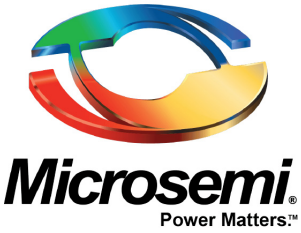
Enhanced Constraint Flow Batch Command Scripting for SmartFusion2/IGLOO2/RTG4

UG0579 User Guide

Microsemi
Power Matters.™



Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense and security, aerospace, and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs, and ASICs; power management products; timing and synchronization devices and precise time solutions; voice processing devices; RF solutions; discrete components; enterprise storage and communications solutions, security technologies, and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees worldwide. Learn more at **www.microsemi.com**.

50200579-3/11.16

Contents

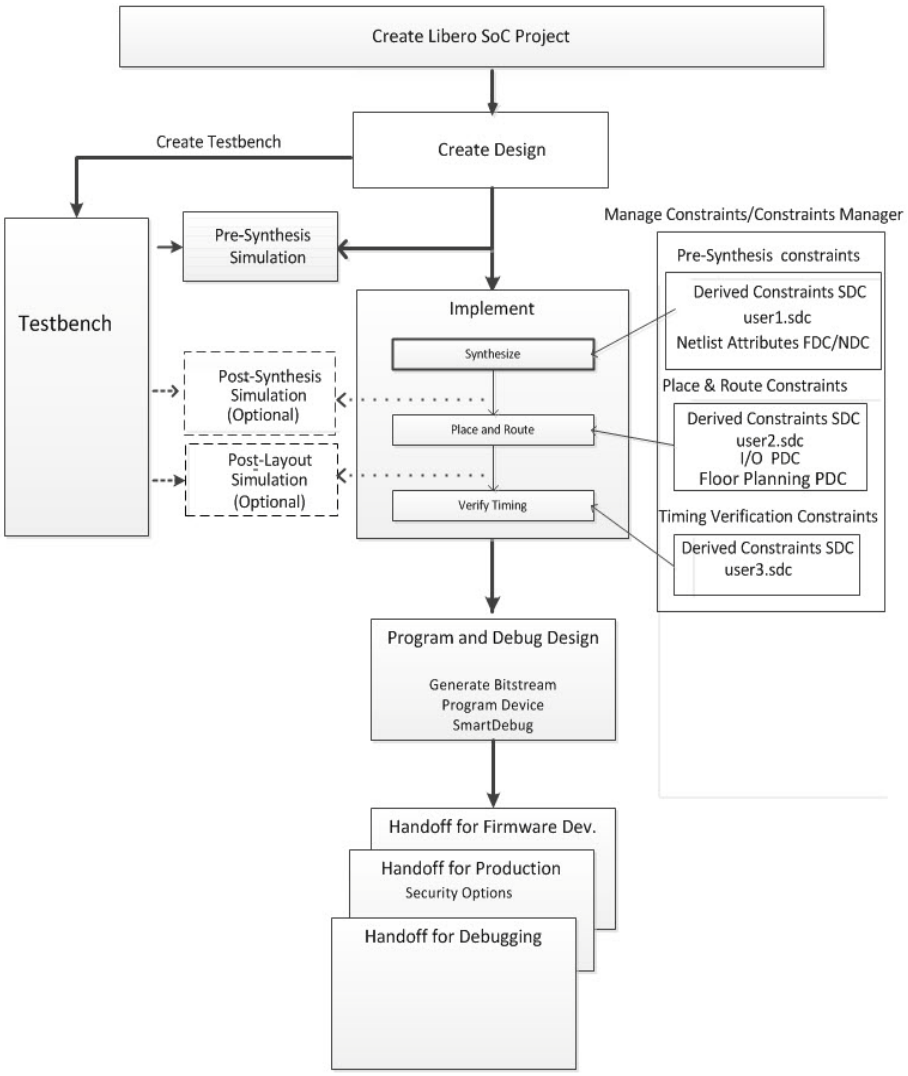
Tables

1 Libero SoC Batch Commands for Enhanced Constraint Flow

1.1 Introduction

Libero SoC has a rich set of Tcl commands to drive the complete design flow from project creation, import of design source files and design constraint files, management of design constraints, through Synthesis, Place-and-Route, Timing and Power Reports generation to programming file generation.

Figure 1 • **Libero SoC Enhanced Design Constraint Flow**



This User Guide illustrates the use of Tcl as batch commands to drive the various steps in the design flow. Tcl commands available exclusively in the Enhanced Constraint Flow (available since Libero SoC v11.7 release for SmartFusion2, IGLOO2, and RTG4 devices) are included. For a general reference of the Libero SoC Tcl commands, refer to the Tcl Command Reference User Guide for details. This User

Guide is also available from Libero SoC software (Help > Reference Manuals > Tcl Command Reference).

2 Create a New Libero SoC Project

2.1 Description

Start a new Libero SoC project for the design. This script creates the new project directory and creates sub-folders to store files imported into the project and the files generated by Libero SoC such as report files, log files and message files.

2.2 Tcl Snippet

```
new_project -location {D:/my_project} \
   -name {my_project}\
   -block_mode 0 \
   -standalone_peripheral_initialization 0 \
   -use_enhanced_constraint_flow 1 \# To turn on Enhanced Constraint Flow
   -hdl {VERILOG} \
   -verilog_mode {VERILOG_2K}\
   -family {SmartFusion2} \
   -die {M2S150TS} \
   -package {1152 FC} \
   -speed {-1} \
   -die_voltage {1.2}\
   -part_range {COM} \
   -adv_options {DSW_VCCA_VOLTAGE_RAMP_RATE:100_MS}\
   -adv_options {IO_DEFT_STD:LVCMOS 2.5V}\
   -adv_options {PLL_SUPPLY:PLL_SUPPLY_25}\
   -adv_options {RESTRICTPROBEPINS:1}\
   -adv_options {RESTRICTSPIPINS:0}\
   -adv_options {SYSTEM_CONTROLLER_SUSPEND_MODE:0}\
   -adv_options {TEMPR:COM}\
   -adv_options {VCCI_1.2_VOLTR:COM}\
   -adv_options {VCCI_1.5_VOLTR:COM}\
   -adv_options {VCCI_1.8_VOLTR:COM}\
   -adv_options {VCCI_2.5_VOLTR:COM}\
   -adv_options {VCCI_3.3_VOLTR:COM}\
   -adv_options {VOLTR:COM}
```

2.3 Relevant Tcl Commands

new_project

The new_project command creates a project directory at the specified disk location. Use this command to specify the device and package the design is targeted to, as well as the HDL type. Other options include voltages, speed and temperature grades. Set the use_enhanced_constraint_flow parameter to “1” or “TRUE” to turn on the Enhanced Constraint Flow for better management of design constraints for the project. For complete information on the Enhanced Constraint Flow, see the Libero User Guide (Enhanced Constraint Flow) or the Online Help (Libero > Help).

