Day-1

Introduction to RTL Design and Synthesis

RTL Design- register-transfer level (**RTL**) is a **design** abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers, and the logical operations performed on those signals.

Various Commands Used in Simulation of an RTL Design

iverilog – Use to load the design(RTL code and test bench) in iverilog and a file a.out is created

```
anshi@rtlworkshop-23062021-02:~/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files$ iverilog good_mux.v
anshi@rtlworkshop-23062021-02:~/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files$ iverilog good_mux.v tb_good_mux.v
```

./a.out – This command dumps the data of the test bench into a Value Change Data File

gtkwave- gtkwave is the waveform viewer whose input is the Value Change Data File.

```
anshi@rtlworkshop-23062021-02:~/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files$ ./a.out
VCD info: dumpfile tb_good_mux.vcd opened for output.
anshi@rtlworkshop-23062021-02:~/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files$ gtkwave tb_good_mux.vcd

GTKWave Analyzer v3.3.86 (w)1999-2017 BSI

[0] start time.
[300000] end time.
```

Synthesis Commands in Yosys

yosys - This command is used to invoke yosys the synthesis tool.

This is the synthesizer used to convert RTL to Netlist

```
anshi@rtlworkshop-23062021-02:~/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files$ yosys
```

read_liberty -lib <library path> - read cells from liberty file as modules into current design. This command reads the .lib file

-lib – Only create empty black box modules

```
yosys> read_liberty -lib ../my_lib/lib/sky130_fd_sc_hd__tt_025C_1v80.lib

2. Executing Liberty frontend.
Imported 428 cell types from liberty file.

yosys> read_verilog good_mux.v

3. Executing Verilog-2005 frontend: good_mux.v
Parsing Verilog input from `good_mux.v' to AST representation.
Generating RTLIL representation for module `\good_mux'.
Successfully finished Verilog frontend.
```

Read_verilog -command to read the design

synth -top <name of the module> - This command runs the default synthesis script

-top -Use the specify as top module

```
yosys> synth -top good_mux

4. Executing SYNTH pass.

4.1. Executing HIERARCHY pass (managing design hierarchy).

4.1.1. Analyzing design hierarchy..
Top module:  \good_mux

4.1.2. Analyzing design hierarchy..
Top module:  \good_mux
Removed 0 unused modules.

4.2. Executing PROC pass (convert processes to netlists).

4.2.1. Executing PROC_CLEAN pass (remove empty switches from decision trees).
Cleaned up 0 empty switches.

4.2.2. Executing PROC_RMDEAD pass (remove dead branches from decision trees).
Marked 1 switch rules as full_case in process $proc$good_mux.v:3$1 in module good_mux.
Removed a total of 0 dead cases.

4.2.3. Executing PROC_PRUNE pass (remove redundant assignments in processes).
Removed 1 redundant assignment.
Promoted 0 assignments to connections.

4.2.4. Executing PROC_INIT pass (extract init attributes).
```

abc – This command uses the ABC tool for technology mapping of yosys internal gate library to a target architecture. The internal gate library consists of the specification of the gates about their area, timing, power consumption, etc.

```
yosys> abc -liberty ../my_lib/lib/sky130_fd_sc_hd__tt_025C_1v80.lib

6. Executing ABC pass (technology mapping using ABC).

6.1. Extracting gate netlist of module `\good_mux' to `<abc-temp-dir>/input.blif'..
Extracted 1 gates and 4 wires to a netlist network with 3 inputs and 1 outputs.

6.1.1. Executing ABC.
Running ABC command: <yosys-exe-dir>/yosys-abc -s -f <abc-temp-dir>/abc.script 2>&1
ABC: ABC command line: "source <abc-temp-dir>/abc.script".
ABC:
ABC: + read_blif <abc-temp-dir>/input.blif
ABC: + read_lib -w /home/anshi/VLSI/sky130RTLDesignAndSynthesisWorkshop/verilog_files/../my_lib/lib/sky130_fd_sc_hd__tt_025C_1v80.lib
ABC: Parsing finished successfully.  Parsing time =     0.16 sec
ABC: Scl_LibertyReadGenlib() skipped cell "sky130_fd_sc_hd__decap_12" without logic function.
ABC: Scl_LibertyReadGenlib() skipped cell "sky130_fd_sc_hd__decap_3" without logic function.
ABC: Scl_LibertyReadGenlib() skipped cell "sky130_fd_sc_hd__decap_4" without logic function.
ABC: Scl_LibertyReadGenlib() skipped cell "sky130_fd_sc_hd__decap_6" without logic function.
ABC: Scl_LibertyReadGenlib() skipped cell "sky130_fd_sc_hd__decap_8" without logic function.
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfbbn_1".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfbbn_2".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfbbp_1".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfrbp_1".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfrbp_2".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfrtn_1".
ABC: Scl_LibertyReadGenlib() skipped sequential cell "sky130_fd_sc_hd__dfrtp_1".
```

Write_verilog-Use to write the netlist.Netlist I s the representation of the design in the form of standard cells present in the .lib file.

```
yosys> write_verilog good_mux_netlist.v

8. Executing Verilog backend.
Dumping module `\good_mux'.
```

-noattr – With this command no attributes are included in the output

```
yosys> write_verilog -noattr good_mux_netlist.v

10. Executing Verilog backend.
Dumping module `\good_mux'.
```

In order to verify the netlist with simulation results we give the netlist file and the same test bench to the iverilog simulator.