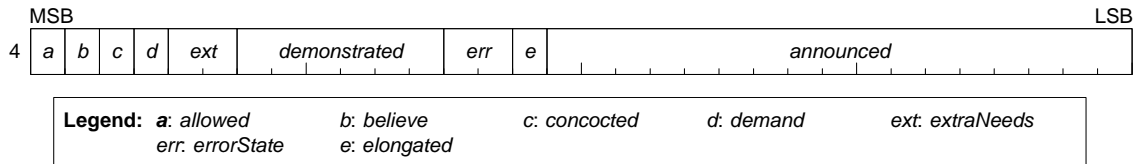


Figure G.5—32-bit field format

Below the fields, a legend is provided. The legend entries are ordered, in that a left-to-right scan of the fields corresponds to a left-to-right-then-top-to-bottom scan of the legend entries. A space should be included after the colon. The dotted line around the legend is a distinct graphics rectangle, which can be sized appropriately.

NOTE—The fields of Figure G.5 were planned to illustrate several of the larger byte-boundary tick marks. Depending on the field alignments, these are oftentimes hidden by the field’s bounding box.

G.2 Standard subclause formats

G.2.1 Common parameters

NOTE—Each clause or annex should use the second subclause to identify common values used within that clause, as illustrated below.

G.2.1.1 Common definitions

The following definitions are used multiple places within this clause.

GROUP_BIT

A constant value specified by the following C-code definition.

```
((uInt8)1) << 40) (G.1)
```

STOMP_CRC

A constant value specified by the following C-code definition.

```
(0xFFFFFFFF) (G.2)
```

NOTE—The fields of Figure 10.3 were planned to illustrate several of the larger byte-boundary tick marks. Depending on the field alignments, these are oftentimes hidden by the field's bounding box.

G.2.1.2 Common variables

The following state machine inputs are used multiple times within this clause.

hopsToCongestion

An input hop-count value generated by the fairness protocols.

myEdgeState

Indicates the adjacent edge condition.

NORMAL—The datapath is operating without a span failure.

INTO_EDGE—The transmit side of the datapath is associated with a failed span.

FROM_EDGE—The receive side of the datapath is associated with a failed span.

NOTE—All enumerated values should be described, even when the definition is obvious to the editor.

myRingletID

An input associated with the ringlet identifier of this datapath entity.

RINGLET_0—Transmitting onto ringlet0.

RINGLET_1—Transmitting onto ringlet1.

rateC

rateCC

Input rate values generated by the fairness protocols (see 9.3.1).

tickTime

A variable representing the start time of the current tick processing interval.

time

An input that represents a constantly incremented elapsed time value.

G.2.1.3 Common routines*Parity(frame)*

Returns the parity of the referenced bits in idle or fairness frames, as described in x.y.

NOTE—Subroutine descriptions should always include their argument, or two tiny spaces between parenthesis when no arguments are provided. The intent is to readily distinguish between routines (which are called) and variable values (which are read and written).

NOTE—The subroutine names and arguments are italics, but parenthesis or special characters are not.

SendSize(frame)

A routine whose behavior is defined by Equation G.1.

```
(frame.ri == myRingletID || \
(myWrapMethod == CENTER_WRAP && myEdgeState == INTO_EDGE))
```

(G.1)
SizeOf(frame)

Returns the number of bytes in the frame.

NOTE—The left edge of the in-line C code is indented using tabs, but nonbreaking spaces are used for line-to-line indent purposes thereafter.

G.2.2 Variables defined in other clauses

NOTE—The use of remote variables should be summarized in a distinct subclause, so that the dependencies between clauses can be easily identified.

This clause references the following global variables defined in Clause 6:

hopsToCongestion

rateC

rateCC

This clause references the following global variables defined in Clause 10:

myEdgeState

myWrapMethod

G.3 State machine definitions

State machine definitions should contain introductory text, an optional flow-chart, variable definitions, and a normative state table, as illustrated in the following subclauses.

