Lab 3 - Implementation

This contains corea design => MBIST , EDT & OCC , SYNTHESIS , SCAN (with wrappers) , ATPG , SIMULATIONS

STEP 1 - MBIST INSERTION

```
set_context dft -rtl -design_id rtl1
set_tsdb_output_directory ../tsdb_outdir

read_cell_library ../../../library/adk.tcelllib
read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary

read_verilog ../design/corea.v

set_current_design corea

set_design_level physical_block

add_clocks clka -period 10ns
add_clocks clkb -period 13.2ns
add_clocks clkc -period 14ns

set_dft_specification_requirements -memory_test on

check_design_rules

#set_config_value /DefaultsSpecification(user)/DftSpecification/use_rtl_cells off

set spec [create_dft_spec]
report_config_dat $spec
process_dft_spec
extract_icl

set spec [create_patterns_spec]
report_config_data $spec
process_patterns_specification
set_simulation_library_sources -v ../../../library/adk.v -y ../../../library/mems/ -extension v
run_testbench_simulations
```

```
/home/vinayakp/aug23/level3/Lab3/corea/1.insert_mbist>>tessent -shell -dofile corea_dft_rtl1.tcl
//  Warning: Tessent user documentation not found
//  Tessent Shell  2021.1    Fri Feb 26 20:45:56 GMT 2021
//                Copyright 2011-2021 Mentor Graphics Corporation
//
//                            All Rights Reserved.
//
//    THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION WHICH
//  IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS AND IS
//                        SUBJECT TO LICENSE TERMS.
//
//  Mentor Graphics software executing under x86-64 Linux on Sun Feb 11 21:14:50 IST 2024.
//  64 bit version
//  Host: vlsiguru (64168 MB RAM, 32191 MB Swap)
//
//  command: set_context dft -rtl -design_id rtl1
//  command: set_tsdb_output_directory ../tsdb_outdir
//  command: read_cell_library ../../../library/adk.tcelllib
//  Reading DFT Library file ../../../library/adk.tcelllib
//  Finished reading file ../../../library/adk.tcelllib
//  command: read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary
//  command: read_verilog ../design/corea.v
//  command: set_current_design corea
//  command: set_design_level physical_block
//  command: add_clocks clka -period 10ns
//  command: add_clocks clkb -period 13.2ns
//  command: add_clocks clkc -period 14ns
//  command: set_dft_specification_requirements -memory_test on
//  command: check_design_rules
//  ------------------------------------------------------------------------------
//  Begin RTL synthesis.
..
//  Synthesized modules=1, Time=1.63 sec.
//  Note: There was 1 module selectively synthesized. There were also 5 sub-modules created by synthesis.
//        Use 'get_module -filter is_synthesized' to see them.
//        You can also use 'set_quick_synthesis_options -verbose on' to have the synthesis step report the
//        synthesized module name in the transcript as it is being synthesized.
//  ------------------------------------------------------------------------------
//  Warning: Rule FN4 violation occurs 2 times
//  Flattening process completed, cell instances=844, gates=4382, PIs=69, POs=64, CPU time=0.01 sec.
//  ------------------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.01 sec.
//  command: #set_config_value /DefaultsSpecification(user)/DftSpecification/use_rtl_cells off
//  command: set spec [create_dft_spec]
//  sub-command: create_dft_specification
//
//  Begin creation of DftSpecification(corea,rtl1)
//    Creation of RtlCells wrapper
//    Creation of IjtagNetwork wrapper
//    Creation of MemoryBist wrapper
//    Creation of MemoryBisr wrapper
//
//  Done  creation of DftSpecification(corea,rtl1)
//
//  command: report_config_data $spec
```

```
DftSpecification(corea,rtl1) {
  IjtagNetwork {
    HostScanInterface(ijtag) {
      Sib(sti) {
        Attributes {
          tessent_dft_function : scan_tested_instrument_host;
        }
        Sib(mbist) {
        }
      }
    }
  }
  MemoryBist {
    ijtag_host_interface : Sib(mbist);
    Controller(c1) {
      clock_domain_label : clka;
      Step {
        MemoryInterface(m1) {
          instance_name : ram1;
        }
      }
    }
    Controller(c2) {
      clock_domain_label : clkb;
      Step {
        MemoryInterface(m1) {
          instance_name : ram2;
        }
      }
    }
  }
}
```

```
//  Begin processing of /DftSpecification(corea,rtl1)
//    --- IP generation phase ---
//    Validation of IjtagNetwork
//    Validation of MemoryBist
//    Processing of RtlCells
//      Generating Verilog RTL Cells
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl1_cells.instrument/corea_rtl1_tessent_posedge_synchr
onizer_reset.v
//
//      Loading the generated RTL verilog files (1) to enable instantiating the contained modules
//      into the design.
//    Processing of IjtagNetwork
//      Generating design files for IJTAG SIB module corea_rtl1_tessent_sib_1
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.icl
//        TCD Scan    : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.tcd_scan
//        CTL         : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.ctl
//        TCD         : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.tcd
//        PDL         : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.pdl
//      Generating design files for IJTAG SIB module corea_rtl1_tessent_sib_2
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_2.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_2.icl
//
//      Loading the generated RTL verilog files (2) to enable instantiating the contained modules
//      into the design.
//    Processing of MemoryBist
//      Generating the IJTAG ICL for the memories.
//      Generating design files for MemoryBist Controller(c1)
//      Generating design files for MemoryBist Controller(c2)
//  Warning: There are warnings issued while generating design files for MemoryBist controller(s).
//           Review the messages in the following generation log files:
//           ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c1.generation_log,
//           ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c2.generation_log
//      Generating design files for Bist Access Port
```

```
//      Loading the generated RTL verilog files (5) to enable instantiating the contained modules
//      into the design.
//      Generating design files for MemoryBist controller assemblies
//    --- Instrument insertion phase ---
//   Inserting instruments of type 'ijtag'
//   Inserting instruments of type 'memory_bist'
//
//    Writing out modified source design in ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design
//    Writing out specification in ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_spec
//
//  Done  processing of DftSpecification(corea,rtl1)
//
//  command: extract_icl
//  Note: Updating the hierarchical data model to reflect RTL design changes.
//  Writing design source dictionary : ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.
design_source_dictionary
//  ------------------------------------------------------------------------
//  Begin RTL synthesis.
//  --------------------
//  Synthesized modules=1, Time=1.64 sec.
//  Note: There was 1 module selectively synthesized. There were also 5 sub-modules created by synthesis.
//        Use 'get_module -filter is_synthesized' to see them.
//        You can also use 'set_quick_synthesis_options -verbose on' to have the synthesis step report the
//        synthesized module name in the transcript as it is being synthesized.
//  ------------------------------------------------------------------------
//  Warning: Rule FN1 violation occurs 1562 times
//  Warning: Rule FN4 violation occurs 8 times
//  Warning: Rule FP13 violation occurs 26 times
//  Flattening process completed, cell instances=961, gates=4817, PIs=76, POs=65, CPU time=0.06 sec.
//  ------------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.00 sec.
//


//  Begin ICL extraction.
//  ---------------------
//  ICL extraction completed, ICL instances=9, CPU time=0.09 sec.
//  ------------------------------------------------------------------------
//  ------------------------------------------------------------------------
//  Begin ICL elaboration and checking.
//  -----------------------------------
//  ICL elaboration completed, CPU time=0.07 sec.
//  ------------------------------------------------------------------------
//  Writing ICL file : ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.icl
//  Writing consolidated PDL file: ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.pdl

//  Writing SDC file: ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.sdc
//  Writing DFT info dictionary: ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.dft_in
fo_dictionary
//  command: set spec [create_patterns_spec]
//  sub-command: create_patterns_specification
//  Creating '/PatternsSpecification(corea,rtl1,signoff)'
//    Getting patterns specifications for the 'ijtag' instrument type
//    Getting patterns specifications for the 'memory_bist' instrument type
//  command: report_config_data $spec
```

```
PatternsSpecification(corea,rtl1,signoff) {
  Patterns(ICLNetwork) {
    ICLNetworkVerify(corea) {
    }
  }
  Patterns(MemoryBist_P1) {
    ClockPeriods {
      clkb : 13.2ns;
      clka : 10.0ns;
    }
    TestStep(run_time_prog) {
      MemoryBist {
        run_mode : run_time_prog;
        reduced_address_count : on;
        Controller(corea_rtl1_tessent_mbist_c1_controller_inst) {
          DiagnosisOptions {
            compare_go : on;
            compare_go_id : on;
          }
        }
        Controller(corea_rtl1_tessent_mbist_c2_controller_inst) {
          DiagnosisOptions {
            compare_go : on;
            compare_go_id : on;
          }
        }
      }
    }
  }


  Patterns(MemoryBist_ParallelRetentionTest_P1) {
    ClockPeriods {
      clkb : 13.2ns;
      clka : 10.0ns;
    }
    TestStep(ParallelRetentionTest) {
      MemoryBist {
        run_mode : hw_default;
        parallel_retention_time : 0;
        reduced_address_count : on;
        Controller(corea_rtl1_tessent_mbist_c1_controller_inst) {
          parallel_retention_group : 1;
          DiagnosisOptions {
            compare_go_id : on;
          }
        }
        Controller(corea_rtl1_tessent_mbist_c2_controller_inst) {
          parallel_retention_group : 1;
          DiagnosisOptions {
            compare_go_id : on;
          }
        }
      }
    }
  }
}
```

```
//  command: process_patterns_specification
//
//  Begin processing of /PatternsSpecification(corea,rtl1,signoff)
//
//    Processing of /PatternsSpecification(corea,rtl1,signoff)/Patterns(ICLNetwork)
//
//      Creation of pattern 'ICLNetwork'
//        Solving ICLNetworkVerify(corea)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl1.patterns_signoff/ICLNetwork.v'
//
//    Processing of /PatternsSpecification(corea,rtl1,signoff)/Patterns(MemoryBist_P1)
//      Processing of TestStep(run_time_prog) instrument 'memory_bist'
//
//      Creation of pattern 'MemoryBist_P1'
//        Solving TestStep(run_time_prog)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl1.patterns_signoff/MemoryBist_P1.v'
//
//    Processing of /PatternsSpecification(corea,rtl1,signoff)/Patterns(MemoryBist_ParallelRetentionTest_P1)
//      Processing of TestStep(ParallelRetentionTest) instrument 'memory_bist'
//
//      Creation of pattern 'MemoryBist_ParallelRetentionTest_P1'
//        Solving TestStep(ParallelRetentionTest)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl1.patterns_signoff/MemoryBist_ParallelRetentionTe
st_P1.v'
//      Writing simulation data dictionary file '../tsdb_outdir/patterns/corea_rtl1.patterns_signoff/simulation.
data_dictionary'
//
//  Done  processing of /PatternsSpecification(corea,rtl1,signoff)
//
//  Writing configuration data file '../tsdb_outdir/patterns/corea_rtl1.patterns_spec_signoff'.
//  command: set_simulation_library_sources -v ../../../library/adk.v -y ../../../library/mems/ -extension v
//  command: run_testbench_simulations
Starting 3 simulations for ./simulation_outdir/corea_rtl1.simulation_signoff
//  Waiting for the simulation(s) to complete
```

