



Using TCL to Implement an Efficient Synthesis Environment

Tim L. Wilson
Rodney Pesavento

Intel Corporation



Purpose

Tim L. Wilson

Intel

Describe the process of converting our existing DC/PT environment to a simple, portable and flexible framework based on TCL



Outline

Tim L. Wilson

Intel

- Major Features
- Seven Steps of Synthesis
- Project Script File Overview
- Target Level Script Files
- Coding Conventions
- PrimeTime Integration
- Tcl Conversion Techniques
- Supporting Scripts and Gmake
- Improvement Suggestions for Synopsys
- Conclusion



Major Features

Tim L. Wilson

Intel

- Library Flexibility - changeable at run time
- Language Independence - VHDL and Verilog
- Project and Target Level Flexibility
- Common Library Setup Between DC and PT
- Very portable with little setup required, all files reside in one directory with two environment variables needed

Seven Steps of Synthesis

Tim L. Wilson

Intel

Dc_setup function
DC variables. Search_path etc

Library Setup
Std cell, embedded blocks

Read in Design Files
Analyze and Elaborate design

Constrain the Design
Clock, port delays, MCPs etc

Compile the Design
ungrouping, scan chain, etc

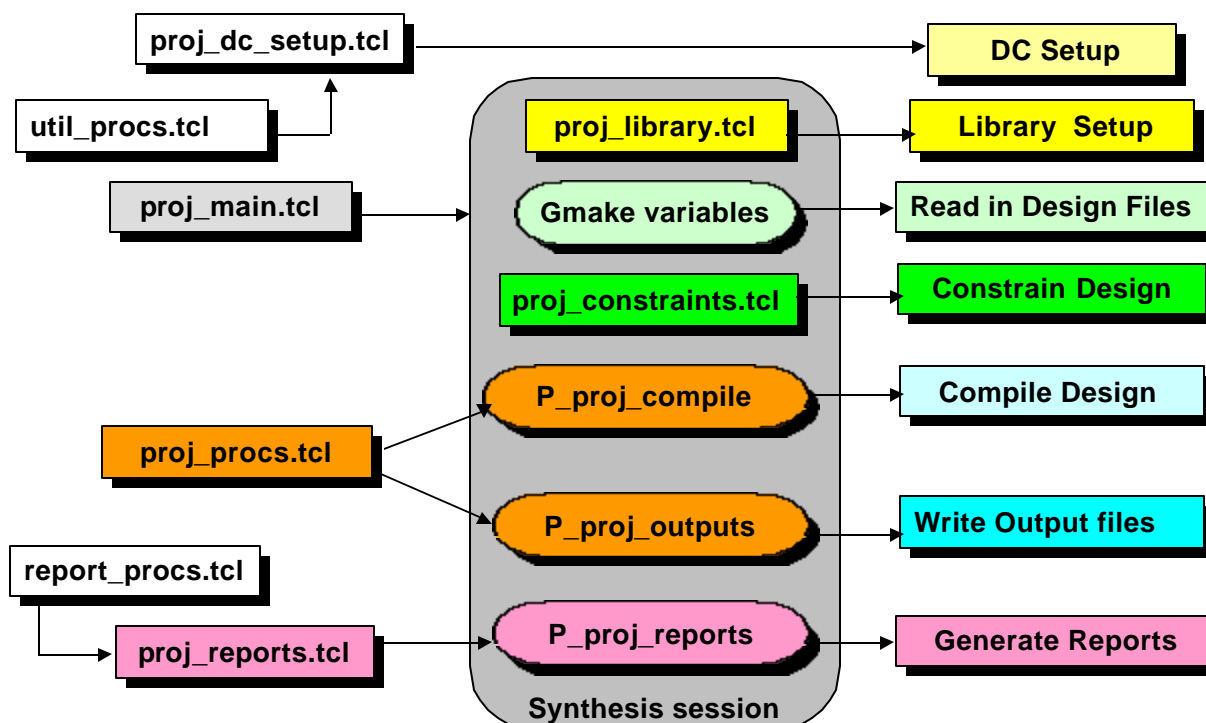
Write the Output Files
dbs, netlist, sdf, sdc, pdef, etc

