# Design Compiler

# Synthesis with Design Compiler

- This manual will go through a step-by-step process for performing synthesis using Design Compiler
- It is assumed that the reader has basic understanding about
  - RTL Design (Verilog, VHDL)
  - Syntax and grammar of Verilog/VHDL
  - Standard cell libraries
  - Basics of synthesis process
- Its is assumed that the reader is not familiar with the design compiler tool
- This manual will go through basic synthesis process to synthesize a AES encryption core

# Design Compiler

- Design Compiler is developed by Synopsys and widely used in industry for ASIC design
- It comprises tools that synthesize your HDL descriptions into optimized, technology-dependent, gate-level designs
- It supports a wide range of flat and hierarchical design styles and can optimize both combinational and sequential designs for speed, area, and power

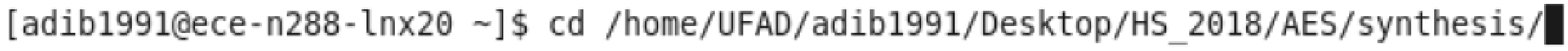
# Necessary Files

- Make sure that the following files are present

| Folder Name | File Name | Description |
|---|---|---|
| AES/rtl/ | aes_128.v<br>round.v<br>table.v | Verilog RTL files for the AES crypto module |
| AES/synthesis/ | run_compiler.tcl | Script file for synthesis<br>All the synthesis commands are located here |
| AES/constraints/ | constraints_AES.tcl<br>Dft_constraints_AES.tcl | Contains constraints of a design |
| SAED_EDK90nm/synopsys/ | saed90nm_typ.db | 90nm Library file |

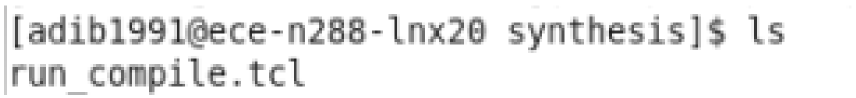
# Setup enviornment

- Change directory to the synthesis folder



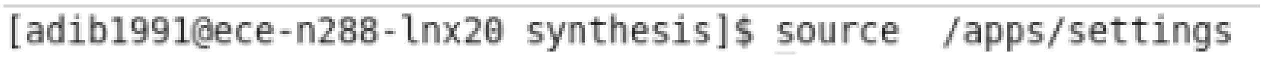
```
[adib1991@ece-n288-lnx20 ~]$ cd /home/UFAD/adib1991/Desktop/HS_2018/AES/synthesis/
```

- Make sure that run_compile.tcl is located in the folder



```
[adib1991@ece-n288-lnx20 synthesis]$ ls
run_compile.tcl
```

- Run the following command: source  /apps/settings

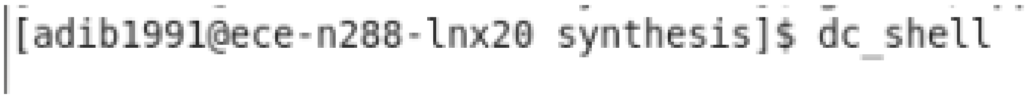


```
[adib1991@ece-n288-lnx20 synthesis]$ source  /apps/settings
```