STEP 2 - EDT INSERTION

```
set_context dft -rtl -design_id rtl2
set_tsdb_output_directory ../tsdb_outdir

read_cell_library ../../../library/adk.tcelllib
read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary


## This will automatically read the modified/newly created rtls of mbist + functional
read_design corea -design_id rtl1
set_current_design corea

## logic_test -> to enable edt/occ insertion
set_dft_specification_requirements -logic_test on

## DFT Signals , source node -> port name in the design
add_dft_signal memory_bypass_en tck_occ_en
add_dft_signal ltest_en int_mode ext_mode
add_dft_signal scan_en test_clock edt_update -source_node {scan_en test_clock edt_update}
add_dft_signal edt_clock shift_capture_clock -create_from_other_signals
add_dft_signal int_ltest_en ext_ltest_en

check_design_rules

## in_wrapper is related to DftSpecification and not related to scan_wrapper concept
set spec [create_dft_spec -sri_sib_list {occ edt}]
## Explanation for below parameters used in EDT config:
     ## longest_chain_range  -> min , max range of chain length
     ## scan_chain_count     -> number of internal scan chains
     ## input_channel_count  -> number of external scan input channels
     ## output_channel_count -> number of external scan output channels

read_config_data -in_wrapper $spec -from_string {
  Occ {
    ijtag_host_interface : Sib(occ);
    Controller(clka) {
       clock_intercept_node : clka;
    }
    Controller(clkb) {
       clock_intercept_node : clkb;
    }
    Controller(clkc) {
      clock_intercept_node : clkc;
    }
  }
  Edt {
    ijtag_host_interface : Sib(edt);
    Controller(c1) {
      longest_chain_range            : 10, 50;
      scan_chain_count               : 15;
      input_channel_count            : 2;
      output_channel_count           : 2;
    }
  }
}


## we are not declaring pre-existing channels port names to edt here, this will done in level4 projects.

report_config_data $spec
process_dft_specification
extract_icl
stop
## This create_pattern_spec/process_pattern_spec is not going to generate any patterns for scan/edt/occ since scan insertion is next step. So, the below commands will only re-generate the patterns related to MBIST and JTAG network ( The reason for this : New Sib modules are added in this run)
set spec [create_patterns_spec]
report_config_data $spec
process_patterns_specification
set_simulation_library_sources -v ../../../library/adk.v -y ../../../library/mems/ -extension v

run_testbench_simulations

write_design_import_script -use_relative_path_to . -replace
```

Line 1: with this design id a directory will be formed into a tsdb output directory, so there will be dft inserted designs, instruments and patterns directory with each of those another subdirectory will be formed called as rtl2