3 Import Files

3.1 Description

Import different types of files into the Libero SoC project. A local copy of the imported file is made and stored in the project location and maintained by Libero SoC.

3.2 Snippet

```
import_files -hdl_source {C:/design/source/ddr3.v}\
   -hdl_source {C:/design/source/design_top.v}\

create_links -hdl_source {C:/design/sources/sub_block.v}

set_root -module {design_top::work}
```

3.3 Relevant Tcl Commands

3.3.1 import_files

The import_files command is used to copy a design file from a disk location outside the Libero Project into the Libero SoC project. Different types of files are supported, including HDL design sources files, HDL stimulus/testbench files, design constraints files (*.sdc, *.pdc, *.ndc), and design blocks (*.cxz). The copied version will be managed locally by Libero SoC and will not be associated with the original file after importation. To keep the association with the original file, do not use the import command. Use the create_links command instead.

For complete information on Libero SoC Tcl Commands, refer to the Libero Soc Tcl Command Reference User Guide. It is also available from the Libero Online Help (Libero > Help > Reference Manuals > Tcl Command Reference).

3.3.2 create_links

The create_links command is used much like the import_files command, but instead of making a local copy, Libero SoC creates a link to the specified file. When the latest version of a file (located and maintained outside of the Libero Soc project) needs to be used for the Libero SoC project, use the create_links command instead of the import_files command. The create_links command supports the same file types as the import_files command.

4 Synthesize the Design

4.1 Description

Configure the Synthesize options and run the Synthesize step to generate an EDIF netlist.

4.2 Tcl Snippet

```
#Associate the user-defined SDC constraint file to synthesis

organize_tool_files -tool {SYNTHESIZE}\
-file {D:/my_project/constraint/design_top_user.sdc} \
-module {design_top::work} \
-input_type {constraint}

#Associate the <top_level>_derived_constraints.sdc file to synthesis, required if the\
#design contains IP Cores such as CoreConfigP, CoreResetP, CCC or OSC.

organize_tool_files -tool {SYNTHESIZE}\
-file {D:/my_project/constraint/design_top_derived_constraints.sdc} \
-module {design_top::work} \
-input_type {constraint}

#Associate the *.fdc file

organize_tool_files -tool {SYNTHESIZE}\
-file {D:/my_project/constraint/design_top_user.fdc} \
-module {design_top::work} \
-input_type {constraint}

#Associate the *.ndc file

organize_tool_files -tool {SYNTHESIZE}\
-file {D:/my_project/constraint/design_top_user.ndc} \
-module {design_top::work} \
-input_type {constraint}

#Configure the Synthesis options

configure_tool -name {SYNTHESIZE} -params {CLOCK_ASYNC:12}\
   -params {CLOCK_DATA:5000}\
   -params {CLOCK_GLOBAL:2}\
   -params {PA4_GB_MAX_RCLKINT_INSERTION:16}\
   -params {PA4_GB_MIN_GB_FANOUT_TO_USE_RCLKINT:300}\
   -params {RAM_OPTIMIZED_FOR_POWER:false}\
   -params {RETIMING:false}\
   -params {SYNPLIFY_OPTIONS:\
       set_option -run_prop_extract 1;\
       set_option -maxfan 10000;\
       set_option -clock_globalthreshold 2;\
       set_option -async_globalthreshold 12;\
       set_option -globalthreshold 5000;\
       set_option -low_power_ram_decomp 0;}\
   -params {SYNPLIFY_TCL_FILE:C:/Users/user1/Desktop/tclflow/synthesis/test.tcl}

#Runs the Synthesis step with the configured options. This command takes no parameters.

run_tool -name {SYNTHESIZE}

#Generate the *derived_constraint.sdc after Synthesis to constrain IP cores in the

#design. Can only be executed after synthesis.

derive_constraints_sdc
```

4.3 Relevant Tcl Documentation

4.3.1 organize_tool_files

This command associates a constraint file to a specific tool. Four types of files can be associated to synthesis:

- *.fdc - netlist attribute constraint file, non-timing-related constraints specifically for SynplifyPro, to direct the synthesis tool on how to handle hierarchy, what to infer for memory, etc. For details, see the Synopsys FPGA Synthesis Synplify Pro ME User Guide.
- *.ndc - netlist design constraint file, non-timing synthesis constraints such as whether or not I/Os are to be combined with registers.
- *.sdc - exclusively timing constraints that the user generates and passes to Synthesis.
- <top_level>_derived_constraints.sdc - a special SDC constraint file Libero SoC generates (if IP Cores are present in the design) to improve the timing performance of the design. This file is available for generation only in the Libero SoC Enhanced Constraint Flow. Passing this SDC file to the synthesis tool ensures that the design object names referenced in the SDC file are preserved in the post-synthesis netlist.

4.3.2 configure_tool

The configure_tool -name {SYNTHESIZE} command takes a parameter:value pair to configure the options.

```
configure_tool -name {SYNTHESIZE} -params {parm:value} [-params {parm:value]
```

Table 1 contains the frequently used parameters. For a complete list of the Synthesize parameters, refer to the Libero SoC Online Help (Libero > Help).