G.3.1 Single-queue transmit state machine

An introduction of the state machine belongs here.

G.3.1.1 Single-queue transmit flow chart

The flow-chart for a state machine oftentimes includes a flow chart, as illustrated in Figure G.6. Text is represented using an 8-point font, except for the 7-point YES/NO labels.

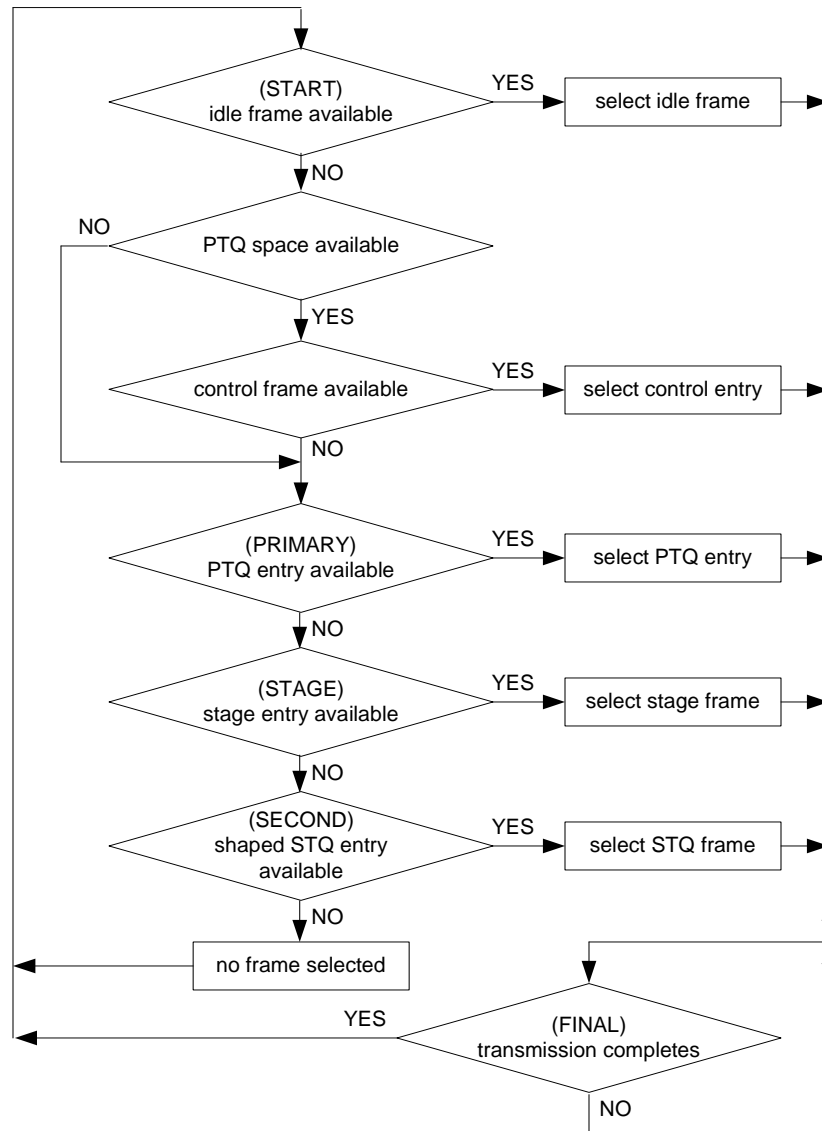


Figure G.6—Flowchart template

NOTE—All names used within the state machine should be described in the immediately preceding subclauses. These terms may cross-reference values set-by and defined-with other state machines, or may defined at the beginning of the clause and cross-reference by multiple state machines.

G.3.1.2 Single-queue transmit state machine definitions

CLASS_A0
See 6.2.2.

(...)

Q_TX_STAGE
See 6.2.1.1.

G.3.1.3 Single-queue transmit state machine variables

creditI
See 6.2.1.2.

(...)

frame
See 6.2.1.2.