Line 9: rtl1 would actually be the design id of mbist insertion of lab3

```
/home/vinayakp/aug23/level3/Lab3/corea/2.insert_edt_occ>>tessent -shell -dofile corea_dft_rtl2.tcl
//  Warning: Tessent user documentation not found
//  Tessent Shell  2021.1    Fri Feb 26 20:45:56 GMT 2021
//                Copyright 2011-2021 Mentor Graphics Corporation
//
//                           All Rights Reserved.
//
//    THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION WHICH
//  IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS AND IS
//                        SUBJECT TO LICENSE TERMS.
//
//  Mentor Graphics software executing under x86-64 Linux on Mon Dec 25 01:36:20 IST 2023.
//  64 bit version
//  Host: vlsiguru (64168 MB RAM, 32191 MB Swap)
//

//  mtnslogicbist_c license will expire in 2 days...
//  command: set_context dft -rtl -design_id rtl2

//  mtijtag_c license will expire in 2 days...
//  command: set_tsdb_output_directory ../tsdb_outdir
//  command: read_cell_library ../../../library/adk.tcelllib
//  Reading DFT Library file ../../../library/adk.tcelllib
//  Finished reading file ../../../library/adk.tcelllib
//  command: read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary
//  command: stop
//  Error: Command 'stop' is unknown
// 'DOFile corea_dft_rtl2.tcl' aborted at line 7
SETUP> set_context dft -rtl -design_id rtl2
SETUP> set_context dft -rtl -design_id rtl2 -verbose
//  Error: Unrecognized argument '-verbose'. ( help set_context )
SETUP> set_context dft -rtl -design_id rtl1
SETUP> set_context dft -rtl -design_id rtl1 -verbose
//  Error: Unrecognized argument '-verbose'. ( help set_context )
SETUP> read_design corea -design_id rtl1
SETUP> read_design corea -design_id rtl1 -verbose
//  sub-command: set_tool_options -reapply_settings_after_reelaboration On
//  sub-command: set_read_design_tag corea
//  sub-command: #### read1 #####

set_design_include_directories \
   {../1.insert_mbist}

set_design_macros -clear

set_design_sources -format verilog -clear
read_verilog \
   {../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/modified_rtl_files/corea.v} \
   -in_library work \
   -format 2001 \
   -no_duplicate_modules_warnings


//  sub-command: #### read2 #####

set_design_include_directories \
   {../1.insert_mbist}

set_design_macros -clear

set_design_sources -format verilog -clear
```

```
read_verilog \
   {../tsdb_outdir/instruments/corea_rtl1_cells.instrument/corea_rtl1_tessent_posedge_synchronizer_reset.v} \
   -in_library work \
   -format 2001 \
   -no_duplicate_modules_warnings


//  sub-command: #### read3 #####

set_design_include_directories \
   {../1.insert_mbist}

set_design_macros -clear

set_design_sources -format verilog -clear
read_verilog \
   {../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.v
    ../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_2.v} \
   -in_library work \
   -format 2001 \
   -no_duplicate_modules_warnings


//  sub-command: #### read4 #####

set_design_include_directories \
   {../1.insert_mbist}

set_design_macros -clear

set_design_sources -format verilog -clear
read_verilog \
   {../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_bap.v
    ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c1_controller.v
    ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c1_interface_m1.v
    ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c2_controller.v
    ../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c2_interface_m1.v} \
   -in_library work \
   -format 2001 \
   -no_duplicate_modules_warnings


//  sub-command: set_read_design_tag ""
//  sub-command: set_design_sources -format verilog -clear
//  sub-command: set_design_include_directories -clear
//  sub-command: set_design_macros -clear
//  sub-command: set_tool_options -reapply_settings_after_reelaboration Off
//  sub-command: read_icl ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea.icl -skip_chi
ld_blocks  -no_notes
//  sub-command: read_core_descriptions ../tsdb_outdir/dft_inserted_designs/corea_rtl1.dft_inserted_design/corea
.tcd
```

- From read 1, it will read the modified netlist so in corea.v file has only module instantiation of SIB, TDR, Controller, BAP and Interface
- From read 2, the tool will read the posedge synchronizer reset as well.
- From read 3, the module definition will available in the instruments directory.
- And from corea level icl file and tcd file

Line 10 : Design corea that we are doing the elaboration

```
SETUP> set_current_design corea
//  ------------------------------------------------------------------------
//  Begin ICL elaboration and checking.
//  ----------------------------------
//  ICL elaboration completed, CPU time=0.09 sec.
//  ------------------------------------------------------------------------
```

Line 13: For EDT and OCC Insertion we have to give as -logic_test on

```
SETUP> set_dft_specification_requirements -logic_test on

//  mtscanpro_c license will expire in 2 days...
```

Line 15 to 19: This static signals will be controlled using TDR

- add_dft_signal <signal_name> : It will insert TDR logic for specified signal.

Note : TDR can be inserted only for static signals (Example TE)

Example :

tck_occ_en : A global DFT control signal that is used to enable the mini-OCC present inside the Sib(sti) node.

ltest_en : A logic test control signal that is used to enable the logic test mode. This signal is force high during all logic test modes.

memory_bypass_en: To bypass memories. This signal is set to 1 by default during logic test

add_dft_signal <signal name> -source node <top_level_port name>

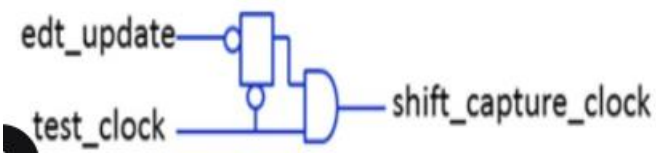
For dynamic signals, we can't insert TDR. So it has to controlled from the top level.

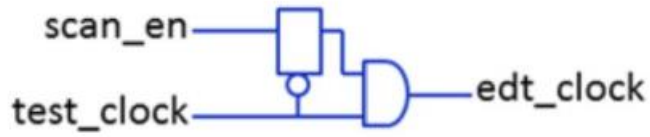
Scanen, test_clock, edt_update will be controlled from top level port

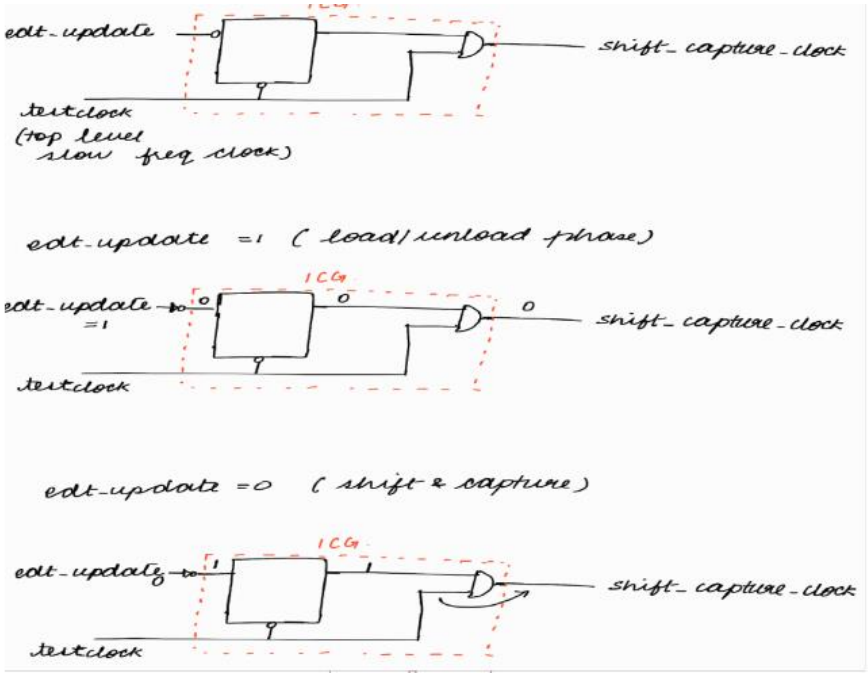
Edt_clock is the clock which will be going to the flops in the edt logic that is when the edt logic has to be pulsed, it has to be pulsed during load and unload and the shift phase. Shift_capture_clock it is the clock which will be going to the scan flops, so this has to be pulsed during shift and catpure phase. So it will be created from top level test clock. The tool will add some one ICG logic when the edt_clock and another ICG logic to shift capture which has to be pulsed.

Command :

