WIND RIVER®
TECHNICAL PUBLICATIONS

HOUSE STYLE GUIDE

8.2

Corporate Headquarters

Wind River
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.
Toll free (U.S.A.): +1-800-545-WIND
Telephone: +1-510-748-4100
Facsimile: +1-510-749-2010

For additional contact information, see the Wind River website:

www.windriver.com

For information on how to contact Customer Support, see:

www.windriver.com/support

Technical Publications

House Style Guide 8.2

13 August 2020

Contents

1

Structure

About Structuring Documents

Most customers do not read our documentation word by word. Instead, they browse or scan the content for the information they need.

If the content is in big chunks of indigestible information, customers cannot easily find what they need when they need it.

Break up long chunks or paragraphs of content into more modular chunks that can easily be scanned.

Indigestible Chunks

Configuring Wind River DCOM to run as a VxWorks component involves configuration inputs at as many as three distinct places in the course of building a VxWorks image. First, you may need to make choices before you compile the component source. These compile time options are

documented in the getting started guide for your Platform. Second, you need to explicitly include Wind River DCOM in your customized VxWorks image project. And finally, you may need to write code to initialize or manage the component at runtime.

When you include the DCOM component in a project, support for both COM and DCOM is incorporated into the project. COM support includes a simple component class registry, the basic IUnknown support for VxWorks-resident COM objects, and the API functions defined in installDir/components/dcom-2.3/h/dcom/comLib.h.

DCOM support adds APIs that are required for remoting interfaces, marshaling, and other support for DCOM support also includes the interface between the main DCOM code and the underlying ORPC layer, which is called the ORPC interface.

Easier to Digest

To configure Wind River DCOM to run as a VxWorks component, you must do the following in the course of building a VxWorks image:

1. Include the necessary configuration options before you compile the component source. Compile-time options are documented in the getting started guide for your Platform.
2. Include Wind River DCOM in your customized VxWorks image project.

   When you include the DCOM component in a project, support for both COM and DCOM is incorporated into the project.

   COM support includes:

   - a simple component class registry
   - the basic IUnknown support for VxWorks-resident COM objects
   - the API functions defined in: *installDir***/components/dcom-2.3/h/dcom/comLib.h**

   DCOM support includes:

   - APIs that are required for remoting interfaces, marshaling, and other support for DCOM
   - the interface between the main DCOM code and the underlying ORPC layer, which is called the ORPC interface
3. Write code to initialize or manage the component at run time.

Titles and Headings

Write headings that are clear and descriptive of the content of the topic or section.

Keep them as short and to the point as possible, but informing users of the content they may expect is more important than keeping a heading to a single line.

Write headings accurately. A common error is to write headings that are either too narrow or too broad and therefore do not accurately reflect the discussion in the chapter or section.

Related Links

Capitalization of Titles and Headings on page 3
Use mixed uppercase and lowercase for the titles in all headings at all levels. Mixed case is easier to read than all uppercase.

About Naming Topics on page 4
Use gerunds in the headings that form task names; use noun phrases for headings that form concept and reference names.

Stylistic Consistency in Headings on page 5
Make headings consistent in grammatical construction and tone, within the constraints of the naming convention.

Lone Headings on page 5
Lone headings are permitted where principles of minimalism require them.

Capitalization of Titles and Headings

Use mixed uppercase and lowercase for the titles in all headings at all levels. Mixed case is easier to read than all uppercase.

Capitalization follows standard practices:

- Capitalize the first and last word of the title.
- Capitalize all other words except the following: articles (the, a, an) prepositions fewer than five letters (into, with, and so forth) conjunctions fewer than five letters (and, but, for, or, nor) the words to and as
- Capitalize literal names—such as commands, libraries, and routines—exactly, no matter what position they occupy in the heading; their capitalization must never be altered.
- Capitalize each element of a hyphenated word as if it were a separate word, except when the element is not the first word and would normally not be capitalized:
  - articles (the, a, an)
  - prepositions (into, with, and so forth)
  - conjunctions fewer than five letters (and, but, for, or, nor)
  - the words to and as

This capitalization standard applies to all headings, including topic and section titles, figure and table headings, and column headings in tables. However, it does not apply to steps.

Capitalizing First, Last, and Intermediate Words

An Overview of Workbench

Programming with VxWorks

About Semaphores

Distinguishing between Nouns and Verbs

Capitalizing Literals

Using the foo( ) Routine

The WIND_BASE Environment Variable

Capitalizing Hyphenated Words

Attach-to-Target

Cross-Development

On-Chip

Plug-in

Pop-up

Real-Time

Run-Time

Third-Party

Related Links

Titles and Headings on page 2
Write headings that are clear and descriptive of the content of the topic or section.

About Naming Topics

Use gerunds in the headings that form task names; use noun phrases for headings that form concept and reference names.

- Name task topics using the gerund verb form (the form ending in -ing).
  - Fetching Certificates at Run Time
  - Preparing to Use TPM
  - Updating WPAN Firmware
- Name concept topics using noun phrases.
  - RPM Repository Server
  - About WebIf
  - About Validating Keys and Certificates
- Name reference topics using noun phrases and “Reference” or a similar term unless the reference nature of the topic is obvious from the topic name.
  - imtool Reference

    SSL API Functions

IDP Services Reference

Related Links

Titles and Headings on page 2
Write headings that are clear and descriptive of the content of the topic or section.

Stylistic Consistency in Headings

Make headings consistent in grammatical construction and tone, within the constraints of the naming convention.

The first level of consistency must be conformance to the topic naming conventions, which apply to first- and second-level headings. For section headings, maintain consistency within a topic. Use all gerunds or all noun phrases and keep constructions parallel.

In a draft version of early release notes, some headings described fixes, while some merely described problems. Consider the following heading:

Large Ring Buffers Did Not Work Correctly

The writer changed the heading to describe a fix rather than a problem.

Large Ring Buffers Now Handled Correctly

This change not only provides consistency with other fix descriptions, it also says what in fact is offered, which is missing from the original. Furthermore, the shift in focus from problems to repairs adds a more positive tone.

Related Links

Titles and Headings on page 2
Write headings that are clear and descriptive of the content of the topic or section.

About Naming Topics on page 4
Use gerunds in the headings that form task names; use noun phrases for headings that form concept and reference names.

Lone Headings

Lone headings are permitted where principles of minimalism require them.

Headings serve specific purposes in structured writing.

- Topic titles, which may be lone headings today but not tomorrow when additional topics are created.
- Section headings within topics, which are required in references for any elements except tables no matter how many (or few) sections exist.

Principles of minimalism tell us not to create topics or sections merely to conform to a rule; only meaningful content is allowed.

This means that a topic can have a single child. It also means that a group of peer topics can have a descriptive heading without any text between the top-level heading and the first topic title.

Related Links

Titles and Headings on page 2
Write headings that are clear and descriptive of the content of the topic or section.

Introductions

Introductions for Groups of Topics

Create introductory topics only when they add value for users.

An introductory topic should give a broad understanding of the information in a group of topics.

- For groups of concepts, provide a high-level overview of the concepts and use subsections to go into detail if needed.
- For groups of tasks, explain what users accomplish by doing the tasks, what tools they need, what concepts or technologies they should be aware of, and where to find that information.

Introductions for Figures and Tables

Follow minimalist principles: if there can be no confusion about the purpose of a figure or table, do not introduce it.

Where an introduction is needed, introduce figures and tables as you would a list or display, with a complete sentence using one of the following phrases:

- ... as follows:
- ... the following:
- For example:
- That is:

NOTE: Do not introduce numbered tables and figures in FrameMaker as you would a list or display. Introduce numbered tables and figures by means of a cross-reference only—do not use a “following” phrase and a colon.

Cross-References within a Topic Set

Cross-references (FrameMaker) and <xrefs> (DITA) generate live links between topics in the same topic set or book.

Cross-references can be placed inline in the text or grouped at the end of a topic in a Related Links section.

- Inline cross-references are introduced as follows:

  For more information, see *Topic Title, p. n*.

- Related links can be listed at the end of the topic in a <related-links> section or generated by a relationship table in DITA. In Frame, place them at the end of the topic with a section heading (Related Links) and no introduction:

  First Topic Title, p. n.

  Second Topic Title, p. n.

Paragraph Structure

The standard purpose of a new paragraph is to signal a new subject, such as a fresh idea, a different set of facts, a change in procedures, or a shift of emphasis.

The first sentence in a paragraph introduces or summarizes the new topic. Readers can skim these topic sentences and decide whether to read the remainder of the paragraph.

Each paragraph should address a single subject. However, long paragraphs, even on a single subject, are hard to read and should be broken up into smaller units. Consider ten lines to be a general limit.

Use parallel construction of sentences, phrases, or lists to help readers understand relationships and differences quickly. In technical documentation, ease of comparison of ideas, terms, and tasks is critical. Variations in phrasing and sentence structure obscure critical information.

Sentence Structure

Complete Sentences

Always write in complete sentences except in specified cases.

Do not use abbreviated English by excluding articles (the, a, an) for brevity.

Exceptions to the complete sentence rule include the following:

- items in definition lists
- definitions in a glossary
- table items

- the construction “For example:” or “That is:” to introduce short code examples or displays

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

Definition Lists on page 12
A definition list, also known as a terms list, gives an item followed by an indented paragraph that defines the term or explains its usage.

Displays on page 14
Display is the term used in technical documentation for a standalone text item set off on a separate line (or lines) from the normal text of a paragraph.

About Capitalizing Sentences

Capitalize only the first letter in a sentence, except for literals and capitalized acronyms.

Do not capitalize text that follows a colon, unless the text is normally capitalized or is an item in a capitalized list (see List Structure on page 8.

Capitalize literals (such as names of commands, libraries, or routines) exactly as these names are capitalized in the system. Avoid using literal names that begin with a lowercase letter as the first word in a sentence. However, if you cannot avoid it, do not capitalize the literal name.

Do not make proper nouns out of names for common hardware or software elements, like routing table and autonomous system. In other words, do not write Routing Table and Autonomous System. This is a common error in material contributed by non-writers.

When defining an acronym, use initial capitals only if the originator of the term uses them. For more information about acronyms and capitalization, see Acronyms on page 36.

List Structure

Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

Lists include unordered lists, ordered lists, and definition or item lists. Lists have the following benefits:

- clarify relationships
- add visual impact
- easy to read
- break up page space
- easy to locate

Data presented in tables adds even more visual impact and clarifies relationships.

Itemized lists are handled with a variety of stylistic and format options, depending on the nature of the information to be conveyed.

Related Links

Grammatical Construction on page 9
Use parallel grammatical construction for lists to avoid distracting readers with meaningless differences.

List Punctuation on page 10
If list items are complete sentences, capitalize the first word and end each sentence with a period. If list items are not complete sentences, do not capitalize and punctuate them as sentences.

About Introducing Lists on page 11
Follow minimalist principles: if the contents of the list are clear from the context, do not write a meaningless sentence just to have an introductory sentence.

Ordered and Unordered Lists on page 12
Use ordered lists, where each item begins with a number or letter, only when you need to list items in a way that conveys sequence or explicit priority.

Definition Lists on page 12
A definition list, also known as a terms list, gives an item followed by an indented paragraph that defines the term or explains its usage.

About Choosing Between Definition Lists and Tables on page 13
While definition list s of can be created using item/follow paragraphs, such lists can also be created with a table.

Ordered and Unordered Lists on page 12
Use ordered lists, where each item begins with a number or letter, only when you need to list items in a way that conveys sequence or explicit priority.

Definition Lists on page 12
A definition list, also known as a terms list, gives an item followed by an indented paragraph that defines the term or explains its usage.

Grammatical Construction

Use parallel grammatical construction for lists to avoid distracting readers with meaningless differences.

- If most items in a given list are best worded as gerunds, then all should be.
  - initializing the shared memory pool and the backplane anchor
  - maintaining the backplane heartbeat
- If most items need to be complete sentences, then all items in the list should be complete sentences.

  INCORRECT:

  - a set of extremely large numbers
  - Each key consists of an RSA key-pair, containing a *private key* and a *public key*.

  CORRECT:

  - The *key* is a set of extremely large numbers.
  - Each key consists of an RSA key-pair, containing a *private key* and a *public key*.

- If items in a given list use verbs, the verbs should agree in verb tense.

  INCORRECT:

  - Boot loaders *will pass* the device owner's certificate content in the boot command line.
  - Applications in user-space *fetch* the root-of-trust from kernel space

  CORRECT:

  - Boot loaders *pass* the device owner's certificate content in the boot command line.
  - Applications in user-space *fetch* the root-of-trust from kernel space

- Lists may consist of verb phrases with a uniform understood subject.

  The boot loader:

  - uses GRUB-IMA and U-Boot to verify the kernel image
  - uses the IMA-Appraise feature to verify executable files

- Lists may consist of noun phrases.
  - a thread of execution; that is, the task's program counter
  - the CPU registers and optionally floating point registers
- Never mix sentence and non-sentence items in the same list.

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

List Punctuation

If list items are complete sentences, capitalize the first word and end each sentence with a period. If list items are not complete sentences, do not capitalize and punctuate them as sentences.

Glossary items and definition lists are exceptions to the complete-sentence rule. See Definition Lists on page 12 and Glossaries on page 16.

Never end list items with a comma or semicolon, then put a period after the last item.

List of Gerund Phrases

The backplane master has the following responsibilities:

- functioning as the gateway to the external network
- allocating the shared memory pool from its dual-ported memory, in some configurations

List of Noun Phrases

A task's context includes the following:

- a stack for dynamic variables and function calls
- I/O assignments for standard input, output, and error
- a delay timer
- signal handlers

- monitoring values for debugging and performance

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

About Introducing Lists

Follow minimalist principles: if the contents of the list are clear from the context, do not write a meaningless sentence just to have an introductory sentence.

Where an introduction is needed, introduce a list with a complete sentence that is a logical lead-in and observe the following rules:

- Employ wording like “as follows” or “the following”, as in the following examples:
  - Topics covered in the remainder of this manual include the following:
  - WindNet OSPFv2 includes the following features:
  - Access rights for the objects in the MIB tables are defined as follows:
  - To include Wind River NAT in a VxWorks image project, take the following steps:
  - When the Generate Includes dialog appears, do the following:
  - The sample source code can be found in the following location:
- Make the introduction a complete sentence in itself.
- Make sure there is agreement between list items and the introduction.
- Never set up list items so that they complete the introductory sentence. For example:

  INCORRECT

  Use Workbench facilities to:
  - create and edit source code
  - configure the RTOS
  - build your code and link it to the RTOS

  CORRECT

  Use Workbench facilities for the following:
  - creating and editing source code
  - configuring the RTOS

- building your code and linking it to the RTOS.

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

Ordered and Unordered Lists

Use ordered lists, where each item begins with a number or letter, only when you need to list items in a way that conveys sequence or explicit priority.

Otherwise, use unordered lists. Unordered lists, where each item begins with a bullet, dash, or no leading punctuation, are more common. Most list items can occur in any order without misleading the reader.

Ordered lists represent the sequence or priority of the items.

NOTE: Lists are not steps. Steps are specific actions that a user performs. The steps may be ordered or unordered. Steps occur only in task topics. While lists may appear in tasks, they must never be used in place of steps.

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

Definition Lists

A definition list, also known as a terms list, gives an item followed by an indented paragraph that defines the term or explains its usage.

Use definition lists for system elements such as parameters, commands, command options, files, routines, or macros. For example:

INCLUDE_MODULE_MANAGER

Provides facilities for managing loaded modules and obtaining information about them. For more information, see the reference entry for **moduleLib**.