- Click here for more discussion on source  /apps/settings command

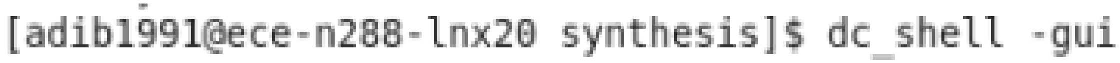
# dc_shell

- Open the design compiler by the following command: dc_shell



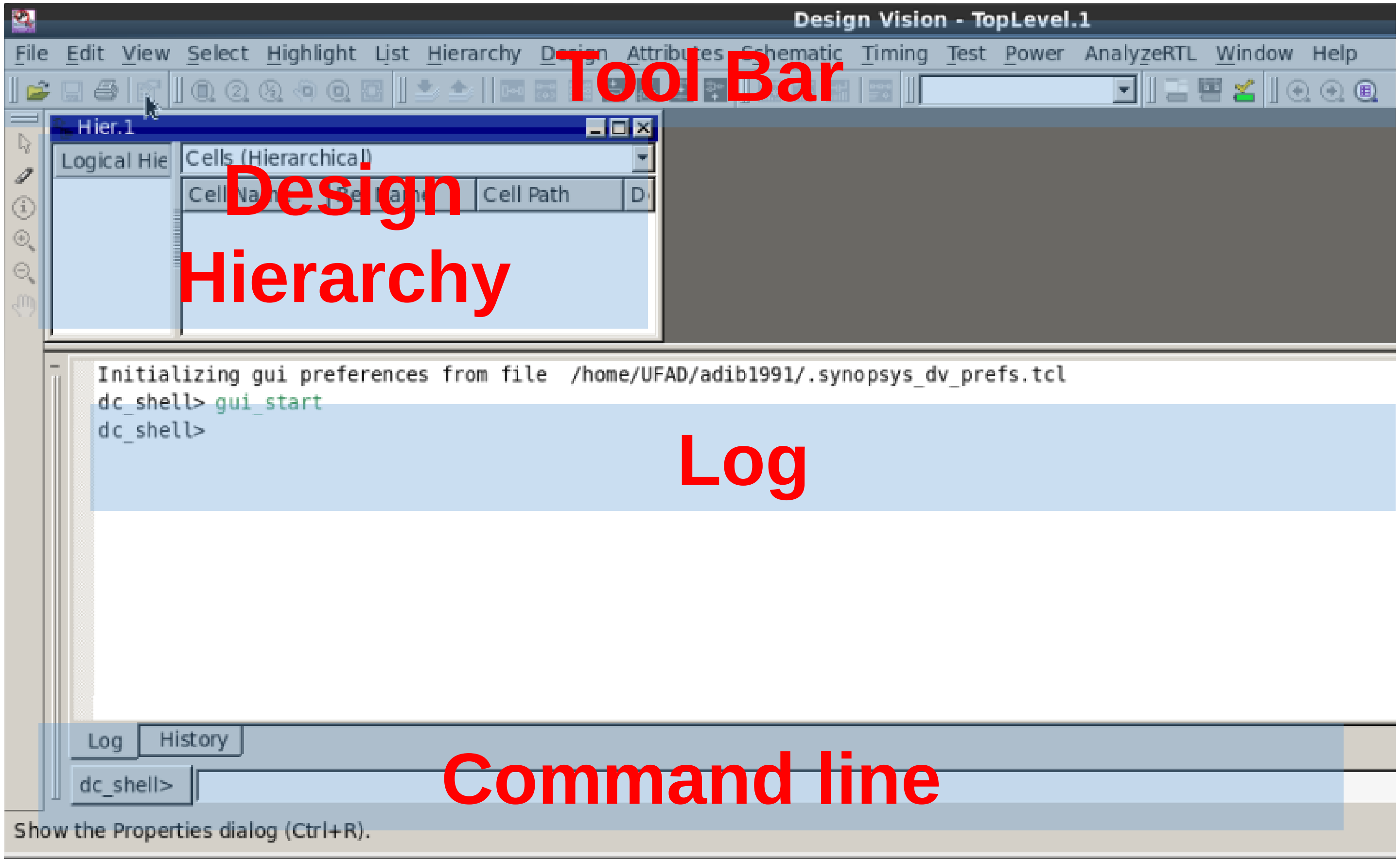
```
[adib1991@ece-n288-lnx20 synthesis]$ dc_shell
```

- You can also use the following command to invoke the gui of design compiler: dc_shell -gui



```
[adib1991@ece-n288-lnx20 synthesis]$ dc_shell -gui
```

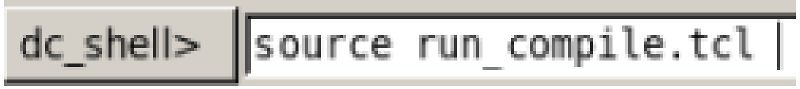
# dc_shell: gui


Design Vision - TopLevel.1
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
Tool Bar
Hier.1
Logical Hie
Cells (Hierarchical)
Cell Na
Ref Name
Cell Path
Design
Hierarchy
Log
Log
History
dc_shell>
Command line
Show the Properties dialog (Ctrl+R).

```
Initializing gui preferences from file  /home/UFAD/adib1991/.synopsys_dv_prefs.tcl
dc_shell> gui_start
dc_shell>
```