add_dft_signal edt_clock shift_capture_clock -create_from_other_signals

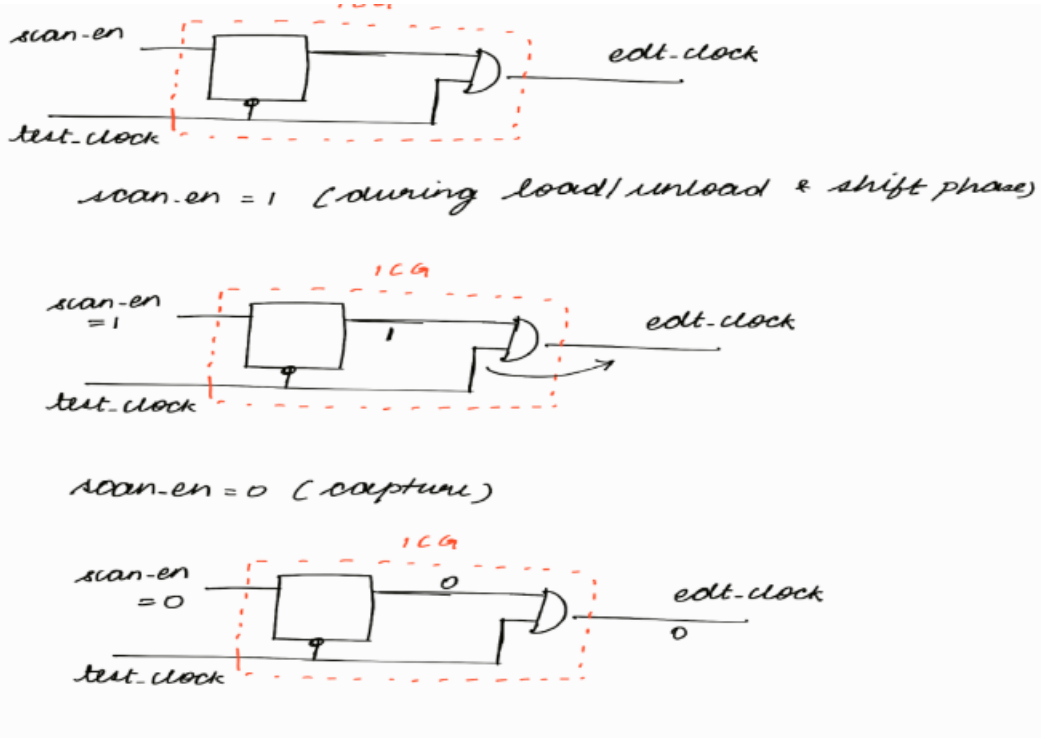




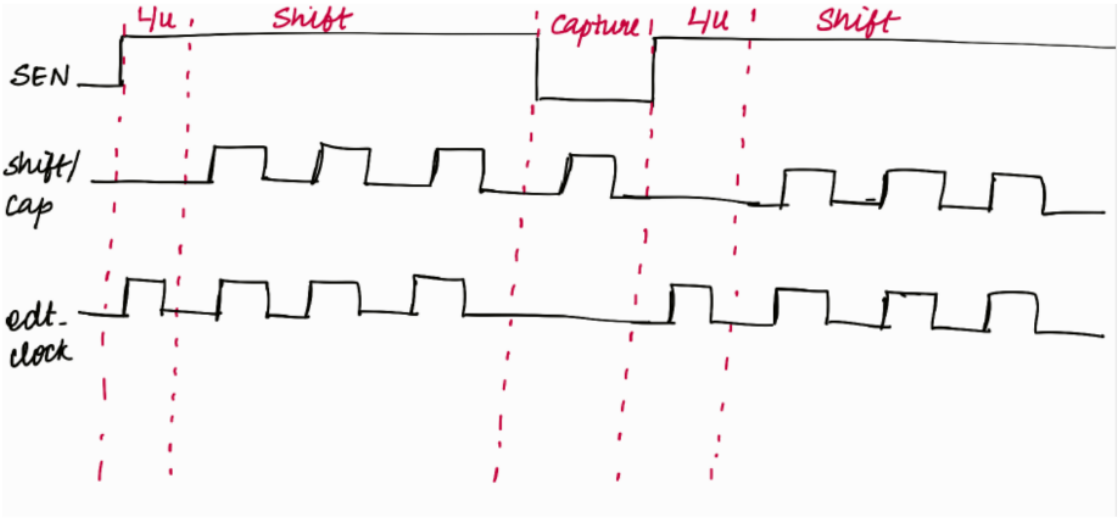


Load and unload will not pulsed during load and unload phase when edt update = 1

Shift and capture will be pulsed during shift and capture phase when edt update = 0, and it will go to the scan clocks, so testclock will be coming out as 1 through and gate.



EDT Clock will be pulsed during load and unload phase & shift phase when SE=1
During capture phase SE = 0 so EDT Clock will not be pulsed during capture phase.




```
SETUP> add_dft_signal memory_bypass_en tck_occ_en
SETUP> add_dft_signal ltest_en int_mode ext_mode
SETUP> add_dft_signal scan_en test_clock edt_update -source_node {scan_en test_clock edt_update}
SETUP> add_dft_signal edt_clock shift_capture_clock -create_from_other_signals
SETUP> add_dft_signal int_ltest_en ext_ltest_en
```

Line 22: checking the DRC's

```
SETUP> check_design_rules
//  ------------------------------------------------------------------------------
//  Begin RTL synthesis.
//  --------------------
//  Synthesized modules=29, Time=5.60 sec.
//  ------------------------------------------------------------------------------
//  Warning: Rule FN4 violation occurs 8 times
//  Flattening process completed, cell instances=29422, gates=45246, PIs=76+1(pseudo port), POs=65, CPU time=0.2
5 sec.
//  ------------------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.09 sec.
//  Begin simulation of auto-generated load_unload procedure.
//  Simulation of load_unload procedure completed, CPU time=0.0 sec.
//  Scan segment = /corea_rtl1_tessent_sib_sti_inst/ltest_so successfully traced with scan_cells = 7.
//  7 scan cells have been identified in 1 scan segment.
//  Warning: 1 edge-triggered clock ports set to stable high. (D7)
//  Note: Adding DFT signal 'async_set_reset_static_disable' with the -create_with_tdr option.
//  Note: Adding DFT signal 'async_set_reset_dynamic_disable' with the -create_from_other_signals option.
//  Note: There were 4 'add_dft_control_points -type async_set_reset' commands inferred from DRC. Use report_dft
 control points to see them.
```

Line 25 to 54:

```
read_config_data -in_wrapper $spec -from_string {
  Occ {
    ijtag_host_interface : Sib(occ);
    Controller(clka) {
       clock_intercept_node : clka;
    }
    Controller(clkb) {
       clock_intercept_node : clkb;
    }
    Controller(clkc) {
      clock_intercept_node : clkc;
    }
  }
  Edt {
    ijtag_host_interface : Sib(edt);
    Controller(c1) {
      longest_chain_range           : 10, 50;
      scan_chain_count              : 15;
      input_channel_count           : 2;
      output_channel_count          : 2;
    }
  }
}
```

- Clock_intercept_node means where we are going to place the OCC.

So, one occ at clka, 2nd occ at clkb and 3rd occ at clkc.
And here OCC will be controlled by SIB OCC.

```
ANALYSIS> ## Explanation for below parameters used in EDT config:
ANALYSIS>      ## longest_chain_range  -> min , max range of chain length
ANALYSIS>      ## scan_chain_count     -> number of internal scan chains
ANALYSIS>      ## input_channel_count  -> number of external scan input channels
ANALYSIS>      ## output_channel_count -> number of external scan output channels
ANALYSIS>
ANALYSIS> read_config_data -in_wrapper $spec -from_string {
>   Occ {
>     ijtag_host_interface : Sib(occ);
>     Controller(clka) {
>        clock_intercept_node : clka;
>     }
>     Controller(clkb) {
>        clock_intercept_node : clkb;
>     }
>     Controller(clkc) {
>       clock_intercept_node : clkc;
>     }
>   }
>   Edt {
>     ijtag_host_interface : Sib(edt);
>     Controller(c1) {
>       longest_chain_range           : 10, 50;
>       scan_chain_count              : 15;
>       input_channel_count           : 2;
>       output_channel_count          : 2;
>     }
>   }
> }
```

Line 59: Whatever we have given from line 32 to 54, same it will show in the terminal as well.

```
ANALYSIS> report_config_data $spec

DftSpecification(corea,rtl2) {
  IjtagNetwork {
    HostScanInterface(sri) {
      Interface {
        design_instance : corea_rtl1_tessent_sib_sti_inst;
        scan_interface : client;
      }
      Sib(sri) {
        Attributes {
          tessent_dft_function : scan_resource_instrument_host;
        }
        Tdr(sri_ctrl) {
          Attributes {
            tessent_dft_function : scan_resource_instrument_dft_control;
          }
        }
        Sib(occ) {
        }
        Sib(edt) {
        }
      }
    }
  }
  OCC {
    ijtag_host_interface : Sib(occ);
    Controller(clka) {
      clock_intercept_node : clka;
    }
    Controller(clkb) {
      clock_intercept_node : clkb;
    }
    Controller(clkc) {
      clock_intercept_node : clkc;
    }
  }
  EDT {
    ijtag_host_interface : Sib(edt);
    Controller(c1) {
      longest_chain_range : 10, 50;
      scan_chain_count : 15;
      input_channel_count : 2;
      output_channel_count : 2;
    }
  }
}
```

- But additionally 2 new SIBs are added i.e. SIB OCC and SIB EDT.
- SIB OCC will control the OCC logic and SIB EDT will control the EDT logic.