Sentence fragments are allowed in the first sentence of the definition. Definitions of elements that perform some kind of action, such as routines or commands, should be complete sentences in imperative mood.

create-key

Create private keys and X.509v3 certificates.

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

About Choosing Between Definition Lists and Tables

While definition list s of can be created using item/follow paragraphs, such lists can also be created with a table.

Consider the following when deciding between an item/follow solution and a table solution:

- The item/follow paragraphs are better for lists of items with long names or long descriptions, whereas tables alone are less suitable for the same.
- Tables allow you to add data columns, if appropriate. See the sample below.
- Always use the paragraph solution when the listing follows a syntax display for a command or routine.

In some cases placing item/follow paragraphs inside a table provides a hybrid solution that sidesteps the shortfalls of either solution by itself. This approach is particularly suited to macro names, which are often long and require an extra column for data values. In a standard table, oversized macro names eat up column width and leave lots of white space, with little space left over for descriptions.

The following is an example of a hybrid solution, showing item/follow paragraphs inside a table.

Table 1-2 PPP Framework Configuration Parameters

Parameter and Description	Default Value
PPP_SYSTEM_DEFAULT_FRAMEWORK_NAME The name associated with this framework as a quoted text string. This name is required as input to some **pfw*()** routines.	SYS_PPP
PPP_CONTROL_JOB_QUEUE_SIZE The size, in bytes, of the job queue associated with the control task. This value limits the amount of memory dedicated to handling control packets. Control packets that arrive when the queue is full are dropped.	2048
PPP_DATA_JOB_QUEUE_SIZE The size, in bytes, of the job queue associated with the data task. This value limits the amount of memory dedicated to handling data packets. Data packets that arrive when the queue is full are dropped.	2048

Related Links

List Structure on page 8
Itemized lists, where items of information are broken out with each new item on a new line, are desirable whenever possible.

Displays

Display is the term used in technical documentation for a standalone text item set off on a separate line (or lines) from the normal text of a paragraph.

Use displays to show URLs, pathnames, command syntax, execution examples, terminal sessions, and code examples. These elements must always be set off in a display, no matter how small they are. Exception: command syntax can also be shown in a definition list.

- Displays draw attention to these elements.
- Displays provide a way to show long pathnames or URLs without the awkward line breaks that often occur in PDF in normal, wrapped paragraphs.

Ideally, but not always, a display is indented one level from the paragraph that introduces it.

INCORRECT:

Boot loader parameters are defined in the *installDir***/vxworks-6.x/target/config/bspname/config.h** file. To change...

CORRECT:

Boot loader parameters are defined in the following file:

*installDir***/vxworks-6.x/target/config/bspname/config.h**

To change...

Introduce displays as you would a list, with a complete sentence using one of the following phrases:

- ... as follows:
- ... the following:
- For example:
- That is:

Related Links

About Introducing Lists on page 11
Follow minimalist principles: if the contents of the list are clear from the context, do not write a meaningless sentence just to have an introductory sentence.

Notes, Cautions, and Warnings

Notes, cautions, and warnings highlight information that readers should not overlook.

Do not overuse these elements. The more they appear, the more their impact becomes diluted.

Note, caution, and warning displays distinguish three levels of alert:

- Note displays contain information that warrants special attention but is not cautionary in nature. Do not use notes for parenthetical information; use them for information that actually warrants the special attention that note displays provide.
- Caution displays contain information that is a mild or moderate warning.

  Examples include:

  - configuring in a way that results in heavy memory requirements
  - programming to use a data region designated for something else
  - moving data such that your software cannot find it where it is expected
  - running hardware or software that is not compatible
- Warning displays convey a severe warning.

  Examples include:

  - heavy loss of data
  - system crashes
  - hardware damage

Glossaries

A glossary is an alphabetical list of terms and their definitions.

A glossary includes terms used in a particular topic set that are either newly introduced, uncommon, or specialized.

- Contrary to normal heading practice, do not capitalize the terms unless they are acronyms or they are otherwise normally capitalized.
- The first "sentence" of the definition should not be a complete sentence; however, it should start with a capital letter and end with a period.
- Uncommon acronyms should include a "see" reference to the spelled-out form.
- When creating index entries for terms where they occur in a glossary, indicate that the entry is for the "glossary definition" to distinguish it from other index entries for the term.

unsupported

A product or element of a product that is included in a release, but has not been tested, and is not guaranteed to work as claimed. Do not use in customer documentation unless approved by Marketing.

2

Grammar

About Using Verbs

Subject/Verb Agreement

In English, agreement between a verb and its subject is complicated by phrases that modify the subject.

The simple case is usually intuitively obvious.

CORRECT:

- One is available.
- Many are available.
- Each is going to...
- All are going to...

When a modifying phrase is added, the phrase may be plural even though the subject is singular. Nevertheless, the verb agrees with the subject, not with the modifier.

INCORRECT:

- *One* of the following *are* available...

CORRECT:

- *One* of the following *is* available...

- *Many* of the following *are* available...

INCORRECT:

- *Each* of the students *are* going to...

CORRECT:

- *Each* of the students *is* going to...
- *All* of the students *are* going to...

Present Tense

Assume that your reader is doing the actions you describe while reading your document.

- Use present tense wherever possible.

  INCORRECT:

  - If there is not enough memory, the routine *will return* ERROR.
  - If..., you *will need to* configure your own PPP framework.

  CORRECT:

  - If there *is* not enough memory, the routine *returns* ERROR.
  - If..., you *must* configure your own PPP framework.

- Avoid past and future tenses unless they are essential to convey the idea.

  CORRECT

  - If you already *performed* this task, continue with the next task in the workflow.
  - WPA is an interim standard that *will be* replaced with the IEEE 802.11i standard when 802.11i is complete.

Active and Passive Voice

Use active voice rather than passive voice in technical documentation.

Passive voice relies on forms of the verb to be, and tends to hide who or what acts and who or what receives the action. This encourages vagueness. In technical documentation, the user wants to know who does what to whom (or to what).

Exceptions

The primary guide for choosing to compose a passive or active sentence should be meaning and emphasis. If the subject is unimportant or not worth mentioning, passive voice is appropriate. For example, notice that the first statement is more direct than the second even though it is in passive voice:

The house was designed, built, and sold within six months.

The architect designed the house, the contractor built it, and the realtor sold it, all within six months.

You can often recast passive sentences as imperatives when an ordinary active sentence is awkward.

Sentences Using Passive and Active Voice

WEAK:

If *the shell is used* to create shared data regions, the optional physical address parameter should not be used with architectures for which the PHYS_ADDRESS type is 64 bits.

STRONGER;

If *you use the shell* to create shared data regions, do not use the optional physical address parameter for architectures with 64-bit PHYS_ADDRESS types.

Can/May and Must/Should

Technical documentation is prescriptive: it tells users how to achieve specific results. Use can and must to avoid indecisiveness and irresolution.

- Use can to indicate that a user is able or permitted to do something in a given context and get expected results.

  INCORRECT:

  If..., you *may wish to* modify your boot loader configuration.

  CORRECT:

  If..., you *can* modify your boot loader configuration.

- Use must to indicate that something is required for the user to get a particular result.

  INCORRECT:

  If there is not enough memory, you *should* modify your application or take other steps to reduce memory usage.

  CORRECT:

  If there is not enough memory, you *must* modify your application or take other steps to reduce memory usage.

May and *should* give an impression of uncertainty, either about requirements or about results.

Split Infinitives

Avoid split infinitives, for example, 'to really succeed', unless the resulting sentence is unnatural or ambiguous.

The infinitive form of a verb uses to, for example:

- to alter
- to confirm
- to perform

The term split infinitive means that you insert a modifier between the to and the following word.

Avoid split infinitives whenever possible. They generally lead to awkward constructions. Usually rephrasing the sentence to avoid the split infinitive makes the meaning clearer. However, a split infinitive is always preferable to a modifier in an unnatural or ambiguous position.

Split Infinitives

INCORRECT:

Use signals to *asynchronously* alter the control flow of a task.

CORRECT:

Use signals to alter the control flow of a task *asynchronously*.

Trailing Prepositions

You can end a sentence with a preposition if this results in a clear and natural statement.

Prepositions are words that show the relationship between people and objects. Prepositions convey the following relationships:

- agency (*by*)
- comparison (*like*, *as*)
- direction (*to*, *toward*, *through*)
- place (*at*, *by*, *on*)
- possession (*of*)
- source (*from*, *out*)
- time (*at*, *before*, *on*)

If you have a choice of constructions that are equally clear and natural, avoid the trailing preposition.

Prepositions

TRADITIONALLY CORRECT:

...the Internet address of the UNIX host *from which to boot*.

CLEARER:

...the Internet address of the UNIX host *to boot from*.

Proper Adjectives

A proper adjective is an adjective that is derived from a proper name.

In the following example, **main( )** is a proper adjective:

Use the **main( )** function to...

In sentences that contain proper adjectives, the proper adjective should always come before the noun.

CORRECT:

Use the **main( )** function to...

INCORRECT:

Use the function **main( )** to...

That/Which

The words *that* and *which* are not interchangeable. They change the meaning of a sentence depending on how you use them.

- If you want to restrict the sentence to refer to a specific case, you can use that with no comma.

  I will shop at a Safeway that is open 24 hours.

  In this example, you are saying that you are only willing to shop at a 24-hour Safeway. This case is called a restrictive or defining clause.

- If you want to provide information only, use which with a comma.

  I will shop at a Safeway, which is open 24 hours.

  In this example, you are saying that you are willing to shop at any Safeway and Safeways can be presumed to be open 24 hours. This case is called a non-restrictive or parenthetical clause.

A useful rule-of-thumb is:

- Use which with a comma when you could substitute parentheses without changing the meaning of the sentence.
- Use that with no comma when substituting parentheses would change the meaning of the sentence by removing a restriction.

Example from VxWorks documentation:

This example could be written with parentheses.

The sections below give an overview of VxWorks tasking functions, which are found in the VxWorks library taskLib.

The clause *which are found...* implies that all VxWorks tasking functions are found in the library taskLib.

Consider the same example reworked:

The sections below give an overview of the VxWorks tasking functions that are found in the VxWorks library taskLib.

This implies that the overview covers only those tasking functions actually found in **taskLib**; that is, there may be tasking functions in other libraries.

Neither of the above versions is wrong, but the meaning of each is different.

3

Punctuation

Comma

Serial Comma

A comma must precede the conjunction and, or, or nor when it separates the last of a series of three or more words or phrases.

For example:

apples, oranges, and bananas

This last comma is often referred to as the final serial comma.

NOTE: The final serial comma is not optional. Even though some writers and editors consider it optional, Wind River observes it in the interest of both accuracy and consistency.

Complying with the rule uniformly throughout avoids the havoc caused by those situations where the final comma has been omitted and really is indispensable. More so than most

journalism, technical documentation requires a high degree of precision in describing complex processes. The difficulty of dropping the final comma occurs when items in a series are not just single words, but phrases which contain conjunctions not part of the overall series. Consider the following sentence:

2. Basic OS describes the fundamentals of the VxWorks run-time environment, including the multitasking kernel, intertask communication facilities, spawning and manipulation of tasks and interrupt service code.

Is *spawning and manipulation of interrupt service code* a single item of this series, or is interrupt service code a separate item? Or, worse, is spawning one item, and manipulation of tasks and interrupt service code a separate item? Understanding the subject and context suggests that interrupt service code is a separate and final item. A comma after tasks would have made this completely clear. Consistently using the serial comma never allows situations like this to occur.

Comma with That or Which

Wind River house style uses a comma preceding *which* and no comma preceding *that*.

- Use *which* with a comma when the modifying phrase could be enclosed in parentheses without changing the meaning.
- Use *that* without a comma when parenthesis would change the meaning.

Related Links

That/Which on page 21
The words *that* and *which* are not interchangeable. They change the meaning of a sentence depending on how you use them.

Comma in Product Name

Product names that include commas must always include the comma.

However, include other commas in the same sentence only if they are correct even without the comma in the product name.

When writing sentences with names such as “Wind River General Purpose Platform, Linux Edition” avoid the temptation to add a comma after “Edition” regardless of whether it seems more natural. Insert the comma only if necessary per standard grammatical practice, as for a serial list or separator in a compound sentence. For example:

INCORRECT:

Wind River General Purpose Platform, Linux Edition, is a bundled offering consisting of core Wind River technology components.

CORRECT:

Wind River General Purpose Platform, Linux Edition is a bundled offering consisting of core Wind River technology components.

For a list of known problems in Wind River General Purpose Platform, VxWorks Edition, visit the Online Support Web site.

Apostrophe

Avoid using an apostrophe to make nouns possessive.

In most cases, you can recast the sentence to avoid the possessive.

If you must use the possessive:

- Form the possessive of plural nouns ending in s by adding only an apostrophe; for example: the parameters' values
- Form the possessive of singular nouns and names ending in the sound of s, x, or z by adding an apostrophe plus s. Thus, the possessive of VxWorks is:

  VxWorks's

However, always avoid this awkwardness if possible by recasting such phrases. See Possessive Forms on page 54.

Never use an apostrophe to form the plural of an acronym. For example:

INCORRECT:

The following API's are available.

CORRECT:

The following APIs are available.

Quotation Marks

Avoid quotation marks. In most cases, there are appropriate character tags for identifying special text.

Exceptions to this rule are single-character literals and periods or commas with quotations.

With Literal Names or Strings

In general, all literals—names of tools, libraries, routines, menus, and so on— should appear in bold, not in quotation marks. See the FrameMaker Formatting Guide or the Information Model.

However, make exception for literal names that consist of a single punctuation mark or other non-alphanumeric character. Single characters can easily be lost in text, even if properly tagged in bold.

At the first reference, name the character, tag it bold (either nameLiteral or <nameliteral>), and surround it with parentheses, for example:

End the statement with a semicolon (;).

If you have frequent subsequent references, discontinue naming it, but tag the punctuation nameLiteral and enclose it in double-quotes, for example:

Do not forget to include the ";".

In GUI Documentation

Use quotation marks to set apart text delivered in a dialog box or input into a dialog box, but that is not identifying an area or element of the dialog box. For example, if you are telling the reader to fill in an edit box with specific text, then the specific text should appear in quotes; that is, "filled in text".

With Error Messages

When referencing an error message in text (not in a code display), surround the message in quotation marks.

With Period and Comma

Historically, quotation marks have been used to denote the direct quotation of a speaker,. The rules that have evolved with this require that periods and commas following quoted material fall within the final quotation mark. Despite the fact that in many cases this violates logic, it is still a firm convention in American English usage.

However, in software documentation, quotation marks are often used for the more specialized purpose of setting off terms that are under discussion. Occasionally, periods and commas are a significant component of these terms. Therefore, in cases where the item to be quoted is an element of code, an error message, or some similar component, periods and commas should be placed outside the final quotation mark to avoid any ambiguity.

In summary: if you are quoting a speaker or writer—an uncommon requirement —put periods and commas inside the quotes, per standard American practice. In all other cases, put them outside.

Colon

Always use a colon after the phrase or sentence introducing an example, itemized list, unnumbered table, or unnumbered graphic.

The following list shows the correct usage.

The backplane master has the following responsibilities:

- functioning as the gateway to the external network

- allocating the shared memory pool from its dual-ported memory, in some configurations

Related Links

About Introducing Lists on page 11
Follow minimalist principles: if the contents of the list are clear from the context, do not write a meaningless sentence just to have an introductory sentence.

Hyphen

Hyphen with Compound Modifiers

When two or more modifiers precede a noun, use the hyphen to make the meaning clear.

When multiple modifiers precede a noun, meaning can vary depending on whether the modifier acts on the noun or on another modifier. The following rule covers most situations:

- Two or more words used as a single modifier before a noun should be hyphenated, unless the first word is an adverb ending in -ly.
- In most cases hyphenation should not occur if the modifier occurs after the modified word.

