

Department of Electronic and Telecommunication Engineering

University of Moratuwa, Sri Lanka

EN4603 - Digital IC Design



Laboratory Experiment 1

RTL Synthesis

Laboratory Report

Submitted by

Submitted by

C.S.Pallikkonda

180441C

R.M.A.S.Rathnayake

180534N

B.P.Thalagala

180631J

Submitted on

February 17, 2023

Contents

List of Figures	2
List of Tables	2
List of Abbreviations	2
1 Introduction	3
1.1 Practical	3
1.2 DFT Insertion	4
1.3 Associated Genus Commands	5
2 Exercise	8
Bibliography	8

Note:

All the materials related to the report can also be found at <https://github.com/bimalka98/Digital-IC-Design>

List of Figures

1	Top level diagram of the Universal Asynchronous Receiver Transmitter (UART) design	3
2	Overview of Design For Testability (DFT) insertion flow of 'Scan Test' in Cadence Genus software.	4

List of Tables

List of Abbreviations

ASIC Application Specific Integrated Circuit

ATPG Automatic Test Pattern Generation

DEF Design Exchange Format

DFT Design For Testability

GPDK Generic Process Design Kit

HDL Hardware Description Language

RTL Register-Transfer Level

RX Receiver

TCL Tool Command Language

TX Transmitter

UART Universal Asynchronous Receiver Transmitter

1 Introduction

1.1 Practical

In this practical, we will be using *Cadence Genus - Ver. 18.10* to perform **DFT** insertion into an example Register-Transfer Level (**RTL**) design, a **UART** transceiver (shown in Figure 1). As inputs to Genus, we will provide

1. Source Verilog files
2. Technology libraries provided by the fabrication plant (here, 45 nm educational Generic Process Design Kit (**GPDK**) given by Cadence) : (.lib, .lef, .tch)
 - Library Timing (.lib) files specify timing (cell delay, cell transition time, setup and hold time requirement) and power characteristics of standard cells. Slow and fast libraries characterize standard cells with maximum and minimum signal delays, which could occur from process variations.
 - Tch files are binary files that accurately characterize library elements, that include capacitance and resistance.
 - Library Exchange Format (LEF) specify design rules, metal capacitances, layer information... etc.
3. Timing constraints

and will obtain the Scan-test compatible netlist (Verilog files), timing constraints (.sdc file), scanDesign Exchange Format (**DEF**) file and a set of files to be used as inputs to Cadence Modus for Automatic Test Pattern Generation (**ATPG**) as output. We will then analyze, compare and comment on the area and ports of the design at various stages of the Scan test insertion design flow.