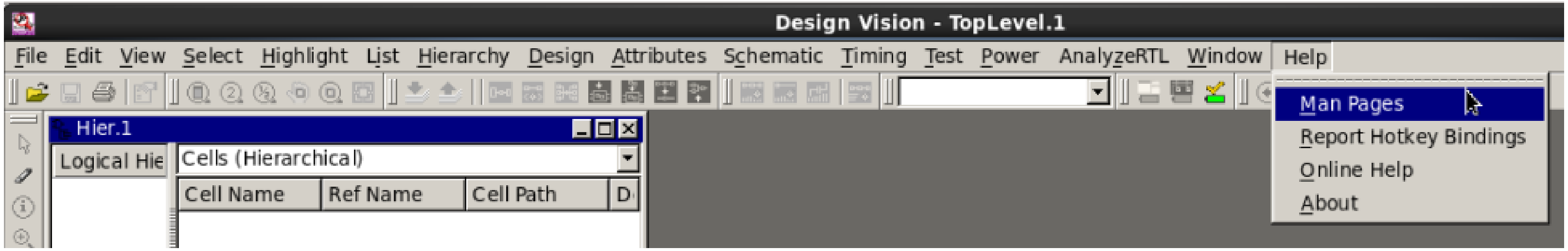
# run_compile.tcl

- Sourcing the run_compile.tcl file will perform the synthesis



```
dc_shell> source run_compile.tcl
```

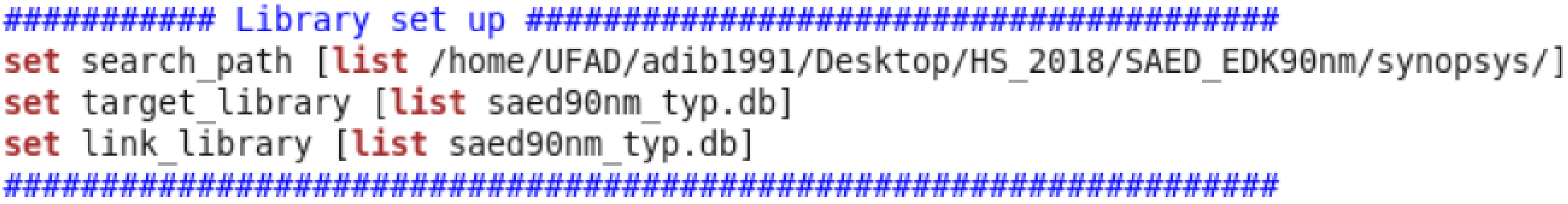
- I will go through each command step-by-step to explain each command's operation
- The detailed description of all commands can be found here
- Also, you invoke the man pages as follows: Help → Man pages



Design Vision - TopLevel.1
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
Hier.1
Logical Hie
Cells (Hierarchical)
Cell Name
Ref Name
Cell Path
D
Man Pages
Report Hotkey Bindings
Online Help
About

# Library Set-up

- Type the following commands

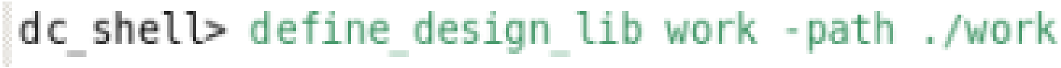


```
############ Library set up ##########################################
set search_path [list /home/UFAD/adib1991/Desktop/HS_2018/SAED_EDK90nm/synopsys/]
set target_library [list saed90nm_typ.db]
set link_library [list saed90nm_typ.db]
#####################################################################
```

- **search_path**: Specifies directories that the tool searches for files
- **target_library:** the ASIC technology which the design is mapped
- **link_library:** Specifies the list of design files and libraries used to interpret the input files
- Click here to see the difference between target_library and link_library