Generate the Report Files
area, timing, cell usage, etc

Project Script File Overview

Tim L. Wilson
Intel

6





Target Level Script Files

Tim L. Wilson
Intel

- Enable individual targets to override project scripts
 - <target>_dc_setup.tcl
 - <target>_procs.tcl
 - <target>_main.tcl
 - <target>_<library>_dont_use.tcl
 - <target>_clocks.tcl
 - <target>_constraints.tcl
 - <target>_exceptions.tcl
 - <target>_wire_load.tcl
 - <target>_compile.tcl
 - <target>_outputs.tcl
 - <target >_reports.tcl

Proj_main execution priority

Tim L. Wilson
Intel

8

- Target procedures or scripts take priority over the project defaults
- if (target procedure exists) execute target procedure
else if (target script exists) source target script
else execute project procedure or script
- Example of compile phase in proj_main

```
echo "##### Beginning Compile Phase #####"

if {{P_exec_if_exists P_${G_DESIGN_NAME}_compile "MAIN"}} {
} elseif {[P_source_if_exists \
    "./scripts/${G_DESIGN_NAME}_compile.tcl" "MAIN"]}] {
} else {P_proj_compile}
```




Coding Conventions

Tim L. Wilson
Intel

- Procedure files (*_procs.tcl) contain only procedure declarations
- Executable files do not include procedures
- All global variables begin with “G_”.
- Procedure specific guidelines
 - All procedures begin with “P_”.
 - Procedures always return a value (1-success, 0-failure, or string data).
 - Do not invoke Perl or C-shell scripts
 - Procedures use the built-in “define_proc_attributes” for help
 - Assign default values to optional procedure arguments
 - Target scripts should never re-define project procedures
 - Follow project / target naming conventions



PrimeTime Integration

Tim L. Wilson
Intel

- proj_pt_setup.tcl file sourced from ~/.synopsys_pt.setup
- Uses same util_procs.tcl and proj_procs.tcl files
- Sources proj_library.tcl to set up library variables
 - not all library setup is consistent between PT and DC
 - must skip some commands such as set_dont_use
 - checks Tcl variable \${synopsys_program_name} to determine if PT is being run and skips certain DC commands
- PT specific TCL procedures contained in proj_pt_procs.tcl



Tcl Conversion Techniques

Tim L. Wilson

Intel

- Procedures have different scoping rules for variables
- There is a difference between lists and collections
- Use “proc_define_attributes” in procedure definition
- Take advantage of built-in Tcl commands
 - printvar
 - help / help -v
 - info commands (i.e info body)
 - error_info
- Some DC and PT commands have slightly different behavior
 - link
 - write_sdf



Gmake Integration

Tim L. Wilson

Intel

- Gmake used to determine file dependencies
- Calls Perl script Synbatch which in turn executes dc_shell-t
- Gmake Variables (example: target cnt_top with two sub blocks)
 - RECURSEDIRS
 - SUBFUB_cnt_top := ../src/cnt1.v ../src/cnt2.v
 - \$(DB)cnt_top.db : ../src/cnt_top.v \$(SUBFUB_cnt_top)
 - all.local : \$(DB)cnt_top.db
- Example of a gmake / synbatch session

```
%> gmake HOST=edison
```

```
synbatch ../src/cnt_top.v SUBFUB="../src/cnt1.v ../src/cnt2.v" \  
LIB=DEFAULT HOST=edison
```



Supporting Script - Synbatch

Tim L. Wilson
Intel

- Synbatch - Perl interface between gmake and dc_shell-t
 - Calls dc_shell-t with variables set from Makefile
 - Allows remote batch jobs to be launched
 - Contains some limited job control (Ctrl-C)
- Example of a gmake / synbatch session

```
%> gmake HOST=edison
synbatch ../src/cnt_top.v SUBFUB="../src/cnt1.v ../src/cnt2.v" \
LIB=DEFAULT HOST=edison

dc_shell-t -x `set G_LIBRARY_NAME DEFAULT; \
  SET G_DESIGN_FILENAME ../src/cnt_top.v; \
  set SUBFUB "../src/cnt1.v ../src/cnt2.v"; source proj_main.tcl;`
```



Improvement Suggestions for Synopsys

14

Tim L. Wilson
Intel

- Suppress_error and suppress_warning operation
- Allow report commands to return a string to a Tcl variable
- Fix the get_license command so that it does not return an error if one already has the license
- Remove command differences between Design Compiler and PrimeTime



Conclusion

Tim L. Wilson
Intel

- Project turned into a major rewrite of our environment
- Very portable with only a few files and two environment variables needed
- Currently being used by multiple projects across multiple sites at Intel