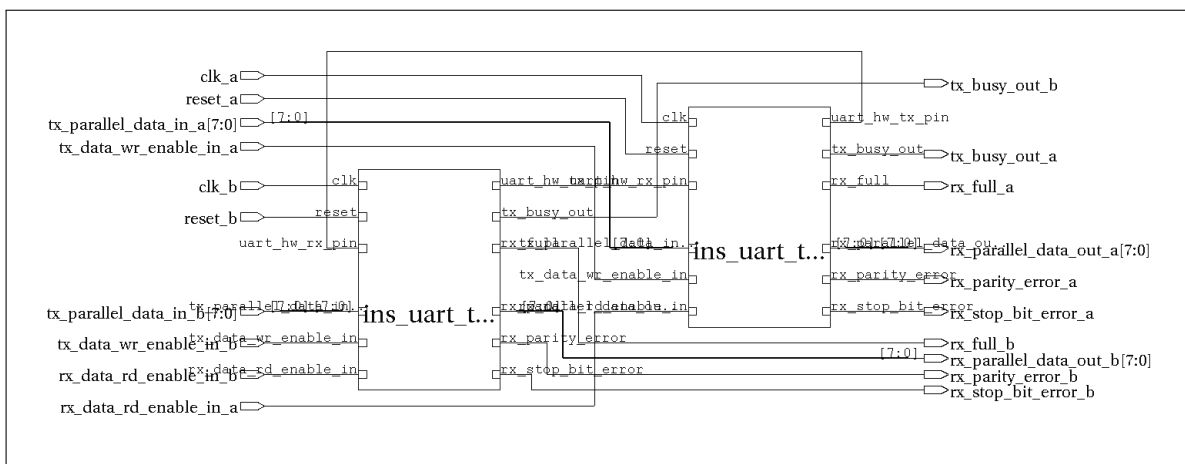


Figure 1: Top level diagram of the **UART** design

1.2 DFT Insertion

In the context of Application Specific Integrated Circuit (ASIC) design, DFT refers to the process of inserting additional logic to improve the testability of the design. Here the testability is defined as the ability to communicate to the internal nodes through the primary input and output ports. The goal in DFT is to cover a maximum number of faults with a minimum amount of additional logic, as the insertion of additional logic directly translates to additional cost per chip. Although there are various DFT techniques in digital IC design, in this experiment, we will be focusing on inserting the 'Scan Test' into an example design.

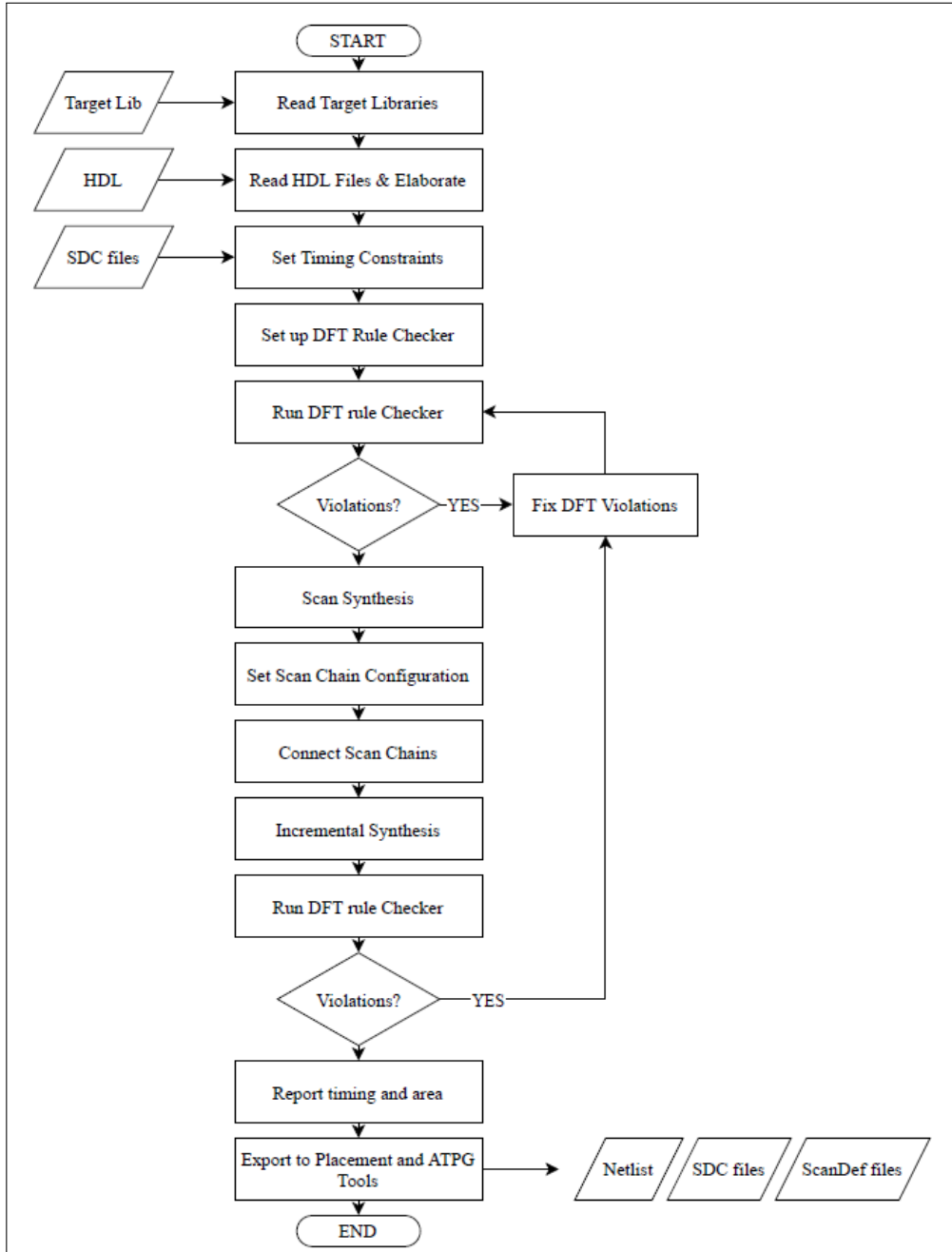


Figure 2: Overview of DFT insertion flow of 'Scan Test' in Cadence Genus software.