Table 1 • **Synthesis Options**

Parameter Name	Value	Description
CLOCK_ASYNC	Integer	Specifies the threshold value for asynchronous pin promotion to a global net. The default is 12.
CLOCK_GLOBAL	Integer	Specifies the threshold value for Clock pin promotion. The default is 2.
CLOCK_DATA	Integer value between 1000 and 200,000.	Specifies the threshold value for data pin promotion. The default is 5000.
RAM_OPTIMIZED_FOR_POWER	Boolean {true | false | 1 | 0}	Set to true or 1 to optimize RAM for Low Power; RAMS are inferred and configured to ensure the lowest power consumption. Set to false or 0 to optimize RAM for High Speed at the expense of more FPGA resources.
RETIMING	Boolean {true | false | 1 | 0}	Set to true or 1 to enable Retiming during synthesis. Set to false or 0 to disable Retiming during synthesis.
PA4_GB_COUNT	Integer	The number of available global nets is reported. Minimum for all dies is “0”. Default and Maximum values are die-dependent: 005/010 die: Default = Max = 8 025/050/060/090/150 die: Default=Max=16 RT4G075/RT4G150: Default=24, Max=48. Note: For RTG4, default is 48.
PA4_GB_MAX_RCLKINT_INSERTION	Integer	Specifies the maximum number of global nets that could be demoted to row-globals. Default is 16, Min is 0 and Max is 50.

Table 1 • **Synthesis Options**

Parameter Name	Value	Description
PA4_GB_MIN_GB_FANOUT_TO_USE_RCLKINT	Integer	Specifies the Minimum fanout of global nets that could be demoted to row-globals. Default is 300. Min is 25 and Max is 5000.
SYNPLIFY_OPTIONS	String	Specifies additional synthesis-specific options. Options specified by this parameter override the same options specified in the user Tcl file if there is a conflict.
SYNPLIFY_TCL_FILE	String	Specifes the absolute or relative path name to the user Tcl file containing synthesis-specific options.

4.3.3 run_tool

The run_tool -name {SYNTHESIZE} commands runs the Synthesis command with the options set in the configure_tool {SYNTHESIZE} command. It takes no parameters.

4.3.4 derived_constraints_sdc

Execute this command only if there are IP cores such as CoreConfigP, CoreResetP, CCC, or OSC in the design.

This command generates two files:

- <top_level>_derived_constraints.sdc to be passed to Synthesis, Place and Route, and Timing Verification.
- <top_level>_derived_constraints.pdc to be passed to Place and Route only.

5 Pre-Synthesis Simulation

5.1 Description

Set up and run pre-synthesis RTL simulation. This script assumes ModelSim ME as the default simulator. For the complete ModelSim operation information, refer to the ModelSim User's Manual.

5.2 Tcl Snippet

```
#Set the top level of the design for simulation

set_root -module "design_top::work"

#Import the testbench file into the project for simulation

import_files -stimulus {D:/2Work/tesbench.v}

#Associate the testbench with the top level of the design

organize_tool_files -tool {SIM_PRESYNTH} \
   -file {D:/2Work/tesbench.v}\
   -module {design_top::work}\
   -input_type {stimulus}

#Configure the Simulation Options

set_modelsim_options -use_automatic_do_file {1}\
   -sim_runtime {700 us}\
   -tb_module_name {design_toptb}\
   -tb_top_level_name {design_toptb_0}\
   -include_do_file 0\
   -type {typ}\
   -resolution {1fs}\
   -add_vsim_options {-novopt}\
   -display_dut_wave 1\
   -log_all_signals 1\
   -do_file_args {}\
   -dump_vcd 1\
   -vcd_file "my.vcd" #dump out vcd file for smartpower

#Run the Pre-synthesis Simulation

run_tool -name {SIM_PRESYNTH}
```

5.3 Relevant Tcl Documentation

5.3.1 import_files

Use the import_files command to import the stimulus file for simulation. The imported stimulus file is copied to the Libero SoC project under the stimulus folder.

```
import_files -stimulus {D:/2Work/tesbench.v}
```

5.3.2 organize_tool_files

This command associates the stimulus files to the simulation tool for the specific simulation run: pre-synthesis simulation, post-synthesis simulation, or post-layout simulation.

```
organize_tool_files -tool {SIM_PRESYNTH} \
   -file {D:/2Work/tesbench.v}\
   -module {mydesign::work}\
   -input_type {stimulus}
```

5.3.3 set_modelsim_options

The set_modelsim_options command sets the runtime options for ModelSim. Common parameters are shown below. For a complete list, see the Libero SoC Tcl Commands Reference Guide.

Table 2 • **ModelSim Options**

Common Parameters	Note	Example Value
-use_automatic_do_file	When set (1), enable the use of the do file created by Libero.	1 or 0
-sim_runtime	Sets the length of the simulation runtime.	700 ns
-tb_module_name	Identifies the testbench driving the design.	testbench
-tb_top_level_name	Identifies the top-level instance name in the testbench.	DUT
-include_do_file	When set (1), will use the do file name specified with the -included_do_file parameter.	0 or 1
-type	Set what timing values are to be used in the simulation.	typ, min, max
-resolution	Sets the time scale for the simulation.	1fs, 1ns, etc.
-add_vsim_options	Calls ModelSim with the options listed.	-display_dut_wave <value>
-display_dut_wave	When set (1), will cause ModelSim to display all signals in the design. 0 will display only the testbench.	1 or 0

5.4 run_tool

The run_tool command runs the simulation with the configured options. Use “SIM_PRESYNTH” for pre-synthesis simulation.

```
run_tool -name {SIM_PRESYNTH}
```

This command takes no other parameters.

6 Manage Constraints

6.1 Description

In the FPGA design world, design constraints files are as important as design source files. Libero SoC provides a rich set of Tcl commands for the management of four different types of design constraints files in the project:

- Timing SDC Constraint files (*.sdc)
- I/O Physical Design Constraint files (*.pdc)
- Floorplanning Physical Design Constraint files (*.pdc)
- Netlist Attributes Constraints files (*.fdc or *.ndc)

To take advantage of the Tcl commands to manage the constraint files, turn on the Enhanced Constraint flow option (set to “1”) when first creating the project as follows:

```
new_project -location {D:/my_project}\
   -name {my_project}\
   -use_enhanced_constraint_flow 1 #turn on Enhanced Constraint Flow
```

Note: The Enhanced Constraint Flow option is available only for SmartFusion2, IGLOO2, and RTG4 devices.