The following three phrases illustrate the correct application of this general rule:

partly finished task

half-finished task

the task was half finished

Many situations are less clear than this example. There is no rule of thumb that governs all cases, and thorough coverage is beyond the scope of this document. At a minimum, if there is any possibility whatsoever that a compound modifier can be misinterpreted, by all means hyphenate it.

backward compatibility model

Strictly speaking, this phrase could be interpreted in either of the following two ways:

- *backward (compatibility model)*
- *(backward compatibility) model*

In this particular case, our audience would generally not think this is about (1), a compatibility model that is backward. They would understand it to refer to (2), a model about backward compatibility. However, where context is absent or the audience is not familiar with the term backward compatibility, this distinction might not be clear.

The need to convey this distinction is the original motivation behind tying words together with a hyphen, and why we write the following:

backward-compatibility model

compound modifiers with based

priority-based task scheduling

ROM-based system

Some publications do not hyphenate compounds formed with based because the word is never used alone as a modifier; however, the Wind River standard is to hyphenate it for consistency with the general rules of compound hyphenation.

compound modifiers with numbers

- *seven-second delay*
- *32-bit Internet address*
- *third-party licensor notice*

suspended compounds

- *character- and block-oriented devices*
- *8- and 32-bit machines*

handling multiple modifiers

What does the following example mean?

firewall extension handler service registration module

Perhaps it is a firewall module that registers services, with this particular module registering extension-handler services. In that case, a logical hyphenation scheme would be the following:

firewall extension-handler service-registration module

You can often improve readability further by recasting some modifiers with prepositional phrases, as in the following:

firewall service-registration module for extension handlers

Hyphenated Compound Reference

Many commonly used compounds should be hyphenated but are often overlooked.

Compound Modifiers with Example Nouns

This list is representative but by no means complete.

Compound Modifier	Noun that Follows (Example)
32-bit	Internet address
backward-compatibility	model
block-oriented	device
C-expression	interpreter
character-oriented	device
command-line	interface
device-independent	interface

Compound Modifier	Noun that Follows (Example)
DOS-like	file system
error-free	communication
floating-point	operation
format-driven	I/O routines
function-call	protocol
higher-level	I/O functions
low-level	system calls
memory-resident	buffer
open-source	software
priority-based	scheduling
ROM-based	system
run-time	environment
S-record	format
system-wide	symbol table
target-specific	routines
third-party	licensor notice
UNIX-compatible	file system
UNIX-format	object modules

Dash

Em-dash

If a dash is necessary to separate a thought or list from the rest of a sentence, use the em-dash, and do not separate it from text with space characters:

The other new object classes—system, region pool, physical region, logical region—are discussed in 6.4 Memory Configuration, p. 108.

En-dash

A space character should separate an en-dash from text that precedes or follows it. Use en-dashes in the following constructions:

- in the title line (NAME section) of reference entries, to separate the module or function name from its short definition, for example:

ramDevCreate( ) – create a RAM disk device

- to separate special terms from their definitions, when not using a definition or item list, for example:

  mutex – mutual exclusion

- as the mark in dash-list items

Pathname

Slash in Pathnames

Use forward slashes when your content applies to both UNIX and Windows systems.

When your content applies to UNIX only or Windows only, use forward slashes for UNIX and back-slashes for Windows. Use forward slashes when the content applies to both, because Windows can accept forward slashes while UNIX cannot accept back-slashes.

Long Pathnames

You can use one of several methods to avoid long pathnames wrapping inappropriately.

Pathnames present special issues when they are long. They can cause right margins to be excessively ragged and they can be difficult for readers to absorb. For this reason, do not include pathnames in a sentence or paragraph, but set them off in a display. For information on displays, see Displays on page 14.

If you have written sentences or paragraphs that include inline pathnames, converting these to displays typically requires reworking the surrounding text.

INCORRECT:

Boot loader parameters are defined in the *installDir*/**vxworks-6.x/target/config/bspName/config.h** file. To change...

CORRECT:

Boot loader parameters are defined in the following file:

installDir/**vxworks-6.x/target/config/bspName/config.h**

To change...

In cases where several paths contain a common root, the root portion can be separated and included within an introductory sentence, as in the following example:

The following files are located in *installDir*/**vxworks-6.3/target/src**:

posix/pthreadLib.c

util/bootLib.c

In general, indent displays one level in from the introductory paragraph. However, in cases where a pathname is long enough to cause a line break in the display, indent it to the same level as the introductory paragraph, as in the following example:

You can give your users the ability to choose their help style by importing the commands into your project from the following directory:

installDir/**webcli-4.4/target/src/wrn/wmc/examples/commands.rcp**

If a pathname is still too long to fit on a single line, wrap the line using a forced return (SHIFT +ENTER) in FrameMaker or a zero-width space ()in DITA. Insert the special character only where there is a slash separator (/) and insert it after the slash, so that the slash appears at the end of the first line. For example:

The boot loader files (for example, bootApp_romCompressed.hex) are at the following location:

installDir/**vxworks653-2.x/target/proj/wrSbc750gx_bootApp/ PPC604gnu_romCompressed**

In FrameMaker, assign the **FP_printonly** condition to the forced return character. This allows it to be stripped when the display is converted to HTML and no line break is therefore needed.

Occasionally, you may need to describe the relationship between two paths, as, for example, when you need to indicate where something needs to be copied or how something needs to be changed. In such cases, the displays can be separated with to, as in the following example:

Base your project on an existing BSP. Copy the BSP from its original location to an appropriate VxWorks 6.x directory. For example, copy the following directory:

installDir/**Tornado/target/config/mv5100**

to:

installDir/**vxworks-6.x/target/config/55mv5100**

4

Terminology and Usage

Standard Usages

Wind River has standards for such usages as acronyms, spelling, numbers in text, emphasis, and IP addresses.

- For usage of problem words, see Problem Words Reference on page 55.
- For usage of common technical terms, see Special Terms on page 96.
- For a list of common acronyms, see Acronyms on page 36.
- For Wind River usage when words have multiple standard spellings, see:
  - Spelling List on page 74
  - *Special Spelling Issues*

- For a comprehensive list of special terminology used in Wind River technical documentation, see the *Wind River Technical Publications Technical Glossary*.

Related Links

Numbers on page 32
Spell out numbers from one to twelve. For 13 and above, type the numeral.

Percent on page 32
Use the % symbol unless the word must be spelled out for context or clarification.

Emphasis on page 32
Use italics sparingly to add stress; otherwise they lose their power to attract attention.

IP Addresses on page 33
Do not use actual Wind River IP addresses in documentation. Doing so risks a security breach.

Numbers

Spell out numbers from one to twelve. For 13 and above, type the numeral.

For example:

The OS has *twelve* registers. Each register has *256* bits.

Related Links

Standard Usages on page 31
Wind River has standards for such usages as acronyms, spelling, numbers in text, emphasis, and IP addresses.

Percent

Use the % symbol unless the word must be spelled out for context or clarification.

For example:

If the resource usage exceeds 80% then the following must be applied.

NOTE: If the word must be spelled, the proper spelling is **percent**.

Related Links

Standard Usages on page 31
Wind River has standards for such usages as acronyms, spelling, numbers in text, emphasis, and IP addresses.

Emphasis

Use italics sparingly to add stress; otherwise they lose their power to attract attention.

The most effective technique for emphasis comes from precise wording and simple construction. One time-honored device is to place words or phrases to be emphasized towards the end of a sentence. See Active and Passive Voice on page 18.

To emphasize text, use only the appropriate inline markup:

Tool	Inline Markup
DITA	`<i>text</i>`
FrameMaker	emphasis character tag

The current output standard displays emphasis in italics. Never apply italic formatting directly (for example, by using the "I" button or equivalent keystrokes in FrameMaker).

Related Links

Standard Usages on page 31
Wind River has standards for such usages as acronyms, spelling, numbers in text, emphasis, and IP addresses.

IP Addresses

Do not use actual Wind River IP addresses in documentation. Doing so risks a security breach.

Use addresses from the following range (TEST-NET-3) for addresses that represent publicly available IP addresses:

203.0.113.0 through 203.0.113.255

Wind River documentation has many examples of IP addresses on internal networks (see, for example, the VxSim networking documents). In this case, use an address from the following private network range:

192.168.0.0 through 192.168.255.255

For more information on IP addresses in general, see:

https://en.wikipedia.org/wiki/IPv4

Related Links

Standard Usages on page 31
Wind River has standards for such usages as acronyms, spelling, numbers in text, emphasis, and IP addresses.

Special Terms

Always define special terms when they are introduced if there is any doubt as to whether they are generally accepted or understood.

When introducing a special term, or referring to a term as a word, always show the term in italics by applying the character tag word in FrameMaker or the inline element <term> in DITA.

The following terms are used variously in our industry, or in some cases are a source of confusion:

API

The term API (application programming interface) refers to a collection of functions. Do not use it to mean a single function or call. It is incorrect to say "The doorClose( ) API closes the door." Instead, say "The doorClose( ) function closes the door."

argument

See parameter.

argument list

The set of arguments that is given to a command or routine when it is executed. Compare parameter list.

baud, baud rate

Strictly speaking, baud is a rate, which means that baud rate is redundant. However, baud rate is common parlance and used widely, and we have interface parameters with names like baudrate; therefore, the term baud rate is acceptable.

decrypt

See encrypt.

deprecated

A feature that has been deprecated is still supported in the current release, but its use is discouraged and it will not be supported in future releases. Declaring that a feature has been deprecated serves as a warning to customers that they should seek an alternative solution. Do not be specific about when or in what release support for a deprecated feature will be discontinued.

Note that it is not correct to say that a feature will be deprecated in the future, because future discontinuance is built into the definition of deprecate. Use the present perfect and say a feature has been deprecated. Do not say a feature is deprecated.

disc

Use only to refer to a CD or DVD, not a hard disk.

disk

Use only to refer to a hard disk or floppy disk, not a CD or DVD.

encrypt

Do not use unencrypt to mean the opposite of encrypt; use decrypt instead. However, it is acceptable to say something is unencrypted, but that should only mean that something is simply not in an encrypted state, rather than actively deciphered or decoded.

endianness

This is a legitimate term that refers to the byte order characteristic of specific processors—big-endian and little-endian.

function

For C language, use the term function rather than routine. This is a change to previous usage.

Do not use this word to indicate the purpose of some software component or to describe how something operates. For this purpose, use *functionality* or *capability*.

function pointer

Continue to use the term function pointer. This is the most commonly understood term for this element, and it also corresponds with the FUNCPTR data type and other similarly named data types.

interpreter

See shell.

interrupt service routine, ISR, interrupt handler

Although these terms are synonymous, use interrupt service routine and ISR for consistency.

operating systems

When possible, spell out this term. When you must abbreviate, use OSs.

parameter

The terms parameter and argument are similar. However, argument is more narrowly used to refer either to command-line options passed to a command or to the values or options that may be given to a function when it is called. The term parameter is used not only to refer to arguments but more broadly to refer to options that may be set in any number of ways, as, for example, in a configuration file.

parameter list

The set of possible parameters or argument types that may be given to a command or routine when it is executed. Compare argument list.

release

Use release to indicate the numbered iteration of a product. In other words, “VxWorks 6.2” refers to release 6.2. Use version to refer to other variants of a product, such as host or architecture, but do not use version to indicate a release. Use version also to refer to host or target variants of a book.

Although these words are often used synonymously in the industry, the term release more narrowly reflects the notion of the product’s level of revision as it is made publicly available, while version is more loosely used to mean not only release, but variation by host, target, architecture, and other differentiating factors.

Note that other departments at Wind River are not always consistent and occasionally use version to mean release.

routine

See function.

run-time, run time

Do not use the word run-time as a noun meaning run-time component. The word is an adjective, as in run-time environment, run-time version, and run-time activity. The term run time (with a space) is a noun meaning the time period in which a program runs or the amount of time needed to run a program (in contrast to compile time.)

shell

A shell is the command-line interface for human interaction with a system. An interpreter is the software that converts what the user enters into action. An interpreter is similar to a compiler, except that the compiler works on files and the interpreter works on individual commands or on scripts containing individual commands.

VxWorks provides a shell that runs on the host and a kernel shell that runs on the target device. UNIX and Linux provide shells. Each shell provides a default interpreter (for example, csh or Bash) but may support other interpreters as well.

Correct:

- Use this set of commands with the C interpreter and that set of commands with the Tcl interpreter.
- Switch from the Tcl interpreter to the C interpreter by entering "C" at the shell prompt.

- There are several standard shells available in Linux: sh, bash, dash, csh. Other shells are available as well, but not as popular.
- Type exit and a carriage return to exit the C interpreter and return to the shell.

Incorrect:

- command interpreter shell
- command shell interpreter
- shell interpreter
- "Type the ls command to the interpreter."

supported/unsupported

See supported/unsupported.

unencrypted

See encrypt.

USB flash drive

Always use USB flash drive when referring to a USB data storage device with flash memory. Do not use the terms “stick” or “thumb-drive”.

A USB flash drive should not be confused with Live USB. Live USB is a USB flash drive or hard disk drive that contains an operating system and can be booted. It may also be used for product installation.

version

See discussion of release, above.

Acronyms

This section covers proper usage of letter abbreviations, whether they are pronounced as words (ANSI) or letters (API).

All acronyms should be defined at first usage, except for widely recognized acronyms such as those included in the *Common Acronyms* table in Acronyms That Can Be Used without Defining on page 37. At first usage, the full definition should be given, followed by the acronym in parentheses; for example:

Internet Control Message Protocol (ICMP)

board support package (BSP)

real-time process (RTP)

- If the first usage of an acronym appears in a heading, do not define it in the heading; rather, define it in the text that follows the heading.
- In topic-based writing, the first usage in a topic must be defined. In chapter-based writing, it is acceptable to define it the first time it is used in a book or chapter.

When giving the full definition, do not artificially capitalize letters in mid-word to indicate how the acronym was derived. In particular, note that definitions for many acronyms should not be

capitalized at all (see examples in the common acronyms list in *Common Acronyms* table in Acronyms That Can Be Used without Defining on page 37. In general, acronyms that stand for proper nouns should be capitalized, while those that stand for regular nouns should not be. If you are uncertain of an acronym's definition or whether it should be capitalized, look it up in standard references such as:

- *Microsoft Manual of Style for Technical Publications (Microsoft Press)*
- *Read Me First! A Style Guide for the Computer Industry (Sun Technical Communications)*

Related Links

Acronyms That Can Be Used without Defining on page 37

Acronym Plurals on page 38
Form the plural of an acronym by adding lower-case "s". For example, the plural of CPU is CPUs.

Acronym Special Cases on page 39
Certain acronyms are Wind River-specific or otherwise used in specific ways in Wind River documentation.

Commonly Misused Acronyms Reference on page 40
Certain terms are often treated incorrectly as lowercase acronyms in Wind River documentation.

