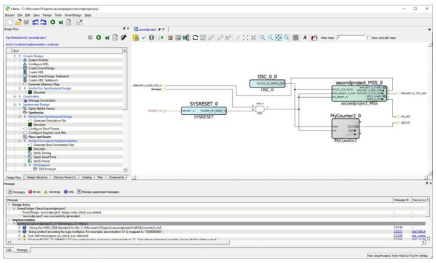
Manuals+ — User Manuals Simplified.



MICROCHIP Libero SoC Simulation Library Software User Guide

Libero SoC Simulation
Library Setup Instructions

Contents

Introduction

(Ask a Question)

The purpose of this document is to describe the procedure to set up the simulation environment using a Libero SoC project as the input. This documentation corresponds to the pre-compiled libraries provided for use with Libero SoC v11.9 and newer software releases. The libraries provided are compiled for Verilog. VHDL users require a license allowing mixed-mode simulation.
The compiled simulation libraries are available for the following tools:

- Aldec Active-HDL
- Aldec Riviera-PRO
- Cadence Incisive Enterprise and Xcelium
- Siemens QuestaSim
- Synopsys VCS

To request a library for a different simulator, contact **Microchip Technical Support**.

Libero SoC Integration

(**Ask a Question**)

Libero SoC supports simulation using ModelSim ME by generating a run.do file. This file is used by ModelSim ME/ModelSim Pro ME to set up and run the simulation. To use other simulation tools, you can generate the ModelSim ME/ModelSim Pro ME run.do and modify the Tcl script file to use the commands that are compatible with your simulator.

1.1 Libero SoC Tcl File Generation (Ask a Question)

After creating and generating design in Libero SoC, start a ModelSim ME/ModelSim Pro ME simulation under all design phases (presynth, postsynth, and post-layout). This step generates the run.do file for the ModelSim ME/ModelSim Pro ME for each design phase.

Important: After starting each simulation run, rename the auto-generated run.do file under the simulation directory to prevent Libero SoC from overwriting that file. For example, the files can be renamed to presynth_run.do, postsynth_run.do and postlayout_run.do.

Aldec Setup for Active-HDL and Riviera-Pro (Ask a Question)

The run.do file used by the ModelSim ME/ModelSim Pro ME can be modified and used for simulation using the Aldec simulators.

2.1 Environment Variable (Ask a Question)

Set your environment variable to your license file location:
LM_LICENSE_FILE: must include a pointer to the license server.

2.2 Download Compiled Library (Ask a Question)

Download the libraries for the Aldec Active-HDL and the Aldec Riviera-PRO from the Microchip website.

2.3 Converting run.do for Aldec simulation (Ask a Question)

The run.do files generated by Libero SoC for simulations using the Active-HDL and Riviera-Pro tool can be used for simulations using Active-HDL and Riviera-Pro with a single change. The following table lists the Aldec-equivalent commands to modify in the ModelSim run.do file.

Table 2-1. Aldec Equivalent Commands

ModelSim	Active-HDL
vlog	alog
vcom	acom
vlib	alib
vsim	asim
vmap	amap

Following is a sample run.do related to Aldec simulators.

1. Set the location of the current working directory.
   set dsn <simulation directory>
2. Set a working library name, map its location, and then map the location of Microchip FPGA family precompiled libraries (for example, SmartFusion2) on which you are running your design.
   alib presynth
   amap presynth presynth
   amap SmartFusion2 <location of the precompiled libraries>
3. Compile all the necessary HDL files used in the design with the required library.
   alog –work presynth temp.v (for Verilog)
   alog –work presynth testbench.v
   acom –work presynth temp.vhd (for Vhdl)
   acom –work presynth testbench.vhd
4. Simulate the design.
   asim –L SmartFusion2 –L presynth –t 1ps presynth.testbench
   run 10us

2.4 Known Issues (Ask a Question)
This section lists the known issues and limitations.

- Libraries compiled using Riviera-PRO are platform specific (i.e. 64-bit libraries cannot be run on 32-bit platform and vice versa).
- For designs containing SERDES/MDDR/FDDR, use the following option in your run.do files while running simulations after compiling their designs:
  – Active-HDL: asim –o2
  – Riviera-PRO: asim –O2 (for presynth and post-layout simulations) and asim –O5 (for post- layout simulations)
  The Aldec setup for Active-HDL and Riviera-Pro has the following pending SARs. For more information, contact **Microchip Technical Support**.
- SAR 49908 – Active-HDL: VHDL Error for Math block simulations
- SAR 50627 – Riviera-PRO 2013.02: Simulation errors for SERDES designs
- SAR 50461 – Riviera-PRO: asim -O2/-O5 option in simulations

Cadence Incisive Setup (Ask a Question)

You need to create a script file similar to the ModelSim ME/ModelSim Pro ME run.do to run the Cadence Incisive simulator. Follow these steps and create script file for NCSim or use the script file provided to convert the ModelSim ME/ModelSim Pro ME run.do files into the configuration files needed to run the simulations using NCSim.

Important: Cadence has stopped releasing new versions of the Incisive Enterprise simulator and started supporting Xcelium simulator.

3.1 Environment Variables (Ask a Question)
To run the Cadence Incisive simulator, configure the following environment variables:

1. LM_LICENSE_FILE: must include a pointer to the license file.

2. cds_root: must point to the home directory location of the Cadence Incisive Installation.
3. PATH: must point to the bin location under the tools directory pointed by cds_root that is, $cds_root/tools/bin/64bit (for a 64-bit machine and $cds_root/tools/bin for a 32-bit machine).

   There are three ways of setting up the simulation environment in case of a switch between 64-bit and 32-bit operating systems:

Case 1: PATH Variable
Run the following command:
set path = (install_dir/tools/bin/64bit $path) for 64bit machines and
set path = (install_dir/tools/bin $path) for 32bit machines
Case 2: Using the -64bit Command-line Option
In the command-line specify -64bit option in order to invoke the 64bit executable.
Case 3: Setting the INCA_64BIT or CDS_AUTO_64BIT Environment Variable
The INCA_64BIT variable is treated as boolean. You can set this variable to any value or to a null string.
setenv INCA_64BIT

 **Important: The** INCA_64BIT environment variable does not affect other Cadence tools, such as IC tools. However, for Incisive tools, the INCA_64BIT variable overrides the setting for the CDS_AUTO_64BIT environment variable. If the INCA_64BIT environment variable is set, all the Incisive tools run in 64-bit mode. setenv CDS_AUTO_64BIT INCLUDE:INCA

 **Important: The** string INCA must be in uppercase. All executables must be run in either 32-bit mode or in 64-bit mode, do not set the variable to include one executable, as in the following:
setenv CDS_AUTO_64BIT INCLUDE:ncelab

Other Cadence tools, such as IC tools, also use the CDS_AUTO_64BIT environment variable to control the selection of 32-bit or 64-bit executables. The following table shows how you can set the CDS_AUTO_64BIT variable to run the Incisive tools and IC tools in all modes.
Table 3-1. CDS_AUTO_64BIT Variables

CDS_AUTO_64BIT Variable	Incisive Tools	IC Tools
setenv CDS_AUTO_64BIT ALL	64 bit	64 bit
setenv CDS_AUTO_64BIT NONE	32 bit	32 bit
setenv CDS_AUTO_64BIT EXCLUDE:*ic_binary*	64 bit	32 bit
setenv CDS_AUTO_64BIT EXCLUDE:INCA	32 bit	64 bit

 **Important:** All Incisive tools must be run in either 32-bit mode or in 64-bit mode, do not use EXCLUDE to exclude a specific executable, as in the following: setenv CDS_AUTO_64BIT EXCLUDE:ncelab
If you set the CDS_AUTO_64BIT variable to exclude the Incisive tools (setenv CDS_AUTO_64BIT EXCLUDE:INCA), all Incisive tools are run in 32-bit mode. However, the -64bit command-line option overrides the environment variable.
The following configuration files help you manage your data and control the operation of the simulation tools and utilities:

- Library mapping file (cds.lib)—Defines a logical name for the location of your design.
- Libraries and associates them with physical directory names.