When the Enhanced Constraint Flow option is turned on, a set of Tcl commands are available for you to:

- import constraint files
- create links to constraint files
- generate Derived Constraints
- associate the constraint files to specific design tools
- check constraints
- generate Constraint Coverage Reports

6.2 Tcl snippet

```
#Import Constraint Files

import_files \
   -sdc {C:/design/sources/design_top.sdc} #import SDC Timing Constraint file

import_files \
   -io_pdc {C:/design/constraints/design_top_io.pdc} #import I/O PDC file

import_files \
   -fp_pdc {C:/design/constraints/design_top_placement.pdc} #import floorplanning PDC

#Create links to constraint files located and maintained outside of Libero SoC project

create_links -sdc {C:/common/constraints/common.sdc}#link constraint file

#Generate the SDC and PDC constraints for IP Cores, available only after synthesis

derive_constraints_sdc

#Associate the SDC timing constraint file derived_constraints.sdc to different tools

organize_tool_files -tool {SYNTHESIS} -file
   {D:/my_project/constraint/M3_MDDR_top_derived_constraints.sdc}

organize_tool_files -tool {PLACEROUTE} -file
   {D:/my_project/constraint/M3_MDDR_top_derived_constraints.sdc}

organize_tool_files -tool {VERIFYTIMING} -file
   {D:/my_project/constraint/M3_MDDR_top_derived_constraints.sdc}

# Associate the physical design constraint *derived_constraints.pdc to Place and Route.
```

```
organize_tool_files -tool {PLACEROUTE} -file\
   {D:/my_project/constraint/fp/mddr_top_derived_constraints.pdc}

# Associate user-defined.sdc constraints to different tools

organize_tool_files -tool {SYNTHESIZE}\
   -file {D:/my_project/constraint/design_top_user.sdc} \
   -module {design_top::work} \
   -input_type {constraint}

organize_tool_files -tool {PLACEROUTE} \
   -file {D:/my_project/constraint/design_top_user.sdc} \
   -module {design_top::work} \
   -input_type {constraint}

organize_tool_files -tool {VERIFYTIMING} \
   -file {D:/my_project/constraint/design_top_user.sdc}\
   -module {design_top::work} \
   -input_type {constraint}

#

#Checking Constraint Files

check_sdc_constraints -tool {synthesis}

check_sdc_constraints -tool (designer)

check_sdc_constraints -tool {timing}

check_pdc_constraints -tool {designer}

check_fdc_constraints -tool (synthesis)

#Generate Constraint Coverage Reports

generate_sdc_constraint_coverage -tool {PLACEROUTE}

generate_sdc_constraint_coverage -tool {VERIFYTIMING}
```

6.3 Relevant Tcl Commands

6.3.1 import_files

See "import_files" on page 10

6.3.2 create_links

See "create_links" on page 10

6.3.3 organize_tool_files

Use this command to associate the constraint file to a design tool. SDC timing constraint file can be associated with Synthesis, Place-and-Route, and Timing Verifications. I/O PDC and Floor Planning PDC files can be associated with Place-and-Route only. Netlist Attributes Constraint files (*.fdc or *.ndc) files can be associated with Synthesis only.

6.3.4 check_pdc_constraints

Use this Tcl command to check PDC constraint files associated with the Place-and-Route step. The design needs to be in the post-synthesis state for this command to execute. This command is available in the Enhanced Constraint Flow only.

6.3.5 check_sdc_constraints

Use this Tcl command to check SDC constraints associated with Synthesis, Place-and-Route, and Timing Verifications. To check the SDC constraint files associated with Place-and-Route and Timing Verifications, the design needs to be in the post-synthesis state. This command is available in the Enhanced Constraint Flow only.

This command checks the syntax of the SDC constraint file. It accepts the "/" as the hierarchy separator and pin separator. Do not use ":" as the pin separator in the SDC constraint file. It causes the command to exit with error.

This command also checks for any design object mismatches between the SDC constraint file and the post-synthesis netlist. A design object such as an instance/cell name that exists in the SDC constraint file but missing in the post-layout netlist causes the check command to exit with error.

6.3.6 check_fdc_constraints

Use this Tcl command to check the FDC constraint files associated with the Synthesis tool. This command is available in the Enhanced Constraint Flow only.

6.3.7 derive_constraints_sdc

This Tcl command generates SDC Timing Constraints for design components used in your design. Clock constraints are generated for OSC, CCC, MSS, and SERDES for SmartFusion2/IGLOO2 devices and CCC and SERDES for RTG4 devices. Set_false_path constraints are generated for certain timing paths passing through the CoreResetP core (if used in your design) and set_max_delay and set_min_delay constraints are generated for certain timing paths passing through the CoreConfigP core (if used in the design). The generated SDC constraint file is named "<top_level>_derived_constraints.sdc" and is listed in the Timing tab. By default, this SDC file is associated with Synthesis, Place-and-Route, and Timing Verification.

Example of the <top_level>_derived_constraints.sdc:

```
create_clock -name {mddr_top_sb_0/FABOSC_0/I_RCOSC_25_50MHZ/CLKOUT} -period 20\
   [ get_pins { mddr_top_sb_0/FABOSC_0/I_RCOSC_25_50MHZ/CLKOUT } ]

create_clock -name {mddr_top_sb_0/mddr_top_sb_MSS_0/CLK_CONFIG_APB} -period 40\
   [ get_pins { mddr_top_sb_0/mddr_top_sb_MSS_0/MSS_ADLIB_INST/CLK_CONFIG_APB } ]

create_generated_clock -name {mddr_top_sb_0/CCC_0/GL0} -multiply_by 4 -divide_by 2\
   -source [ get_pins { mddr_top_sb_0/CCC_0/CCC_INST/RCOSC_25_50MHZ } ]\
   -phase 0 [ get_pins { mddr_top_sb_0/CCC_0/CCC_INST/GL0 } ]

set_false_path -ignore_errors -through [ get_nets \
   {mddr_top_sb_0/CORECONFIGP_0/INIT_DONE mddr_top_sb_0/CORECONFIGP_0/SDIF_RELEASED}]

set_false_path -ignore_errors -through [ get_nets \
   { mddr_top_sb_0/CORERESETP_0/ddr_settled\
   mddr_top_sb_0/CORERESETP_0/count_ddr_enable\
   mddr_top_sb_0/CORERESETP_0/release_sdif*_core\
   mddr_top_sb_0/CORERESETP_0/count_sdif*_enable}]

set_false_path -ignore_errors -from [ get_cells \
   { mddr_top_sb_0/CORERESETP_0/MSS_HPMS_READY_int } ]\
   -to [ get_cells { mddr_top_sb_0/CORERESETP_0/sm0_areset_n_rcosc\
   mddr_top_sb_0/CORERESETP_0/sm0_areset_n_rcosc_q1 } ]

set_false_path -ignore_errors -from [ get_cells \
   { mddr_top_sb_0/CORERESETP_0/MSS_HPMS_READY_int\
   mddr_top_sb_0/CORERESETP_0/SDIF*_PERST_N_re } ] \
   -to [ get_cells { mddr_top_sb_0/CORERESETP_0/sdif*_areset_n_rcosc* } ]

set_false_path -ignore_errors -through \
   [ get_nets { mddr_top_sb_0/CORERESETP_0/CONFIG1_DONE\
   mddr_top_sb_0/CORERESETP_0/CONFIG2_DONE mddr_top_sb_0/CORERESETP_0/SDIF*_PERST_N\
   mddr_top_sb_0/CORERESETP_0/SDIF*_PSEL mddr_top_sb_0/CORERESETP_0/SDIF*_PWRITE\
   mddr_top_sb_0/CORERESETP_0/SDIF*_PRDATA[*]\
   mddr_top_sb_0/CORERESETP_0/SOFT_EXT_RESET_OUT\
   mddr_top_sb_0/CORERESETP_0/SOFT_RESET_F2M mddr_top_sb_0/CORERESETP_0/SOFT_M3_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_MDDR_DDR_AXI_S_CORE_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_FDDR_CORE_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_SDIF*_PHY_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_SDIF*_CORE_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_SDIF0_0_CORE_RESET\
   mddr_top_sb_0/CORERESETP_0/SOFT_SDIF0_1_CORE_RESET } ]
```

```
set_max_delay 0 -through [ get_nets \
   { mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PSEL\
   mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PENABLE } ] \
   -to [ get_cells { mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PREADY*\
   mddr_top_sb_0/CORECONFIGP_0/state[0] } ]

set_min_delay -24 -through [ get_nets \
   { mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PWRITE\
   mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PADDR[*]\
   mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PWDATA[*]\
   mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PSEL\
   mddr_top_sb_0/CORECONFIGP_0/FIC_2_APB_M_PENABLE } ]
```