Line 60: process_dft_specification, the tool will validate all the test HW which was generated and it will insert the HW into our design and it will also write out all the output files into the tsdb output directory and individually the icl decription for all the sibs and tdrs.

```
ANALYSIS> process_dft_specification
//
//  Begin processing of /DftSpecification(corea,rtl2)
//    --- IP generation phase ---

//  mttestkompress_c license will expire in 2 days...
//    Validation of IjtagNetwork
//    Validation of OCC
//    Validation of EDT
//    Processing of RtlCells
//      Generating Verilog RTL Cells
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_cells.instrument/corea_rtl2_tessent_posedge_synchr
onizer_reset.v
//
//      Loading the generated RTL verilog files (1) to enable instantiating the contained modules
//      into the design.
//    Processing of IjtagNetwork
//      Generating design files for IJTAG SIB module corea_rtl2_tessent_sib_1
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_1.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_1.icl
//      Generating design files for IJTAG SIB module corea_rtl2_tessent_sib_2
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_2.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_2.icl
//      Generating design files for IJTAG Tdr module corea_rtl2_tessent_tdr_sri_ctrl
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_tdr_sri_ctrl.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_tdr_sri_ctrl.i
cl
//
//      Loading the generated RTL verilog files (3) to enable instantiating the contained modules
//      into the design.
//    Processing of OCC
//      Generating design files for OCC module corea_rtl2_tessent_occ
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.icl
//        IJTAG PDL   : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.pdl
//        TCD         : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.tcd
//        TCD Scan    : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.tcd_scan
//        CTL         : ../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.ctl
//
//      Loading the generated RTL verilog files (1) to enable instantiating the contained modules
//      into the design.
//    Processing of EDT
//      Generating design files for EDT Tdr module corea_rtl2_tessent_edt_c1_tdr
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1_tdr.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1_tdr.icl
//      Generating design files for EDT module corea_rtl2_tessent_edt_c1
//        Verilog RTL : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1.v
//        IJTAG ICL   : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1.icl
//        IJTAG PDL   : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1.pdl
//        TCD         : ../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1.tcd
//
//      Loading the generated RTL verilog files (2) to enable instantiating the contained modules
//      into the design.
//    --- Instrument insertion phase ---
//   Inserting instruments of type 'ijtag'
//   Inserting instruments of type 'occ'
//   Inserting instruments of type 'edt'
//
//    Writing out modified source design in ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design
//    Writing out specification in ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_spec
//
//  Done  processing of DftSpecification(corea,rtl2)
//
/DftSpecification(corea,rtl2)
```

Line 61: The Corea ICL file will be generated. Why we are generating it? Because in the future runs the tool will be understand the connectivity from the top level.

```
INSERTION> extract_icl
//  Note: Updating the hierarchical data model to reflect RTL design changes.
//  Writing design source dictionary : ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.
design_source_dictionary

//  mtijtag_c license will expire in 2 days...
//  ------------------------------------------------------------------------
//  Begin RTL synthesis.
//  --------------------
//  Synthesized modules=1, Time=2.22 sec.
//  Note: There was 1 module selectively synthesized. There were also 5 sub-modules created by synthesis.
//        Use 'get_module -filter is_synthesized' to see them.
//        You can also use 'set_quick_synthesis_options -verbose on' to have the synthesis step report the
//        synthesized module name in the transcript as it is being synthesized.
//  ------------------------------------------------------------------------
//  Warning: Rule FN1 violation occurs 1790 times
//  Warning: Rule FN4 violation occurs 19 times
//  Warning: Rule FP13 violation occurs 28 times
//  Flattening process completed, cell instances=1051, gates=5088, PIs=80, POs=67, CPU time=0.07 sec.
//  ------------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.00 sec.
//  ------------------------------------------------------------------------
//  Begin ICL extraction.
//  ---------------------
//  ICL extraction completed, ICL instances=18, CPU time=0.12 sec.
//  ------------------------------------------------------------------------
//  ------------------------------------------------------------------------
//  Begin ICL elaboration and checking.
//  -----------------------------------
//  ICL elaboration completed, CPU time=0.10 sec.
//  ------------------------------------------------------------------------
//  ------------------------------------------------------------------------
//  Begin ICL elaboration and checking.
//  -----------------------------------
//  ICL elaboration completed, CPU time=0.10 sec.
//  ------------------------------------------------------------------------
//  Writing ICL file : ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.icl
//  Writing consolidated PDL file: ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.pdl

//  Writing SDC file: ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.sdc
//  Writing DFT info dictionary: ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.dft_in
fo dictionary
```

Line 64: Patterns Specs will be written for MBIST and IJTAG only. It will not generate the patterns for EDT and OCC, functionality of the EDT and OCC it will tested during the ATPG time. So here we are regenerating the patterns during the MBIST and IJTAG Network.

- **Why we are doing it? Because additionally we are added few more SIBs, one SIB for EDT logic, one SIB for OCC logic, so as of new SIB modules are added into our design we are again validating our MBIST and IJTAG patterns and we are checking whether it is functioning properly or not.**

```
SETUP> set spec [create_patterns_spec]
//  sub-command: create_patterns_specification
//  Creating '/PatternsSpecification(corea,rtl2,signoff)'
//    Getting patterns specifications for the 'ijtag' instrument type
//    Getting patterns specifications for the 'memory_bist' instrument type
/PatternsSpecification(corea,rtl2,signoff)
```

```
SETUP> report_config_data $spec

PatternsSpecification(corea,rtl2,signoff) {
  AdvancedOptions {
    ConstantPortSettings {
      scan_en : 0;
    }
  }
  Patterns(ICLNetwork) {
    ICLNetworkVerify(corea) {
    }
  }
  Patterns(MemoryBist_P1) {
    ClockPeriods {
      clkb : 13.2ns;
      clka : 10.0ns;
    }
    TestStep(run_time_prog) {
      MemoryBist {
        run_mode : run_time_prog;
        reduced_address_count : on;
        Controller(corea_rtl1_tessent_mbist_c1_controller_inst) {
          DiagnosisOptions {
            compare_go : on;
            compare_go_id : on;
          }
        }
        Controller(corea_rtl1_tessent_mbist_c2_controller_inst) {
          DiagnosisOptions {
            compare_go : on;
            compare_go_id : on;
          }
        }
      }
    }
  }
  Patterns(MemoryBist_ParallelRetentionTest_P1) {
    ClockPeriods {
      clkb : 13.2ns;
      clka : 10.0ns;
    }
    TestStep(ParallelRetentionTest) {
      MemoryBist {
        run_mode : hw_default;
        parallel_retention_time : 0;
        reduced_address_count : on;
        Controller(corea_rtl1_tessent_mbist_c1_controller_inst) {
          parallel_retention_group : 1;
          DiagnosisOptions {
            compare_go_id : on;
          }
        }
        Controller(corea_rtl1_tessent_mbist_c2_controller_inst) {
          parallel_retention_group : 1;
          DiagnosisOptions {
            compare_go_id : on;
          }
        }
      }
    }
  }
}
```