- Variables file (hdl.var)—Defines variables that affect the behavior of simulation tools and utilities.

3.2 Download Compiled Library (Ask a Question)
Download the libraries for Cadence Incisive from Microsemi's website.
3.3 Creating the NCSim Script File (Ask a Question)
After creating a copy of the run.do files, perform these steps to run your simulation using NCSim:

1. Create a cds.lib file that defines the libraries that are accessible and their location. The file contains statements that map library logical names to their physical directory paths. For example, if you are running presynth simulation, the cds.lib file is written as shown in the following codeblock.
   DEFINE presynth ./presynth
   DEFINE COREAHBLITE_LIB ./COREAHBLITE_LIB
   DEFINE smartfusion2 <location of Smartfusion2 precompiled libraries on disk>
2. Create a hdl.var file, an optional configuration file that contains configuration variables, that determines how your design environment is configured. The following variable files are included:
   – Variables that are used to specify the work library where the compiler stores compiled objects and other derived data.
   – For Verilog, variables (LIB_MAP, VIEW_MAP, WORK) that are used to specify the libraries and views to search when the elaborator resolves instances.
   – Variables that allow you to define compiler, elaborator, and simulator command-line options and arguments.
   In case of presynth simulation example shown above, say we have three RTL files: a.v, b.v, and testbench.v, which needs to be compiled into presynth, COREAHBLITE_LIB, and presynth libraries respectively. The hdl.var file can be written as shown in the following codeblock.
   DEFINE WORK presynth
   DEFINE PROJECT_DIR <location of the files>
   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/a.v => presynth)
   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/b.v => COREAHBLITE_LIB)
   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/testbench.v => presynth)
   DEFINE LIB_MAP ($LIB_MAP, + => presynth)
3. Compile the design files using ncvlog option.
   ncvlog +incdir+<testbench directory> –cdslib ./cds.lib –hdlvar ./hdl.var –logfile
   ncvlog.log –update –linedebug a.v b.v testbench.v
4. Elaborate the design using ncelab. The elaborator constructs a design hierarchy based on the instantiation and configuration information in the design, establishes signal connectivity, and computes initial values for all objects in the design. The elaborated design hierarchy is stored in a simulation snapshot, which is the representation of your design that the simulator uses to run the simulation.
   ncelab –Message –cdslib ./cds.lib –hdlvar ./hdl.var –logfile ncelab.log –errormax 15 –
   access +rwc –status worklib.<name of testbench module>:module
   Elaboration During Post-layout simulation
   In case of post-layout simulations, first the SDF file needs to be compiled before elaboration using the ncsdfc command.
   ncsdfc <filename>.sdf –output <filename>.sdf.X
   During elaboration use the compiled SDF output with –autosdf option as shown in the following codeblock.
   ncelab -autosdf –Message –cdslib ./cds.lib –hdlvar ./hdl.var –logfile ncelab.log –errormax

15 –access +rwc –status worklib.<name of testbench module>:module –sdf_cmd_file ./

sdf_cmd_file

The sdf_cmd_file must be as shown in the following codeblock.

COMPILED_SDF_FILE = “<location of compiled SDF file>”

5. Simulate using ncsim. After elaboration a simulation snapshot is created, which is loaded by ncsim for simulation. You can run in batch mode or GUI mode.

   ncsim –Message –batch/-gui –cdslib ./cds.lib –hdlvar ./hdl.var –logfile ncsim.log –

   errormax 15 –status worklib.<testbench module name>:module

 **Important:** All the above three steps of compiling, elaborating, and simulating can be put into a shell script file and sourced from command-line. Instead of using these three steps, design can be simulated in one step using ncverilog or irun option as shown in the following codeblock.

ncverilog +incdir+<testbench location> -cdslib ./cds.lib –hdlvar ./hdl.var <all RTL
files used in the design>
irun +incdir+<testbench location> -cdslib ./cds.lib –hdlvar ./hdl.var <all RTL files
used in the design>

3.3.1 Known Issues (Ask a Question)

Testbench Workaround

Using the following statement for specifying the clock frequency in the testbench generated by user, or the default testbench generated by Libero SoC does not work with NCSim.

always @(SYSCLK)
#(SYSCLK_PERIOD / 2.0) SYSCLK <= !SYSCLK;

Modify as follows to run simulation:

always #(SYSCLK_PERIOD / 2.0) SYSCLK = ~SYSCLK;

 **Important: Compiled** libraries for NCSim are platform specific (i.e. 64 bit libraries are not compatible with 32 bit platform and vice versa).

Postsynth and Post-layout Simulations Using MSS and SERDES While running postsynth simulations of designs containing the MSS block or the post-layout simulations of designs using SERDES, the BFM simulations do not work if the –libmap option is

not specified during elaboration. This is because during elaboration, MSS is resolved from the work library (because of the default binding and the worklib being postsynth/post-layout) where it is just a Fixed Function.

The ncelab command must be written as shown in the following code block to resolve the MSS
block from the SmartFusion2 precompiled library.

ncelab -libmap lib.map -libverbose -Message -access +rwc cfg1
and the lib.map file must be as follows:
config cfg1;
design <testbench_module_name>;
default liblist smartfusion2 <worklib>;
endconfig

This resolves any cell in the SmartFusion2 library before looking in the work library i.e. postsynth/ post-layout.

The –libmap option can be used by default during elaboration for every simulation (presynth, postsynth, and post-layout). This avoids simulation issues that are caused due to resolution of instances from libraries.

ncelab: *F,INTERR: INTERNAL EXCEPTION

This ncelab tool exception is a caveat for designs containing FDDR in SmartFusion 2 and IGLOO 2 during postsynth and post-layout simulations using –libmap option.

 **Important:** This issue has been reported to Cadence support team (SAR 52113).

3.4 Sample Tcl and Shell Script Files (Ask a Question)

The following files are the configuration files needed for setting up the design and shell script file for running NCSim commands.