This command also generates a PDC file for certain Microsemi IP cores such as the CoreConfigP. This PDC file constrains the placement of the IP cores in a fixed and optimal region Libero SoC creates specifically to ensure that placement of these IP cores do not cause timing violations. The PDC file is named <top_level_>derived_constraints.pdc and is listed in the Floor Planner tab of the Constraint Manager. By default, this PDC file is associated with Place-and-Route.

Example of <top_level_derived_constraints.pdc>:

```
define_region -name {auto_coreconfigp} -type inclusive 1104 159 1451 299

assign_region {auto_coreconfigp} {mddr_top_sb_0/CORECONFIGP_0}
```

This command is available only in the Enhanced Constraint Flow (Enabled Enhanced Constraint Flow option turned on).

Use the organize_tool_files command to pass the *derived_constraints.sdc to Synthesis, Place and Route, and Timing Veification.

```
organize_tool_files -tool {SYNTHESIS} -file
   {D:/2Work/alex_MDDR_SimDRAM/constraint/M3_MDDR_top_derived_constraints.sdc}

organize_tool_files -tool {PLACEROUTE} -file
   {D:/2Work/alex_MDDR_SimDRAM/constraint/M3_MDDR_top_derived_constraints.sdc}

organize_tool_files -tool {VERIFYTIMING} -file
   {D:/2Work/alex_MDDR_SimDRAM/constraint/M3_MDDR_top_derived_constraints.sdc}
```

Use the organize_tool_files command to pass the *derived_constraints.pdc to Place and Route.

```
organize_tool_files -tool {PLACEROUTE} -file\
   {D:/2Work/alex_MDDR_SimDRAM/constraint/fp/mddr_top_derived_constraints.pdc}
```

6.3.8 generate_sdc_constraint_coverage

Use this tcl command to generate a constraint coverage report for the SDC constraint file associated with Place-and-Route, or Verify Timing. To execute this Tcl command, run Synthesis first. The design must be in the post-synthesis state. Completion of the Place-and-Route step, however, is not needed to execute this command. The generated coverage reports (*.xml) are displayed in the Reports tab and also placed in the <prj_folder>/designer/<module>/*.xml file.

7 Post-Synthesis Simulation

7.1 Description

Runs the gate-level simulation.

7.2 Tcl Snippet

```
#Set the top level of the design for simulation

set_root -module "design_top::work"

#Associate the testbench with the top level of the design

organize_tool_files -tool {SIM_POSTSYNTH} \
   -file {D:/2Work/tesbnech.v}\
   -module {prep1::work}\
   -input_type {stimulus}

#Configure the Simulation Options

set_modelsim_options -use_automatic_do_file {1}\
   -sim_runtime {700 us}\
   -tb_module_name {MDDR_TB}\
   -tb_top_level_name {MDDR_TB}\
   -include_do_file 0\
   -type {typ}\
   -resolution {1fs}\
   -add_vsim_options {-novopt}\
   -display_dut_wave 1\
   -log_all_signals 1\
   -do_file_args {}\
   -dump_vcd 1\
   -vcd_file "my.vcd" #dump out vcd file for smartpower

#Run the Pre-synthesis Simulation with the configured options

run_tool -name {SIM_POSTSYNTH}
```

7.3 Relevant Tcl Documentation

7.3.1 import_files

Use the import_files command to import the stimulus file for simulation. The imported stimulus file is copied to the Libero SoC project under the stimulus folder.

```
import_files -stimulus {D:/2Work/tesbench.v}
```

7.3.2 organize_tool_files

This command associates the stimulus files to the simulation tool for the specific simulation run: pre-synthesis simulation, post-synthesis simulation, or post-layout simulation.

```
organize_tool_files -tool {SIM_POSTSYNTH} \
   -file {D:/2Work/tesbench.v}\
   -module {design_top::work}\
   -input_type {stimulus}
```

7.3.3 set_modelsim_options

See "set_modelsim_options" on page 15 for details

7.3.4 run_tool

The run_tool command runs the simulation with the configured options. Use "SIM_POSTSYNTH" for post-synthesis simulation.

```
run_tool -name {SIM_POSTSYNTH}
```

This command takes no other parameters.

8 Place and Route the Design

8.1 Description

The Place-and-Route step places the logic elements in the gate-level netlist and interconnect them according to the DRC rules of the physical device. It also produces a pin-out report which is used to interface with the parts outside the physical device.

