Draft Proposal for Unifying Path Names

Peter Ashenden

16 April 2004

1. Introduction

Path names occur in various forms in VHDL, both as it stands in IEEE Std 1076-2002 and in proposed revisions:

1. In expanded names
2. In the 'path_name and 'instance_name attributes
3. In the DefName and FullName prperties proposed for VHPI
4. In proposed cross module references (XMRs)
5. In proposed PSL descriptions

Path names are used for two purposes:

- As an informative identifier for a named entity
- To uniquely identify a named entity

It would be desirable to have a uniform syntax for all of these path names, while preserving backward compatibility as much as is possible. Current proposals for VHPI and XMRs are based on the path name syntax used in the 'path_name attribute. However, use of the colon character as a delimiter is problematic for PSL, which needs to deal with mixed VHDL/Verilog descriptions. A similar problem may arise with other occurrences of path names in a mixed-language design. Expanded names use the dot character as a delimiter, which is inconsistent with the 'path_name attribute.

This proposal examines the forms of path name that are required in different contexts and suggests a syntax that is uniform across those contexts.

2. Path Name Contexts

Three contexts for path names have been identified:

1. Library unit path names

   This form of name identifes a named entity in an uninstantiated design unit that resides in a library. The named entity may be the design unit itself, or a named entity nested within one or more declarative regions.

2. Design hierarchy path names

   This form of name identifies an instance of a named entity in the elaborated design hierarchy. The named entity may be the root design entity itself; or, recursively, a named entity declared within the design entity, a named entity declared with a nested declarative region, a named entity declared within an instance of a protected type, or a named entity declared within a design entity bound to a component instantiation statement.

3. Wildcard path names

This form of name identifies zero or more instances of a named entity in the elaborated design hierarchy. It consists of a head part that denotes a design entity in a library and a tail part that denotes a path to an instance of a named entity in any instance of the design entity.

In addition to supporting these contexts, a name syntax should support specification of relative path names, with the name of a declarative region or region instance giving the scope for interpretation of the relative name.

2.1. Expanded names

Expanded names occur in context (1) and are used as unique identifiers of named entities. They are relative path names, interpreted relative to the innermost visible occurrence of the first element of the path name. For example, an expanded name A.B.C.D is interpreted by finding the innermost visible occurrence of A, then treating B.C.D as a relative path from the region denoted by A.

2.2. Path_name and Instance_name attributes

The 'path_name and 'instance_name attributes occur in context (2) and are used as informative identifiers of named entities. They don't uniquely identify an instance of a named entity. There are at least two forms of ambiguity. First, if a library containing a package in the design hierarchy has the same name as the root design entity, the 'path_name attributes for two named entities can be the same. For example ":L:P:X" might be a named entity X in a package P in library L, or a named entity X in a region P in the root design entity L. Second, for a named entity in a subprogram, there is no way to distinguish between different instances occurring in different activations of the subprogram. Similarly, for a named entity in a protected type body, there is no way to distinguish between different instances occurring in different shared variables of the protected type.

2.3. VHPI name properties

The DefName property in the current VHPI proposal occurs in context (1) and is intended to uniquely identify a named entity. It is of the form

@ <library_name>
. <primary_unit_name> [: <architecture_name> | body]
{ . <item_name> [<signature>] }

The combination of the leading '@' character and the use of '.' delimiters indicates that it is a library unit path name.

The FullName property in the current VHPI proposal occurs in context (2) and is intended to uniquely identify a named entity. It has various forms. If the named entity is statically elaborated and occurs within an elaborated package, the path name is of the form

@ <library_name> :
<package_name> :
[<item_name> [<signature>]]

If the named entity is statically elaborated and occurs within the design hierarchy other than in a package, the path name is of the form

: <root_entity_name > :
{ <region_name> : }
[<item_name> [<signature>]]

Each <region_name> is the label of a statement (block, generate, process, component instantiation) in the hierarchy between the root and the named entity. In the case of a for-generate, the <region_name> includes the value of the generate parameter.

The VHPI proposal does not currently address the path name of a named entity within a shared variable of a protected type. It could do so by including the variable name as a <region_name>. Since a shared variable can occur in a package, the form of path name for a statically elaborated named entity in a package would have to be augmented to include an optional <region_name> after the <package_name>.

If the named entity is dynamically elaborated (ie, it is declared within a subprogram that is activated), the path name is of the form

<ancestor_full_name> { <call_depth> } :
[<item_name> [<signature>]]

where the <parent_full_name> is the hierachical path name to the process from which the subprogram was called, and the <call_depth> indicates how many stack frames down from the process to get to the named entity.