Cds.lib

```
NE smartfusion2 /scratch/krydor/tmpspace/users/me/nc-vlog64/SmartFusion2
DEFINE COREAHBLITE_LIB ./COREAHBLITE_LIB
DEFINE presynth ./presynth
```

Hdl.var

```
DEFINE WORK presynth
DEFINE PROJECT_DIR /scratch/krydor/tmpspace/sqausers/me/3rd_party_simulators/Cadence/IGLOO2/
ENVM/M2GL050/envm_fic1_ser1_v/eNVM_fab_master
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_addrdec.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_defaultslavesm.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_masterstage.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_slavearbiter.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_slavestage.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_matrix2x16.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite.v => COREAHBLITE_LIB )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB/CCC_0/SB_CCC_0_FCCC.v =>
presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreConfigMaster/
2.0.101/rtl/vlog/core/coreconfigmaster.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreConfigP/4.0.100/rtl/
vlog/core/coreconfigp.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreResetP/5.0.103/rtl/
vlog/core/coreresetp_pcie_hotreset.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreResetP/5.0.103/rtl/
vlog/core/coreresetp.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB/FABOSC_0/SB_FABOSC_0_OSC.v =>
presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB_HPMS/SB_HPMS.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB/SB.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/SERDES_IF_0/
SB_top_SERDES_IF_0_SERDES_IF.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/SB_top.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/testbench.v => presynth )
DEFINE LIB_MAP ( $LIB_MAP, + => presynth )
```

Commands.csh

```
ncvlog +incdir+../../component/work/SB_top -cdslib ./cds.lib -hdlvar ./hdl.var -logfile
ncvlog.log -errormax 15 -update -linedebug
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_addrdec.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/
coreahblite_defaultslavesm.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_masterstage.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_slavearbiter.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_slavestage.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_matrix2x16.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite.v
../../component/work/SB/CCC_0/SB_CCC_0_FCCC.v
../../component/Actel/DirectCore/CoreConfigMaster/2.0.101/rtl/vlog/core/coreconfigmaster.v
../../component/Actel/DirectCore/CoreConfigP/4.0.100/rtl/vlog/core/coreconfigp.v
../../component/Actel/DirectCore/CoreResetP/5.0.103/rtl/vlog/core/coreresetp_pcie_hotreset.v
../../component/Actel/DirectCore/CoreResetP/5.0.103/rtl/vlog/core/coreresetp.v
../../component/work/SB/FABOSC_0/SB_FABOSC_0_OSC.v ../../component/work/SB_HPMS/SB_HPMS.v
```

../../component/work/SB/SB.v ../../component/work/SB_top/SERDES_IF_0/
SB_top_SERDES_IF_0_SERDES_IF.v
../../component/work/SB_top/SB_top.v ../../component/work/SB_top/testbench.v
ncelab -Message -cdslib ./cds.lib -hdlvar ./hdl.var
-work presynth -logfile ncelab.log -errormax 15 -access +rwc -status presynth.testbench:module
ncsim -Message -batch -cdslib ./cds.lib -hdlvar ./
hdl.var -logfile ncsim.log -errormax 15 -status presynth.testbench:module

3.5 Automation (Ask a Question)

The following script file converts the ModelSim run.do files into configuration files needed to run simulations using NCSim.

Script File Usage

perl cadence_parser.pl presynth_run.do postsynth_run.do
postlayout_run.do Microsemi_Family
Location_of_Cadence_Precompiled_libraries

Cadence_parser.pl

```
#!/usr/bin/perl -w

##########################################################################################################
#Usage: perl questa_parser.pl presynth_run.do postsynth_run.do postlayout_run.do
Microsemi_Family Precompiled_Libraries_location#

##########################################################################################################
use POSIX;
use strict;
my ($presynth, $postsynth, $postlayout, $family, $lib_location) = @ARGV;
&questa_parser($presynth, $family, $lib_location);
&questa_parser($postsynth, $family, $lib_location);
&questa_parser($postlayout, $family, $lib_location);
sub questa_parser {
my $ModelSim_run_do = $_[0];
my $actel_family = $_[1];
my $lib_location = $_[2];
my $state;
if ( -e “$ModelSim_run_do” )
{
open (INFILE,”$ModelSim_run_do”);
my @ModelSim_run_do = <INFILE>;
my $line;
if ( $ModelSim_run_do =~ m/(presynth)/)
{
`mkdir QUESTA_PRESYNTH`;
open (OUTFILE,”>QUESTA_PRESYNTH/presynth_questa.do”);
$state = $1;
} elsif ( $ModelSim_run_do =~ m/(postsynth)/)
{
`mkdir QUESTA_POSTSYNTH`;
open (OUTFILE,”>QUESTA_POSTSYNTH/postsynth_questa.do”);
$state = $1;
} elsif ( $ModelSim_run_do =~ m/(postlayout)/ )
{
`mkdir QUESTA_POSTLAYOUT`;
open (OUTFILE,”>QUESTA_POSTLAYOUT/postlayout_questa.do”);
$state = $1;
```

```
} else
{
print “Wrong Inputs given to the file\n”;
print “#Usage: perl questa_parser.pl presynth_run.do postsynth_run.do postlayout_run.do
\”Libraries_location\”\n”;
}
foreach $line (@ModelSim_run_do)
{
#General Operations
$line =~ s/..\/designer.*simulation\///g;
$line =~ s/$state/$state\_questa/g;
#print OUTFILE “$line \n”;
if ($line =~ m/vmap\s+.*($actel_family)/)
{
print OUTFILE “vmap $actel_family \”$lib_location\”\n”;
} elsif ($line =~ m/vmap\s+(.*_LIB)/)
{
$line =~ s/..\/component/..\/..\/component/g;
print OUTFILE “$line \n”;
} elsif ($line =~ m/vsim/)
{
$line =~ s/vsim/vsim -novopt/g;
print OUTFILE “$line \n”;
} else
{
print OUTFILE “$line \n”;
}
}
close(INFILE);
close(OUTFILE);
} else {
print “$ModelSim_run_do does not exist. Rerun simulation again \n”;
}
}
```

Cadence Xcelium Setup (Microchip Login)

You need to create a script file similar to the ModelSim ME/ModelSim Pro ME run.do to run the Cadence Xcelium simulator. Follow these steps and create script file for Xcelium or use the script file provided to convert the ModelSim ME/ModelSim Pro ME run.do files into the configuration files needed to run simulations using Xcelium.
4.1 Environment Variables (**Ask a Question**)
To run the Cadence Xcelium, configure the following environment variables:

1. LM_LICENSE_FILE: must include a pointer to the license file.
2. cds_root: must point to the home directory location of Cadence Incisive Installation.
3. PATH: must point to the bin location under the tools directory pointed by cds_root (i.e. $cds_root/tools/bin/64bit (for a 64 bit machine and $cds_root/tools/bin for a 32 bit machine).

There are three ways of setting up the simulation environment in case of a switch between 64-bit and 32-bit operating systems:
Case 1: PATH Variable
set path = (install_dir/tools/bin/64bit $path) for 64bit machines and
set path = (install_dir/tools/bin $path) for 32bit machines
Case 2: Using the -64bit Command-line Option
In the command-line specify -64bit option in order to invoke the 64-bit executable.

Case 3: Setting the INCA_64BIT or CDS_AUTO_64BIT Environment Variable
The INCA_64BIT variable is treated as boolean. You can set this variable to any value or to a null string.
setenv INCA_64BIT

Important: The INCA_64BIT environment variable does not affect other Cadence tools, such as IC tools. However, for Incisive tools, the INCA_64BIT variable overrides the setting for the CDS_AUTO_64BIT environment variable. If the INCA_64BIT environment variable is et, all Incisive tools run in 64-bit mode.
setenv CDS_AUTO_64BIT INCLUDE:INCA

Important: The string INCA must be in uppercase. All executables must be run in either 2-bit mode or in 64-bit mode, do not set the variable to include one executable, as in the following:
setenv CDS_AUTO_64BIT INCLUDE:ncelab
Other Cadence tools, such as IC tools, also use the CDS_AUTO_64BIT environment variable to control the selection of 32-bit or 64-bit executables. The following table shows how you can set the CDS_AUTO_64BIT variable to run the Incisive tools and IC tools in all modes.