# Working Directory

- Type the following command

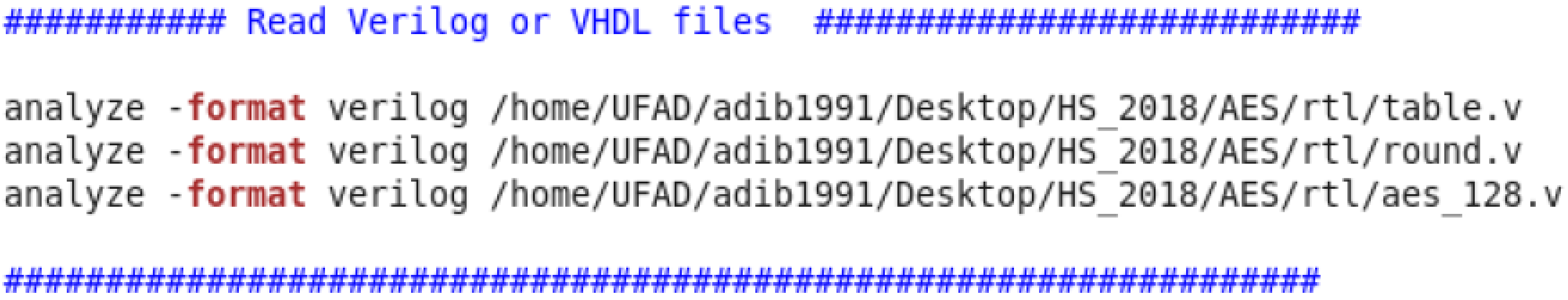


```
dc_shell> define_design_lib work -path ./work
```

- **define_design_lib:** Creates a directory and stores intermediate representations of designs in that directory

# Read files

- Type the following commands to read the Verilog or VHDL files

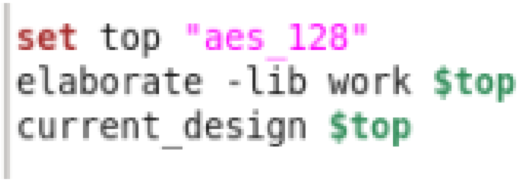


```
########### Read Verilog or VHDL files  ############################

analyze -format verilog /home/UFAD/adib1991/Desktop/HS_2018/AES/rtl/table.v
analyze -format verilog /home/UFAD/adib1991/Desktop/HS_2018/AES/rtl/round.v
analyze -format verilog /home/UFAD/adib1991/Desktop/HS_2018/AES/rtl/aes_128.v

##################################################################
```

- **analyze:** command reads the design files
- All the design files need to be read, otherwise the un-read designs will be used as black-box
- You can also use the command to **-autoread** mode to automatically read all the files;  to learn more click here

# Top module

- Type the following commands to read and elaborate the top module

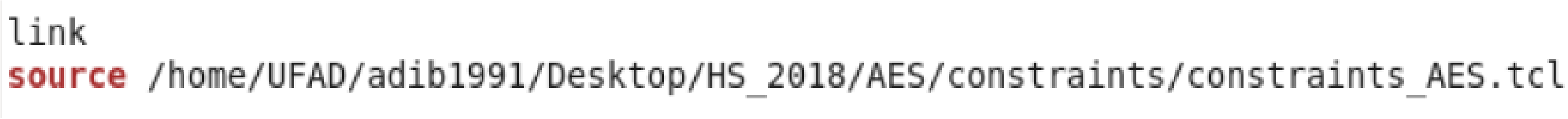


```
set top "aes_128"
elaborate -lib work $top
current_design $top
```

- $top is a tcl variable. Use set command to define it
- **elaborate**: command builds a design from its intermediate representation
- **current_design:** command sets the name of top module

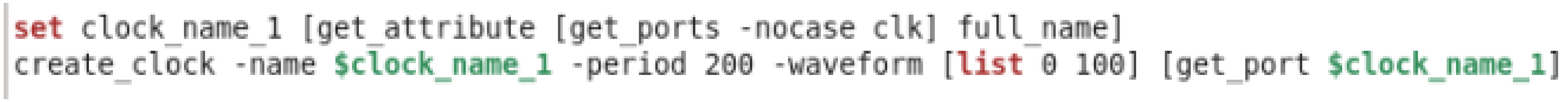
# Link and Constraints

- Use link command to resolve design references
- Source the constraints_AES.tcl file



```
link
source /home/UFAD/adib1991/Desktop/HS_2018/AES/constraints/constraints_AES.tcl
```