8.2 Tcl snippet

```
#Pass the user-defined SDC and PDC constraint file to Place and Route

organize_tool_files -tool {PLACEROUTE} -file
   {D:/2Work/my_project/constraint/user.sdc}\
   -module {design_top::work} -input file {constraint}

organize_tool_files -tool {PLACEROUTE} -file \
   {D:/2Work/my_project/constraint/fp/user.pdc}\
   -module {design_top::work} -input {constraint}

#Pass the user-defined *_io.pdc constraint file to fix placement of I/Os

organize_tool_files -tool {PLACEROUTE} -file \
   {D:/2Work/my_project/constraint/io/*_io.pdc}\
   -module {design_top::work} -input {constraint}

#Pass the <design_top>_derived_constraints sdc file to Place and Route

organize_tool_files -tool {PLACEROUTE} -file \
   {D:/2Work/my_project/constraint/design_top_derived_constraints.sdc}\
   -module {design_top::work} -input {constraint}

#Pass the <design_top>_derived_constraints pdc file to Place and Route.

organize_tool_files -tool {PLACEROUTE} -file \
   {D:/2Work/my_project/constraint/fp/design_top_derived_constraints.pdc}\
   -module {design_top::work} -input {constraint}

#Configure the options for Place and Route

configure_tool    -name {PLACEROUTE}      \

    -params {TDPR:1}        \

    -params {PDPR:0}        \

    -params {EFFORT_LEVEL:0}    \

    -params {INCRPLACEANDROUTE:0}

#Runs the Place and Route step

run_tool -name {PLACEROUTE}
```

8.3 Relevant Tcl Commands

8.3.1 organize_tool_files

The command passes the SDC and PDC constraint files to Place and Route. If the design contains IP cores such as CoreConfigP, CoreResetP, CCC or OSC, constraining the Place and Route step with the <design_top>_derived_constraints.sdc improves the timing performance of the design. If the <design_top>_derived_constraints.sdc file is to be passed to Place and Route, Microsemi recommends that the same <design_top>_derived_constraints.sdc file be passed to synthesis for the synthesis step first prior to running Place and Route (see organize_tool_files). This will ensure that all the design object names referenced in the <design_top>_derived_constraints.sdcfile are preserved in the post-synthesis

netlist. If this file is passed to Place and Route but not to the synthesis step before it, Place and Route may fail because the tool cannot find in the post-synthesis netlist the design object names referenced in the SDC constraint file.

Use the organize_tool_files command to pass other user-defined SDC timing (*.sdc) constraint files, I/O Placement Constraint (*_io.pdc) files and Floorplanning Constraint (*_fp.pdc) files.

8.3.2 configure_tool

The Configure_tool command is used to configure the Place and Route options.

```
configure_tool -name {PLACEROUTE} -params {parm:value} [-params {parm:value}]
```

This command takes parameter:value pairs to configure the options. Table 3 lists the frequently used parameters.

For a complete list of the parameters for this command, refer to the online help (Libero > Help)

Table 3 • **Place and Route Options**

Parameter Name	Value	Descriptions
TDPR	Boolean {true | false | 1 | 0}	Set to true or 1 to enable Timing-Driven Place and Route. Default is 1.
PDPR	Boolean {true | false | 1 | 0}	Set to true or 1 to enable Power-Driven Place and Route. Default is false or 0.
EFFORT_LEVEL	Boolean {true | false | 1 | 0}	Set to true or 1 to enable High Effort Layout to optimize design performance. Default is false or 0.
INCRPLACEANDROUTE	Boolean {true | false | 1 | 0}	Set to true or 1 to use previous placement data as the initial placement for the next run. Default is false or 0.
REPAIR_MIN_DELAY	Boolean {true | false | 1 | 0}	Set to 1 to enable Repair Minimum Delay violations for the router when TDPR option is set to true or 1. Default is false.
MULTI_PASS_LAYOUT	Boolean {true | false | 1 | 0}	Set to true or 1 to enable Multi-Pass Layout Mode for Place and Route. Default is false or 0.

9 Post-Layout Simulation

9.1 Description

Runs the post-layout simulation on the back-annotated netlist with SDF timing information.

9.2 Tcl snippet

```
set_root -module "design_top::work"

#import the stimulus file

import_files -stimulus {D:/2Work/tesbench.v}

#Associate the testbench with the top level of the design

organize_tool_files -tool {SIM_POSTLAYOUT} \
   -file {D:/2Work/my_project/stimulus/tesbench.v}\
   -module {design_top::work}\
   -input_type {stimulus}

#Generate the *_ba.sdf and *_ba.v/*_ba.vhd files

run_tool -name {EXPORTSDF}

#Configure the Simulation Options

set_modelsim_options -use_automatic_do_file {1}\
   -sim_runtime {700 us}\
   -tb_module_name {MDDR_TB}\
   -tb_top_level_name {MDDR_TB}\
   -include_do_file 0\
   -type {typ}\
   -resolution {1fs}\
   -add_vsim_options {-novopt}\
   -display_dut_wave 1\
   -log_all_signals 1\
   -do_file_args {}\
   -dump_vcd 1\
   -vcd_file "my.vcd" #dump out vcd file for smartpower

#Run the Post-Layout Simulation with the configured options

run_tool -name {SIM_POSTLAYOUT}
```

9.3 Relevant Tcl Commands

9.3.1 import_files

Use the import_files command to import the stimulus file for simulation. The imported stimulus file is copied to the Libero SoC project under the stimulus folder.

```
import_files -stimulus {D:/2Work/tesbench.v}
```

9.3.2 organize_tool_files

This command associates the stimulus files to the simulation tool for the specific simulation run: pre-synthesis simulation, post-synthesis simulation, or post-layout simulation.

```
organize_tool_files -tool {SIM_POSTLAYOUT} \
   -file {D:/2Work/tesbench.v}\
   -module {design_top::work}\
   -input_type {stimulus}
```

9.3.3 set_modelsim_options

See "set_modelsim_options" on page 15 for details.

9.3.4 run_tool

This command executes the Post-Layout Simulation step with the configured simulation options.

```
run_tool -name {SIM_POSTLAYOUT}
```

This command takes no other parameters.

10 Verify Timing

10.1 Description

Run Timing Verification to generate Timing Reports to ensure the timing requirements of the design are met.

