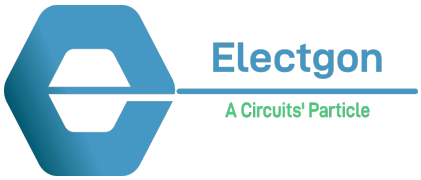


Compiling Xilinx Libraries for QuestaSim

ELECTGON
www.electgon.com
contact@electgon.com

08.06.2019

1 Using ISE Flow

In order to make functional simulation for your design, all design components shall be defined in the main working library. For designs that are based on FPGA resources or primitives, these primitives shall be defined in the main working library also. If we need then to simulate a design that is using for instance output buffer of a Xilinx FPGA, we have to compile this instance in order to take effect in the functional simulation. In this guide, we will see how to use Xilinx primitives in a functional simulation using QuestaSim.

Xilinx already provides a tool for that purpose. Using the tool compxlib you can compile Xilinx libraries to be used by Questa. The syntax for this command is *compxlib [options]*

For example, the following command can be used

```
compxlib -s questa -arch all -l all -lib all -p /path/to/Questa_executable -dir
    /path/to/output_compiled_libraries
```

Option	Declaration
-s	is used to specify the tool for which the libraries are compiled.
-arch	is used to specify which FPGA family is targeted for compilation. You can define the device family explicitly or use 'all' to compile for all Xilinx device families. The following are the available keyword families for that option acr2 - aspartan3 - aspartan3a - aspartan3adsp - aspartan3e - aspartan6 - kintex7 - kintex7l - qrvirtex4 - qvirtex4 - qvirtex5 - qspartan6 - qvirtex6 - spartan3 - spartan3a - spartan3adsp - spartan3e - spartan6 - virtex4 - virtex5 - virtex6 - virtex6l - virtex7 - virtex7l - xa9500xl - xbr - xc9500 - xc9500xl - xpla3.
-l	is used to specify for which language the compilation shall be. Possible values are verilog, vhdl, all.
-lib	is used to specify which Xilinx is needed to be compiled. Available values are: Unisim – simprim – uni9000 – xilinxcorelib – coolrunner – edk – all You can use all to compile all these libraries. If you need to compile some of them, for instance two libraries only, you can specify it explicitly like compxlib –lib unisim –lib simprim …
-p	is used to specify the path of the target simulator executables.
-dir	is used to specify where the output of the compilation process should be. If not specified the output will reside in the following location Linux $XILINX/language/target_simulator/version/{lin|lin64} Windows %XILINX%\language\target_simulator\version\{nt|nt64}
Other Options	There are other options that can be used with this command: -64bit -cfg –e –exclude_superseded –exclude_sublib –f –info –log –source_lib –verbose –w For more details about these options, please refer to Xilinx Command Line Tools User Guide.

Running previous example will take long time (around 1 hour) because the target is compiling all libraries of all FPGA families for all languages. The output is as shown in the figure 1.

Figure 1: Compile Output

What is important from this output is the modelsim.ini file. In this file, location of the compiled libraries is specified. Now in order to use these compiled libraries all what you have to do is to copy this file into your main simulation directory (directory from which you run a do file or directory in which your simulation project resides).

2 Using Vivado Design Flow

Since Xilinx is replacing currently the ISE flow with the new one Vivado, previous execution command can be done using Vivado design flow using TCL command as follows

```
vivado -mode tcl -source vivado_compileXilinx_lib.tcl
```

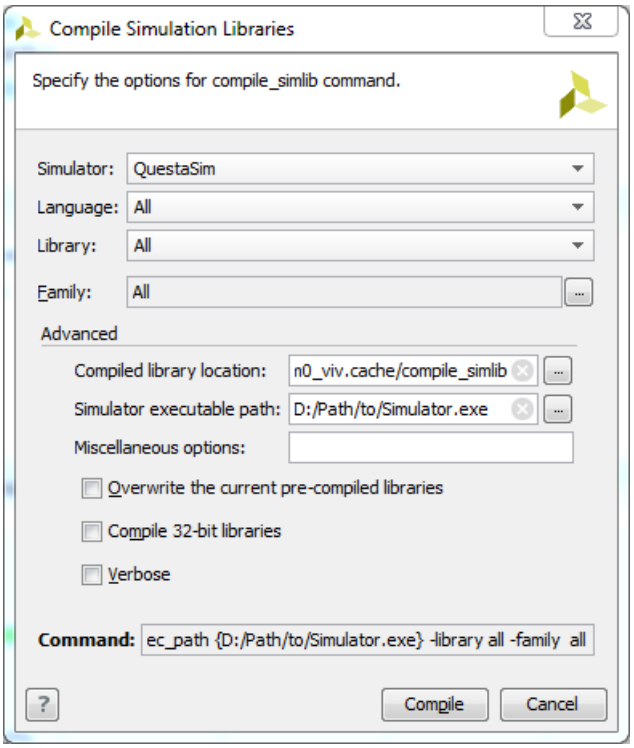
where vivado_compileXilinx_lib.tcl is a tcl file that shall include your tcl command of vivado that can compile simulation libraries

```
compile_simlib -language all -dir {/path/to/output_Compiled_lib} -simulator
    questa -simulator_exec_path {/path/to/questa_executable} -library all
    -family  all
```

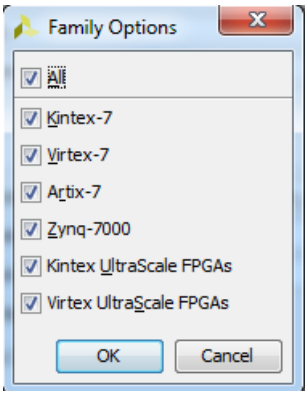
Note that compile_simlib has the following available options

Option	Declaration
-directory	Directory path for saving the compiled results.
-family	Selects device architecture. Default: all
-force	Overwrites the pre-compiled libraries
-language	Compiles libraries for this language. Default: all
-library	Selects library to compile. Default: all
-print_library_info	Prints pre-compiled library information
-simulator	Compiles libraries for this simulator
-simulator_exec_path	Uses simulator executables from this directory
-source_library_path	If specified, this directory is searched for the library source files before searching the default path(s) found in the environment variable XILINX_VIVADO for Vivado
-32bit	Performs the 32-bit compilation
-quiet	Ignores command errors
-verbose	Suspends message limits during command execution

Compile process can be executed also using GUI tool of Xilinx like Vivado. This can be simply done by clicking in the top toolbar on Tools>>Compile Simulation Libraries. The following pop-up shall appear to define your parameters



Note that in Vivado only the following architectures are supported



In some cases, some developers have old designs that were built to work with old Xilinx architectures (Spartan 6 and older FPGAs). They include some primitives that belong to these old architectures. At the same time they have other designs that are built for recent architectures (7 series and later). The requirement then is to have one compiled libraries instead of compiling the libraries for old architectures and recent architecture (having two compiled operations will result in two different modelsim.ini file which may be confusing for the simulation environment settings).

In this case Vivado flow can be used to compile for old architectures also if it has been executed with this option -ise_install_path

In our example the command to do that shall be

```
compile_simlib -language all -dir {/path/to/output_Compiled_lib}
    -ise_install_path {/path/to/ISE_DS/ISE} -simulator questa
    -simulator_exec_path {/path/to/questa_executable} -library all -family  all
```