Table 4-1. CDS_AUTO_64BIT Variables

CDS_AUTO_64BIT Variable	Incisive Tools	IC Tools
setenv CDS_AUTO_64BIT ALL	64-bit	64-bit
setenv CDS_AUTO_64BIT NONE	32-bit	32-bit
setenv CDS_AUTO_64BIT EXCLUDE:ic_binary	64-bit	32-bit
setenv CDS_AUTO_64BIT EXCLUDE:INCA	32-bit	64-bit

Important: All Incisive tools must be run in either 32-bit mode or in 64-bit mode, do not use EXCLUDE to exclude a specific executable, as in the following:
setenv CDS_AUTO_64BIT EXCLUDE:ncelab
If you set the CDS_AUTO_64BIT variable to exclude the Incisive tools (setenv CDS_AUTO_64BIT EXCLUDE:INCA), all Incisive tools are run in 32-bit mode. However, the -64bit command-line option overrides the environment variable.
The following configuration files help you manage your data and control the operation of the simulation tools and utilities:

- Library mapping file (cds.lib) defines a logical name for the location of your design.
- Libraries and associates them with physical directory names.
- Variables file (hdl.var) defines variables that affect the behavior of simulation tools and utilities.

4.2 Download Compiled Library (**Ask a Question**)
Download the libraries for Cadence Xcelium from Microsemi's website.
4.3 Creating the Xcelium script file (**Ask a Question**)
After creating a copy of the run.do files, perform the following steps to run your simulation using Xcelium script file.

1. Create a cds.lib file that defines which libraries are accessible and where they are located.

   The file contains statements that map library logical names to their physical directory paths. For example, if you are running presynth simulation, the cds.lib file can be written as shown in the following codeblock.

DEFINE presynth ./presynth

DEFINE COREAHBLITE_LIB ./COREAHBLITE_LIB

DEFINE smartfusion2 <location of Smartfusion2 precompiled libraries on disk>

2. Create a hdl.var file which is an optional configuration file that contains configuration variables, that determines how your design environment is configured. These include:

   – Variables that are used to specify the work library where the compiler stores compiled objects and other derived data.

   – For Verilog, variables (LIB_MAP, VIEW_MAP, WORK) that are used to specify the libraries and views to search when the elaborator resolves instances.

   – Variables that allow you to define compiler, elaborator, and simulator command-line options and arguments.

   In case of presynth simulation example shown above, say we have 3 RTL files a.v, b.v, and testbench.v, which needs to be compiled into presynth, COREAHBLITE_LIB, and presynth libraries respectively. The hdl.var file can be written as shown in the following codeblock.

   DEFINE WORK presynth

   DEFINE PROJECT_DIR <location of the files>

   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/a.v => presynth)

   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/b.v => COREAHBLITE_LIB)

   DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/testbench.v => presynth)

   DEFINE LIB_MAP ($LIB_MAP, + => presynth)

3. Compile the design files using ncvlog option.

   xmvlog +incdir+<testbench directory> –cdslib ./cds.lib –hdlvar ./hdl.var –logfile

   ncvlog.log –update –linedebug a.v b.v testbench.v

4. Elaborate the design using ncelab. The elaborator constructs a design hierarchy based on the instantiation and configuration information in the design, establishes signal connectivity, and computes initial values for all objects in the design. The elaborated design hierarchy is stored in a simulation snapshot, which is the representation of your design that the simulator uses to run the simulation.

   Xcelium –Message –cdslib ./cds.lib –hdlvar ./hdl.var –logfile ncelab.log –errormax 15 –

   access +rwc –status worklib.<name of testbench module>:module

   Elaboration During Post-layout simulation

   In case of post-layout simulations, first the SDF file needs to be compiled before elaboration using the ncsdfc command.

   Xcelium <filename>.sdf –output <filename>.sdf.X

   During elaboration use the compiled SDF output with –autosdf option as shown in the following codeblock.

   xmelab -autosdf –Message –cdslib ./cds.lib –hdlvar ./hdl.var –logfile ncelab.log –errormax

   15 –access +rwc –status worklib.<name of testbench module>:module –sdf_cmd_file ./

   sdf_cmd_file

   The sdf_cmd_file must be as shown in the following codeblock.

   COMPILED_SDF_FILE = “<location of compiled SDF file>”

5. Simulate using Xcelium. After elaboration a simulation snapshot is created which is loaded by Xcelium for simulation. This can be run in batch mode or GUI mode.

   xmsim –Message –batch/-gui –cdslib ./cds.lib –hdlvar ./hdl.var –logfile xmsim.log –

   errormax 15 –status worklib.<testbench module name>:module

   Cadence Xcelium Setup

Important: All the above three steps of compiling, elaborating and simulating can be put into a shell script file and sourced from command-line. Instead of using these three steps, design can be simulated in one step using ncverilog or xrun option as shown in the following codeblock.

xmverilog +incdir+<testbench location> -cdslib ./cds.lib –hdlvar ./hdl.var <all RTL files used in the design>

xrun +incdir+<testbench location> -cdslib ./cds.lib –hdlvar ./hdl.var <all RTL files used in the design>

4.3.1 Known Issues (Ask a Question)

Testbench Workaround

Using the following statement for specifying the clock frequency in the testbench generated by user or the default testbench generated by Libero SoC does not work with Xcelium.

always @(SYSCLK)
#(SYSCLK_PERIOD / 2.0) SYSCLK <= !SYSCLK;

Modify as follows to run simulation:

always #(SYSCLK_PERIOD / 2.0) SYSCLK = ~SYSCLK;

Important: Compiled libraries for Xcelium are platform specific (i.e. 64 bit libraries are not compatible with 32 bit platform and vice versa).

Postsynth and Post-layout Simulations using MSS and SERDES

While running postsynth simulations of designs containing MSS block, or post-layout simulations of designs using SERDES, the BFM simulations do not work if –libmap option is not specified during elaboration. This is because during elaboration, MSS is resolved from the work library (because of the default binding and the worklib being postsynth/post-layout) where it is just a Fixed Function.

The ncelab command must be written as shown in the following code block to resolve the MSS block from the SmartFusion2 precompiled library.

xmelab -libmap lib.map -libverbose -Message -access +rwc cfg1

and the lib.map file must be as follows:

config cfg1;
design <testbench_module_name>;
default liblist smartfusion2 <worklib>;
endconfig

This must resolve any cell in the SmartFusion2 library before looking in the work library i.e. postsynth/post-layout. The –libmap option can be used by default during elaboration for every simulation (presynth, postsynth and post-layout). This avoids simulation issues that are caused due to resolution of instances from libraries.

xmelab: *F,INTERR: INTERNAL EXCEPTION

This ncelab tool exception is a caveat for designs containing FDDR in SmartFusion2 and IGLOO2 during postsynth and post-layout simulations using –libmap option.

Important: This issue has been reported to Cadence support team (SAR 52113).

4.4 Sample Tcl and shell script files (Ask a Question)

The following files are the configuration files needed for setting up the design and shell script file for running Xcelium commands.