- This file applies constraints into the synthesis process
- For example, the command shown below, sets the maximum clock period of 200ns



```
set clock_name_1 [get_attribute [get_ports -nocase clk] full_name]
create_clock -name $clock_name_1 -period 200 -waveform [list 0 100] [get_port $clock_name_1]
```

- Click here to see all the constraint commands

# Compile

- compile command performs logic-level and gate-level synthesis and optimization on the current design
- Has the following options
  - **-power_effort:** Effort for power optimization
  - **-area_effort:** Effort for area optimization
  - **-scan:** Replaces all sequential elements with scan elements (DfT) during optimization
- If scan-chain inserted design is required, then the additional commands shown here
- If a flatten design is needed, then the additional commands shown here

# Save Files

- Always use the change_names -rules [verilog|vhdl] -hierarchy command whenever you want to write out a Verilog/VHDL
- Use write_file command to save the following files
- .v: gate-level netlist

  ```
  write -format verilog -hierarchy -output "aes_128_netlist.v"
  ```
- .sdc: timing constraint file for P&R

  ```
  write_sdc aes_128_netlist.sdc
  ```
- .sdf: timing file for Verilog simulation

  ```
  write_sdf -version 1.0 aes_128_netlist.sdf
  ```
- .spf: test protocol file for Tetramax (for DfT design)

# Synthesis Report

- Use following commands to:
- Report area
  **report_area -hierarchy > “aes_128_report.out”**
- Report reference
  **report_reference >> “aes_128_report.out”**
- Reports dynamic and static power for the design
  **report_power >> “aes_128_report.out”**
- Reports timing information about a design
  **report_timing >> “aes_128_report.out”**

# Some Useful Commands

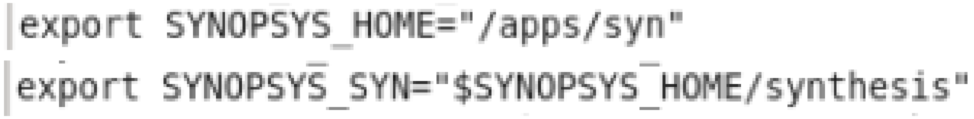
| Command Names | Description |
|---|---|
| get_object_name | Returns a list of names of the objects in a collection |
| all_fanin | Reports pins, ports, or cells in the fanin of specified sinks |
| all_fanout | Returns a set of pins, ports, or cells in the fanout of the specified sources |
| all_registers | Returns a collection of sequential cells or pins in the current design |

- You can use the following commands to get approximated switching activity

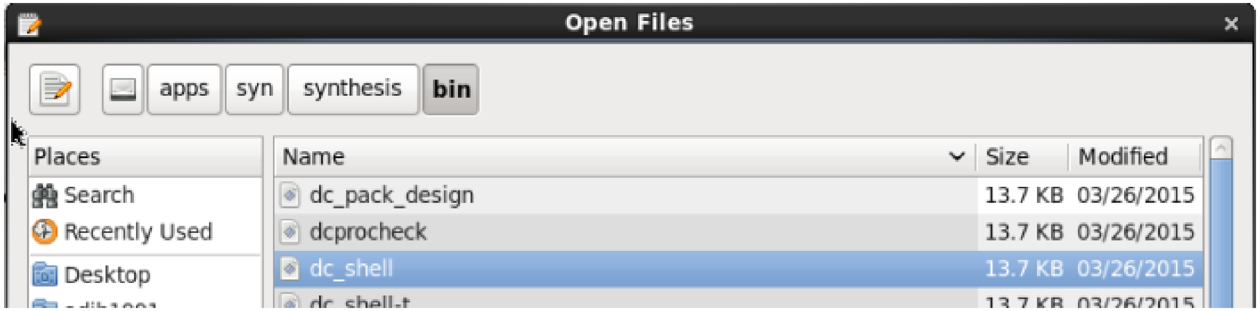
```
report_power -net -analysis_effort high -nosplit > report.txt
```