G.3.1.4 Single-queue transmit state machine routines

Dequeue(queue)
See G.2.1.3.

(...)

SpaceInPTQ()
See G.2.1.3.

G.3.1.5 Single-queue transmit state table

The single-queue transmit frame selection state machine specified in Table G.1 implements the functions necessary for selecting which frame to transmit in a single-queue MAC. (...).

Table G.1—Single-queue transmit selection states

Current state		Row	Next state	
state	condition		action	state
START	PHY_READY.indication() == READY	1	—	CONT
	—	2	—	START
CONT	(frame = Dequeue(Q_TX_IDLE)) != NULL	3	creditI -= SizeOf(frame); Enqueue(Q_TX_COUNT, frame);	START
	(frame = DequeueControl(Q_TX_STAGE)) == NULL	4	—	PTQ
	SizeOfMacControl() > SpaceInPTQ()	5	—	START
	—	6	creditM -= SizeOf(frame); Enqueue(Q_TX_COUNT, frame);	
PTQ	(frame = Dequeue(Q_TX_PTQ))!= NULL	7	—	COUNT
	—	8	—	STAGE
COUNT	frame.sc != CLASS_A0	9	creditD -= SizeOf(frame);	START
	—	10	Enqueue(Q_TX_COUNT, frame);	
STAGE	(frame = Dequeue(Q_TX_STAGE))!= NULL	11	Enqueue(Q_TX_COUNT, frame);	START
	—	12	classC	

NOTE—
 1) Input conditions should not be OR'd, but split into two rows.
 2) White space should, when possible, be placed between operators and text.
 If such white space would result in a (oftentimes more confusing) line wrap, it can be eliminated.
 3) Common rows should be straddled for clarity within the figure and described in close-together text.

Row 1: The physical layer has indicated that it is ready.

Row 2: Wait for the physical layer to be ready.

NOTE—A paragraph (not line feed) is appropriate between descriptions relating to distinct states.

Row 9: Transit frames using unreserved bandwidth subtract from the downstream shaper credits.

Row 10: Transit frames using reserved bandwidth do not subtract from any shaper credits.

Row 11: The stage queue is selected when an entry is available in the queue.

Row 12: No frame is selected when no frame transmissions are possible.

G.4 Service primitives

The following abbreviated subclause illustrates styles used when specifying service primitives.

G.4.1 MA_DATA.request

G.4.1.1 Function

The MA_DATA.request primitive defines the transfer of data from a MAC client entity to a single peer entity, or to multiple peer entities in the case of group addresses.

G.4.1.2 Semantics of the service primitive

The semantics of the primitives are as follows:

```
MA_DATA.request
(
    destination_address,
    service_class,
    mac_protection           // optional
)
```

The parameters of the MA_DATA.request are described below:

destination_address
Specifies either an individual or group MAC address, different from the local MAC address, to be used to create the da (destination MAC address) field of the transmitted frame,

NOTE—A table is sometimes necessary to adequately describe the enumerated service primitive parameters, as illustrated for the service_class parameter below.

service_class
Indicates the class of service requested by the MAC client, as described in Table G.2, which is used by the MAC entity to select the value of the sc field, which is described in 5.2, and to indicate the requested MAC treatment of the transmitted frame, as described in 7.3.

Table G.2—service_class values

service_class value	Corresponding sc value	Description
SC_CLASSA	CLASS_A0 or CLASS_A1	classA
SC_CLASSB	CLASS_B	classB
SC_CLASSC	CLASS_C	classC

NOTE—Simple descriptions of enumerated values should be appended as a tab-indented list, as is done with the mac_protection specification below.

mac_protection

Indicates a choice of whether the MAC provides protection for the frame, as described in 4.3.

If the mac_protection parameter is omitted, a TRUE value is assumed.

TRUE—The MAC shall provide protection for the frame.

FALSE—The MAC shall not provide protection for the frame.

G.4.1.3 When generated

The MA_DATA.request primitive is invoked by the client entity whenever data is to be transferred to a peer entity or entities.