Cds.lib

DEFINE smartfusion2 /scratch/krydor/tmpspace/users/me/nc-vlog64/SmartFusion2
DEFINE COREAHBLITE_LIB ./COREAHBLITE_LIB
DEFINE presynth ./presynth
Hdl.var
DEFINE WORK presynth
DEFINE PROJECT_DIR /scratch/krydor/tmpspace/sqausers/me/3rd_party_simulators/Cadence/IGLOO2/ENVM/M2GL050/envm_fic1_ser1_v/eNVM_fab_master
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/

vlog/core/coreahblite_addrdec.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_defaultslavesm.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_masterstage.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_slavearbiter.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_slavestage.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite_matrix2x16.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/
vlog/core/coreahblite.v => COREAHBLITE_LIB)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB/CCC_0/SB_CCC_0_FCCC.v =>
presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreConfigMaster/
2.0.101/rtl/vlog/core/coreconfigmaster.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreConfigP/4.0.100/rtl/
vlog/core/coreconfigp.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreResetP/5.0.103/rtl/
vlog/core/coreresetp_pcie_hotreset.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/Actel/DirectCore/CoreResetP/5.0.103/rtl/
vlog/core/coreresetp.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB/FABOSC_0/SB_FABOSC_0_OSC.v =>
presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB_HPMS/SB_HPMS.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB/SB.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/SERDES_IF_0/
SB_top_SERDES_IF_0_SERDES_IF.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/SB_top.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, ${PROJECT_DIR}/component/work/SB_top/testbench.v => presynth)
DEFINE LIB_MAP ($LIB_MAP, + => presynth)

Commands.csh

ncvlog +incdir+../../component/work/SB_top -cdslib ./cds.lib -hdlvar ./hdl.var -logfile
ncvlog.log -errormax 15 -update -linedebug
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_addrdec.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/
coreahblite_defaultslavesm.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_masterstage.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_slavearbiter.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_slavestage.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite_matrix2x16.v
../../component/Actel/DirectCore/CoreAHBLite/4.0.100/rtl/vlog/core/coreahblite.v
../../component/work/SB/CCC_0/SB_CCC_0_FCCC.v
../../component/Actel/DirectCore/CoreConfigMaster/2.0.101/rtl/vlog/core/coreconfigmaster.v
../../component/Actel/DirectCore/CoreConfigP/4.0.100/rtl/vlog/core/coreconfigp.v
../../component/Actel/DirectCore/CoreResetP/5.0.103/rtl/vlog/core/coreresetp_pcie_hotreset.v
../../component/Actel/DirectCore/CoreResetP/5.0.103/rtl/vlog/core/coreresetp.v
../../component/work/SB/FABOSC_0/SB_FABOSC_0_OSC.v ../../component/work/SB_HPMS/SB_HPMS.v
../../component/work/SB/SB.v ../../component/work/SB_top/SERDES_IF_0/
SB_top_SERDES_IF_0_SERDES_IF.v
../../component/work/SB_top/SB_top.v ../../component/work/SB_top/testbench.v
ncelab -Message -cdslib ./cds.lib -hdlvar ./hdl.var
-work presynth -logfile ncelab.log -errormax 15 -access +rwc -status presynth.testbench:module
ncsim -Message -batch -cdslib ./cds.lib -hdlvar ./
hdl.var -logfile ncsim.log -errormax 15 -status presynth.testbench:module

4.5 Automation (**Microchip Login**)

The following script file converts ModelSim run.do files into configuration files needed to run simulations using

Xcelium.

Script File Usage

perl cadence_parser.pl presynth_run.do postsynth_run.do
postlayout_run.do Microsemi_Family
Location_of_Cadence_Precompiled_libraries

Cadence_parser.pl

```
#!/usr/bin/perl -w

##########################################################################################################
#Usage: perl questa_parser.pl presynth_run.do postsynth_run.do postlayout_run.do
Microsemi_Family Precompiled_Libraries_location#

##########################################################################################################
use POSIX;
use strict;
my ($presynth, $postsynth, $postlayout, $family, $lib_location) = @ARGV;
&questa_parser($presynth, $family, $lib_location);
&questa_parser($postsynth, $family, $lib_location);


&questa_parser($postlayout, $family, $lib_location);
sub questa_parser {
my $ModelSim_run_do = $_[0];
my $actel_family = $_[1];
my $lib_location = $_[2];
my $state;
if ( -e “$ModelSim_run_do” )
{
open (INFILE,”$ModelSim_run_do”);
my @ModelSim_run_do = <INFILE>;
my $line;
if ( $ModelSim_run_do =~ m/(presynth)/)
{
`mkdir QUESTA_PRESYNTH`;
open (OUTFILE,”>QUESTA_PRESYNTH/presynth_questa.do”);
$state = $1;
} elsif ( $ModelSim_run_do =~ m/(postsynth)/)
{
`mkdir QUESTA_POSTSYNTH`;
open (OUTFILE,”>QUESTA_POSTSYNTH/postsynth_questa.do”);
$state = $1;
} elsif ( $ModelSim_run_do =~ m/(postlayout)/ )
{
`mkdir QUESTA_POSTLAYOUT`;
open (OUTFILE,”>QUESTA_POSTLAYOUT/postlayout_questa.do”);
$state = $1;
} else
{
print “Wrong Inputs given to the file\n”;
print “#Usage: perl questa_parser.pl presynth_run.do postsynth_run.do postlayout_run.do
\”Libraries_location\”\n”;
}
foreach $line (@ModelSim_run_do)
{
#General Operations
$line =~ s/..\/designer.*simulation\///g;
$line =~ s/$state/$state\_questa/g;
```

```
#print OUTFILE “$line \n”;
if ($line =~ m/vmap\s+.*($actel_family)/)
{
print OUTFILE “vmap $actel_family \”$lib_location\”\n”;
} elsif ($line =~ m/vmap\s+(.*._LIB)/)
{
$line =~ s/..\/component/..\/..\/component/g;
print OUTFILE “$line \n”;
} elsif ($line =~ m/vsim/)
{
$line =~ s/vsim/vsim -novopt/g;
print OUTFILE “$line \n”;
} else
{
print OUTFILE “$line \n”;
}
}
close(INFILE);
close(OUTFILE);
} else {
print “$ModelSim_run_do does not exist. Rerun simulation again \n”;
}
}
```

Siemens QuestaSim Setup/ModelSim Setup (Ask a Question)

The run.do files, generated by the Libero SoC for simulations using the ModelSim Microsemi Editions, can be used for simulations using the QuestaSim/ModelSim SE/DE/PE with a single change. In the ModelSim ME/ModelSim Pro ME run.do file, the precompiled libraries location needs to be modified.

Important:
By default, the simulation tool other than the ModelSim Pro ME performs design optimization during simulation that can impact the visibility into simulation artifacts such as design objects and input stimulus.
This is typically helpful in reducing simulation runtime for the complex simulations, using verbose, self-checking testbenches. However, the default optimizations might not be appropriate for all simulations, especially in cases where you expect to graphically inspect the simulation results using the wave window.
To address issues caused by this optimization, you must add appropriate commands and related arguments during simulation to restore visibility into the design. For tool-specific commands, see the documentation of the simulator in-use.

5.1 Environment Variables (Ask a Question)

Following are the required environment variables.

- LM_LICENSE_FILE: must include the path to the license file.
- MODEL_TECH: must identify the path to the home directory location of QuestaSim installation.
- PATH: must point to the executable location pointed by MODEL_TECH.