# source /apps/settings

- You are sourcing the settings file located in the /apps folder
- You can view the file using: gedit /apps/settings



```
export SYNOPSYS_HOME="/apps/syn"
export SYNOPSYS_SYN="$SYNOPSYS_HOME/synthesis"
```

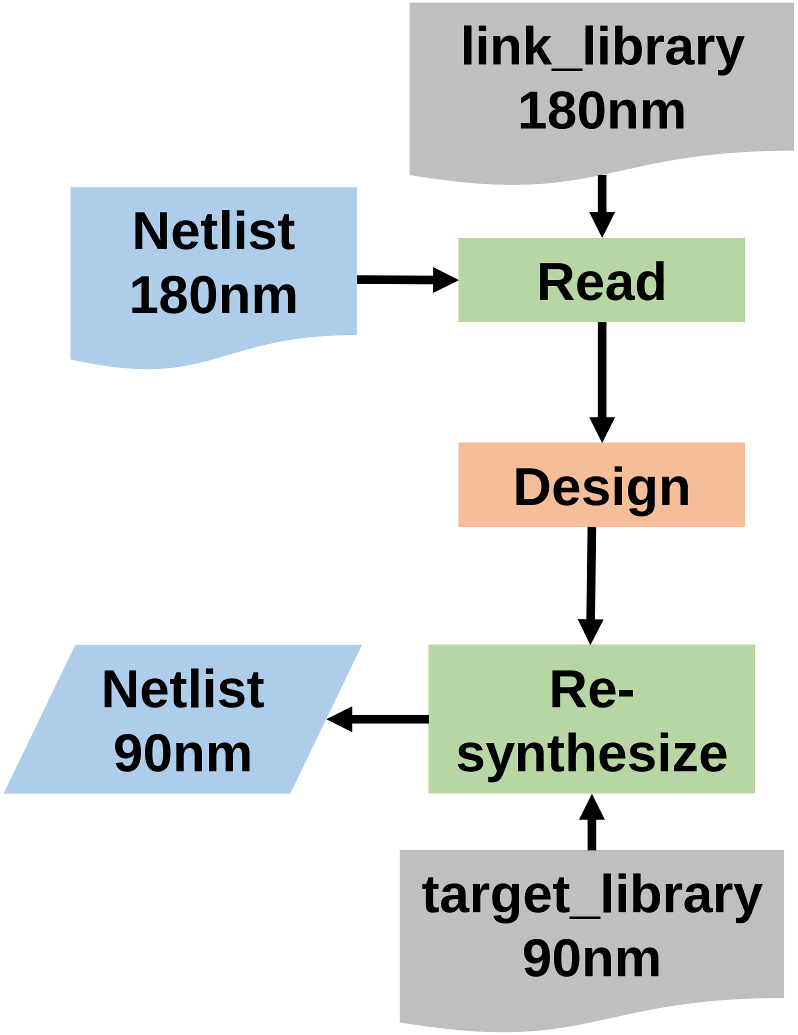


| Name | Size | Modified |
|---|---|---|
| dc_pack_design | 13.7 KB | 03/26/2015 |
| dcprocheck | 13.7 KB | 03/26/2015 |
| dc_shell | 13.7 KB | 03/26/2015 |
| dc_shell-t | 13.7 KB | 03/26/2015 |

- export command is used to export a variable or function to the environment of all the child processes running in the current shell
- This allows us to invoke dc_shell and other tools

# target_library and link_library

- Say you have a design synthesized with 180nm library and you want to re-synthesize it to 90nm library
- Then you need to write
  set link_library [list 180nm.db]
  set target_library [list 90nm.db]
- The figure describes the process