Acronyms That Can Be Used without Defining

Table 1 **Common Acronyms**

Acronym	Definition
ASCII	American Standard Code for Information Interchange
ANSI	American National Standards Institute
API	application programming interface
CPU	central processing unit
DOS	Disk Operating System
EOF	end-of-file
FCC	Federal Communications Commission
FTP	File Transfer Protocol
GB	gigabyte(s) (For usage, see Acronym Special Cases on page 39.)
GiB	gibibyte(s) (For usage, see Acronym Special Cases on page 39)
GUI	graphical user interface
IEEE	Institute of Electrical and Electronics Engineers
I/O	input/output

Acronym	Definition
IP	Internet Protocol
KB	kilobyte(s) (For usage, see Acronym Special Cases on page 39)
KiB	kilibyte(s) (For usage, see Acronym Special Cases on page 39.)
LAN	local-area network
MB	megabyte(s) (For usage, see Acronym Special Cases on page 39.)
MiB	mibibyte(s) (For usage, see Acronym Special Cases on page 39.)
NFS	Network File System
PDF	Portable Document Format
PPP	Point-to-Point Protocol
PTY	pseudo terminal device
RAM	random access memory
ROM	read-only memory
RSH	Remote Shell
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TTY	teletypewriter (terminal device)
URL	Uniform Resource Locator
WAN	wide-area network

Related Links

Acronyms on page 36
This section covers proper usage of letter abbreviations, whether they are pronounced as words (ANSI) or letters (API).

Acronym Plurals

Form the plural of an acronym by adding lower-case "s". For example, the plural of CPU is CPUs.

Treat the plurals of acronyms that end in S the same way. For example, the plural of OS is OSs.

- Do not use an apostrophe.

  WRONG: *CPU's*
- Do not use upper-case "S".

WRONG: *CPUS*

Related Links

Acronyms on page 36
This section covers proper usage of letter abbreviations, whether they are pronounced as words (ANSI) or letters (API).

Acronym Special Cases

Certain acronyms are Wind River-specific or otherwise used in specific ways in Wind River documentation.

KB, MB, GB, KiB, MiB, GiB

Use these abbreviations only when expressing a measurement with numerals. Put a space between the numeral and the acronym unless the measurement is an adjective that precedes a noun, in which case separate with a hyphen, per the standard for compound modifiers. Examples:

- *The remaining disk space is 400 MB.*
- *The application requires a 10-GiB hard disk.*

These acronyms are both singular and plural; for example, do not add s to KB to mean kilobytes.

Lowercase and Mixed-Case Acronyms

Wind River uses several special lowercase and mixed-case acronyms, or acronym-like constructions, most of which are company-defined terms.

Table 2 **Lowercase and Mixed-Case Acronyms**

Acronym	Definition
bss	segment of memory that holds uninitialized variables
fd[1]	file descriptor
cdromFs	CD-ROM file system
dosFs	DOS-like file system
rawFs	raw file system
TrueFFS	True Flash File System
mbuf	
zbuf	type of data buffer that can be shared between separate software modules

[1] The acronym fd is not really a lowercase acronym, but a standard placeholder that has come to be used as a word in VxWorks documentation. For historical reasons, continue applying word to this term, except in code examples where it requires placeholder.

Acronym	Definition
varbind	

Related Links

Acronyms on page 36
This section covers proper usage of letter abbreviations, whether they are pronounced as words (ANSI) or letters (API).

Commonly Misused Acronyms Reference

Certain terms are often treated incorrectly as lowercase acronyms in Wind River documentation.

standard I/O library

Do not use stdio as an acronym for the C-language standard I/O library. If you need a shorthand way of referring to the library, refer to the associated header file, **stdio.h**.

ioctl( )

Do not use *ioctl* as an acronym when referring to the special **ioctl( )** functions; just use the routine name **ioctl( )**.

TTY and PTY

Do not use *tty* and *pty* as alternatives for these acronyms.

ROMFS

When referring to the file system, spell it uppercase. However, use *romfs* for literals associated with it (directory names, device names, and so on.).

Related Links

Acronyms on page 36
This section covers proper usage of letter abbreviations, whether they are pronounced as words (ANSI) or letters (API).

Function Names and Other Proper Code Elements

When referring to function names and other proper code elements always use the formatting specified in Proper Adjectives on page 20.

Processor and Board Names

Processor Names

Processor names usually correspond as closely as possible to the names given by their suppliers.

However, in naming a family of processors, Wind River style sometimes deviates from the manufacturer's.

- Do not use the term "x86" to refer to the Intel family of 32-bit processors; use "IA-32".
- Do not use the name "Pentium" to refer to Pentium processors generically; use "Pentium" only when identifying specific processors, such as Pentium III, Pentium M, and so on.

Note that there are still a variety of places in code, pathnames, and similar where you might see naming conventions such as "x86-win32" or "i86". When referring to these elements, treat them as literals and show them exactly as they appear in the software.

The table shows the correct names for most supported target architectures.

Table 3 **Processor Names**

Supplier	Family Name	Processors
ARM Ltd., Freescale Semiconductor, other ARM licensees	ARM	ARM v5 and ARM v6 cores, including ARM926EJ-S (ARM9E family) and ARM1136JF-S (ARM11 family)
Intel	IA-32 (For Intel 32-bit processors. Do not use the term "x86".)	Intel386 family processors, Intel486 family processors; Pentium, Pentium II, Pentium III, Pentium 4, Pentium D, and Pentium M family processors; Xeon and Celeron variants
Intel	Intel XScale	Intel IXP425
Broadcom, MIPS Technologies, NEC, PMC-Sierra, Toshiba	MIPS	BCM1250; MIPS32 4Kc, MIPS32 4KEc, MIPS32 24Kc, MIPS64 5Kc, MIPS64 5Kf; VR5000 Series (VR5500); RM9XXX (E9000 Core); TX4938, TX4927
Freescale Semiconductor, IBM, AMCC	PowerPC	MPC7XX (MPC745, MPC755), MPC7XXX (MPC7410, MPC7445, MPC7455), MPC8XX (MPC823, MPC823E, MPC850, MPC855T), MPC82XX (MPC8272,

Supplier	Family Name	Processors
		MPC8280), MPC83XX (MPC8349E), MPC85XX (MPC8540, MPC8560); PowerPC 604e, 405GP, 405GPr, 440EP, 440GP, 440GX, 970
Renesas Technologies (formerly Hitachi)	Renesas SuperH	SH-4 and SH-4A cores (SH7750, SH7751)

Board Names

Board names usually correspond as closely as possible to the names given by their suppliers.

For example, MV135 is not an acceptable name for the MVME135. The table shows the correct names for some example VxWorks targets. For a complete list of currently supported targets and their correct names, see the list maintained by Marketing.

In VxWorks target-specific documentation, when two or more boards are covered by the same BSP, and the portion of their names that differs is separated by a slash (/) or a hyphen (-), the portions that differ can be repeated, each separated by a comma and a space (see the examples in the table).

Table 4 **Example Board Names**

Supplier	Board
Force	SYS68K/CPU-21, -29, -32
Heurikon	HK68/V10, VF, VE
Motorola	MVME166, MVME167

Product Names

The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

The following topics describe standards for referring to Wind River products, covering the following issues:

- formal and short-form names for products
- whether and when to use product names
- when and how to use trademark symbols
- when and how to use release numbers

The master product-names file defines variables for each product name. It is maintained regularly whenever new products are added or existing products are renamed. The file also provides the latest information on appropriate trademark symbols, indicates which names

should be preceded with the article “the”, and defines the basenames for clearcase document directories and PDF file names.

Related Links

Generic References to Platform Products on page 43
When referring to “Platform products” generically, capitalize just the first “P”. Or, you can write “Wind River Platforms”.

Alternatives to Large Product Names on page 43
Many Wind River product names are large and cumbersome; avoid overuse of the long product name.

Considerations in Sentences on page 44
Be aware that changes in product names may affect how they are used in sentences.

Trademarks and Copyrights on page 44
Do not insert trademark (™, ®) or copyright (©) symbols in body text.

Release Numbers on page 45
Product release numbers must sometimes be shown when referencing a product name, a pathname, or a book title.

Generic References to Platform Products

When referring to “Platform products” generically, capitalize just the first “P”. Or, you can write “Wind River Platforms”.

For first usage of “Wind River Platforms”, a registered trademark symbol is required after “Wind River”. (It should not replace the space.)

However, typically the first-usage trademark for “Wind River” will already have been taken care of on the title page by virtue of its appearance in a Platform product name like Wind River® Platform for Whatever.

Related Links

Product Names on page 42
The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

Alternatives to Large Product Names

Many Wind River product names are large and cumbersome; avoid overuse of the long product name.

In general, to counteract issues of readability and obtrusiveness with product names that are large, ask yourself the following questions when composing text:

- Is it really necessary to use a product name here?
- Is there a generic alternative to the product name that could be used here?

- Is there a sanctioned short-form name and is there a variable for it? See **var_product_names.fm**.

Related Links

Product Names on page 42
The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

Considerations in Sentences

Be aware that changes in product names may affect how they are used in sentences.

Some names require the use of the article "the"; others do not. See the "The" columns in var_product_names.fm. If a product name changes, be sure that you not only change the variable or keyword, but also check the sentence structure.

Related Links

Product Names on page 42
The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

Trademarks and Copyrights

Do not insert trademark (™, ®) or copyright (©) symbols in body text.

The boilerplate copyright/trademarks template (Copyright.fm) provides the legalese that covers usage of trademarked names and intellectual property. Be aware that the list of Wind River products covered on the standard copyright/trademarks page is not exhaustive and is not intended to be.

Use trademark symbols on covers and title pages only. Do not try to use trademarks symbols with the first mention of the full name in a body paragraph or in a running header.

The following two Wind River names must always carry a registered trademark symbol (®) when they appear on covers or title pages:

- *Wind River®*
- *VxWorks®*

Other Wind River names that carry either the registered trademark symbol or the claimed trademark symbol are no longer under development. However, for reference, these names and their correct trademark symbols are listed in the following file:

T:\templates\var_product_names.fm

NOTE: The ® symbol is used only for trademarks that are registered. The ™ symbol is used for trademarks that are unregistered but claimed. If you are uncertain whether Wind River has registered the trademark for a product you are documenting, ask the product manager.

When product names appear in a plain-text file, such as a readme file, trademark symbols must still appear at first usage. Substitute ® with (R) and ™ with (TM). For example: Wind River(R) Platform for Network Equipment.

In unusual cases, third-party agreements may require the explicit listing of third-party trademarks. Consult with your Marketing project manager if you are uncertain. Copyrighted materials should be cited in bibliographical sections or appendices.

For details, see *Referring to Other Content*.

Related Links

Product Names on page 42
The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

Release Numbers

Product release numbers must sometimes be shown when referencing a product name, a pathname, or a book title.

Depending on circumstances, it may be most appropriate to show a product's entire release number, a partial release number, or no release number at all. This section discusses those circumstances, and how release numbers should appear.

When to Include Release Numbers

References to release numbers should be as general (vague) as possible without compromising the precision required to ensure proper understanding and use of the products in question. The order of preference is as follows:

1. Do not mention a release number, if possible.
2. If a release number is necessary, but it is satisfactory to mention only the top-level release number, include the minor-level release number as a lowercase "x" tagged with the placeholder character tag. For example:
   - Foo is compatible with VxWorks 6.*x*.
   - The file is located in *installDir*/**vxworks-6.***x*/**target/config**.
   - The file is located in *installDir*/**Workbench-2.x/wrwb/2.***x*/**plugins**.
   - The file is located in *installDir*/**components/webcli-4.***x*/**config**.
3. Indicate additional levels of the release number as necessary. For example:
   - Foo is compatible with VxWorks 6.2.
   - See *Wind River Foo Programmer's Guide, 4.6*.

The full release number is often required when discussing migration issues.

Using a Comma with Release Numbers

While it is preferable to show a product name and release number simply as "Foo 1.2", in some cases this can be ambiguous and a comma is required for clarity.

Consider the following example:

Wind River ICE for Wind River Workbench 2.2

In the above, does "2.2" refer to a release number for Wind River ICE or for Wind River Workbench? It should refer to the former, which is the same as saying it should refer to the entire string "Wind River ICE for Wind River Workbench". To make this clear in such situations, add a comma before the release number. For example:

Wind River ICE for Wind River Workbench, 2.2

Also consider the following example:

Wind River General Purpose Platform, VxWorks Edition 3.1

In the above, does “3.1” refer to a release number or an edition number? It should refer to the former. In other words it should refer to the entire string “Wind River General Purpose Platform, VxWorks Edition”. Again, to avoid ambiguity, add the comma before the release number. For example:

Wind River General Purpose Platform, VxWorks Edition, 3.1

The purpose of the comma is to remove as much ambiguity as possible about what the release number applies to—the entire product name or just the final portion.

For information about indicating a product release number associated with the reference to another Wind River manual, see *Referring to Other Content*.

Related Links

Product Names on page 42
The complete list of standard Wind River product names appears in the master product-names file **T:\docs\templates\var_product_names.fm**.

Placeholders

A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Always ensure that there is a clear distinction between placeholders and literal names of software elements by tagging them correctly.

NOTE: Placeholders are typically associated with command arguments, variable elements in pathnames, and function parameters. For function parameters, assigning placeholders has been a source of some confusion, because from the standpoint of code authors, these words are literal (they are the names of variables in the source code they wrote). We italicize function parameters because when we show a function declaration, they are placeholders for whatever arguments the application programmer provides. Thus, we treat them as placeholders.

WARNING: Do not treat environment variables as placeholders. For example, the environment variable **WIND_HOME** may be accessed as **$(WIND_HOME)**, but neither of these expressions are placeholders from the standpoint of documentation—they are literal strings understood by the software.

Related Links

Placeholder Format on page 47
User the proper character tags or elements to tag placeholders. Placeholders will be rendered in italics in the output.

Placeholder Naming Conventions on page 48
If you must create a new placeholder, you must choose a name based on the principles of clarity, consistency, and conciseness, in priority order.

Placeholders Made Up of Multiple Words on page 49
If a placeholder name is to be made up of multiple words, join them together in "camel-caps" style.

Placeholders That Represent Separate Elements on page 49
Use underscores if a placeholder represents separate elements normally connected by underscores.

Multiple Installation Directories on page 49
In some cases you need to distinguish between multiple installation directories, in which case installDir is not sufficient.

Standard Placeholder Names Reference on page 50
Where possible, use the standard names for common placeholders.

Placeholder Format

User the proper character tags or elements to tag placeholders. Placeholders will be rendered in italics in the output.

Do not use the angle-brackets "<" and ">" to indicate a placeholder, even if a developer gives you text that indicates placeholders in this way. Note that plain-text files such source-code documentation that is to be processed by apigen are an exception to this rule. In such cases the markup for placeholders is in fact "<" and ">". However, when authoring documentation in FrameMaker, DITA, or any format that supports italics, do not use the angle brackets for placeholders.

Use one of the following methods to tag your placeholders:

Authoring Tool	Method
FrameMaker	Use the placeholder character tag.
DITA / oXygen	Use the **<varname>** element.

Use the same placeholder format for a syntax, code, or execution example display that otherwise appears in a fixed-width font; for example:

```
% eval 'installDir/wrenv.sh -p platform -o print_env -f shell'
```

In FrameMaker, be sure not to apply the placeholder tag to space characters; doing so alters the width of the space characters, which should match the fixed-width default character spacing.

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Placeholder Naming Conventions

If you must create a new placeholder, you must choose a name based on the principles of clarity, consistency, and conciseness, in priority order.

In many cases, a standard name may already exist that is equivalent to the placeholder you need, in which case, you should use that name; see Standard Placeholder Names Reference on page 50.

Clarity

Ensure above all that the name you choose is clear and in no way obscure or vague for the context in which it is to be used. While maintaining clarity, then strive for the consistency and conciseness. Keep in mind that in a given situation, clarity might require that you deviate from a standard placeholder. For example, instead of using a standard name like *projName*, you might need to clarify that you mean a new project and therefore would need to use the following instead:

installDir/**workspace/***newProjName*/**project.properties**

Consistency

Ensure that the name is consistent with placeholder names that serve the same purpose in the rest of your book, and if possible with like names in other related books and the documentation set as a whole. If you use *projDir* in one instance, and *projectDir* in another, readers might well wonder whether these are the same or different directories.