Line 65: The tool will write out all the testbench file and pattern file into tsdb out directory

```
SETUP> process_patterns_specification

//  mttsmemorybist_c license will expire in 2 days...
//
//  Begin processing of /PatternsSpecification(corea,rtl2,signoff)
//
//    Processing of /PatternsSpecification(corea,rtl2,signoff)/Patterns(ICLNetwork)
//
//      Creation of pattern 'ICLNetwork'
//        Solving ICLNetworkVerify(corea)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl2.patterns_signoff/ICLNetwork.v'
//
//    Processing of /PatternsSpecification(corea,rtl2,signoff)/Patterns(MemoryBist_P1)
//      Processing of TestStep(run_time_prog) instrument 'memory_bist'
//
//      Creation of pattern 'MemoryBist_P1'
//        Solving TestStep(run_time_prog)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl2.patterns_signoff/MemoryBist_P1.v'
//
//    Processing of /PatternsSpecification(corea,rtl2,signoff)/Patterns(MemoryBist_ParallelRetentionTest_P1)
//      Processing of TestStep(ParallelRetentionTest) instrument 'memory_bist'
//
//      Creation of pattern 'MemoryBist_ParallelRetentionTest_P1'
//        Solving TestStep(ParallelRetentionTest)
//
//      Writing pattern file '../tsdb_outdir/patterns/corea_rtl2.patterns_signoff/MemoryBist_ParallelRetentionTe
st_P1.v'
//      Writing simulation data dictionary file '../tsdb_outdir/patterns/corea_rtl2.patterns_signoff/simulation.
//
//  Done  processing of /PatternsSpecification(corea,rtl2,signoff)
//
//  Writing configuration data file '../tsdb_outdir/patterns/corea_rtl2.patterns_spec_signoff'.
SETUP>
```

Line 66: the libraries which is carried out for simulation that we are reading the adk.v
So, -v for file and -y stands for directory

```
SETUP> set_simulation_library_sources -v ../../../library/adk.v -y ../../../library/mems/ -extension v
SETUP> run_testbench_simulations
Starting 3 simulations for ./simulation_outdir/corea_rtl2.simulation_signoff
//  Waiting for the simulation(s) to complete

unscheduled 0 queued 0 running 0 pass 3 fail 0
```

Line 70: It will write out a file which has list of all the files which is required to do the synthesis.

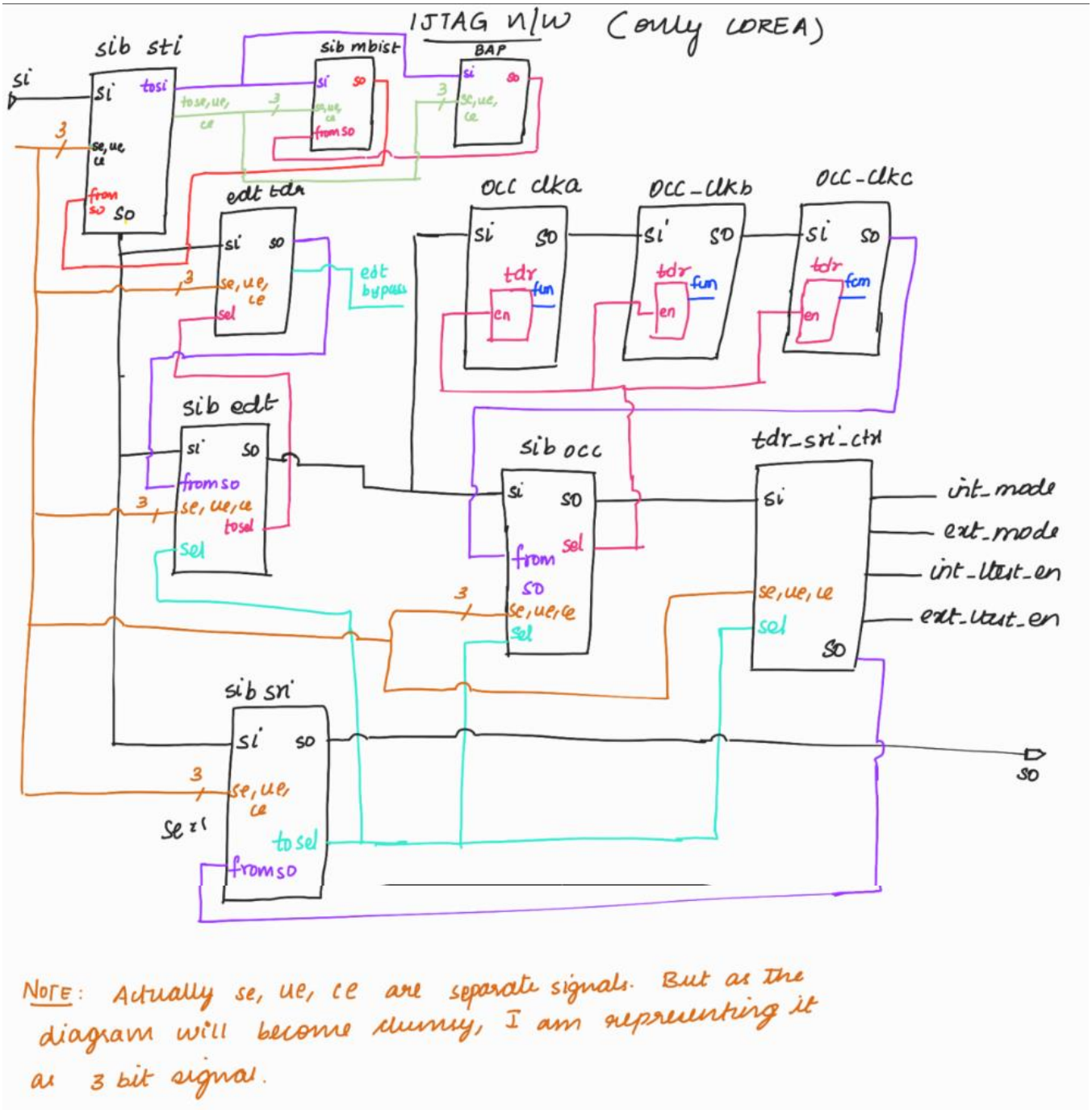
- **So in order to create the file which contains input file of synthesis run.**

```
SETUP> write_design_import_script -use_relative_path_to . -replace
//  Writing file 'corea.dc_shell_import_script'.
```

- So -relative path to means all the paths written in that file will be in relative path in the current directory.

- **Design Import script**

```
set_app_var hdlin_enable_upf_compatible_naming true
set_app_var hdlin_mux_size_only 2

# From file ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.design_source_dictionary

analyze -format verilog -library work \
      -vcs [list  \
        +incdir+{.} \
        +incdir+{../1.insert_mbist} ] { \
      "../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/modified_rtl_files/corea.v" \
   }

analyze -format verilog -library work \
      -vcs [list  \
        +incdir+{.} \
        +incdir+{../1.insert_mbist} ] { \
      "../tsdb_outdir/instruments/corea_rtl1_cells.instrument/corea_rtl1_tessent_posedge_synchronizer_reset.v" \
   }

analyze -format verilog -library work \
      -vcs [list  \
        +incdir+{.} \
        +incdir+{../1.insert_mbist} ] { \
      "../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_1.v" \
      "../tsdb_outdir/instruments/corea_rtl1_ijtag.instrument/corea_rtl1_tessent_sib_2.v" \
   }

analyze -format verilog -library work \
      -vcs [list  \
        +incdir+{.} \
        +incdir+{../1.insert_mbist} ] { \
      "../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_bap.v" \
      "../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c1_controller.v" \
       "../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c1_interface_m1.v" \
       "../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c2_controller.v" \
       "../tsdb_outdir/instruments/corea_rtl1_mbist.instrument/corea_rtl1_tessent_mbist_c2_interface_m1.v" \
    }

analyze -format verilog -library work \
       -vcs [list  \
         +incdir+{.} ] { \
       "../tsdb_outdir/instruments/corea_rtl2_cells.instrument/corea_rtl2_tessent_posedge_synchronizer_reset.
v" \
    }

analyze -format verilog -library work \
       -vcs [list  \
         +incdir+{.} ] { \
       "../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_1.v" \
       "../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_sib_2.v" \
       "../tsdb_outdir/instruments/corea_rtl2_ijtag.instrument/corea_rtl2_tessent_tdr_sri_ctrl.v" \
    }

analyze -format verilog -library work \
       -vcs [list  \
         +incdir+{.} ] { \
       "../tsdb_outdir/instruments/corea_rtl2_occ.instrument/corea_rtl2_tessent_occ.v" \
    }

analyze -format verilog -library work \
      -vcs [list  \
        +incdir+{.} ] { \
      "../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1.v" \
      "../tsdb_outdir/instruments/corea_rtl2_edt.instrument/corea_rtl2_tessent_edt_c1_tdr.v" \
   }
```




```
set_attribute_value [get_module -of_type design] -name synthesize_before_analysis
set_attribute_value qcsram* -name preserve_boundary -silent
set_attribute_value qcrf* -name preserve_boundary -silent
set_system_mode analysis
```