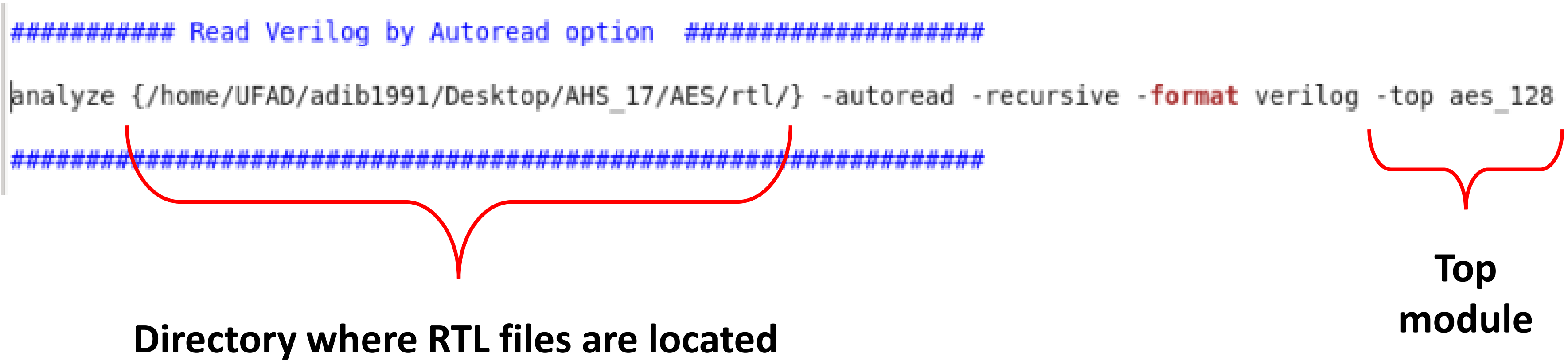
# Constraint Commands

- The following table shows the commands used in the constraints_AES.tcl file

| Command Names | Description |
|---|---|
| create_clock | Create clock for the current design |
| set_clock_latency | Specifies the clock latency for clocks |
| set_input_delay | Sets input delay on pins or input ports relative to a clock signal |
| set_driving_cell | sets attributes on the specified input to associate an external driving cell with the ports |
| set_max_capacitance | sets the **max_capacitance** attribute to a specified value on the specified clocks, ports or designs |
| set_operating_conditions | Defines the operating conditions for the current design. |

# Autoread

- use the **-autoread** mode for script-free analysis of whole HDL source directories or directory trees
- All HDL source files located in those directories are processed
- Use the following code for **-autoread**



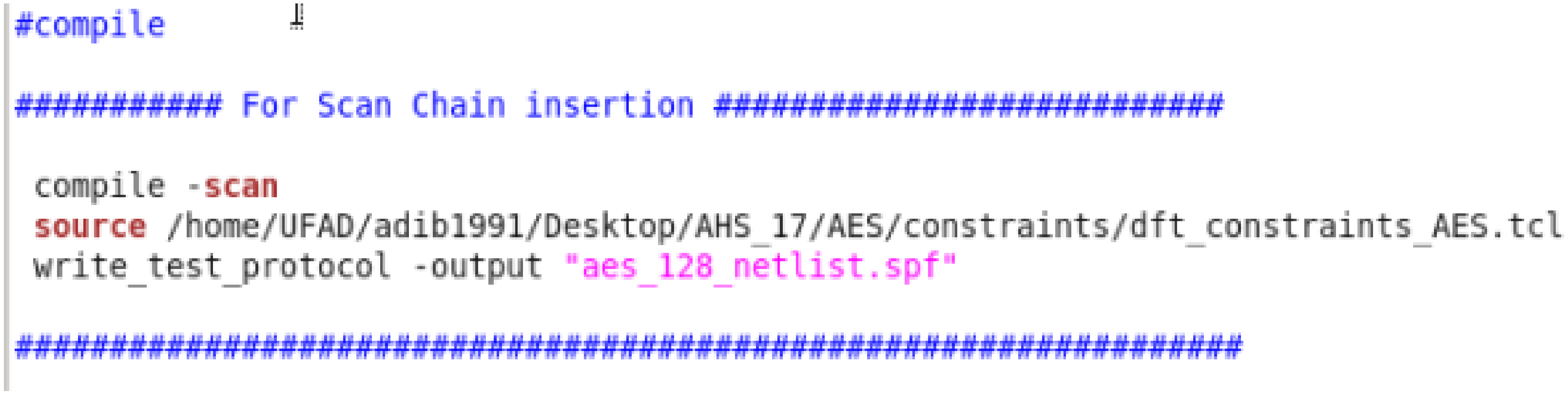
```
########### Read Verilog by Autoread option  ####################

analyze {/home/UFAD/adib1991/Desktop/AHS_17/AES/rtl/} -autoread -recursive -format verilog -top aes_128

#################################################################
```

