# HDL Primitives

## HDL Primitives

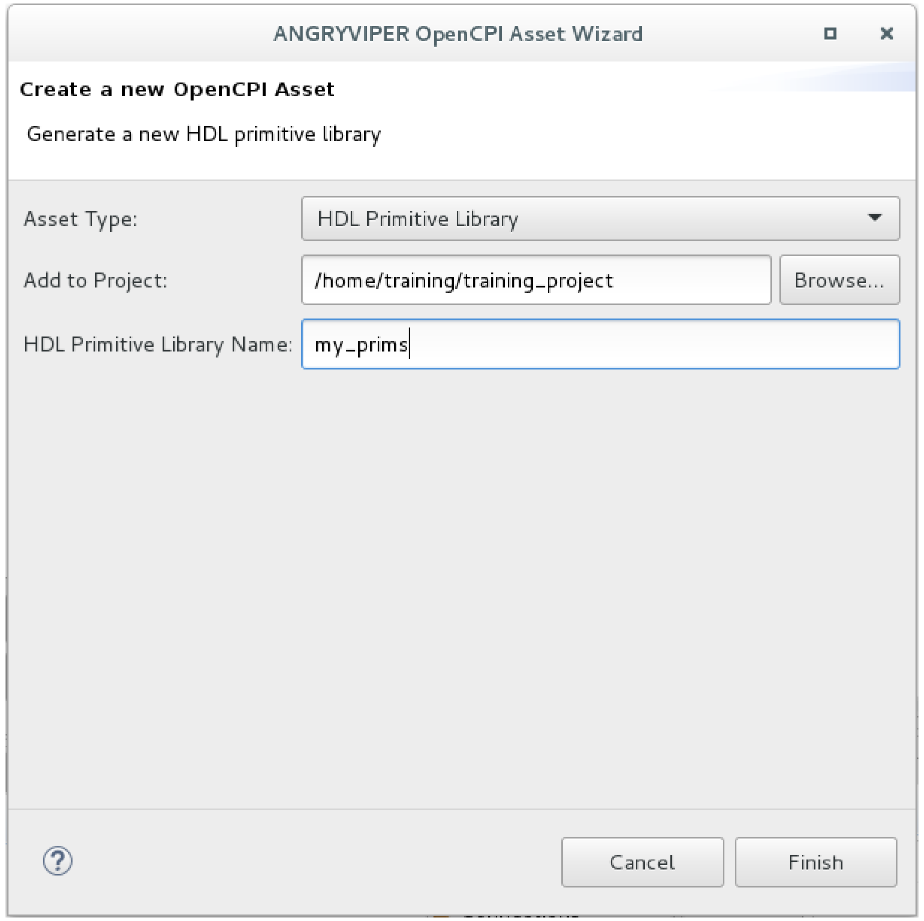
- HDL Primitives and Primitive Libraries are the preferred way in OpenCPI to store/present reusable HDL for use in multiple workers
- Including HDL Primitives requires modifying Makefiles **(not supported in IDE yet)**
  - May be referenced in
    - *Project.mk* file using *HdlLibraries*,
    - Component library Makefile using *HdlLibraries*,
    - Worker Makefiles using *Libraries*
- Not required, but recommended practice
  - Alternatively, if used in a single worker, source could be added to the worker’s directory and to the *SourceFiles* makefile variable
- **May NOT have the same name as an HDL Application Worker**

## HDL Primitives - continued

- HDL Primitive Library
  - Collection of modules
  - Built from HDL source code (ex: FIR, CIC, etc.)
- HDL Primitive Core
  - Single modules built from HDL source code or generated by vendor tools
    - Vendor Dependent (ex. Xilinx/Coregen or Vivado, Altera/MegaWizard)
  - Prebuilt Cores from 3rd party (.qxp/.edf/.ngc)
- Primitives may depend on each other
  - Core depends on primitive libraries
  - Primitive libraries depend on primitive libraries
    - use *PrimitiveLibraries* to define the build order in the hdl/primitives/Makefile
    - use *Libraries* when one primitive depends on another in the hdl/primitives/<my_prim>/Makefile
  - Circular dependencies are not supported

# HDL Primitive Libraries

- Library creation supported in IDE, but population not supported yet
  - Makefile editing still required
- Directory for the primitive library created at *hdl/primitives/<my_prims>/*
- Makefile contains, at a minimum, *include $(OCPI_CDK_DIR)/include/hdl/hdl-library.mk*
- Libraries build in per-target directories named *target-<hdl-target>*
- Library must include the VHDL file *<libname>_pkg.vhd* used for component declarations and unique type declarations
- Can have multiple **_pkg.vhd* files and separate package body **-body.vhd* files


ANGRYVIPER OpenCPI Asset Wizard

**Create a new OpenCPI Asset**

Generate a new HDL primitive library

Asset Type: HDL Primitive Library

Add to Project: /home/training/training_project

HDL Primitive Library Name: my_prims

## HDL Primitive Libraries - continued

- VHDL package name doesn’t have to be the same as the library’s name
- Ex:  Single package for a primitive library *mylib*
  - *library mylib; use mylib.mylib.all;*
- Ex:  Multiple packages in the primitive library *mylib*; different package names
  - *library mylib; use mylib.mypkg.all;*
- *SourceFiles*
  - used to define build dependency order within a primitive library
  - required when a multilevel directory structure is used within the primitive library
- Log output of tools found in *target-<hdl-target>/<libname>-<tool>.out*
- Exportable results for all primitive libraries & cores found in *hdl/primitives/lib*

# HDL Primitive Cores

- Directory for the primitive core created at *hdl/primitives/<my_core>/*
- Makefile contains *include $(OCPI_CDK_DIR)/include/hdl/hdl-core.mk*
- Can be a mix of source and prebuilt files
  - Pre-synthesized core files (Xilinx Vivado *.edf, Xilinx ISE *.ngc or Altera *.qxp)
- VHDL instantiation must have a package file <*corename*>*_pkg.vhd*
- Verilog instantiation must have a “black box” empty module definition file <*corename*>*_bb.v*
- Optional Makefile/ocpidev variable *Top*
  - Specifies the top module name of core when different than ocpidev name <*my_core*>
- Optional Makefile/ocpidev variable *PreBuiltCore*
  - Specifies a file that is not source code