G.4.1.4 Effect of receipt

The receipt of the MA_DATA.request primitive causes the MAC entity to create a data frame or an extended data frame, fill in the fields whose values are given or determined by the parameters of this request, and pass the properly formed frame to the transmit state machines (as shown in Figure xx), for transfer to the peer MAC sublayer entity or entities.

G.4.1.5 Additional comments

The MAC does not reflect frames back to the client. If a client issues an MA_DATA.request primitive with a destination_address value equal to its local MAC address, the request is rejected.

Annex H (informative) Using word

H.1 Special characters

For the convenience of the editors, commonly-used special characters are described within Table H.1.

Table H.1—Commonly used special characters

Symbol	Name	Row	Press	Description
(not visible)	em space	1	Insert/Symbol/Special Characters	Space characters
	en space	2		
	¼ em space	3		
	nonbreaking space	4		
...	ellipsis	5	Insert/Symbol/Special Characters	Special characters
©	copyright	6		
™	trademark	7		
®	registered trademark	8		
–	en dash	9	Insert/Symbol/Special Characters	Printable hyphens and dash
—	em dash	10		
–	non-breaking hyphen	11	Insert/Symbol/Special Characters	Hyphenation control
		12	CTRL+SHIFT+HYPHEN	

H.2 Inserting template into an existing document

You can insert the template into a pre-existing document. To do this perform the following steps:

- a) Open the pre-existing document.
- b) Go to Tools|Templates and Add-Ins.
- c) Click the **Attach...** button.
- d) Attach the appropriate template (*IEEETemplate*).
- e) Click Automatically update document styles.
- f) Click the **Organizer** button.
- g) On the right side of the dialog, click the **Close File** button.
- h) Click the **Open File** button.
- i) Select the appropriate (*IEEETemplate*) template.
- j) In the **Style** tab, select all styles in the (*IEEETemplate*) template.
- k) Copy and overwrite all the styles from the template to your document. This will be sure that all style names, even the ones not presently used by the document, are copied.
- l) Click *Close*.

Annex I (informative) IEEE specifics

I.1 IEEE references

So that common references can be included within this document, cross references are provided for cut-and-paste uses:

AEIC publications are available from the Association of Edison Illuminating Companies, 600 N. 18th Street, P. O. Box 2641, Birmingham, AL 35291-0992, USA.

ANSI publications are available from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

API historical materials can be obtained (for a fee) from the American Petroleum Institute Library, 1200 L Street NW, Washington, DC 20005, USA.

API publications are available from the Publications Section, American Petroleum Institute, 1200 L Street NW, Washington, DC 20005, USA.

ARINC publications are available from ARINC Research Corporation, Document Section, 2551 Riva Rd., Annapolis, MD 21401.

ASHRAE publications are available from the Customer Service Department., American Society of Heating, Refrigerating and Air Conditioning Engineers, 1791 Tullie Circle, NE, Atlanta, GA 30329, USA.

ASME publications are available from the American Society of Mechanical Engineers, 22 Law Drive, Fairfield, NJ 07007, USA.

ASTM publications are available from the American Society for Testing and Materials, 100 Barr Harbor Drive, West Conshohocken, PA 19428-2959, USA.

AWEA publications are available from the American Wind Energy Association, Standards Program, 777 North Capital Street NE, #805, Washington, DC 20002, USA.

CCITT publications are available from the International Telecommunications Union, Sales Section, Place des Nations, CH-1211, Genève 20, Switzerland/Suisse. They are also available in the United States from the

U.S. Department of Commerce, Technology Administration, National Technical Information Service (NTIS), Springfield, VA 22161, USA.

CFR publications are available from the Superintendent of Documents, U.S. Government Printing Office, P.O. Box 37082, Washington, DC 20013-7082, USA.