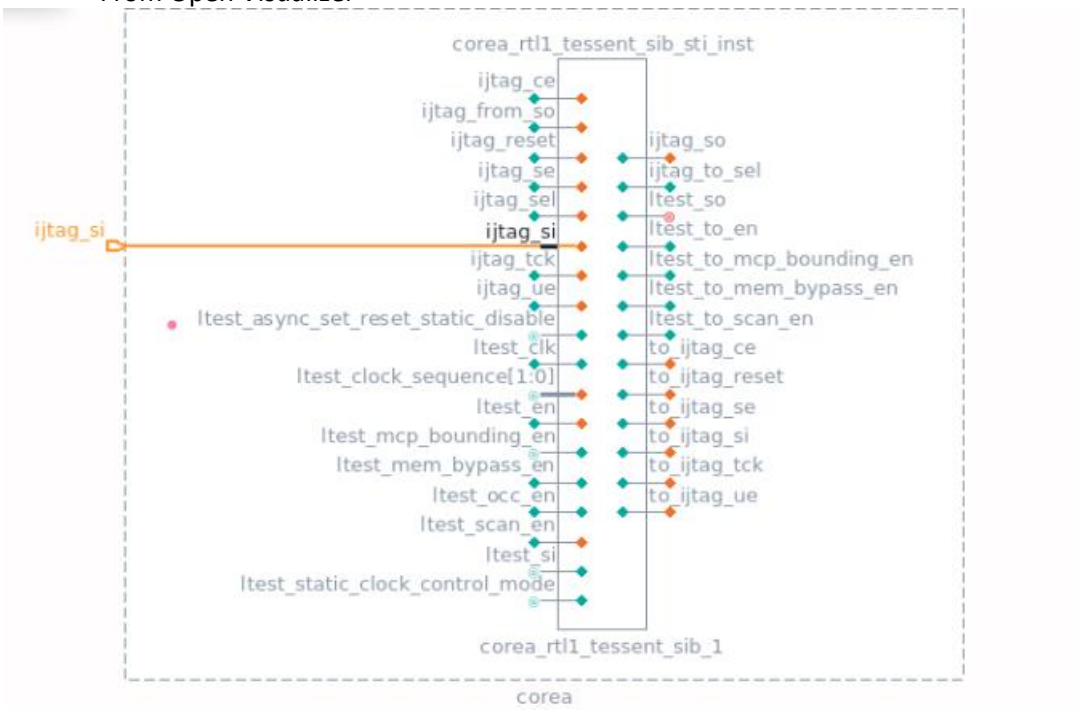
- In the newer version of tool, before if we start trace any logic in the visualizer we can use this additional command otherwise tool will trace only till certain amount but not further.

```
SETUP> set_attribute_value [get_module -of_type design] -name synthesize_before_analysis
{SYNC_1R1W_16x8 corea corea_rtl1_tessent_mbist_bap corea_rtl1_tessent_mbist_bap_ctl_sib corea_rtl1_tessent_m
_bap_sib corea_rtl1_tessent_mbist_bap_tdr corea_rtl1_tessent_mbist_c1_controller corea_rtl1_tessent_mbist_c1
troller_add_format corea_rtl1_tessent_mbist_c1_controller_add_gen corea_rtl1_tessent_mbist_c1_controller_ctl
p corea_rtl1_tessent_mbist_c1_controller_data_gen corea_rtl1_tessent_mbist_c1_controller_fsm corea_rtl1_tess
mbist_c1_controller_pointer_cntrl corea_rtl1_tessent_mbist_c1_controller_repeat_loop_cntrl corea_rtl1_tessen
ist_c1_controller_signal_gen corea_rtl1_tessent_mbist_c1_interface_m1 corea_rtl1_tessent_mbist_c1_interface_
TATUS corea_rtl1_tessent_mbist_c2_controller corea_rtl1_tessent_mbist_c2_controller_add_format corea_rtl1_te
t_mbist_c2_controller_add_gen corea_rtl1_tessent_mbist_c2_controller_ctl_comp corea_rtl1_tessent_mbist_c2_co
ller_data_gen corea_rtl1_tessent_mbist_c2_controller_fsm corea_rtl1_tessent_mbist_c2_controller_pointer_cntr
rea_rtl1_tessent_mbist_c2_controller_repeat_loop_cntrl corea_rtl1_tessent_mbist_c2_controller_signal_gen cor
tl1_tessent_mbist_c2_interface_m1 corea_rtl1_tessent_mbist_c2_interface_m1_STATUS corea_rtl1_tessent_posedge
chronizer_reset corea_rtl1_tessent_sib_1 corea_rtl1_tessent_sib_2 corea_rtl2_tessent_edt_c1 corea_rtl2_tesse
dt_c1_bypass_logic corea_rtl2_tessent_edt_c1_compactor corea_rtl2_tessent_edt_c1_controller corea_rtl2_tesse
dt_c1_decompressor corea_rtl2_tessent_edt_c1_onehot_decoder_3_to_7 corea_rtl2_tessent_edt_c1_onehot_decoder_
_8 corea_rtl2_tessent_edt_c1_spatial_compactor_7_w_output_lockup corea_rtl2_tessent_edt_c1_spatial_compactor
_output_lockup corea_rtl2_tessent_edt_c1_tdr corea_rtl2_tessent_edt_c1_xor_decoder corea_rtl2_tessent_occ co
rtl2_tessent_occ_control corea_rtl2_tessent_occ_shift_reg corea_rtl2_tessent_occ_sib corea_rtl2_tessent_pose
synchronizer_reset corea_rtl2_tessent_sib_1 corea_rtl2_tessent_sib_2 corea_rtl2_tessent_tdr_sri_ctrl}
SETUP> set_attribute_value qcsram* -name preserve_boundary -silent
SETUP> set_attribute_value qcrf* -name preserve_boundary -silent
SETUP> set_system_mode analysis
//  Warning: 4 modules with attribute 'synthesize_before_analysis' were not synthesized. Use this command to
ess them:
//           get_modules * -filter "synthesize_before_analysis && (type==cell || type==primitive || !has_rtl
is_blackbox || has_no_definition || exclude_from_synthesis)"
//  ------------------------------------------------------------------------
//  Begin RTL synthesis.
//  --------------------
//  Synthesized modules=46, Time=6.09 sec.
//  Note: There were 46 modules selectively synthesized. There were also 29 sub-modules created by synthesis
//        Use 'get_module -filter is_synthesized' to see them.
//        You can also use 'set_quick_synthesis_options -verbose on' to have the synthesis step report the
//        synthesized module names in the transcript as they are being synthesized.
//  ------------------------------------------------------------------------
//  Warning: Rule FN4 violation occurs 19 times
//  Flattening process completed, cell instances=30127, gates=46490, PIs=80, POs=67, CPU time=0.28 sec.
//  ------------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.10 sec.
```