10.2 Tcl Snippet

```
configure_tool -name {VERIFYTIMING}\
   -params {MAX_TIMING_SLOW_LV_HT:1}\
   -params {MIN_TIMING_FAST_HV_LT:1}\
   -params {MAX_TIMING_FAST_HV_LT:1}\
   -params {MIN_TIMING_SLOW_LV_HT:1}\
   -params {CONSTRAINTS_COVERAGE:1}\
   -params {FORMAT:XML}

run_tool -name {VERIFYTIMING}
```

10.3 Relevant Tcl Commands

10.3.1 configure_tool

The configure_tool -name {VERIFYTIMING} Tcl command configures Libero SoC to generate different timing reports based on process, operating temperature and voltage. It also specifies the format of the timing reports. This command can also generate a Constraint Coverage report of the SDC timing constraints associated with Timing Verification.

This command takes parameter:value pairs to configure the options. lists the frequently used parameters. For a complete list of the parameters for this command, refer to the online help (Libero > Help.

Table 4 • **Verify Timing Parameters and Values**

Parameter Name	Value	Description
MAX_TIMING_SLOW_LV_HT	{1,0}	Max Delay Static Timing Analysis report based on Slow process, Low Voltage, and High Temperature operating conditions. 0 turns report generation OFF and 1 turns it ON. Default is 1.
MIN_TIMING_FAST_HV_LT	{1,0}	Min Delay Static Timing Analysis report based on Fast process, High Voltage, and Low Temperature operating conditions.
MAX_TIMING_FAST_HV_LT	{1,0}	Max Delay Static Timing Analysis report based on Fast process, High Voltage, and Low Temperature operating conditions. 0 turns report generation OFF and 1 turns it ON. Default is 0.

Table 4 • **Verify Timing Parameters and Values**

Parameter Name	Value	Description
MIN_TIMING_SLOW_LV_HT	{1,0}	Min Delay Static Timing Analysis report based on Slow process, Low Voltage, and High Temperature operating conditions. 0 turns report generation OFF and 1 turns it ON. Default is 0.
MAX_TIMING_VIOLATIONS_SLOW_LV_HT	{1,0}	Max Delay Static Timing Analysis violation report based on Slow process, Low Voltage, and High Temperature operating conditions. 0 turns report generation OFF and 1 turns it ON. Default is 0.
MIN_TIMING_VIOLATIONS_FAST_HV_LT	{1,0}	Min Delay Static Timing Analysis violation report based on Fast process, High Voltage, and Low Temperature operating conditions. 0 turns Report generation OFF and 1 turns it ON. Default is 0.
MAX_TIMING_VIOLATIONS_FAST_HV_LT	{1,0}	Max Delay Static Timing Analysis violation report based on Fast process, High Voltage, and Low Temperature operating conditions. 0 turns Report generation OFF and 1 turns it ON. Default is 0.
MIN_TIMING_VIOLATIONS_SLOW_LV_HT	{1,0}	Min Delay Static Timing Analysis violation report based on Slow process, Low Voltage, and High Temperature operating conditions. 0 turns report generation OFF and 1 turns it ON. Default is 0.
CONSTRAINTS_COVERAGE	{1,0}	If set to 1, Libero SoC generates the Constraint Timing Coverage report
FORMAT	{TXT,XML}	Selects the format for the Timing Report: Text (txt) or XML (xml) format

10.3.2 run_tool

The run_tool runs the VERIFYTIMING Command with the configured options and takes no other parameters.

```
run_tool -name {VERIFYTIMING}
```

11 Verify Power

11.1 Description

Run Power Verification to generate Power Reports to ensure the power requirements of the design are met.

11.2 Tcl Snippet

```
run_tool -name {VERIFYPOWER} [-script {filename}]
```

11.3 Relevant Tcl Commands

11.3.1 run_tool

The run_tool -name {VERIFYPOWER} Tcl command generates power reports. While it has no configuration parameters that can be directly set from a Libero SoC Tcl script, there is a rich scripting capability that can be utilized from a separate script. A sample script “mypower.tcl” is given below:

```
smartpower_report_power \
   -powerunit "uW" \
   -frequnit "MHz"\
   -opcond "Typical"\
   -opmode "Active"\
   -toggle "TRUE"\
   -rail_breakdown "TRUE"\
   -battery_life "TRUE" \
   -style "Text"\
   -power_summary "TRUE"\
   -activity_sortby "Source"\
   {D:/work/power_report.txt} #name of the generated power report.
```

The name of this script can be passed as a parameter value to the run_tool command.

```
run_tool -name {VERIFYPOWER} -script “D:/work/mypower.tcl”
```

For a complete listing of the SmartPower-specific Tcl commands that can be included in the script and passed to Libero SoC, please refer to SmartPower User Guide.

12 Exporting Files

12.1 Description

This section describes the commands for extracting design information and generating files for programming. A sample script is given as an example. For details, see the Libero SoC Tcl Commands Reference Guide in Libero SoC Help (Help > Reference Manual > Tcl Command Reference)

12.2 Tcl Snippet

```
#Export Pin Reports for Board Layout and Design
export_pin_reports \
         -export_dir {D:\my_project\design_top} \
         -pin_report_by_name 1 \
         -pin_report_by_pkg_pin 1 \
         -bank_report 1 \
         -io_report 1
#Export Bitstream for Programming
export_bitstream_file \
         -file_name {design_top} \
         -export_dir {D:\2Work\my_project\export} \
         -format {STP|CHAIN_STP|DAT|SPI} \
         -master_file {1|0}\
         -master_file_components {SECURITY|FABRIC|ENVM} \
         -encrypted_uek1_file {1|0} \
         -encrypted_uek1_file_components {FABRIC|ENVM} \
         -encrypted_uek2_file {1|0| \
         -encrypted_uek2_file_components {FABRIC|ENVM} \
         -trusted_facility_file {1|0} \
         -trusted_facility_file_components {FABRIC|ENVM} \
         -add_golden_image {1|0} \
         -golden_image_address {<golden_image_address>} \
         -golden_image_design_version {<golden_image_design_version} \
         -add_update_image {1|0}\
         -update_image_address {<update_image_address>} \
         -update_image_design_version {<update_image_design_version>} \
         -serialization_stapl_type {SINGLE|MULTIPLE} \
         -serialization_target_solution {FLASHPRO_3_4_5| generic_STAPL_player}
#Export_job_data for Programming
export_job_data \
         -file_name {design_top} \
         -export_dir {D:\2Work\my_project\export} \
```

```
        -components {SECURITY|FABRIC|ENVM}
#Export_prog_job for Programming
export_prog_job \
        -job_file_name {design_top}
        -export_dir {D:\2Work\my_project\export} \
        -bitstream_file_type {TRUSTED_FACILITY|MASTER|UEK1|UEK2} \
        -bitstream_file_components {SECURITY|FABRIC|ENVM}
```

