Sample Tcl script to run Libero SoC flow in batch mode for SmartFusion2

ID: SL5619
Devices: SmartFusion2
Tools: Libero SoC
Keywords: Tcl batch flow

```
# ---------------------------------------------------------
# The following Tcl sample script exercise the following:
#
# 1- Project Creation or Opening
# 1.1- Create a new Libero project
# 1.2- Open an existing Libero project
# 2- import an edif netlist (same can be done for any HDL netlist)
# 3- import SDC and PDC (or mark them as used)
# 4- Running Synthesis
# 5- compile (Default options)
# 6- timing driven place and route
# 7- export reports: timing, violation, max delay paths, pin report.
# 8- export .stpl file.
# 9- save and close the design
# ---------------------------------------------------------

# ---------------------------------------------------------
# 1.1 Create a new project
# ---------------------------------------------------------
new_project\
            -location {<project location path>/<project>} \
            -name {<project name>} \
            -project_description {} \
            -hdl {VERILOG} \
            -family {SmartFusion2} \
            -die {M2S050T} \
            -package {896 FBGA} \
            -speed {-1} \
            -die_voltage {1.2} \
            -adv_options {DSW_VCCA_VOLTAGE_RAMP_RATE:100_MS} \
            -adv_options {IO_DEFT_STD:LVCMOS 2.5V} \
            -adv_options {RESTRICTPROBEPINS:1} \
            -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}

# ---------------------------------------------------------
# 1.2 Open existing Libero project
# ---------------------------------------------------------
open_project -file {<project path>/<project_name>.prjx} \
                -do_backup_on_convert 1 \
         -backup_file {<path>/<project_name>.zip}
```

```
# ---------------------------------------------------------
# 2. Import EDIF (same can be done for any HDL netlist)
# ---------------------------------------------------------
import_files -edif {<path>/<file>.edn}

# ---------------------------------------------------------
# 3. Import constraints
#
# -io_pdc is for io constraints and attributes
# -fp_pdc is for floorplanning constraints
#  These constraints must be in seperate files and imported
#  with their individual commands
#  This applies to SmartFusion2 only
# ---------------------------------------------------------
import_files -io_pdc {<path>/<file_name>_io.pdc}
import_files -fp_pdc {<path>/<file_name>_fp.pdc}
import_files -sdc    {<path>/<file_name>.sdc}

# ---------------------------------------------------------
# 4. Running Synthesis
# The command also shows how to organize constraint files for Synthesis
#  I in step # 2 the file that was imported is an .edn, then the synthesis
#  step is not required.
# ---------------------------------------------------------
organize_constraints -file {./synthesis/<file_name>.sdc} \
        -mode {new} -designer_view {Impl1} \
        -module {<design_name>::work} \
        -tool {synthesis}


organize_tool_files -tool {SYNTHESIZE} \
        -file {./synthesis/<file_name>.sdc} \
        -module {<design_name>::work} \
        -input_type {constraint}

run_tool -name {SYNTHESIZE}


# ---------------------------------------------------------
# 5. compiling and specifying which constraint files to use
# sd1: is a SmartDesign component.
# ---------------------------------------------------------
organize_tool_files -tool {COMPILE} \
            -file {<path>/<file_name>_io.pdc} \
            -file {<path>/<file_name>_fp.pdc} \

            -file {<path>/<file_name>.sdc} \
            -module {sd1::work} \
            -input_type {constraint}

run_tool  -name {COMPILE}

# ---------------------------------------------------------
# 6. place and route
# ---------------------------------------------------------
configure_tool -name {PLACEROUTE} \
        -params {EFFORT_LEVEL:false} \
        -params {INCRPLACEANDROUTE:false} \
        -params {PDPR:false} -params {TDPR:true}
```

```
run_tool -name {PLACEROUTE}

# ---------------------------------------------------------
# 7. exporting reports
# This automatically exports a number of reports
# under <project>/designer/<design_name>
# some are xml, some are text
# ---------------------------------------------------------
run_tool -name {VERIFYTIMING}

# pin report goes to <project>/designer/<design_name>/export
run_tool -name {EXPORTPIN}

# ---------------------------------------------------------
# 8. export programming file (stp)
# stpl file under <project>/designer/<design_name>/export
# ---------------------------------------------------------
run_tool -name {EXPORTPROGRAMMINGFILE}

# ---------------------------------------------------------
# 9. Save and close project
# ---------------------------------------------------------
close_project -save 1
```

Last Modified: 5/23/2013
If you have any questions or concerns about this document, please contact Actel Customer Support:
soc_tech@microsemi.com | 1.650.318.4460 | 1.800.262.1060 (USA toll-free)