Also strive to choose placeholder names that are consistent with the general naming practices used elsewhere in your book and in the standard. For example, a name like *projectDir* would not be consistent with the existing standard for the similar placeholder *projName*.

Conciseness

Placeholders should be as short as possible while still clear and consistent. This is particularly important given that long path names and command lines are often unavoidable. Remove any words that are unnecessary. For example, *myTargetBoard* could likely just as well be

targetBoard. Where possible, abbreviate words, as long as doing so does not compromise clarity.

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Placeholders Made Up of Multiple Words

If a placeholder name is to be made up of multiple words, join them together in "camel-caps" style.

That is, capitalize the first letter of each word except the first; for example, *installDir*. Use the camel-caps style even if the placeholder is part of an all-caps literal, such as a constant or component name; for example, **INCLUDE**_*commandName*_**CMD**. If the initial portion of a placeholder is actually a literal—as in *objArch*—capitalize the placeholder portion as if it were the second word in a normal placeholder.

In rare cases, clarity may demand a placeholder name longer than five words. In such cases, separate the words with underscore characters and do not use capitalization; for example, *new_buffer_name_for_func_pointer*.

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Placeholders That Represent Separate Elements

Use underscores if a placeholder represents separate elements normally connected by underscores.

For example, in the following path, **mv5100_diab** is a literal that would be represented by the placeholder *bspName_toolChain:*

installDir/vxworks-6.*x*/target/proj/mv5100_diab

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Multiple Installation Directories

In some cases you need to distinguish between multiple installation directories, in which case installDir is not sufficient.

In most cases, you will use *installDir* to indicate the root directory where the user has installed the software. However, in some cases, such as in migration information, a manual may need to make reference to more than one product, and the installation directories may be different for the different products. In such cases, use *installDir* as a template when coining placeholder names for other installation directories.

Ensure that the *installDir* placeholder reflects that difference by taking the following approach:

- Always use the placeholder *installDir* for the current product, regardless of whether an older product is mentioned. Docs, even migration guides that describe older products, are always written from the perspective of a current product.
- In cases where you need to refer to an installation directory for an older product, use a special version of *installDir* that IDs the older product.

  For example, to refer to a Tornado installation directory in a Workbench migration guide, use a placeholder called *installDirTor*.

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Standard Placeholder Names Reference

Where possible, use the standard names for common placeholders.

arch

Architecture family, typically appearing in directory pathnames.

bspName

Wind River name for a board support package (BSP), typically appearing in directory pathnames. Use this instead of *bspname*, *bsp_name*, *platform*, or *yourTargetBoard*. Example:

installDir/**vxworks-6.***x*/**target/config/***bspName*/**target.ref**

bspName_toolChain

This case uses an underscore that represents the underscore in a multipart placeholder that standards for a BSP name such as mv5100_diab. Example:

installDir/vxworks-6.*x*/target/config/*bspName_toolChain*

comPort

COM port. Use instead of comport or com_port. Example:

```
tgtsvr -V -d comPort -bps speed -B wdbserial ipAddr
```

cpuType

CPU name, typically appearing in compiler command syntax and more specific than arch. Example:

```
make CPU=cpuType TOOL=toolChain
```

hostType

Host type, typically appearing in pathnames for host-specific executables. Do not use host_type. Example:

installDir/**workbench-2.x/foundation/4.x/hostType/bin/wtxregd**

installDir

Root directory where the user has installed the software, typically appearing in pathnames. Do not use **WIND_BASE** or similar environment variables.

ipAddr

IP address. Use instead of *ipaddress* or *ip_address*. Example:

```
tgtsvr -V -d comPort -bps speed -B wdbserial ipAddr
```

projName

Project name. Use this placeholder instead of *projname*, *projectName*, or *project_name*. Example:

installDir/**vxworks-6.***x*/**target/proj/***projName*

runtimeDirJboss

Follows the *installDir* format by replacing install with runtime. Example:

/wind/*runtimeDirJboss*/**jboss-4.0.3SP1/server/default/dsmconfig**

toolChain

Compiler tool chain type, often appearing in compiler command syntax. Example:

```
make CPU=cpuType TOOL=toolChain
```

Related Links

Placeholders on page 46
A placeholder is meant to indicate text that is not to be interpreted literally, but rather as a stand-in for some element that will vary depending on context.

Unacceptable Usages

In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

This means some common usages are not acceptable in technical writing.

Related Links

Excluded Articles on page 52
Do not use abbreviated English by excluding articles (the, a, an).

Contractions on page 53
Do not use contractions; for example, *don't*, *doesn't*, *can't*, *it's*.

Latin Words and Abbreviations on page 53
Do not use Latin words and abbreviations. Choose English-language alternatives.

Possessive Forms on page 54
In general, avoid using the possessive. Rewording the sentence usually communicates more clearly and gracefully.

Colloquialisms and Jargon on page 54
Colloquialisms are unacceptable.

Hype on page 54
Do not praise the product or try to sell it.

Vague Modifiers on page 55
Avoid quantifying modifiers that are imprecise: a few is somewhat definitive, but quite a few and a considerable number can mean almost any number.

Problem Words Reference on page 55
Certain words are often misused in technical writing. Wind River has standards for how they should be used.

Excluded Articles

Do not use abbreviated English by excluding articles (the, a, an).

Always use articles as required by standard English usage.

However, there is one major exception: reference-entry title lines (under NAME) may be too long to fit comfortably on a single line, particularly when the routine or library name is long. This may

cause the title to wrap in the table of contents. In such cases, abbreviating by excluding articles is an acceptable and necessary practice.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Contractions

Do not use contractions; for example, *don't*, *doesn't*, *can't*, *it's*.

Contractions are informal and are often misunderstood by non-native English speakers.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Latin Words and Abbreviations

Do not use Latin words and abbreviations. Choose English-language alternatives.

The following substitutes are useful:

- *etc.*:

  and so on, and so forth, among others

- *e.g.*:

  for example, such as

- *i.e.*:

  that is

- *via*:

  through, by way of, using, with

One reason for this rule is that readers commonly confuse the distinction between *e.g.* and *i.e.*

Note that for example and that is should precede, not follow, the phrase to which they refer and should generally be preceded by a semicolon. They should also be followed by a comma or colon. For example:

This parameter must be a power of 2; that is: 1, 2, 4, 8, and so on.

Avoid constructions that use and so on (and its relatives) unless necessary. Lists preceded with such as are often a reasonable alternative. Never conclude a *such as* list with *and so on* (or its relatives).

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Possessive Forms

In general, avoid using the possessive. Rewording the sentence usually communicates more clearly and gracefully.

Consider the following worst-case example:

VxWorks's standard I/O package...

This could just as well be written less awkwardly as follows:

The VxWorks standard I/O package...

If you must use an apostrophe, use it correctly. For more information, see Apostrophe on page 24.

Keep in mind that the possessive form of *it* is always *its*. The word *it's* is a contraction which is not permitted by the standard. For more information, see Contractions on page 53.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Colloquialisms and Jargon

Colloquialisms are unacceptable.

Avoid jargon. However, technical terminology is acceptable and desirable.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Hype

Do not praise the product or try to sell it.

Maintain a neutral tone; the purpose of manuals is to describe the technology clearly and completely.

Avoid words like *cheaply*, *conveniently*, *easily*, *powerful*, and *intuitive*.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Vague Modifiers

Avoid quantifying modifiers that are imprecise: a few is somewhat definitive, but quite a few and a considerable number can mean almost any number.

Avoid such qualifying adverbs as *quite*, *rather*, and *very*. In theory, they strengthen what they modify; in practice, they dilute it. Ironically, the impact increases when these qualifiers are removed. Consider the following use of *very* and *quite*:

Because VxWorks primitives are very fast, the overhead associated with low-level calls to the I/O system is quite small.

With the qualifiers removed, the sentence reads as follows:

Because VxWorks primitives are fast, the overhead associated with low-level calls to the I/O system is small.

The statement is now more direct and to the point, with no hint of equivocation.

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

Problem Words Reference

Certain words are often misused in technical writing. Wind River has standards for how they should be used.

and/or

Avoid the use of this stilted construction. Whether "or" is exclusive or inclusive is nearly always evident from context.

like/as

Use like to compare objects or ideas. Use as to show similarity in actions. In typical fashion, the tobacco industry provides a good example of what not to do:

Winston tastes good like a cigarette should.

Although the form of this colloquialism is widely used, avoid it in formal English.

that/which

These relative pronouns are sometimes interchangeable, but not always. The confusion between the two generally stems from confusion about the two types of clauses they introduce, the restrictive/defining clause versus the non-restrictive/parenthetical clause. For details, see That/Which on page 21.

data

Treat the word data as singular. Historically, this word has been considered plural, but is now well established as either a singular and plural noun.

he/she (pronoun gender)

Is a programmer male or female? While the spirit of equality should be respected, the use of his/her or he/she and similar constructions are graceless and self-conscious and should be avoided where possible. The simplest way around this is to make the antecedent noun plural (change programmer to programmers), so that the pronoun can be they, them, or their.

however

Do not position the word however in the middle of a clause (an interruption) or at the end of a clause (too late to offer much value). However is a powerful word for clarifying contrast; making it the first word in a contrasting clause is the best way to prompt the reader that what you are about to do is show the other side of the coin.

iconize

Do not use. This is a colloquial term for minimize.

it

When the word *it* requires no antecedent it is known as an *indefinite it*. Avoid starting sentences with an indefinite it that adds nothing to the meaning; for example:

- *It will be seen that...*
- *It is the purpose of...*
- *It is important to consider that...*

Avoid using an indefinite it along with other instances of *it* that do require antecedents, in the same paragraph as well as the same sentence:

It is up to the user to know how this could affect it.

please

Do not use the word *please* when instructing users to do something.

google

Do not use the word *google* as a verb. Use *search online* to indicate that something should be searched for from a browser. Only use *Google* as a proper noun.

recommend

INCORRECT:

We recommend...

It is recommended...

CORRECT:

Wind River recommends...

should

Avoid using the word *should* in ways the suggest equivocation or lack of firmness. Do not is considerably stronger than should not. For example:

WEAK:

The **-nostdinc** flag should not be used with the current release since it prevents the compilers from finding headers such as **stddef.h**.

STRONG:

Do not use the **-nostdinc** flag with the current release. It prevents...

In cases where a should sentence cannot be reworded in the imperative, the word must is a stronger choice than should.

so

Do not introduce a clause with the word so (without that). Doing so is a colloquialism. Even though its meaning in conversation may be made clear by inflection, in print it can be ambiguous. Consider the following example:

In the example, no close( ) routine has been specified, so no driver routines are called.

Does the writer mean *therefore no driver routines are called or so that no driver routines are called*? The comma suggests the former, but explicitly using *thus*, *therefore*, or *so that* would have left no doubt as to the intended meaning.

supported/unsupported

To say that a feature or function is supported is to say that it is part of Wind River's product or service and that Customer Support is available. The converse is true for *unsupported*: it is not part of our product or service and Customer Support is not available.

Do not use *unsupported* to mean that something does not work; in such cases, be direct about the fact that it does not work

use

Do not overuse the word *use*. Look for alternative words, such as *with*, or verbs that are more descriptive. For example, a routine can be *called* instead of *used*.

user

Do not write user in place of you; do not speak of the reader in the third person.

INCORRECT:

The user should...

CORRECT:

You must ...

we/our

Do not use first-person pronouns, even the impersonal we. For example:

INCORRECT:

As we have seen in the examples above...

Our online support site provides...

We recommend...

Contact us for...

CORRECT:

As illustrated in the examples above...

Wind River Online Support provides...

Wind River recommends...

Contact Wind River...

Contact your sales representative...

wish

Do not use the word *wish*; use *want* instead.

you, your

Addressing examples to the reader by writing *you* and *your* is acceptable; in fact, in distinctly tutorial documentation, these words are desirable. However, apply them sparingly and do not overuse. While judicious use of *you/your* can engage the reader, excessive or unnecessary use can easily become a distraction.

Avoid using a you construction when a simple imperative statement does the job more succinctly. For example, instead of "you configure the facility by...", write "Configure the facility by..."

Related Links

Unacceptable Usages on page 52
In technical writing, you must avoid any usage that detracts from the clarity and accuracy of the content.

5

Graphics

Standard Graphics Tools

The standard graphics tools for Wind RIver Technical Publications are Visio 2007and PaintShop Pro.

Visio 2007

Visio 2007 is a vector drawing tool by Microsoft. It is located on a virtual server. To access it, you must connect to the server using the Remote Desktop application.

Paint Shop Pro

PaintShop Pro is a raster and vector graphics editor for Microsoft Windows. It was originally published by Jasc Software. In October 2004, Corel purchased Jasc Software and the distribution rights to Paint Shop Pro. PSP functionality can be extended by Photoshop-compatible plugins.

Graphics Standards

The Wind River graphics standards help you create images that are consistent not only within the same document, but also across the documentation set.

They also allow for better reuse of images.

Terms

The discussion of graphics standards below uses the following terms:

callout

In a figure, text coupled with a line (called a leader) that points to some part of a drawing or imported bitmap.

leader

The straight line leading from a callout to that which it identifies. A leader should never be an arrow.

label

Text applied directly on a specific element that it identifies.

General Guidelines

While this topic describes standards that are primarily for diagrams, much of them apply to figures that contain screenshots as well, because screenshot figures often include callouts and other diagram elements.

Lay out diagrams and screenshot callouts in a logical, clear, comprehensible, and consistent fashion. If you can add aesthetic orderliness, do so. Copy and paste commands are valuable for achieving this. Zooming in to 200%, 400%, or greater can help you clean up edges and spaces in a drawing. Use the graphic alignment and distribution functions to arrange objects and text blocks neatly.

Line Widths

Apply the following line widths to the various elements of drawings:

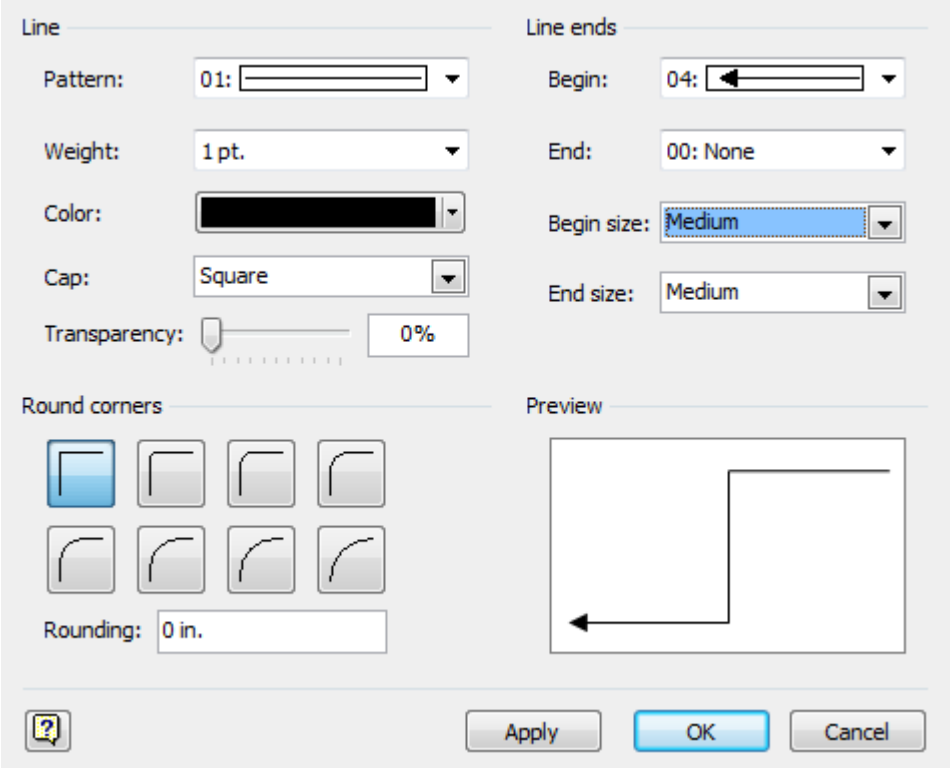
- 0.2 pt – line width for callout leaders and braces
- 0.4 pt – line width for shapes
- 1.0 pt – line width for arrows, particularly in flow diagrams

Arrows

Arrows and connector lines in flow diagrams should be formatted in Visio as follows:

Option		Value
Line		
	Pattern	01
	Weight	1 pt
	Cap	Squares
Line Ends		
	Begin	00: None (for no arrow) 04 (for beginning arrow)
	End	00: None (for no arrow) 04 (for end arrow)

Option		Value
	End size	Medium (default)
	Begin size	Choose size as appropriate based on the scale of the drawing.



