

=====
PGI Document Notice and License

© Copyright 2008 The Portland Group, Inc. (PGI) All right reserved.

Permission to use, copy, and distribute the contents of this document, in any medium for any purpose and without fee or royalty is hereby granted, provided that you include the following on ALL copies of the document, or portions thereof:

1. A link or URL to the original document located at:

<http://www.pgroup.com/doc/pgexplain.xsd>

2. The copyright notice listed above.

When space permits, inclusion of the full text of this NOTICE should be provided. We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

The right to create modifications or derivatives is granted provided that that you indicate any modification that you make by changing the namespace so as to distinguish your altered version from the PGI version. We further request you include a statement regarding who made the modifications, when, and what changes were made.

THIS DOCUMENT IS PROVIDED "AS IS," AND THE COPYRIGHT HOLDER MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

THE COPYRIGHT HOLDER WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name and trademarks of the copyright holder may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with copyright holders.

CCFF is the Common Compiler Feedback Format, initially defined and implemented by PGI. PGI compilers add a section to object and executable files. The CCFF information can be extracted into a file or read directly from the section of the object or executable file. The CCFF information is stored as an XML file, whose structure we describe here. In the BNF below, we show the XML element names using the standard angle brackets (<ccff> or </ccff>), and use ALL CAPS for BNF nonterminals. Ellipses ("...") are used where literal information is to be placed.

A <ccffbin> element can contain many <ccff> elements. A well formed XML file has a single outermost element, so <ccffbin> serves that purpose. The intent is that a single <ccffbin> will appear in an executable file; it will contain all the <ccff> elements collected from the object files used in the link.

```
CCFFBIN ::= <ccffbin> CCFF* </ccffbin>
```

A <ccff> element contains information for a single compilation unit or file. The compiler will append a section with a <ccff> element to the object file. The <ccff> element may have a version="..." attribute, where ... is the literal version number.

```
CCFF ::= <ccff version="..."> SOURCE? BUILD? UNITS? </ccff>
```

The <source> element contains information about the source file. It contains the elements:

- <sourcename> - source file name
- <sourcepath> - directory path to the source file
- <sourcedir> the current directory, to which the <sourcepath> may be relative.

```
SOURCE ::= <source> SOURCENAME? SOURCEPATH? SOURCEDIR? </source>
```

example:

```
<source>
  <sourcename>a.c</sourcename>
  <sourcepath>../src</sourcepath>
  <sourcedir>/home/user/application/obj</sourcedir>
</source>
```

The <build> element contains information about the compiler that built the object. It contains the elements:

- <buildcompiler> - name of the compiler, perhaps the command line name
- <buildvendor> - name of the vendor
- <buildoptions> - command line options used for this object

<buildversion> - compiler version
<buildhost> - information about the host used to compile the file
<buildtarget> - information about the compiler target
<buildlanguage> - the source language
<builddate> - date of the compile

```
BUILD ::= <build> BUILDCOMPILER? BUILDVENDOR? BUILDOPTIONS?  
        BUILDVERSION? BUILDHOST? BUILDTARGET? BUILDLANGUAGE?  
        BUILDDATE? </build>
```

example:

```
<build>  
  <buildcompiler>PGC</buildcompiler>  
  <buildvendor>PGI</buildvendor>  
  <buildoptions>-c -fast</buildoptions>  
  <buildversion>7.2-3</buildversion>  
  <buildhost>Linux</buildhost>  
  <buildtarget>x86-64</buildtarget>  
  <buildlanguage>C</buildlanguage>  
  <builddate>07/08/2008</builddate>  
</build>
```

The <units> element contains zero or more <unit> elements. Each <unit> element contains information about a single program unit (C or C++ function, or Fortran subroutine, function, or module). Within a <unit> element, there is a <unitinfo> element containing information about the program unit. The <unitinfo> contains the elements:

<unitname> - program unit name
<unitabiname> - name used in the assembler, as decorated by the ABI
<unitlines> - line numbers in the source file
<unittype> - type of program unit, for Fortran

The <unit> element also contains a <messages> element. The <messages> element contains zero or more <message> elements and zero or more <inlined> elements, in any order.

```
UNITS ::= <units> UNIT* </units>  
UNIT ::= <unit> UNITINFO MESSAGES </unit>  
UNITINFO ::= <unitinfo> UNITNAME UNITABINAME? UNITLINES?  
            UNITTYPE? </unitinfo>  
MESSAGES ::= <messages> [MESSAGE | INLINED]* </messages>
```

example:

```
<units>  
  <unit>  
    <unitinfo>  
      <unitname>test</unitname>  
      <unitabiname>test_</unitabiname>  
      <unitlines>1-10</unitlines>
```

```
    <unittype>subroutine</unittype>
  </unitinfo>
  <messages>
    ...
  </messages>
</unit>
</units>
```

A `<message>` contains the elements:

- `<messageline>` - line number to which the message corresponds
- `<messagevar>` - name of a variable, if the message is about a variable
- `<messagefunc>` - name of a function or routine, if the message is about a function or routine
- `<messageid>` - a message ID or key, used to index the message repository
- `<messagetext>` - text of the message; if the message repository is available, this is redundant
- `<messageargs>` - arguments and values used for substitution in the message text and message explanatory information in the repository.