1.3 Associated Genus Commands

Note: *Genus is a Tool Command Language (TCL) based tool and therefore .tcl scripts can be created to execute a series of commands instead of typing each command individually. The entire interface of Genus is accessible through TCL, and true TCL syntax and semantics are supported.*

Followings are the commands used in this lab, and the entire script could be executed once using a single .tcl file. However, it was encouraged to execute the commands one-by-one in order to understand the process of synthesizing an RTL design.

```
# 1. Link Technology Library
source ../scripts/setup.tcl

# 2. Read HDLs
read_hdl [glob ../input/rtl/*.v]

# 3. Elaborate the top module
elaborate uart_top

# Uniquify the top module
uniquify uart_top

# 5. Set timing constraints
source ../input/constraints.tcl

# Now that we have set the target libraries,
# read the design files and set the design constraints,
# we may now proceed to the DFT insertion steps.
# *****

# 7. Set the DFT scan style in order to configure the DFT rule checker
set_db dft_scan_style muxed_scan

# 8. Set the prefix for names of additional modules/ports
set_db dft_prefix dft_

# 9. Define shift_enable signal
define_shift_enable -name SE -active high -create_port SE

# 10. Run DFT rule checker
check_dft_rules

# If all steps have been followed correctly up until this point,
# we should not see any DFT violations
# *****

# 11. Synthesize the design into the generic logic netlist,
# and then map to the technology library
```

```

# Synthesize to generic logic with medium effort
set_db syn_generic_effort medium
syn_generic

# Map to technology library and re-synthesize with medium effort
set_db syn_map_effort medium
syn_map

# 12. Write the scan synthesized netlist as uart_top_1.v
write_hdl > ../output/uart_top_1.v

# 13. Generate the reports after scan synthesis.
report_area > ../report/after_scan_synthesis/area.log
report_timing -nworst 10 > ../report/after_scan_synthesis/timing.log
report_port * > ../report/after_scan_synthesis/ports.log
report_power > ../report/after_scan_synthesis/power.log

# 14. Set scan configuration
define_scan_chain -name top_chain_a -sdi scan_in_a -sdo scan_out_a
    -non_shared_output -create_ports -domain clk_a
define_scan_chain -name top_chain_b -sdi scan_in_b -sdo scan_out_b
    -non_shared_output -create_ports -domain clk_b

# 15. Preview the scan chains/Scan Stitching
connect_scan_chains -preview -auto_create_chains

# 16. Preview the scan chains/Scan Stitching
connect_scan_chains -auto_create_chains

# 17. Perform incremental synthesis to generate the netlist
# of the scan connected design
syn_opt -incr

# 18. Perform DFT rule check after scan connecting
check_dft_rules

# 19. Report scan setup and scan chain information
report_scan_setup > ../report/scan_setup.log
report_scan_chains > ../report/scan_chains.log

# 20. Write the DFT (scan test) inserted netlist and constrains.
write_hdl > ../output/uart_top_2.v
write_sdc > ../output/uart_top_2.sdc

# 21. Write the scanDEF file
write_scandef > ../output/uart_top_2_scanDEF.scandef

# 22. Generate the reports after scan connect
report_area > ../report/after_scan_connect/area.log
report_timing -nworst 10 > ../report/after_scan_connect/timing.log

```

```
report_port * > ../report/after_scan_connect/ports.log
report_power > ../report/after_scan_connect/power.log

# 23. Write the scripts required for the ATPG tool
write_dft_atpg -library
    ../input/libs/gsclib045/timing/slow_vdd1v0_basicCells.lib
```


2 Exercise

This section documents the observations made in step 4 and 5 of the practical guide, with screenshots and explanations.

Bibliography