Directory where RTL files are located

Top module

# Scan Chain Insertion

- Comment out the compile command of the original script
- Uncomment the following commands



```
#compile

########### For Scan Chain insertion ###########################

 compile -scan
 source /home/UFAD/adib1991/Desktop/AHS_17/AES/constraints/dft_constraints_AES.tcl
 write_test_protocol -output "aes_128_netlist.spf"

##################################################################
```

- The details of the dft/scan chain insertion command can be in this slide

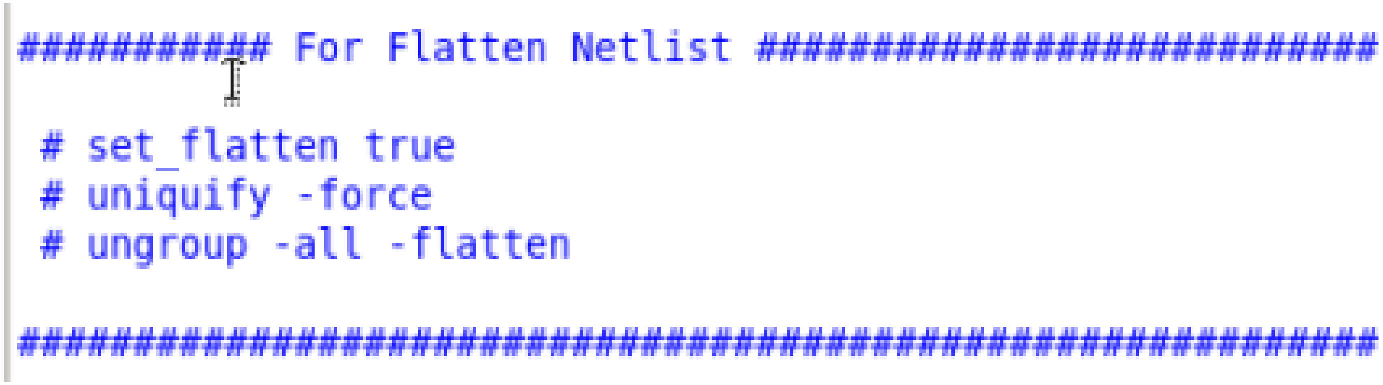
# DfT Constraint Commands

- The following table shows the commands used for the dft/scan chain insertion

| Command Names | Description |
|---|---|
| compile -scan | Replace sequential elements with its scan equivalent elements |
| set_scan_configuration | Specifies the scan chain design. |
| set_dft_signal | Specifies the DFT signal types for DFT insertion |
| set_dft_configuration | Sets the DFT configuration for the current design |
| create_test_protocol | Creates a test protocol based on user specifications |
| dft_drc | Checks the current design against test design rules |
| insert_dft | Inserts DFT logic in the current design |

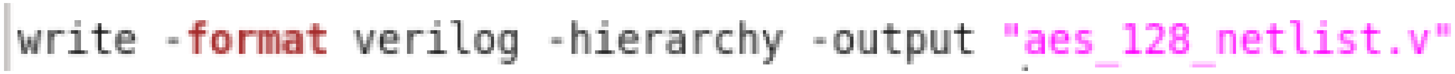
# Flatten Netlist

- Uncomment the following commands for generating a flatten netlist



```
########### For Flatten Netlist ###########################

 # set_flatten true
 # uniquify -force
 # ungroup -all -flatten

###########################################################
```

- Remove –hierarchy from the following command



```
write -format verilog -hierarchy -output "aes_128_netlist.v"
```