Colors

Do not apply colors to elements in drawings.

Figure Labels

Use Arial font. Tailor the size as necessary, generally from 8 to 12 points, but never less than 7 points. For emphasis, you can apply bold manually.

Do not create white labels on black or dark gray objects.

Callouts

If a callout is a complete sentence, start it with a capital and end it with a period. However, most callouts are simply noun phrases that identify elements; do not treat these as sentences, and capitalize words only as necessary.

Draw callout leaders at angles so that they stand out against the horizontal/vertical orientation of screenshots. If you include numbered callouts, put the number in square brackets.

Drawing Boxes

Do not apply rounded corners to boxes. In other words, do not use the rounded rectangle tool.

Screenshots

Screenshots are essential when describing software tools that have a graphical interface.

However, over-use can be as much of a problem as under-use.

Be judicious in selecting which windows and dialog boxes to show. In any given case, ask yourself whether a particular screenshot is actually necessary to explain the concepts you want to convey.

When setting up GUI elements for screenshots, adjust window size and interior panes to minimize empty space.

Draw attention to the portion of the screenshot that matters, using either of the following techniques:

- Trim out certain areas that are irrelevant to the topic and not necessary to show context. Do not trim by cropping the image in FrameMaker, because this will not carry over into HTML. Instead, do any cropping in the original image, using PaintShop Pro.
- Gray out selected areas using PaintShop Pro. This is a particularly useful technique for showing context but is also handy when trimming is not an option. In PaintShop Pro, select areas to gray out and use the brightness/contrast controls (SHIFT+B) to increase brightness and reduce contrast. The gray areas in the following example were created by setting brightness to 90 and contrast to -80.

Figure 1: Example Screenshot with Grayed Areas

Screenshots in a manual should reflect a consistent, standard window theme (the look-and-feel made up of colors, border styles, and typefaces) as follows:

- For Windows screenshots, the theme should be the standard window theme for Windows 7 in the default blue color scheme.
- For Solaris screenshots, use the standard CDE window manager.
- For Linux screenshots, use the Gnome window manager.

Never use a custom theme for screenshots.

If the text within the screenshot is relevant, ensure that the screenshot has sufficiently high resolution and is imported with a dpi setting that allows the text to be read—not only in the text editor, but in PDF and HTML. Capture Windows screenshots using PaintShop Pro. Capture Solaris screenshots using FrameMaker or xv.

Capture Linux screenshots using Gimp.

When capturing screen images, save the results in **.png** format.

Store imported images in an images subdirectory of the main document directory.

In FrameMaker, import screenshots by reference. This means that your FrameMaker file does not contain the screenshot data, but rather points to the graphics file where it resides. Graphics that are imported directly into the document result in larger files and are harder to maintain and troubleshoot.

For screenshot files, choose meaningful names created with lowercase letters with words separated by dashes. Do not include space characters in filenames.

6

Documenting the User Interface

Screen Component Labels

Whenever possible, use the label of a screen component (Ok, Save As, File) but do not mention the screen component itself (button, option, command.)

Examples:

CORRECT:

Select **File** > **New**.

Click **Ok**.

INCORRECT:

From the **File** menu, select the **New** command.

Click the **Ok** button.

Add the component itself only when it will avoid confusion, and with items that have no label, such as the  (Close) button. If present, omit the trailing ellipsis (...) from screen components.

Examples:

INCORRECT:

Click Next...

CORRECT:

Click Next...

Check Boxes

Check boxes present users with yes/no or true/false selections.

Unlike option buttons, any number of (non-conflicting) check boxes can be selected at a time.

Select and *clear* check boxes. Check box is always two words.

Examples:

CORRECT:

Select **Print Only** to File.

Clear **Skip Blank Pages** before printing.

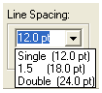
INCORRECT:

Check the Thumbnails *check box*.

Do not use *check* and *uncheck*, as check can also mean verify, and some contain an X instead of a checkmark.

Combo Boxes

Combo boxes allow users to select one of the displayed values or type a new value.

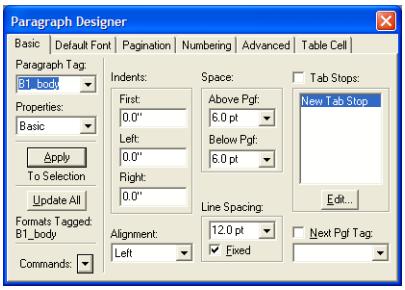


Example:

Select or type the desired line spacing.

Dialog Boxes

Dialog boxes prompt for user input, and may contain check boxes, combo boxes, and many other controls.



Open and *close* dialog boxes. *Select* options in dialog boxes. Dialog boxes *appear* and *display* information.

Examples:

CORRECT:

Open the Paragraph Designer.

The initial wizard page *appears*.

The screen *displays* a message if you do not log on correctly.

INCORRECT:

Bring up the Paragraph Designer dialog box.

Dismiss the Layout dialog.

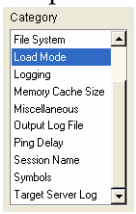
Use *appears* as an intransitive verb; use *displays* as a transitive verb.

Use the passive *is displayed* when the subject is unimportant. Example:

A message is displayed if you do not log on correctly.

Drop-down Lists

Drop-down lists allow users to select one of the displayed items; they may not type a new value.



Select an item from a drop-down list.

Example:

Select the appropriate paragraph tag from the drop-down list.

Menus

Menus can contain one or more submenus. When referring to a specific menu, menu is lowercase.

Select menus and menu items; do not use *click*. *Enable* or *disable* menu items.

Examples:

CORRECT:

Select **File** > **New**.

To *enable* bar functionality, *select* foo.

INCORRECT:

Click **File** | **New**.

Choose **New** from the **File** menu.

Mouse Actions

Mention the action; do not mention the mouse itself unless the user must do something unusual.

Examples:

Point to the window border.

Drag the folder to the Desktop.

Select Ohio and Illinois in the States list.

The term *drag-and-drop* is acceptable as an adjective, but not as a verb or noun. For the verb form, simply use the term *drag*. To convey the concept as a noun, use the term as a modifier; for example, *drag-and-drop operation* or *drag-and-drop feature*.

Highlight text with the mouse. Highlighted items are selected. Examples:

Highlight the contents of the text box to select it.

If you highlight a method in the Outline View, its text declaration is selected in the Editor.

Mouse Buttons

A mouse can have one, two, or three buttons, called mouse button, middle mouse button, and right mouse button respectively.

Click, double-click, or right-click mouse buttons. Always hyphenate double-click and right-click. Examples:

CORRECT:

Double-click the icon.

Right-click to see the shortcut menu.

INCORRECT

Double-click on the icon.

Do not combine keyboard and mouse actions with a plus sign. Examples:

CORRECT:

Hold down **SHIFT** and *click* the right mouse button.

INCORRECT

Press **SHIFT+right-click**.

Use *rotate*, not *roll*, to refer to rolling the IntelliMouse wheel. Examples:

CORRECT:

Rotate the mouse wheel forward to scroll up in the document.

INCORRECT

Roll the mouse wheel to scroll the document.

Option Buttons

An option button allows the user to choose a single option within a limited set of mutually exclusive options.

Choose option buttons. Use *option buttons* rather than *radio buttons*, but do not mention either one unless necessary for clarity. Examples:

CORRECT:

Choose **Portrait** for vertical page layout, **Landscape** for horizontal.

INCORRECT:

Choose the **Portrait** option for vertical page layout.

Radio Buttons

Do not use *radio buttons*. Use *option buttons* instead.

For more information, see Option Buttons on page 69.

Screen Buttons

Screen buttons cause an application to perform some action when clicked.

Click screen buttons. Use *the __ button* only with buttons that have no labels, such as the  (Close) and  (Minimize) buttons.

Examples:

CORRECT:

Click **Ok**.

Click the **Close** button.

INCORRECT:

Click on **Cancel**.

Reserve *press* for keyboard keys.

Scroll Bars

Scroll bars allow users to choose the direction and distance to scroll through information in a window or list box.

Scroll bar is always two words.

Example:

Use the scroll bar to adjust the view in your screen.

When it is necessary to distinguish it from the rest of the scroll bar, the movable box on the scroll bar is the scroll box.

Use *scroll* with a preposition, not as a transitive verb.

Examples:

CORRECT:

Scroll through the document to get to the end.

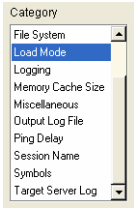
INCORRECT:

Scroll the document to get to the end.

Scroll Lists

Scroll lists are like drop-down lists but without the drop-down feature.

Scroll lists allow users to select one of the displayed items; they may not type a new value.



Select an item from a scroll list.

Example:

Select the appropriate category from the scroll list.

Text Boxes

Text boxes allow the user to type any value or text string.

Type in text boxes.

Examples:

CORRECT:

Type the filename in the text box.

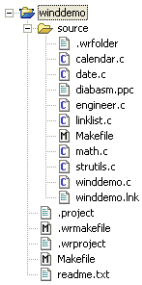
Click in the text box to place the cursor before *typing*.

INCORRECT:

Enter text in the text box.

Tree Widgets

Tree widgets display hierarchically-organized data.



Click + to expand folders, click − to collapse them.

Examples:

Expand the winddemo tree to see the files in the project.

Windows

Open and *close* windows. Use *active* or *open*, not *current*, when referring to windows.

Examples:

Open the window by selecting **File** > **Open**.

Close the window when you are finished.

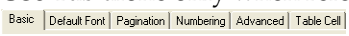
You can view your document in the *open* window.

Use *lower left/right*, not *bottom left/right* (hyphenate as adjectives; i.e., *lower-left corner*.)

NOTE: If you are writing about Workbench, be careful not to use the word *window* for *view*. There are no windows in Workbench, only views.

Tabs

Use Tab alone only when referring to a tab in a dialog box.



Example:

Select the Pagination tab.

Otherwise, clarify the meaning with a descriptor (such as TAB key, tab stop.) Do not use tab as a verb. Examples:

CORRECT:

Use the **TAB** key to move through a dialog box.

Set a *tab stop* on the ruler.

INCORRECT:

You can *tab* through a dialog box.

Set a *tab* on the ruler.

Keyboard Keys

Spell the names of keyboard keys in all caps, regardless whether your keyboard uses all caps or not, as in **ESC**, **ENTER**, **SHIFT**, **TAB**, **DELETE**, **BACKSPACE**, **ALT**, and **CTRL**.

In FrameMaker, apply the character tag **nameLiteral-uc**.

In DITA, apply the **<uicontrol>** element.

Do not use **RETURN** for **ENTER**.

Refer to the arrow keys and space bar as **LEFTARROW**, **UPARROW**, **DOWN ARROW**, **RIGHTARROW**, and **SPACEBAR**.

The syntax for showing key combinations (keys that must be pressed at the same time) is to join them with the plus sign with no space separation, key+key—for example, **CTRL+D** or **SHIFT+F**. Do not use the UNIX convention of using a circumflex for control keys.

The syntax for showing key sequences (keys that must be pressed in succession) is to separate them with commas and hard spaces—for example, **ALT, F, P**. *Press* keyboard keys. Do not use *strike*, *push*, or *hit* when referring to keys.

Examples:

Press **ENTER**.

Press **CTRL+F**, then type the text.

Command-Line Prompts

For command-line example displays, use the standard system prompts.

The standard system prompts are as follows:

%

For UNIX shell commands.

$

For Linux shell commands.

#

For UNIX and Linux shell commands that must be executed as root/superuser.

C:\>

For Windows commands.

->

For standard VxWorks and host shell prompts.

In rare cases, a command-line example may be exactly the same in both UNIX and Windows. Rather than showing the example twice with each prompt, show just the UNIX version.

In the uncommon case where all or nearly all command examples in a manual use % to mean both UNIX and Windows, explain the dual use of % in a section explaining documentation conventions specific to your book. However, in general, you will need to say so explicitly when a % example applies to both UNIX and Windows.

7 Spelling

About Spelling in Technical Documentation

There are often many ways to spell technical terms; however, Wind River documentation attempts to maintain consistency across all products.

Spelling variations occur either because the terminology has not been fully standardized in the industry or because there are multiple possibilities depending on dictionary or context.

The standards in this document are the single, authoritative source of the current standard.

Spelling List

The spelling list defines the spelling standard for terms that are used frequently in Wind River documentation and which have a tendency to be spelled inconsistently

Table 5 **Spelling List**

and so forth, among others	etc.
autoscaling	auto scaling, auto-scaling
back end	backend

back up (v.)	backup, back-up
backup (n., adj.)	back up, back-up
backward	backwards
baseline	base line
basename (of filename)	base name
bit-field	bit field
boot line	bootline, boot-line
boot loader	bootloader
boot ROM	bootrom, boot rom, bootROM
bring up (v.)	bringup, bring-up
bring-up (n., adj.)	bringup, bring up
bps	BPS, baud
caching	cacheing
cacheable	cachable
callback	call-back
cannot	can not
CD-ROM	CDROM, cdrom
check in (v.)	checkin, check-in
checkin (n., adj.)	check in, check-in
check out (v.)	checkout, check-out
checkout (n., adj.)	check out, check-out
clean up (v.)	cleanup, clean-up
cleanup (n., adj.)	clean up, clean-up
coprocessor	co-processor
countdown	count-down
cross-compiler	cross compiler
cross-development	cross development
cross-reference	cross reference
data type	datatype
decrypt	unencrypt, de-encrypt
dialog	dialogue

email	e-mail, E-mail, Email
endianness	endianess
Ethernet	ethernet
Excelan	Excellan
extensible	extendable, extendible
fax	FAX
fd (See Acronym Special Cases on page 39.	FD
file name	filename
file system	filesystem
flash (n.)	Flash, flash (v.) or flashing (gerund)
floating-point	floating point
follow up (v.)	followup, follow-up
followup (n., adj.)	follow up, follow-up
for example	e.g.
FTP	ftp
HP-UX	HP/UX, HPUX
hard copy	hardcopy
home page	homepage
hot swap	hotswap
I/O	i/o, IO, io
ID	id
Internet	internet
inter-process	interprocess
intertask	inter-task
inline	in-line
ioctl( )	IOCTL, IOctl, IOCtl, ioctl
Iostreams (no bold)	IOStreams, iostreams, IoStreams
KB	Kb, Kbyte
log in (v.)	login, log-in
login (n., adj.)	log in, log-in
log out (v.)	logout, log-out

logout (n., adj.)	log out, log-out
lowercase	lower-case
MB	Mb, Mbyte
MC680x0	M68000, M68k
MC68020...	M68020, 68020...
MS-DOS	MSDOS, MS DOS
make up (v.)	makeup, make-up
makeup (n., adj.)	make up, make-up
motherboard	mother-board, mother board
multi-core	multicore, Multicore
multiprocessor	multi-processor
multitasking	multi-tasking
multi-user	multiuser
multi-WAN, Multi-WAN	multiwan, Multiwan, multiWAN
mux	MUX
nonvolatile	non-volatile
nonzero	non-zero
offload	off-load
on-board	on board, onboard
online	on-line, on line
overrun	over-run, over run
overwrite	over-write, over write
PAL	pal
pathname	path name
Platform product	platform product
plug-in	plugin
pop-up	popup
POSIX	Posix
power up (v.)	powerup, power-up
powerup (n., adj.)	power up, power-up
preemptive	pre-emptive

print out (v.)	prinout, print-out
printout (n., adj.)	print-out, print-out
real-time, Real-Time	realtime, Real-time
reentrant	re-entrant
ROMFS	romfs (unless a literal)
RSH	rsh
run-time (adj.)	runtime, run time
run time (n.)	runtime, run-time
SBus	S-Bus, Sbus
scalable	scaleable
SCSI	Scsi, scsi
set up (v.)	set-up
setup (n., adj.)	set-up
shell script	shellscript
shut down (v.)	shutdown, shut-down
shutdown (n., adj.)	shut down, shut-down
single-stepping	single stepping
spinlock	spin-lock
standalone	stand-alone
start up (v.)	startup, start-up
startup (n., adj.)	start up, start-up
stdio.h	stdio
subclass	sub-class
subdirectory	sub-directory
SunOS	SUN OS
superclass	super-class
task ID	task id
Tcl	TCL, tcl
TFTP	tftp
that is	i.e.
timeout	time-out

timestamp	time stamp, time-stamp
title bar	titlebar
TTY	tty
turn off (v.)	turnoff, turn-off
turnoff (n., adj.)	turn off, turn-off
underrun	under-run, under run
UNIX	Unix
uppercase	upper-case, upper case
Usersí Group	Userís Group, Users Group
user name	username
user space	userspace
VxWorks	VxWORKS, VXWORKS, vxWorks
Web	web
website	Web site, web site
Wind River	WindRiver
work around (v.)	workaround, work-around
workaround (n., adj.)	work-around
write-through, write-back	writethrough, writeback

Special Spelling Issues

Setup Program

In general, refer to *Setup* as a proper noun (capitalized) and without bold or italics.