For both the DefName and FullName properties, implicitly defined labels are used for unlabelled statements whose labels are needed as path name elements. The labels are of the form _Pn for process statements and _Ln for for-loop statements.

The VHPI name properties also provide for identifying an element of an array or record or a slice of a record by appending an index, element name or range using the same syntax as VHDL names. Indices and range bounds have to be literals.

2.4. XMR names

Steve to advise...

2.5. PSL names

Erich to advise...

3. Abstract syntax for path names

3.1. Library unit path name

A library unit path name that needs to uniquely identify a named entity can be formed from the following elements:

<library_name>
<primary_unit_name> [<secondary_unit_name>]
{ <item_name> [<signature>] }

The <library_name> is the logical name of the library containing the design unit. The <primary_unit_name> is the either the simple name of the entity if the named entity is declared in an entity declartion or architecture body, or the simple name of the package if the named entity is declared in a package declaration or body. The <secondary_unit_name> is either the simple name of the architecture if the named entity is declared in an architecture body, or an indicator of some sort if the named entity is declared in a package body. The sequence of item names are names of nested declarative regions within which the named entity is declared. If any

of those is a subprogram, the signature is included. The simple name of the named entity (if the named entity is other than the design unit itself) is the last item name in the sequence.

3.2. Design Hierarchy path name

A design hierachy path name that needs to uniquely identify an instance of a named entity must include elements that identify parent region instances, whether those instances be statically or dynamically elaborated.

For a statically elaborated named entity declared within a package or instantiated within a shared variable that is declared within a package, the elements in the path name are

<library_name>
  <package_name>
  [<region_name>]
  [<item_name> [<signature>]]

The <library_name> is the logical name of the library containing the package, and the <package_name> is the simple name of the package. The <region_name> is included if the named entity is within a shared variable; in that case, the <region_name> is the simple name of the shared variable. The simple name of the declared entity (if the named entity is other than the package itself) is the <item_name> at the end of the path name.

For a statically elaborated named entity declared within a design entity or instantiated within a shared variable that is declared within a design entity, the elements of the path name are

<root_entity_name>
  { <region_name>}
  [<item_name> [<signature>]]

Each <region_name> is the label of a statement (block, generate, process, component instantiation), or simple name of a shared variable, in the hierarchy between the root and the named entity. In the case of a for-generate statement, the <region_name> includes the value of the generate parameter. In the case of an unlabeled process statement, the <region_name> is a surrogate identifier, unique in the region in which the process statement occurs, but deterministically generated so that the process can be identified. The simple name of the declared entity (if the named entity is other than the root entity) is the <item_name> at the end of the path name.

For a dynamically elaborated named entity, a design hierarchy path only has validity for the lifetime of the named entity. At that time, there is a statically elaborated ancestor region from which dynamic elaboration commenced. The ancestor region is either a statically elaborated declarative region, if the subprogram is called as part of elaborating a declaration or an interface element in a port map or generic map; or it is a process statement or equivalent, if the program is called from a statement. In either case, the design hierarchy path name of the ancestor can be used as a head part of the named entity’s path name to uniquely identify the site at which dynamic elaboration commenced. For that given site, there may be multiple activations of the subprogram containg the named entity, so some indication destinquishing among them is required. (In the VHPI FullName property, this is the call depth.) Putting all of this together, the elements of the path name are

<ancestor_full_name> { <activation_indicator> }
  [<item_name> [<signature>]]

Note that this is different from the 'path_name attribute, in which the path given leads to the declarative region in which the subprogram is declared. The site at which dynamic elaboration

starts might be further nested within concurrent statements in that region (in which the subprogram is visible), leading to a longer path to the dynamic elaboration site.

3.3. Wildcard path names

Erich to review and advise...

A wildcard path name can be formed from the following elements

<library_name>
  <entity_name> [<architecture_name>]
  { <region_name>}
  [<item_name> [<signature>]]

The is a hybrid of a library unit and a design hierachy path name, and matches zero or more instances of a named entity in a design hierarchy. The <library_name> and <entity_name> specify an entity declaration; the wildcard path name only matches named entities for which the specified entity declaration is an ancestor in the design hierachy. If the <architecture_name> is included, matching is further constrained to require the design entity formed by the entity and the architecture to be an ancestor.

The remainder of the path specifies a chain of containing regions in the design hierarchy down to a named entity. The path matches any named entity whose simple name is the same as the item name and which is contained in the specified chain of regions starting from an instance of the specified design entity. If the chain of regions and the <item_name> is omitted, the path matches just the design entity instance.

4. Issues

- unlabelled statements
  - process statements and equivalents
  - loop statements (to get to loop parameter)

- library name for package may be same as top-level entity

- delimiter between path elements

- should there be a trailing delimiter for a name that denotes a region?