0.1 Compiled IP Directory Substructure

This section describes the substructure of a compiled IP directory. This directory seems to be rather exhaustive, but there are only a few items that are mandatory. Many information is only required for a certain set of compiled simulation IP (e.g. Full Chip Simulation models) or in cases ... Mandatory information is:

- the model's source code
- the model's object code
- the build environment; at a minimum a Makefile that is specific for the type of compiled IP

...

Path name	Description	???
`<parent>/`	Parent directory	
Model Source Code		
`source/`	Model Source Directory	
`*.c, *.cpp, *.h`	C (*.c) or C++ (*.cpp) source and local header files; can be grouped within a set of subdirectories).	
`include/`	Model Include Directory	
`*.h`	Include files describing the external interface of the compiled IP.	
Model Object Code		
`<platform>/`	Platform specific object code and data (identifier must be compliant with the list defined in ...)	
`*.o`	Object code	
`*.a, *.lib`	Archives and static libraries needed for static linking	
`*.so, *.sl, *.dll`	Shared libraries	
`*.tab`	PLI table file; must be provided for PLI and ...	
Build Environment		
`makefile.<csip_type>`	Standalone Makefile used to compile the source files into the appropriate type of object code.	
`...`	...	
Regression Test Suite		
`...`	...	
Simulator Integration		
`iftype`	...	
`*.v, *.V`	Verilog wrapper for the compiled IP This item is optional. ...	
`config/`	Configuration files needed by the simulator interface ...	
`caf.<simulator>`	Simulator specific arguments to	

Table 1: Directory Structure

Notes:

EXAMPLE

1. Model specific source code must be stored in the **`source`** directory, unless it is ...
2. Every source directory (**`source`**, **`...`**) might have a module substructure to denote and structure the various modules or utilities contained. Also, third party software might impose its own, specific substructure within a module directory. It is recommended to provide an appropriate build script for each module within these directories.
3. Header files shall be stored in the `include` directory if and only if they are required for the integration of the corresponding IP within other modules. Files that are only referred from files within this block must reside in conjunction with their respective source files in the **`source`** directory. This minimizes the externally visible information and clearly defines the external interface of a component. As such, it is in line with object oriented (OO) principles of information hiding and reduces possible namespace problems or the chance for clashes in header file naming.
4. The **`include`** directory must not have a substructure; it must be empty, if there a no header files required to define the interface. The files stored in this directory can be copies of include files in the **`source`** directory, in this case the Makefile must take care of copying them into the **`include`** directory. This still allows modularity below the **`source`** directory.
5. The only purpose of the Build Environment is to compile the C/C++ code intended to integrate into the simulation. It is not related to the code required to integrate this code into the simulation. ...
6. The Build Environment might consist of a single Makefile; the ... The name of the Makefile is dependent on the type of compiled simulation IP:

   a.

   b.

   c.