When referring to the executable, spell the word lowercase and apply **<filepath>** character tag as with any other executable name: **setup** or **setup.exe**.

Hyphenated and Compound Words

Watch out for verbs that take a prepositional adverb; for example, start plus up.

These combinations are often referred to as phrasal verbs.

A common error, even among technical writers, is to treat such verbs as closed compounds. However, the verb and the prepositional adverb must never be combined when these words are used as verbs.

They can only be combined, and must be combined, when used as nouns or adjectives. Although practice varies with many of these words as to whether the noun and adjective compounds are formed with or without a hyphen, the house style is never to use a hyphen.

The following usage examples illustrate this distinction:

- Verb, always two words:

  INCORRECT:

  - Startup the system by turning on the power.
  - Please checkin my files.

  CORRECT:

  - Start up the system by turning on the power.
  - Please check in my files.

- Adjective, always one word:

  INCORRECT:

  - Be sure to supply the correct start up parameters.
  - Please review the check-in schedule.

  CORRECT:

  - Be sure to supply the correct startup parameters.
  - Please review the checkin schedule.

- Noun, always one word:

  INCORRECT:

  - There is a brief delay before system start-up.
  - What time is tomorrow's check in?

  CORRECT:

  - There is a brief delay before system startup.
  - What time is tomorrow's checkin?

One way to remember the one-word/two-word distinction is the subtle difference in the way the verb forms and the noun/adjective forms are pronounced. When using these words as verbs, speakers tend to stress the verb and the prepositional adverb equally. When using them as nouns or adjectives, speakers tend to stress the first portion of the word.

Another way to remember is to keep in mind how you would form the past tense of these verbs. Past tense usage as verbs enforces the rule that the verb and the adverb remain separate. You would never say "checkined" -- you would only say "checked in".

Makefile

In most cases, the term makefile should be used as a generic word, appearing with an article (a or the), containing no special capitalization, and appearing in the default font.

If you are referring to it as a specific filename, it must not appear with an article; it must be capitalized exactly like the filename; and it must appear in bold (with the **nameLiteral** character tag in FrameMaker or the **<filepath>** element in DITA).

INCORRECT:

Edit the **Makefile** to redefine the build options.

CORRECT:

Edit the makefile to redefine the build options.

Edit **Makefile** to redefine the build options.

Run-time and Run Time

The words run-time and run time are commonly used and misused in Wind River documentation.

- The word run-time is hyphenated because it is a compound modifier, as in run-time environment, run-time version, or run-time activity.
- The term run time (with a space) is a noun meaning the time period in which a program runs or the amount of time needed to run a program. (Compare with compile time.)
- Do not use the word run-time as a noun meaning run-time component.

Platform Products

Always capitalize the word Platform when it is in reference to Platform products.

Platform products is a sanctioned general term; you can also write Wind River Platforms.

Do not capitalize the word *platform* when it is used generically for any of its other meanings that are not with reference to Platform products.

Prefix Hyphenation

Follow standard practice and do not hyphenate between prefixes and the root word, even if it means leaving vowels adjacent.

For example:

- *reentrant*
- *reinitialize*
- *reuse*
- *preallocate*
- *decouple*

If the addition of a prefix alters the meaning of the word in unintended ways, do use the hyphen, as for example, to distinguish *re-creation* from *recreation*, *re-collect* from *recollect*.

Company Name

For the company name: use “Wind River” only, except for legal references, in which case, use “Wind River Systems, Inc.”

In FrameMaker, except for legal references, always insert the *CompanyName* variable to write the company name.

“WindRiver” with unspaced “camel-caps” should never appear in text. It was once displayed as part of the company logo, but never permitted in text. Currently, it is no longer valid even as a logo.

8

Referring to Other Content

General Style

Names of documents or manuals, including Wind River manuals, should appear in italics, using the **title** tag in FrameMaker or the **<cite>** element in DITA.

References can appear inline or in a separate section at the end of a topic. The preferred method is to list references at the end. Place them in a non-bulleted list in either a Related Links section (live links) or an Additional Information section (non-linked references).

All inline cross-references should be phrased in the following general style:

For information on topic, see *crossRef*.

For details on topic, see *crossRef*.

For example:

For information on installation, see the *Developer Install Guide*.

If the topic is clear, it is also acceptable to say the following:

For further information, see...

For details, see...

Do not use the following form:

See the *Developer Install Guide* for information on installation.

Do not add the word chapter or section to cross-references.

If you need to provide two or more inline cross-references, present them in an unordered list. Do not use an inline serial list.

INCORRECT:

For more information, see *2.2 Titles and Headings, p. 6*, and *4.7 Cross-References, p. 127*.

CORRECT:

For more information, see:

- *2.2 Titles and Headings, p. 6*
- *4.7 Cross-References, p. 127.*

References to Other Books

References to documents or topic sets most often follow the formal style.

However, for certain types of books a less formal, generic style is more efficient and appropriate.

Formal Style

Cross-references that refer to chapters in a separate book are treated differently, as are cross-references in reference (man page) documentation.

Such cross-references should follow the format:

bookTitle: *chapterName*

They should not specify a chapter number, nor should they point to a section below the chapter level. This practice avoids the maintenance issues associated with trying to keep such numbering synchronized across documents that most likely will not be released at the same time or have an identical life span. For example:

For more information, see the *VxWorks Programmer's Guide: I/O System*.

As this example illustrates, include the article *the* before the volume title when appropriate.

Do not include the book's release number except in cases where the user could be confused without it. If a release number proves to be necessary, separate it from the title with a comma and set it in italics, also; for example:

Wind River OSPFv2 Programmer's Guide, 1.2

If the book is host or OS specific indicate this only if the distinction is necessary in order to avoid confusion. Show the OS or host version in parentheses and in italics, also; for example:

Wind River Data Monitor User's Guide (Windows Version)

Wind River Workbench User's Guide (VxWorks Version), 3.0

A common mistake is failing to include the entire title, particularly for release notes and getting started books. For example:

INCORRECT:

See the *Release Notes* for Wind River Platforms.

See the *Getting Started* for the General Purpose Platform

CORRECT:

See the *Wind River Platforms Release Notes*.

See *Wind River General Purpose Platform Getting Started*.

Generic Style

A generic cross-reference is one that does not mention a document by its formal title. The generic style is acceptable when you make reference to the following types of documentation:

- release notes
- getting started guides
- installation guides

The following are examples of generic style cross-references:

For information on changes introduced with this release, see the release notes for the General Purpose Platform.

For information on migrating your application to this release, see the getting started guide.

Formal and Generic Styles Compared

FORMAL:

See the *Wind River Platforms Release Notes*.

See *Wind River General Purpose Platform Getting Started*.

GENERIC:

See the release notes for Wind River Platforms.

See the getting started guide for the General Purpose Platform.

References to External Sources

As with Wind River manuals, names of external documents or manuals should appear in italics, using the title tag in FrameMaker or the **<cite>** element in DITA.

RFCs

Treat Requests for Comments (RFCs) as follows:

RFC 1825: *Security Architecture for the Internet Protocol*

Standards or Specifications

When providing the name of a specification or standard, present it in its full form. For example:

American National Standard for Information Systems – Programming Language – C, ANSI X3.159-1989

When citing a particular chapter or section of a standard, separate it from the title with a colon as in the following:

American National Standard for Information Systems – Programming Language – C, ANSI X3.159-1989: Input/Output (stdio.h)

Shortened Forms

Do not use shortened forms, unless you initially cite the full name and indicate that you are also going to refer to the item by a short form. Indicate this as in the following example:

Intelligent I/O (I2O) Architecture Specification, version 1.5 (also referred to as I2O Specification, v. 1.5)

Copyrighted Materials

Copyrighted materials should be cited in bibliographical sections or appendices. Follow the conventions described in the *Microsoft Manual of Style for Technical Publications*, which is based on the *The Chicago Manual of Style*. The standard is the same in each, but the Microsoft description is more accessible and less detailed.

References to API Reference Entries

Cross-references to API reference entries should merely give the name of the entry and not specify whether it is online or in a book such as the VxWorks API Reference.

For example:

For more information, see the reference entry for **rpcLib**.

The above example also demonstrates the preferred sentence structure of cross-references. Start the sentence by specifying the purpose of the reference, and end it with the word see followed by the reference. Do not use consult or refer to.

Do not make reference-entry cross-references live, as in a link to an entry in an API appendix.

In rare cases where an API reference entry appears in only a single book, do include the title of the book. For example:

For more information on **envoy_ax_ma_handler( )**, see the *Wind River SNMP API Reference*.

Web Links

When including a Web link, ensure the address is visible to the reader—that is, not hidden behind a title or other text.

Do not include the “http://” prefix, unless it is essential to indicate that the link is a Web address.

If, based on your information or experience with a particular Web site, you have reason to believe the full link will be reasonably stable for the anticipated life-span of this release of the document, use the full path to the link. However, if you have no information or experience that suggests a particular full link will be stable within the document lifespan, take the conservative approach and provide only the top-level address.

Whenever you update a document, you should reverify all mentioned links, both deep links and top-level links.

Third-Party Vendors

When mentioning third-party vendors in documentation, do not supply addresses, phone numbers, or URLs (which are time sensitive).

Always check with Marketing to verify the validity of third-party vendors.

“Borrowing” Material from Outside Sources

You cannot “swipe” material, either graphics or text, from another company's documentation or Web site.

How or whether such material can be republished depends on the copyright notices that accompany it. If the copyright notice allows republishing at all, it will probably require the owner's written permission to do so and an attribution of the source.

If you have outside material you believe is necessary to republish (and simply citing the reference is not sufficient), check the copyright notices accompanying such material and send e-mail to contracts@windriver.com to get help interpreting the legalese. Summarize the issue and a Wind River attorney will be assigned to help you.

9

Indexing

About Indexing

Good indexes are key to the usefulness of technical manuals. Because Wind River customers use manuals so heavily as reference books, indexes are essential.

All programmer's guides and user's guides that are large enough to qualify as chapter books should have an index. However, getting started guides and release notes should not have indexes.

All standalone API reference books should have a keyword index. Standards and tools for generating keyword indexes are described elsewhere; for further information, see *Reference-Entry Format*.

Indexing Guidelines

Hierarchy

The hierarchy of your index makes the index more usable than an unstructured index.

- Give consideration to the organization of the table of contents when creating an indexing hierarchy, as top-level headings can become main index entries; the next level of headings become first-level subentries (if not their own main index entries), and so on. However, this is not a hard and fast rule.

- *see also* entries follow first-level entries only.
- Reduce index congestion caused by cross-indexing (having multiple index entries for a single subject). While one subject, say BOOTP, appears as its own main entry, it also appears as a subentry under a different main index entry, booting. As shown in the following example, the level of indexing detail is greater under the main entry for BOOTP than under the booting subentry. Rather than repeat under booting:BOOTP all the second- and third-level entries that appear under the main BOOTP main, a see also refers users from booting to BOOTP.

  booting
  see also BOOTP
  ...
  BOOTP 325-331
  boot parameters 328-329
  database 326-328
  instructions, step-by-step booting 329-330
  protocols, using with other 349-350
  code example 350
  registering targets 326-328
  server 326
  multiple servers 331
  troubleshooting 330
  bootrom.hex 46,237
  bootrom_uncmp 458
  i386/i486 621
- Avoid fourth-level entries.

Too-General Main Index Entries

Avoid creating main index entries that are too broad, would encompass too many references, and are not likely to be considered as an index entry by a user; for example, *routines*, *libraries*, and *system*.

General mentions of subjects, particularly in overviews or introductions, may not warrant index markers. If a term is introduced and defined in one location but discussed in depth in another, it might be helpful to create markers as follows:

datagram sockets 141-153
defined 132

See and See Also Entries

See and *see also* entries direct readers to other entries in the index.

Note that they are not meant for cross-references to chapters, sections, or other books; however, they may point to specific reference entries.

The basic difference between see and see also is as follows:

- *See* is non-optional. It means that no information is provided here, you must look where directed.
- *See also* is optional. It is a suggestion that what you are looking for may also be found at a related topic.

General Guidelines for "see" and "see also" Entries

- If possible, avoid using see or see also to refer to a subentry rather than a main entry. However, if reference to a subentry is necessary, then separate the main entry from the subentry with an en-dash surrounded by spaces, as in the following:

  MMU:
  see also virtual memory – VxVMI option

- If the see or see also entry has multiple references, list them alphabetically, for example:

  remote file access
  see also FTP; NFS; RSH; TFTP

- When see is a single-line entry, precede see with a comma, for example:

  Internet Protocol, *see* IP

- See also entries must be the first subentry of a main item. Avoid using see also in a second-level or lower entry.

See Entries

Use *see* in the following circumstances:

- When the item indexed is not our preferred term and should instead direct the reader to the index entry for the preferred term; for example:

  interrupt handler, see interrupt service routine

- For terms that refer to an acronym (or, in unusual cases, acronyms that refer to the term). For rules about indexing acronyms and their definitions, seeAcronyms on page 36.

  board support package, *see* BSP

- For technology that is known by different names, and the index uses only one as the main reference, which corresponds to the preferred reference in text.

  80836, *see* i386/i486
  x86, *see* i386/i486

- For multiple entries for a single subject where there is no clear preference but subentries make it necessary to favor a single entry (to avoid redundancy and maintenance headaches). Conversely, do not use see in such cases if no subentries are involved; for the convenience of the reader, provide the page number with each entry.