The structure of the message text and message args allow for substitution of names or values into a message.

The `<inlined>` element contains information and messages for routines inlined into this program unit. It contains a `<inlineinfo>` element and a nested `<messages>` element. The `<inlineinfo>` element contains the elements:

- `<inlinename>` - name of the inlined function
- `<inlinemangledname>` - for C++, the mangled name of the inlined function
- `<inlineline>` - line number of the call where the routine was inlined
- `<inlinefile>` - file name containing the inlined function
- `<inlinesrcline>` - source line of the inlined function in inlinefile
- `<inlinelevel>` - integer level of inlining, where 1 means inlined directly into this routine, 2 means inlined into a level-1 routine, and so on

```
MESSAGE ::= <message> MESSAGELINE? MESSAGEVAR? MESSAGEFUNC?
MESSAGEID MESSAGETEXT? MESSAGEARGS? </message>
```

```
INLINED ::= <inlined> INLINEINFO MESSAGES? </inlined>
```

```
INLINEINFO ::= <inlineinfo> INLINENAME INLINEMANGLEDNAME?
INLINELINE? INLINEFILE? INLINESRCLINE? INLINELEVEL?
</inlineinfo>
```

The complete BNF is show below:

```
CCFFBIN ::= <ccffbin> CCFF* </ccffbin>

CCFF ::= <ccff version="..."> SOURCE? BUILD? UNITS? </ccff>

SOURCE ::= <source> SOURCENAME? SOURCEPATH? SOURCEDIR? </source>

SOURCENAME ::= <sourcename> TEXT </sourcename>
SOURCEPATH ::= <sourcepath> TEXT </sourcepath>
SOURCEDIR ::= <sourcedir> TEXT </sourcedir>

BUILD ::= <build> BUILDCOMPILER? BUILDVENDOR? BUILDOPTIONS?
        BUILDVERSION? BUILDHOST? BUILDTARGET? BUILDLANGUAGE?
        BUILDDATE? </build>

BUILDCOMPILER ::= <buildcompiler> TEXT </buildcompiler>
BUILDVENDOR ::= <buildvendor> TEXT </buildvendor>
BUILDOPTIONS ::= <buildoptions> TEXT </buildoptions>
BUILDVERSION ::= <buildversion> TEXT </buildversion>
BUILDHOST ::= <buildhost> TEXT </buildhost>
BUILDTARGET ::= <buildtarget> TEXT </buildtarget>
BUILDLANGUAGE ::= <buildlanguage> TEXT </buildlanguage>
BUILDDATE ::= <builddate> TEXT </builddate>

UNITS ::= <units> UNIT* </units>

UNIT ::= <unit> UNITINFO MESSAGES? </unit>

UNITINFO ::= <unitinfo> UNITNAME UNITABINAME? UNITLINES?
            UNITTYPE? </unitinfo>

UNITNAME ::= <unitname> TEXT </unitname>
UNITABINAME ::= <unitabiname> TEXT </unitabiname>
UNITLINES ::= <unitlines> TEXT </unitlines>
UNITTYPE ::= <unittype> TEXT </unittype>

MESSAGES ::= <messages> [MESSAGE|INLINED]* <messages>

MESSAGE ::= <message> MESSAGELINE? MESSAGEVAR? MESSAGEFUNC?
            MESSAGEID MESSAGETEXT? MESSAGEARGS? </message>

MESSAGELINE ::= <messageline> TEXT </messageline>
MESSAGEVAR ::= <messagevar> TEXT </messagevar>
MESSAGEFUNC ::= <messagefunc> TEXT </messagefunc>
```

```
MESSAGEID ::= <messageid> TEXT </messageid>
MESSAGETEXT ::= <messagetext> TEXT </messagetext>
MESSAGEARGS ::= <messageargs> TEXT </messageargs>

INLINED ::= <inlined> INLINEINFO MESSAGES? </inlined>

INLINEINFO ::= <inlineinfo> INLINENAME INLINEMANGLEDNAME?
  INLINELINE? INLINEFILE? INLINESRCLINE? INLINELEVEL?
  </inlineinfo>

INLINENAME ::= <inlinename> TEXT </inlinename>
INLINEMANGLEDNAME ::= <inlinemangledname> TEXT
  </inlinemangledname>
INLINELINE ::= <inlineline> TEXT </inlineline>
INLINEFILE ::= <inlinefile> TEXT </inlinefile>
INLINESRCLINE ::= <inlinesrcline> TEXT </inlinesrcline>
INLINELEVEL ::= <inlinelevel> TEXT </inlinelevel>
```