12.3 Relevant Tcl Commands

12.3.1 export_pin_reports

The export_pin_reports command configures and exports a pin report file to a specified folder/directory location.

```
export_pin_reports -export_dir {absolute path to folder location}\
   -pin_report_by_name {1|0}\
   -pin_report_by_pkg_pin {1|0}\
   -bank_report {1|0} \
   -io_report {1|0}
```

Table 5 • **Parameters and Values**

Parameter Name	Value	Description
pin_report_by_name	1 | 0	Set to “1” to have the pin report sorted by pin name. Default is “1”
pin_report_by_pkg_pin	1 | 0	Set to “1” to have the pin report sorted by package pin number, “0” to not sort by package pin number. Default is “1”.
bank_report	1 | 0	Set to “1” to generate the I/O bank report, “0” to not generate the report. Default is “1”
io_report	1 | 0	Set to “1” to generate the I/O report, “0” to not generate the report. Default is “1”.
export_dir	string	Absolute or relative path to the folder for pin report file

For details, refer to the Libero Online Help (Libero > Help).

12.3.2 export_bitstream_file

This command configures the parameters for the bitstream to be exported from Libero SoC.

Table 6 • **export_bitstream_file Parameters and Values**

Parameter Name	Parameter Value	Descriptions
file_name	string	Specifies the name of the bitstream file. If omitted, design name is used
export_dir	string	Location where the bitstream file is exported. If omitted, design export folder is used.
format	string, {STP | CHAIN_STF | DAT | SPI}	Specifies the bitstream file formats to be exported. Space is used as a delimiter. If omitted, STP (STAPL) will be exported.
master_file	boolean, (1 | 0}	set to “1” to export master_file and “0” not to export master_file

Table 6 • **export_bitstream_file Parameters and Values**

Parameter Name	Parameter Value	Descriptions
master_file_components	string, {SECURITY| FABRIC| ENVM}	Specifies the components in the design that will be saved to the bitstream file. The value can be any one or any combination of SECURITY, FABRIC, and ENVM.
encrypted_uek1_file	boolean, {1,0}	set to “1” to export the file or “0” not to export
encrypted_uek1_file_components	string {FABRIC| ENVM}	set to any one or both of FABRIC and ENVM
encrypted_uek2_file	boolean, {1,0}	set to “1” to export the file or “0” not to export
encrypted_uek2_file_components	string {FABRIC| ENVM}	set to any one or both of FABRIC and ENVM
trusted_facility_file	boolean {1,0}	set to “1” to export the file or “0” not to export
trusted_facility_file_components	string {FABRIC| ENVM}	set to any one or both of FABRIC and ENVM
add_golden_image	boolean {1,0}	set to “1” to enable or “0” to disable golden SPI image in SPI directory
golden_image_address	string, a hexadecimal value of 1 to 4 bytes (inclusive),e.g. {0A1F}	specifies the hexadecimal address for golden image
golden_image_design_version	string, e.g.12	specifies the decimal value for golden image design version
add_update_image	boolean {1,0}	sets to “1” to enable and “0” to disable update of image address
update_image_address	string, a hexadecimal value of 1 to 4 bytes (inclusive), e.g. {AF}	specifies the hexadecimal value for update image address
update_image_design_version	string, for example 15	specifies the decimal value for update image design version
serialization_stapl_type	string, {SINGLE | MULTIPLE}	specifies SINGLE or MULTIPLE for the type of STAPLE files
serialization_target_solution	string {Flashpro_3_4_5 | generic_STAPL_player	specifies the programming hardware, either FlashPro 3/4/5 programmer from Microsemi or a third_party generic programmer

12.3.3 export_job_data

The export_job_data Tcl command configures the parameters for the Job Manager Data Container (JDC) to be exported from Libero and used by the Job Manager

Table 7 • **export_job_data Parameters and Values**

Parameter Name	Parameter Value	Descriptions
file_name	string	Specifies the name of the job data file. If omitted, the design name is used
export_dir	string {<path>}	Specifies the location where the file is saved. If omitted, the Libero SoC export folder is used.

Table 7 • **export_job_data Parameters and Values**

Parameter Name	Parameter Value	Descriptions
components	string, {SECURITY| FABRIC| ENVM}	Specifies the components of the design that is saved to the file. The value can be any one or a combination of SECURITY, FABRIC or ENVM if they are available in the design. If the parameter is omitted, all available components of the design are saved. **Note:** The SECURITY component must be selected if user security is initialized for the current Libero SoC design.

12.3.4 export_prog_job

This Tcl command configures the parameters for the programming job to be exported.

Table 8 • **export_prog_job Parameters and Values**

Parameter Name	Parameter Value	Descriptions
job_file_name	string	Specifies the name of the file. The name must start with the design name. If omitted, the design name is used.
export_dir	string	Specifies the location where the job is saved. Any folder can be specified. The default folder is the Libero SoC export folder.
bitstream_file_type	string,{TRUSTED_FACILITY | MASTER | UEK1 | UEK2}	Only available in SmartFusion2 and IGLOO2. Not available for RTG4. Specifies the bitstream file to be included in the programming job. Only one bitstream file can be included in a programming job.
bitstream_file_components	string, {SECURITY | FABRIC | ENVM}	Only available in SmartFusion2 and IGLOO2. Not available for RTG4. Specifies the list of components to be included in the programming job. Valid values can be any one of or a combination of SECURITY, FABRIC, or ENVM. Components should be delimited by space. **Note:** The SECURITY option is available only when MASTER is selected in the bitstream_file_type parameter

13 Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

13.1 Customer Service

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**

13.2 Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

13.3 Technical Support

For Microsemi SoC Products Support, visit http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support.

13.4 Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at www.microsemi.com/soc.

13.5 Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

13.5.1 Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

13.5.2 My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

13.5.3 Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office.

Visit About Us for sales office listings and corporate contacts.

Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

13.6 ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; Enterprise Storage and Communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif. and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

5-02-00579-3/11.16