5.2 Converting run.do for Mentor QuestaSim (Ask a Question)

The run.do files generated by Libero SoC for simulations using ModelSim Microsemi Editions can be used for simulations using QuestaSim/ModelSim_SE with a single change.

Important: All the designs which are simulated using QuestaSim must include -novopt option along with vsim command in the run.do script files.

5.3 Download the Compiled Library (Ask a Question)

Download the libraries for Mentor Graphics QuestaSim from Microsemi’s website.

Synopsys VCS Setup (Ask a Question)

The flow recommended by Microsemi relies on the Elaborate and Compile flow in VCS. This document includes a script file that uses the run.do script files generated by Libero SoC and generates the setup files needed for VCS simulation. The script file uses the run.do file to do the following.

- Create a library mapping file, which is done using the synopsys_sim.setup file located in the same directory where VCS simulation is running.
- Create a shell script file to elaborate and compile your design using VCS.

6.1 Environment Variables (Ask a Question)

Set the appropriate environment variables for VCS based on your setup. The environment variables needed as per the VCS documentation are:

- LM_LICENSE_FILE: must include a pointer to the license server.
- VCS_HOME: must point to the home directory location of the VCS installation.
- PATH: must include a pointer to the bin directory below the VCS_HOME directory.

6.2 Download Compiled Library (Ask a Question)

Download the libraries for Synopsys VCS from Microsemi's website.

6.3 VCS Simulation Script File (Ask a Question)

After setting up VCS and generating the design and the different run.do files from Libero SoC, you must:

1. Create the library mapping file synopsys_sim.setup; this file contains pointers to the location of all the libraries to be used by the design.

   Important: The file name must not change and it must be located in the same directory where simulation is running. Here is an example for such a file for presynthesis simulation.

   WORK > EFAULT

   SmartFusion2 : <location of the SmartFusion2 pre-compiled libraries>

   presynth : ./presynth

   DEFAULT : ./work
2. Elaborate the different design files, including the testbench, using the vlogan command in VCS. These commands may be included in a shell script file. Following is an example of the commands that are needed to elaborate a design defined in rtl.v with its testbench defined in

   testbench.v.

   vlogan +v2k -work presynth rtl.v

   vlogan +v2k -work presynth testbench.v
3. Compile the design using VCS using the following command.

   vcs –sim_res=1fs presynth.testbench

   Note: The timing resolution of simulation must be set to 1fs for correct functional simulation.
4. Once the design is compiled, start simulation using the following command.

   ./simv
5. For back-annotated simulation, the VCS command must be as shown in the following codeblock.

   vcs postlayout.testbench –sim_res=1fs –sdf max:<testbench_module_name>.<DUT instance name>:<sdf file path> –gui –l postlayout.log

6.4 Limitations/Exceptions (Ask a Question)

Following are the limitations/exceptions of Synopsys VCS setup.

- VCS simulations can be run only for Verilog projects of Libero SoC. The VCS simulator has strict VHDL language requirements that are not met by the Libero SoC auto-generated VHDL files.
- You must have a $finish statement in the Verilog testbench to stop the simulation whenever you want to.

Important: When simulations are run in GUI mode, run time can be specified in the GUI.

6.5 Sample Tcl and Shell Script Files (Ask a Question)

The following Perl automates the generation of the synopsys_sim.setup file as well as the corresponding shell script files needed to elaborate, compile, and simulate the design.

If the design uses an MSS, copy the test.vec file located in the simulation folder of the Libero SoC project into the VCS simulation folder. The following sections contain sample run.do files generated by Libero SoC, including the corresponding library mapping and shell script files needed for VCS simulation.

6.5.1 Pre-synthesis (Ask a Question)

Presynth_run.do

```
quietly set ACTELLIBNAME SmartFusion2
quietly set PROJECT_DIR “/sqa/users/me/VCS_Tests/Test_DFF”
if {[file exists presynth/_info]} {
echo “INFO: Simulation library presynth already exists”
} else {
vlib presynth
}
vmap presynth presynth
vmap SmartFusion2 “/captures/lin/11_0_0_23_11prod/lib/ModelSim/precompiled/vlog/smartfusion2”
vlog -work presynth “${PROJECT_DIR}/component/work/SD1/SD1.v”
vlog “+incdir+${PROJECT_DIR}/stimulus” -work presynth “${PROJECT_DIR}/stimulus/SD1_TB1.v”
vsim -L SmartFusion2 -L presynth -t 1fs presynth.SD1_TB1
add wave /SD1_TB1/*
add log -r /*
run 1000ns
```

presynth_main.csh

```
#!/bin/csh -f
set PROJECT_DIR = “/sqa/users/Me/VCS_Tests/Test_DFF”
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k -work presynth “${PROJECT_DIR}/component/
work/SD1/SD1.v”
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k “+incdir+${PROJECT_DIR}/stimulus” -work
presynth “${PROJECT_DIR}/stimulus/SD1_TB1.v”
/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs presynth.SD1_TB1 -l compile.log
./simv -l run.log
```

Synopsys_sim.setup

```
WORK > DEFAULT
SmartFusion2 : /VCS/SmartFusion2
presynth : ./presynth
DEFAULT : ./work
```

6.5.2 Post-synthesis (Ask a Question)

postsynth_run.do

```
quietly set ACTELLIBNAME SmartFusion2
quietly set PROJECT_DIR “/sqa/users/Me/VCS_Tests/Test_DFF”
if {[file exists postsynth/_info]} {
echo “INFO: Simulation library postsynth already exists”
} else {
vlib postsynth
}
```

```
vmap postsynth postsynth
vmap SmartFusion2 "//idm/captures/pc/11_0_1_12_g4x/Designer/lib/ModelSim/precompiled/vlog/
SmartFusion2"
vlog -work postsynth "${PROJECT_DIR}/synthesis/SD1.v"
vlog "+incdir+${PROJECT_DIR}/stimulus" -work postsynth "${PROJECT_DIR}/stimulus/SD1_TB1.v"
vsim -L SmartFusion2 -L postsynth -t 1fs postsynth.SD1_TB1
add wave /SD1_TB1/*
add log -r /*
run 1000ns
log SD1_TB1/*
exit
```

Postsynth_main.csh

```
#!/bin/csh -f
set PROJECT_DIR = "/sqa/users/Me/VCS_Tests/Test_DFF"
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k -work postsynth "${PROJECT_DIR}/synthesis/
SD1.v"
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k "+incdir+${PROJECT_DIR}/stimulus" -work
postsynth "${PROJECT_DIR}/stimulus/SD1_TB1.v"
/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs postsynth.SD1_TB1 -l compile.log
./simv -l run.log
```

Synopsys_sim.setup

```
WORK > DEFAULT
SmartFusion2 : /VCS/SmartFusion2
postsynth : ./postsynth
DEFAULT : ./work
```

6.5.3 Post-layout (Ask a Question)

postlayout_run.do

```
quietly set ACTELLIBNAME SmartFusion2
quietly set PROJECT_DIR "E:/ModelSim_Work/Test_DFF"
if {[file exists ../designer/SD1/simulation/postlayout/_info]} {
echo "INFO: Simulation library ../designer/SD1/simulation/postlayout already exists"
} else {
vlib ../designer/SD1/simulation/postlayout
}
vmap postlayout ../designer/SD1/simulation/postlayout
vmap SmartFusion2 "//idm/captures/pc/11_0_1_12_g4x/Designer/lib/ModelSim/precompiled/vlog/
SmartFusion2"
vlog -work postlayout "${PROJECT_DIR}/designer/SD1/SD1_ba.v"
vlog "+incdir+${PROJECT_DIR}/stimulus" -work postlayout "${PROJECT_DIR}/stimulus/SD1_TB1.v"
vsim -L SmartFusion2 -L postlayout -t 1fs -sdfmax /SD1_0=${PROJECT_DIR}/designer/SD1/
SD1_ba.sdf postlayout.SD1_TB1
add wave /SD1_TB1/*
add log -r /*
run 1000ns
```