CISPR documents are available from the International Electrotechnical Commission, 3, rue de Varembe, Case Postale 131, CH 1211, Genève 20, Switzerland/Suisse. They are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

CSA publications are available from the Canadian Standards Association (Standards Sales), 178 Rexdale Blvd., Etobicoke, Ontario, Canada M9W 1R3.

ECMA publications are available from the European Computer Manufacturers Association, 114 rue du hone, CH-1204, Geneva, Switzerland/Suisse.

EGSA publications are available from the Electrical Generating Systems Association, 10251 W. Sample d., Suite B, Coral Springs, FL 33065, USA.

EIA publications are available from Global Engineering, 1990 M Street NW, Suite 400, Washington, DC, IPS publications are available from the National Technical Information Service (NTIS), U. S. Dept. of ommerce, 5285 Port Royal Rd., Springfield, VA 22161.

ICEA publications are available from ICEA, P.O. Box 411, South Yarmouth, MA 02664, USA.

ICRU publications are available from the National Council on Radiation Protection and Measurements, 7910 Woodmont Avenue, Suite 800, Bethesda, MD 20814, USA.

IEC publications are available from IEC Sales Department, Case Postale 131, 3, rue de Varembe, CH-1211, benève 20, Switzerland/Suisse. IEC publications are also available in the United States from the Sales

Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

Internet Requests for Comments (RFCs) are available from the DDN Network Information Center, SRI lternational, Menlo Park, CA 94025 USA. They are also available on the World Wide Web at the following URL: <http://www.internic.net/ds/rfc-index.html>

IPC publications are available from the Institute for Interconnecting and Packaging Electronic Circuits (IPC), 7380 N. Lincoln Ave., Lincolnwood, IL 60646.

ISO publications are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembe, CH-1211, Genève 20, Switzerland/Suisse. ISO publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

JEDEC publications are available from JEDEC, 2001 I Street NW, Washington, DC 20006, USA.

MIL publications are available from Customer Service, Defense Printing Service, 700 Robbins Ave., Bldg. 4D, Philadelphia, PA 19111-5094.

NASA publications are available from NASA Center for Aerospace Information (CASI), ATTN: Document Orders, 800 Elkridge Landing Rd., Linthicum Heights, MD 21090-2934.

NBS publications are available from the Superintendent of Documents, U.S. Government Printing Office, P.O. Box 37082, Washington, DC 20013-7082, USA.

NCRP publications are available from the National Council on Radiation Protection and Measurements, 7910 Woodmont Avenue, Suite 800, Bethesda, MD 20814, USA.

NEMA publications are available from the National Electrical Manufacturers Association, 1300 N. 17th St., Ste. 1847, Rosslyn, VA 22209, USA.

NFPA publications are available from Publications Sales, National Fire Protection Association, 1 Battery-march Park, P.O. Box 9101, Quincy, MA 02269-9101, USA.

UL standards are available from Global Engineering, 1990 M Street NW, Suite 400, Washington, DC, 20036, USA.

US Regulatory Guides are available from the Superintendent of Documents, US Government Printing Office, P.O. Box 37082, Washington, DC 20013-7082, USA.

This authorized standards project was not approved by the IEEE Standards Board at the time this went to press. It is available from the IEEE Service Center. [NOTE: the reference must include the P-number, title, revision number, and date.]

ANSI XXX-19XX has been withdrawn; however, copies can be obtained from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

The numbers in brackets, when preceded by the letter B, correspond to those in the bibliography in Section XX.

The numbers in brackets correspond to those of the references in XX.

IEEE Std XXX-19XX has been withdrawn; however, copies can be obtained from the IEEE Standards Department, IEEE Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

As this standard goes to press, IEEE Std SSS-199X is not yet published. It is, however, available in manuscript form from the IEEE Standards Department, (908) 562-3800. Anticipated publication date is XXX 199X, at which point IEEE Std XXX-199X will be available from the IEEE Service Center, 1-800-678-4333.

This standard will be available from the Institute of Electrical and Electronics Engineers, Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA, in early 199X.