See Also Entries

Use see also in the following circumstances:

- To refer the reader from one page-numbered entry to other related entries; for example:

  data cache 123
  see also cache

- To refer the reader from a general entry to a main entry of the same or related subject:

  tasks
  see also shared code
  shared code 327

- To refer the reader from a general main entry to an unspecified set of other, more specific main entries.

devices, *see also* specific device types

NOTE: In the above example, because "specific device types" is not an index entry but part of the see instruction, it must also appear in italics.

See Also Reference Entry

See also reference entry is used as a no-page pointer to reference entries (libraries, routines, and so on) appearing either online or in print. If the pointer is to more than one entry, use entries. Example:

remote file access
see also reference entries ftpdLib; ftpLib; nfsDrv; remLib; tftpdLib; tftpLib

See also the discussion under Pointers to API Reference Entries on page 96.

Alphabetization and Order

Both FrameMaker and DITA provide the ability to specify the order of index entries.

DITA

Use **<index-sort-as>** to indicate the virtual spelling that should be used in creating the index.

FrameMaker

- The hyphen, commonly used as a prefix for command-line options, is set to be ignored by our IX template.
- As mentioned previously, use \i at the beginning of a see or see also subentry to force the entry to the top of a subentry list; for example:

```
tasks:hooks:\i<xi>see also</> taskHookLib<$nopage>
```

- In some cases, you will need to use bracketed index commands to achieve proper alphabetization of problem entries. For example, "I/O" should be force-sorted as "IO" as in the following:

```
I/O system:implementing[IO system:implementing]
```

- Try to avoid entries like "kernel:and multitasking" (which will sort on "and"). However, if an entry exists with a second level that does not start with the keyword, force the sort order as follows:

```
kernel:and multitasking[kernel:multitasking]
```

Capitalization

In general, do not capitalize index entries.

Capitalize index entries only if they are proper nouns, program elements that are normally capitalized, or words that are otherwise normally capitalized. Otherwise, do not capitalize them.

Acronyms

Both an acronym and its spelled-out version should appear in the index.

However, designated common acronyms should not include an entry for the spelled-out version.

For the list of common acronyms, see the table under Acronyms on page 36.

Treat non-common acronyms in either of two ways, depending on whether subentries are needed.

If the entries do not have subentries, follow the acronym or spelled-out version by its counterpart in parentheses; for example:

board support package (BSP) *n*
...
BSP (board support package) *n*

If the entries do have subentries, generally the spelled-out form should use see to reference the acronym. The subentries should appear only in the main entry, not in the see entry. In unusual cases where the spelled-out form is the preferred form used in the text, the acronym should be the see entry. In either case, do not use parentheses with the see entry. See also See and See Also Entries on page 89.

board support package, see BSP
...
BSP (board support package)
documentation 429
initialization modules 430
...

Qualifying Multiple Entries

When you have identical index entries that apply to different features, facilities, architectures, and so forth, you must add a qualifier to distinguish the two entries.

You can qualify the entries with parentheses or with a comma.

Qualifying Entries with Parentheses

Index entries of the following types should be qualified with parenthetical inclusions:

- entries that are specific to particular architectures, either because the feature or facility is available only for specific architectures or because the text identified by this index entry is specific to specific architectures; for example:

  SDE-MIPS GNU ToolKit (MIPS)
  cacheClear( ) (MC68040)

  NOTE: For example, indexing a routine in this way does not imply that the routine is for use only with the named architecture. Rather, it means that the discussion to which the index entry points is specific to that architecture.

- entries that are specific to an optional component; for example:

  page states (VxVMI option)

- entries that are specific to a particular facility and would be enhanced by a parenthetical qualifier; for example:

  clusters (dosFs)

- entries that may need to be qualified in order to clarify an ambiguous or vague reference; for example:

  ld linker (UNIX)
  ld( ) (VxWorks)
  trailing slash (GCC_EXEC_PREFIX)
  customer support (Wind River)

However, if an entry has multiple qualifiers that point to text on separate pages (multiple architectures, for instance), create subentries instead of using parentheses, as in the following example:

dbgLib
AM29K 589
MIPS 575
Coldfire 546

Qualifying Index Entries with a Comma

Entries may also be qualified by using a comma. This is actually preferred to the use of parenthetical inclusions and is typically done when the qualifier is normally prefixed as an adjective; for example:

function calls, shell

Handling Function and Array Names

As in regular text, function names in index entries must be followed by a pair of parentheses and array names must be followed by square brackets.

In FrameMaker, use a hard space to separate the parens or brackets.

Arrays, such as *sysPhysMemDesc[ ]*, sometimes appear in text without brackets. For the sake of uniformity, always include the brackets with array names in index entries.

Conjunctions and Prepositions in Subentries

Make every effort to have the subentry lead word be the most important word of the entry, that is, the word that falls flush left.

This is extremely important for ensuring that concepts are sorted where users expect to find them.

INCORRECT:

BOOTP
using with other protocols

CORRECT:

BOOTP
protocols, using with other

Exceptions are *and*, *of*, and *in*; however, these constructions can generally be avoided and therefore should be avoided if possible.

errno
and task contexts

Subentries beginning with *and*, *of*, or *in* require special treatment in order to be sorted on the keyword and not the conjunction or preposition. In the preceding example, the subentry should be sorted on *task*, not *and*.

Gerunds

Use gerunds for index entries describing product functions—for example, *initializing*, *configuring*, and *exiting*.

As in the standard for headings, gerunds are preferred over nouns such as initialization and configuration.

For subentries, using a gerund as the keyword of a phrase should occur only when it is indeed the most important aspect of the entry; for example:

include files
hiding internal details

However, when the gerund is not the concept of interest, use the comma:

interrupt handling
disks, changing

Singular or Plural

Make noun entries plural, unless plural makes no sense, as when there is only one of something.

For example:

RAM disks
POSIX:message queues

However:

POSIX: memory-locking interface

Numeric Entries

Entries that begin with numbers should be indexed under the number (which means the entry will appear in the first grouping of the generated index), but may also appear alphabetized elsewhere in the index, if that seems like somewhere readers might look for the entry.

Proprietary Entities and Other Company Products

Typically proprietary products, when indexed, are indexed in the manner in which they appear in their manufacturer's literature.

Of course, this leaves much room for ambiguity and confusion because many companies refer to products in more than one way. The following have been agreed to by Wind River Tech Pubs:

- Am29K
- MC680x0
- MIPS
- SPARC or SPARClite

Inline Markup

Do not use character tags or inline elements in index entries.

What to Index

Topics from the Table of Contents

The table of contents is a good starting place for developing index topics.

Generally, try to follow the table of contents when developing your index, creating entries that reflect the chapter, section, and subsection organization of the document. If a subject is important enough for a section heading, then it is most likely important enough to index. Of course, index entries cover more than just headings; they cover the text throughout.

Program Elements

Index meaningful occurrences of program elements such as macros and routines.

For example:

- macros
- routines
- tools
- libraries
- directories
- constants
- compiler
- options

- variables
- control
- characters
- commands

... and so on

Some program elements are further identified by words telling the reader what type of element. This is used to differentiate among program elements that are not easily recognizable typographically. Use these identifiers in index entries. For example:

-ACA compiler option
chkdsk utility

Special Terms

Consider indexing words assigned the word character tag or element, because these tend to be special terms.

For example:

Context is indexed as:

contexts:task
tasks:contexts

Event is indexed as:

events, real-time

Local objects is indexed as:

local objects

Glossary Terms

When creating index entries for terms where they occur in a glossary, indicate that the entry is for the "glossary definition" to distinguish it from other index entries for the term.

For example:

overrides
acceptable 77
glossary definition 194
indication 77

Pointers to API Reference Entries

When text refers the reader to a library in an API reference, such as the VxWorks OS Libraries API Reference with a "See the..." or a "See...", then any main index entry for the subject being discussed should have a see also reference entry referring the reader to that library.

Consider the following excerpt:

"Originally designed by Sun Microsystems and available in the public domain, Remote Procedure Call (RPC) is a facility that allows a process on one machine to call a procedure that is executed by another process on another machine. Thus with RPC, a VxWorks task or UNIX process can invoke routines that are executed on other VxWorks or UNIX machines, in any combination. See the RPC documentation (available both publicly and commercially) and the reference entry for rpcLib."

The above should prompt the following index entry:

RPC (Remote Procedure Calls):*see also reference entry* rpcLib

Use such pointers for libraries only. Do not create index pointers to routine reference entries.

A

Bibliography

External documents were consulted in the creation of Wind River style standards and are useful resources for additional information.

1. *Chicago Manual of Style. 15th ed.* Chicago: University of Chicago Press, 2003.
2. Johnson, Edward D. *The Handbook of Good English*. New York: Facts on File Publications, 1982.
3. *Microsoft Manual of Style for Technical Publications*, 3rd ed. Redmond, WA: Microsoft Press, 2004.
4. *Read Me First! A Style Guide for the Computer Industry*. Mountain View, CA: Sun Technical Publications, 1996.
5. Shertzer, Margaret. *The Elements of Grammar*. New York: MacMillan Publishing Co., 1986.
6. Siegal, Allan M., and Silliam G. Connolly. *The New York Times Manual of Style and Usage*. New York: Times Books, 1999.
7. Skillin, Marjorie E. *Words Into Type, 3rd ed.* Englewood Cliffs, NJ: Prentice-Hall, 1974.
8. Strunk, William, and E. B. White. *The Elements of Style. 3rd ed.* New York: MacMillan Publishing Co., 1979.
9. Truss, Lynne. Eats, *Shoots & Leaves, The Zero Tolerance Approach to Punctuation.* New York: Gotham Books, 2003.
10. Watkins, F. C., W. B. Dillingham, and E. T. Martin. *Practical English Handbook.* 3rd ed. New York: Houghton Mifflin, 1971.

B

Glossary

Terms in this glossary pertain to the writing and production documentation itself. For a glossary of standard terms used in Wind River technical publications, see the *Wind River Technical Publications Technical Glossary*.

API reference

A book, book part, or appendix that provides a set of reference entries describing an API. It covers the same subject as a programmer's guide but provides more detailed information. The notion of reference stems from the fact that the entries are not organized topically or hierarchically, but are ordered alphabetically for easy look-up. Compare the glossary entry for *reference entry*.

argument list

The set of arguments that is given to a command or routine when it is executed. Compare the glossary entry for *parameter list*.

callout

In a figure, text coupled with a leader that points to some part of a diagram or screenshot. Compare the glossary entry for *label*.

content template

A template that provides a prescribed organization and boilerplate text for a particular type of document. A content template also provides examples and guidelines about what to include.

context-sensitive help

Online documentation that is context sensitive, meaning that when the user presses a help button or particular keys the help system retrieves information about the currently active facility in a GUI. Compare the glossary entry for *online help*.

deprecated

A term used to indicate that a feature will no longer be supported in a future release. Note that it is not correct to say that a feature will be deprecated in the future. Noting that a feature has been deprecated means that support will be discontinued in the future and serves as a warning to customers that they should seek an alternative solution.

defining clause

A relative clause that limits (or defines) what it modifies. It is also known as a restrictive clause. A defining clause begins with the relative pronoun which or that, but is not separated from the main clause with a comma. Compare the glossary entry for *parenthetical clause*.

definition list

A list style used for system elements such as parameters, commands, command options, files, routines, and macros. Definition lists follow a specific format, where each term appears on its own line and is followed by an indented paragraph that defines the term or explains its usage. For details, see About Introducing Lists on page 11.

display

In technical documentation, a standalone text item set off on a separate line (or lines) from the normal text of a paragraph. Ideally, but not always, a display is indented one level from the paragraph that introduces it. Displays are used to show URLs, pathnames, command syntax, execution examples, terminal sessions, and code examples. They are particularly useful for drawing attention to these elements, but also provide a means of showing long pathnames or URLs without the awkward line breaks that are likely to occur in normal, wrapped paragraphs.

dpi

Dots per inch, a measure of printer resolution. Higher dpi ratings or printer settings indicate a higher or finer resolution. A dpi setting is also a measure used by FrameMaker to indicate how to size imported graphics; the higher the dpi value, the smaller the image will appear, because the ppi (pixels per inch) of the source graphic remains constant. Compare the glossary entry for *ppi*.

format override

The difference between the format of text or tables in a document and the corresponding format in the Paragraph, Character, or Table Catalog (quoting from Adobe Frame manual).

format template

A template that provides a document's layout and formatting styles. Compare the glossary entry for *content template*.

gerund

\The *-ing* form of a verb functioning as a noun; for example, *writing* in *Writing is easy*.

imperative mood

The style of a sentence or clause when it is delivered as a command. The one-line summary description in a reference entry for a routine or command is written in imperative mood.

label

In a diagram, text that is applied directly on the element that it identifies. Compare the glossary entry for *callout*.

leader

The straight line leading from a *callout* to the graphic element it identifies.

literal

A name of a software element that is spelled and capitalized exactly as it appears in a software system. Examples are names of routines, commands, command flags, configuration parameters, files, pathnames, macros, constants, language keywords, environment variables, program variables, operators, and register names. Compare the glossary entry for *placeholder*.

non-restrictive clause

See the glossary entry for *parenthetical clause*.

online help

Documentation provided in electronic format and displayed on-screen, a synonym for online manuals and help system. This definition is broader than *context-sensitive help*.

parameter list

The set of possible parameters or argument types that may be given to a command or routine when it is executed. Compare the glossary entry for *argument list*.

parenthetical clause

A relative clause that adds new information to what it modifies, so-called because a natural test is whether it can be set off in parentheses. It is also known as a non-restrictive clause. A parenthetical clause begins with the relative pronoun which, but never that, and is separated from the main clause with a comma. Compare the glossary entry for *defining clause*.

phrasal verb

A verb that is combined with an prepositional adverb that changes its meaning. Examples are *set up* and *check in*. When used as a verb, the verb and the prepositional adverb must be separate words. The corresponding noun and adjective forms are always single words and do not include a hyphen; for example, *setup* and *checkin*.

placeholder

A text string that is not to be interpreted literally, and represents some value that the user supplies or an element that will vary depending on context. The use of placeholders is confined mostly to command arguments, variable portions of pathnames, and function parameters. Compare the glossary entry for *literal*.

ppi

Pixels per inch, a measure of resolution in a graphical image. Compare the glossary entry for *dpi*.

Platform

An OS-specific collection of Wind River technologies sold as a product and usually, but not always, targeted towards a specific vertical market. Examples include:

Wind River Platform for Network Equipment, Linux Edition
Wind River General Purpose Platform, VxWorks Edition (no vertical market)
Wind River Platform for Safety Critical ARINC 653

prepositional adverb

A word that is usually thought of as a preposition but functions as an adverb, such as the word of in this sentence. When such words modify a verb, they are adverbs and not prepositions. See also the glossary entry for *phrasal verb*.

product-names file

The template file **var_product_names.fm**, which lists all product names and defines correct usage.

restrictive clause

See the glossary entry for *defining clause*.

reference entry

A Wind River term for a unit of documentation that is approximately equivalent to the classical UNIX-style man page. Although the underlying file format is not UNIX troff, the general style and layout is inherited directly from man pages. For further information, see the glossary entry for *Reference-Entry Format*.

split infinitive

An infinitive verb form with some element, usually an adverb, interposed between to and the verb form; for example, to aggressively deny. Avoid split infinitives.

syntax display

A display that shows the execution syntax of a routine, command, or other function. A syntax display is not an execution example, although the two are sometimes confused. See also the glossary entry for *display*.

terminal session

A display showing the execution of a command line and the computer's response.

terms list

See the glossary entry for *definition list*.