Postlayout_main.csh

```
#!/bin/csh -f
set PROJECT_DIR = "/VCS_Tests/Test_DFF"
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k -work postlayout "${PROJECT_DIR}/
designer/SD1/SD1_ba.v"
/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k "+incdir+${PROJECT_DIR}/stimulus" -work
postlayout "${PROJECT_DIR}/stimulus/SD1_TB1.v"
/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs postlayout.SD1_TB1 -sdf

max:SD1_TB1.SD1_0:${PROJECT_DIR}/designer/SD1/SD1_ba.sdf -l compile.log
./simv -l run.log
```

Synopsys_sim.setup

```
WORK > DEFAULT
SmartFusion2 : /VCS/SmartFusion2
postlayout : ./postlayout
```

DEFAULT : ./workVCS

6.6 Automation (Ask a Question)

The flow can be automated using the following Perl script file to convert the ModelSim run.do files into VCS compatible shell script files, create proper directories inside the Libero SoC simulation directory, and then run simulations.

Run the script file using the following syntax.

perl vcs_parse.pl presynth_run.do postsynth_run.do postlayout_run.do

Vcs_parse_pl

```
#!/usr/bin/perl -w
################################################################################
#
#Usage: perl vcs_parse.pl presynth_run.do postsynth_run.do postlayout_run.do
#
################################################################################
my ($presynth, $postsynth, $postlayout) = @ARGV;
if(system("mkdir VCS_Presynth")) {print "mkdir failed:\n";}
if(system("mkdir VCS_Postsynth")) {print "mkdir failed:\n";}
if(system("mkdir VCS_Postlayout")) {print "mkdir failed:\n";}
chdir(VCS_Presynth);
`cp ../$ARGV[0] .` ;
&parse_do($presynth,"presynth");
chdir ("../");
chdir(VCS_Postsynth);
`cp ../$ARGV[1] .` ;
&parse_do($postsynth,"postsynth");
chdir ("../");
chdir(VCS_Postlayout);
`cp ../$ARGV[2] .` ;
&parse_do($postlayout,"postlayout");
chdir ("../");
sub parse_do {
my $vlog = "/cad_design/tools/vcs.dir/E-2011.03/bin/vlogan +v2k" ;
my %LIB = ();
my $file = $_[0] ;
my $state = $_[1];
open(INFILE,"$file") || die "Cant open File Reason might be:$!";
if ( $state eq "presynth" )
{
open(OUT1,">presynth_main.csh") || die "Cant create Command File Reason might be:$!";
}
elsif ( $state eq "postsynth" )
{
open(OUT1,">postsynth_main.csh") || die "Cant create Command File Reason might be:$!";
}
elsif ( $state eq "postlayout" )
{
open(OUT1,">postlayout_main.csh") || die "Cant create Command File Reason might be:$!";
}
else
{
print "Simulation State is missing \n" ;
}
open(OUT2,">synopsys_sim.setup") || die "Cant create Command File Reason might be:$!";
# .csh file
print OUT1 "#!/bin/csh -f\n\n\n" ;
#SET UP FILE
print OUT2 "WORK > DEFAULT\n" ;
print OUT2 "SmartFusion2 : /sqa/users/Aditya/VCS/SmartFusion2\n" ;
while ($line = <INFILE>)
```

```
{

Synopsys VCS Setup


if ($line =~ m/quietly set PROJECT_DIR\s+\"(.*?)\"/)
{
print OUT1 “set PROJECT_DIR = \”$1\”\n\n\n” ;
}
elsif ( $line =~ m/vlog.*\.v\”/ )
{
if ($line =~ m/\s+(\w*?)\_LIB/)
{
#print “\$1 =$1 \n” ;
$temp = “$1″._LIB”;
#print “Temp = $temp \n” ;
$LIB{$temp}++;
}
chomp($line);
$line =~ s/^vlog/$vlog/ ;
$line =~ s/ //g;
print OUT1 “$line\n”;
}
elsif ( ($line =~ m/vsim.*presynth\.(.*)/) || ($line =~ m/vsim.*postsynth\.(.*)/) || ($line
=~ m/vsim.*postlayout\.(.*)/) )
{
$tb = $1 ;
$tb =~ s/ //g;
chomp($tb);
#print “TB Name : $tb \n”;
if ( $line =~ m/sdf(.*)\.sdf/)
{
chomp($line);
$line = $1 ;
#print “LINE : $line \n” ;
if ($line =~ m/max/)
{
$line =~ s/max \/// ;
$line =~ s/=/:/;
print OUT1 “\n\n/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs postlayout.$tb -sdf
max:$tb.$line.sdf -l compile.log\n” ;
}
elsif ($line =~ m/min/)
{
$line =~ s/min \/// ;
$line =~ s/=/:/;
print OUT1 “\n\n/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs postlayout.$tb -sdf
min:$tb.$line.sdf -l compile.log\n” ;
}
elsif ($line =~ m/typ/)
{
$line =~ s/typ \/// ;
$line =~ s/=/:/;
print OUT1 “\n\n/cad_design/tools/vcs.dir/E-2011.03/bin/vcs -sim_res=1fs postlayout.$tb -sdf
typ:$tb.$line.sdf -l compile.log\n” ;
}
#-sdfmax /M3_FIC32_0=${PROJECT_DIR}/designer/M3_FIC32/M3_FIC32_ba.sdf — ModelSim SDF format
#$sdf = “-sdf max:testbench.M3_FIC32_0:${PROJECT_DIR}/designer/M3_FIC32/M3_FIC32_ba.sdf”; -VCS
SDF format
```

```
}
}
}
print
OUT1 “\n\n”
;
if
( $state eq “presynth”
)
{
print
OUT2 “presynth
: ./presynth\n”
;
print
OUT1 “/cad_design/tools/vcs.dir/E-2011.03/bin/vcs
-sim_res=1fs presynth.$tb -l
compile.log\n”
;
}
elsif
( $state eq “postsynth”
)
{
print
OUT2 “postsynth
: ./postsynth\n”
;
print
OUT1 “/cad_design/tools/vcs.dir/E-2011.03/bin/vcs
-sim_res=1fs postsynth.$tb -l
compile.log\n”
;
}
elsif
( $state eq “postlayout”
)
{
print OUT2 “postlayout : ./postlayout\n” ;
}
else
{
print “Simulation State is missing \n” ;
}
foreach $i ( keys %LIB)
{
#print “Key : $i Value : $LIB{$i} \n” ;
print OUT2 “$i : ./$i\n” ;
}
print OUT1 “\n\n” ;
print OUT1 “./simv -l run.log\n” ;
print OUT2 “DEFAULT : ./work\n” ;
close INFILE;
close OUT1;
close OUT2;
}
```

Revision History (Microchip Login

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision	Date	Description
A	12/2023	The following changes are made in this revision: • Document converted to Microchip template. Initial Revision. • Updated section 5. Siemens QuestaSim Setup/ModelSim Setup to include a new note that explains the impact on visibility during si mulation and optimization.