Approved 1394TA documents can be ordered through: 1394 Trade Association, 221 West Sixth Street, Suite 1520, Austin, TX 78701; by contacting taadmin@1394TA.org; or by fetching a pdf copy from the 1394TA web site: <http://www.1394TA.org/abouttech/specifications/techspec.html>.

I.2 IEEE cover pages

Cover pages for IEEE documents are provided in the remainder of this annex. When editing IEEE specifications, these pages should be moved to the front of the document.

IEEE PXXXX/Draft #
Date

Draft Standard ...

Sponsor:
Committee of the IEEE Society

Abstract:

Keywords:

Copyright © <current year> by the Institute of Electrical and Electronics Engineers, Inc.
Three Park Avenue
New York, New York 10016-5997, USA

All rights reserved. This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to change. **USE AT YOUR OWN RISK!** Because this is an unapproved draft, this document must not be utilized for any conformance/compliance purposes. Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only. Prior to submitting this document to another standards development organization for standardization activities, permission must first be obtained from the Manager, Standards Licensing and Contracts, IEEE Standards Activities Department. Other entities seeking permission to reproduce this document, in whole or in part, must obtain permission from the Manager, Standards Licensing and Contracts, IEEE Standards Activities Department.

IEEE Standards Activities Department
Standards Licensing and Contracts
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA

Introduction

(This introduction is not part of IEEE Std xx-200X, title.)

At the time this standard was completed, the working group had the following membership:

name, Chair
name, Secretary

The following three columns of names is implemented as a three-column table, with a white color applied to the between-column lines. When names are available, approximately 1/3 should be placed in each of these three table cells.

To Be Supplied By IEEE

Etc.

Etc.

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

The following three columns of names is implemented as a three-column table, with a white color applied to the between-column lines. When names are available, approximately 1/3 should be placed in each of these three table cells.

To Be Supplied By IEEE

Etc.

Etc.

When the IEEE-SA Standards Board approved this standard on XX Month 200X, it had the following membership:

name, Chair
name, Vice Chair
Judith Gorman, Secretary

The following three columns of names is implemented as a three-column table, with a white color applied to the between-column lines. When names are available, approximately 1/3 should be placed in each of these three table cells.

To Be Supplied By IEEE

Etc.

Etc.

*Member Emeritus

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

Satish K. Aggarwal, NRC Representative
Alan H. Cookson, NIST Representative
Donald R. Volzka, TAB Representative

Editor's name here
IEEE Standards Project Editor

Annex J (informative) C code illustrations

```
// The following illustrate how code can be presented in a landscape fashion
//
//          1          2          3          4          5          6          7          8          9          0          1          1          1          1
//2345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
```

```
#include <assert.h>
#include <stdio.h>

typedef unsigned char    uInt1;    // 1-byte unsigned integer
typedef unsigned short  uInt2;    // 2-byte unsigned integer
typedef unsigned int    uInt4;    // 4-byte unsigned integer
typedef unsigned long long uInt8;  // 8-byte unsigned integer

typedef signed char     sInt1;    // 1-byte signed integer
typedef signed short    sInt2;    // 2-byte signed integer
typedef signed int      sInt4;    // 4-byte signed integer
typedef signed long long sInt8;   // 8-byte signed integer

// Illustrations of useful ASCII-art comments follow
//
// Ethernet packet format:
// +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
// |          |l|m|          destinationMacAddress          |
// +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
// |          |l|m|          sourceMacAddress          |
// +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
// |          vlanCode          | pri |c|          vlanIdentifier          |          typeCode          |
// +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
// |version| ihl |
// +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