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices.

Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at **www.microchip.com/support**. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call 800.262.1060
- From the rest of the world, call 650.318.4460
- Fax, from anywhere in the world, 650.318.8044

Microchip Information

The Microchip Website

Microchip provides online support via our website at **www.microchip.com/**. This website is used to make files and information easily available to customers. Some of the content available includes:

- Product Support – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- General Technical Support – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- Business of Microchip – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to **www.microchip.com/**pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.
Technical support is available through the website at: **www.microchip.com/support**

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”.

  Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at **www.microchip.com/en-us/support/design-help/client-support-services**.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet- Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, TrueTime, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, GridTime, IdealBridge, In-Circuit Serial Programming, ICSP, INICnet, Intelligent

Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, KoD, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net,
PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAMICE, Serial Quad I/O, simpleMAP, SimpliPHY, SmartBuffer, SmartHLS, SMART-I.S., storClad, SQI, SuperSwitcher,
SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, Trusted Time, TSHARC, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, and ZENA are trademarks of Microchip Technology Incorporated
in the U.S.A. and other countries.
SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.
The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.
GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.
All other trademarks mentioned herein are property of their respective companies.
© 2023, Microchip Technology Incorporated and its subsidiaries. All Rights Reserved.
ISBN: 978-1-6683-3694-6
Quality Management System
For information regarding Microchip's Quality Management Systems, please visit **www.microchip.com/quality.**

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC

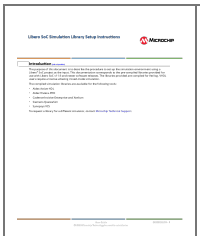
Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
www.microchip.com/support
Web Address:
www.microchip.com
Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455
Austin, TX
Tel: 512-257-3370
Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088
Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075
Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924
Detroit
Novi, MI
Tel: 248-848-4000
Houston, TX
Tel: 281-894-5983
Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380
Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800
Raleigh, NC
Tel: 919-844-7510
New York, NY
Tel: 631-435-6000
San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270
Canada – Toronto
Tel: 905-695-1980
Fax: 905-695-2078

Australia – Sydney
Tel: 61-2-9868-6733
China – Beijing
Tel: 86-10-8569-7000
China – Chengdu
Tel: 86-28-8665-5511
China – Chongqing
Tel: 86-23-8980-9588
China – Dongguan
Tel: 86-769-8702-9880
China – Guangzhou
Tel: 86-20-8755-8029
China – Hangzhou
Tel: 86-571-8792-8115
China – Hong Kong SAR
Tel: 852-2943-5100
China – Nanjing
Tel: 86-25-8473-2460
China – Qingdao
Tel: 86-532-8502-7355
China – Shanghai
Tel: 86-21-3326-8000
China – Shenyang
Tel: 86-24-2334-2829
China – Shenzhen
Tel: 86-755-8864-2200
China – Suzhou
Tel: 86-186-6233-1526
China – Wuhan
Tel: 86-27-5980-5300
China – Xian
Tel: 86-29-8833-7252
China – Xiamen
Tel: 86-592-2388138
China – Zhuhai
Tel: 86-756-3210040

India – Bangalore
Tel: 91-80-3090-4444
India – New Delhi
Tel: 91-11-4160-8631
India – Pune
Tel: 91-20-4121-0141
Japan – Osaka
Tel: 81-6-6152-7160
Japan – Tokyo
Tel: 81-3-6880- 3770
Korea – Daegu
Tel: 82-53-744-4301
Korea – Seoul
Tel: 82-2-554-7200
Malaysia – Kuala Lumpur
Tel: 60-3-7651-7906
Malaysia – Penang
Tel: 60-4-227-8870
Philippines – Manila
Tel: 63-2-634-9065
Singapore
Tel: 65-6334-8870
Taiwan – Hsin Chu
Tel: 886-3-577-8366
Taiwan – Kaohsiung
Tel: 886-7-213-7830
Taiwan – Taipei
Tel: 886-2-2508-8600
Thailand – Bangkok
Tel: 66-2-694-1351
Vietnam – Ho Chi Minh
Tel: 84-28-5448-2100

© 2023 Microchip Technology Inc. and its subsidiaries

Documents / Resources

	MICROCHIP Libero SoC Simulation Library Software [pdf] User Guide DS50003627A, Libero SoC Simulation Library Software, SoC Simulation Library Software, Simulation Library Software, Library Software, Software

References

- { 42 ,18 , , AV }
- Empowering Innovation | Microchip Technology
- Empowering Innovation | Microchip Technology
- Design Help and Other Services | Microchip Technology
- Product Change Notification | Microchip Technology
- Quality | Microchip Technology
- Microchip Lightning Support
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-0501615F-4412-4A88-ACA1-48EE266CB53A&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-090B43B9-E81C-4B12-AC63-3B95B76CCCF2&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-155CD92F-80DC-4FB7-A3EB-8F682DB8B254&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-3092C7B9-A727-4C98-A72C-0DA5ABB22C92&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-3489D02D-6AE3-4E99-91E0-BF098583908C&cover_title=Libero%20SoC%20Simulation%20Library%20Setu

- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-3A7ECBC3-1FCD-47E0-B08D-10C0BB181F5B&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-3D0F1C7E-FEAC-46D7-9A62-76B3F7F408EC&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-5D1FE97C-006A-429B-96CB-2AD7981CF38D&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-61988FFD-FAC9-4AA6-A759-39336066E8ED&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-66638D62-FF4B-4CC7-9975-36E8894FDD6F&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-74AE6250-DEA9-403B-865A-FA105EBD7F4E&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-7939C078-30E3-4B50-84D0-6DF58FEE26BC&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-79AC6E23-1842-4503-B38D-1CFA02113BC8&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-879333D7-EB03-4271-8F08-186B1B03C132&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-88ABB8F9-786F-4AAD-B145-FA238EFC90F9&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-8AF4FDD9-68EF-4D98-BCA3-

FBB92DB7B88A&cover_title=Libero%20SoC%20Simulation%20Library%20Setu

- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-9AB402D3-2A47-48F4-BE1F-4A4487AB3780&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-9BECA924-8A51-4D0B-9B39-FBEF8E968DF9&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-9D131223-9D46-4234-9836-3FCBAD108173&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-A2098E00-EC57-4F5E-BD93-7DA64C9C931B&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-BA9BF755-3CF9-4275-A23F-7C23E73B00C3&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-BC7CEF06-C4D7-4E4B-958A-39E8EE8EAC42&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-C10B2934-67ED-4011-B889-1A6164FBF7B4&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-C2AD862F-160B-4638-AD29-B5E6BBCA9518&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-C2C9749F-490D-4715-8244-29DDA08058B1&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-C8B3C4D4-6AB2-4BA7-A077-3892C4467DEB&cover_title=Libero%20SoC%20Simulation%20Library%20Setu
- microchipsupport.force.com/s/newcase?pub_guid=GUID-C5F2A2E4-8E46-4933-8016-E9D322CA479C&pub_lang=en-US&pub_ver=1&pub_type=User%20Guide&bu=fpga&tpc_guid=GUID-CADCA481-3AF7-4676-80C2-

33BB9EEBE0E7&cover_title=Libero%20SoC%20Simulation%20Library%20Setu

- User Manual

Manuals+, Privacy Policy