```

// IP-V4 packet format
// +-----+
//h2|          vlanCode          | pri |c|  vlanIdentifier  |          typeCode          |
// +-----+-----+-----+-----+-----+-----+
//h3|version| ihl | pre |D|T|R|C|r|  totalLength  |  identification  | fls |  fragmentOffset  |
// +-----+-----+-----+-----+-----+-----+
//h4| timeToLive | protocol |  headerChecksum  |          sourceIpAddress  |
// +-----+-----+-----+-----+-----+-----+
//h5|          destinationIpAddress  |          (ipOptions)  |
// +-----+-----+-----+-----+-----+-----+
//h6|          sourcePort          |  destinationPort  |          sequenceNumber  |
// +-----+-----+-----+-----+-----+-----+
//h7|          acknowledgementNumber  | offs | reserved | bits |          window  |
// +-----+-----+-----+-----+-----+-----+
//h8|          checksum          |  urgentPointer  |          (tcpOptions)  |
// +-----+-----+-----+-----+-----+-----+
//h9|
// |          applicationData  |
// |
// +-----+-----+-----+-----+-----+-----+
// |          checksum          |          (extension)  |
// +-----+-----+-----+-----+-----+-----+

```

```

// IP-V4 packet format
// +-----+
//h0|          |l|m|          destinationMacAddress  |
// +-----+-----+-----+-----+-----+-----+
//h1|          |l|m|          sourceMacAddress  |
// +-----+-----+-----+-----+-----+-----+
//h2|          vlanCode          | pri |c|  vlanIdentifier  |          typeCode          |
// +-----+-----+-----+-----+-----+-----+
//h3|version| ihl | pre |D|T|R|C|r|  totalLength  |  identification  | fls |  fragmentOffset  |
// +-----+-----+-----+-----+-----+-----+
//h4| timeToLive | protocol |  headerChecksum  |          sourceIpAddress  |
// +-----+-----+-----+-----+-----+-----+
//h5|          destinationIpAddress  |          (ipOptions)  |
// +-----+-----+-----+-----+-----+-----+
//h6|          sourcePort          |  destinationPort  |          sequenceNumber  |
// +-----+-----+-----+-----+-----+-----+
//h7|          acknowledgementNumber  | offs | reserved | bits |          window  |
// +-----+-----+-----+-----+-----+-----+
//h8|          checksum          |  urgentPointer  |          (tcpOptions)  |
// +-----+-----+-----+-----+-----+-----+
//h9|
// |          applicationData  |
// |
// +-----+-----+-----+-----+-----+-----+
// |          checksum          |          (extension)  |
// +-----+-----+-----+-----+-----+-----+

```

```
// Options quadlets (assumed to be aligned)
// +-----+
//p0| ip4NULL | ip4OpType | ip4OpLength + ip4OpPointer |
// +-----+-----+-----+-----+-----+
//p1| routeAddress[0] |
// +-----+-----+-----+-----+-----+
// | routeAddress[1] |
// +-----+-----+-----+-----+-----+
// | (etc) |
// +-----+-----+-----+-----+-----+
// | routeAddress[n] |
// +-----+-----+-----+-----+-----+
```



```

// IP-V6 source-route option format
// +-----+-----+-----+-----+-----+-----+-----+-----+
//s0| nextHead | opLength | opType=0 | segment | reserved |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//s1|
// +
// | address [n] |
// |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//s3|
// +
// | address [n-1] |
// |
// +-----+-----+-----+-----+-----+-----+-----+-----+
// |
// | ... |
// |
// +-----+-----+-----+-----+-----+-----+-----+-----+
// |
// | address [0] |
// |
// +-----+-----+-----+-----+-----+-----+-----+-----+

// IPX packet format
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h0| |1|m| destinationMacAddress |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h1| |1|m| sourceMacAddress |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h2| vlanCode |pri|c| vlanIdentifier | typeCode |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h3| checksum | packetLength | control | type | destinationNetworkHi |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h4| destinationNetworkLo | destinationNode |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h5| destinationSocket | sourceNetwork | sourceNodeHi |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h6| sourceNodeLo | sourceSocket |
// +-----+-----+-----+-----+-----+-----+-----+-----+
//h7|
// | applicationData |
// |
// +-----+-----+-----+-----+-----+-----+-----+-----+
// |
// | checksum | (extension) |
// +-----+-----+-----+-----+-----+-----+-----+-----+

```