- The exact hierarchical path for all the sibs

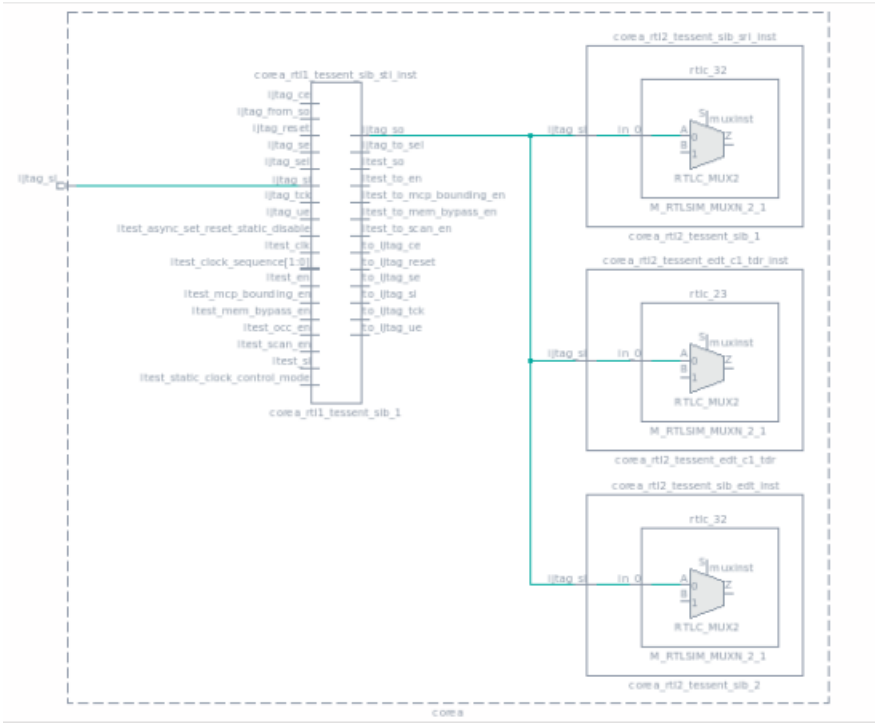
```
ANALYSIS> get_instance *sib* -hier
{corea_rtl1_tessent_sib_mbist_inst corea_rtl1_tessent_sib_sti_inst corea_rtl2_tessent_sib_edt_inst corea_rtl2_te
ssent_sib_occ_inst corea_rtl2_tessent_sib_sri_inst corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_co
ntroller_sib_ctl_bypass_inst corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_inst0 co
ea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_inst1 corea_rtl1_tessent_mbist_bap_inst/
orea_rtl1_tessent_mbist_controller_sib_tdr_bypass_inst corea_rtl1_tessent_sib_sti_inst/rtlcreg_sib corea_rtl1_te
ssent_sib_sti_inst/rtlcreg_sib_latch corea_rtl1_tessent_sib_mbist_inst/rtlcreg_sib corea_rtl1_tessent_sib_mbist_
inst/rtlcreg_sib_latch corea_rtl2_tessent_sib_sri_inst/rtlcreg_sib corea_rtl2_tessent_sib_sri_inst/rtlcreg_sib_l
atch corea_rtl2_tessent_sib_edt_inst/rtlcreg_sib corea_rtl2_tessent_sib_occ_inst/rtlcreg_sib corea_rtl2_tessent_
sib_edt_inst/rtlcreg_sib_latch corea_rtl2_tessent_sib_occ_inst/rtlcreg_sib_latch corea_rtl1_tessent_mbist_bap_in
st/corea_rtl1_tessent_mbist_controller_sib_inst1/rtlcreg_sib corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tesse
t_mbist_controller_sib_inst0/rtlcreg_sib corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_s
ib_inst1/rtlcreg_sib_latch corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_inst0/rtlcr
eg_sib_latch corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_tdr_bypass_inst/rtlcreg_s
ib corea_rtl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_ctl_bypass_inst/rtlcreg_sib corea_r
tl1_tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_tdr_bypass_inst/rtlcreg_sib_latch corea_rtl1_
tessent_mbist_bap_inst/corea_rtl1_tessent_mbist_controller_sib_ctl_bypass_inst/rtlcreg_sib_latch corea_rtl2_tess
ent_occ_clka_inst/occ_control/tdr_sib corea_rtl2_tessent_occ_clkb_inst/occ_control/tdr_sib corea_rtl2_tessent_oc
c_clkc_inst/occ_control/tdr_sib corea_rtl2_tessent_occ_clka_inst/occ_control/tdr_sib/rtlcreg_sib corea_rtl2_tess
ent_occ_clkb_inst/occ_control/tdr_sib/rtlcreg_sib corea_rtl2_tessent_occ_clkc_inst/occ_control/tdr_sib/rtlcreg_s
ib corea_rtl2_tessent_occ_clka_inst/occ_control/tdr_sib/rtlcreg_sib_latch corea_rtl2_tessent_occ_clkb_inst/occ_c
ontrol/tdr_sib/rtlcreg_sib_latch corea_rtl2_tessent_occ_clkc_inst/occ_control/tdr_sib/rtlcreg_sib_latch}
ANALYSIS> a_s_0 corea_rtl1_tessent_sib_mbist_inst corea_rtl1_tessent_sib_sti_inst corea_rtl2_tessent_sib_edt_ins
t  -disp hier
//  Error: Command 'a_s_0 corea_rtl1_tessent_sib_mbist_inst corea_rtl1_tessent_sib_sti_inst corea_rtl2_tessent_s
ib_edt_inst -disp hier' is unknown
ANALYSIS> a_s_0 corea_rtl1_tessent_sib_sti_inst -disp hier
//  Error: Command 'a_s_0 corea_rtl1_tessent_sib_sti_inst -disp hier' is unknown
ANALYSIS> a_s_o corea_rtl1_tessent_sib_sti_inst -disp hier
```

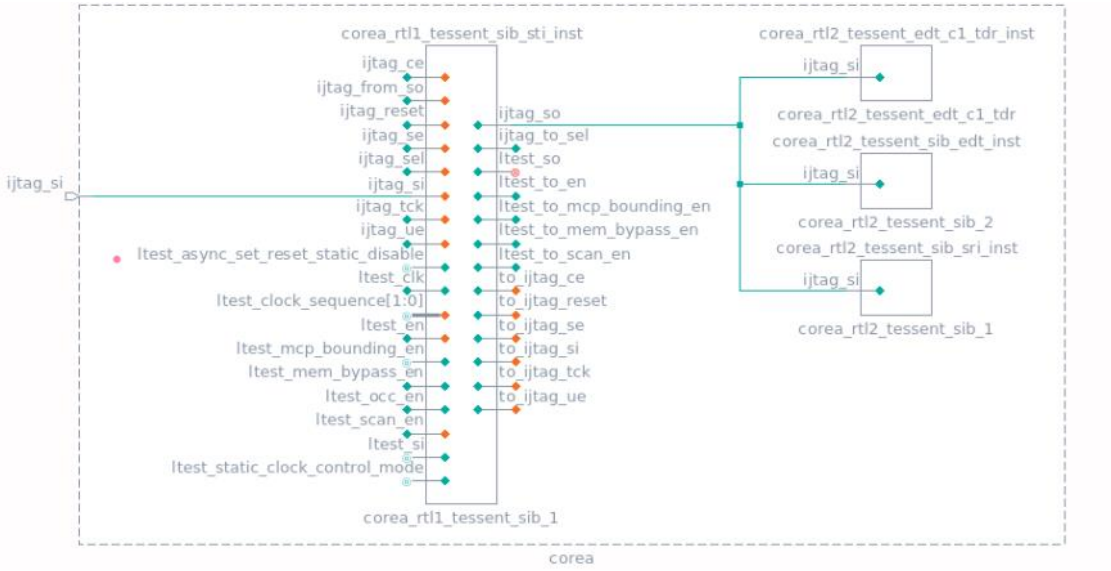
- From Open Visualizer

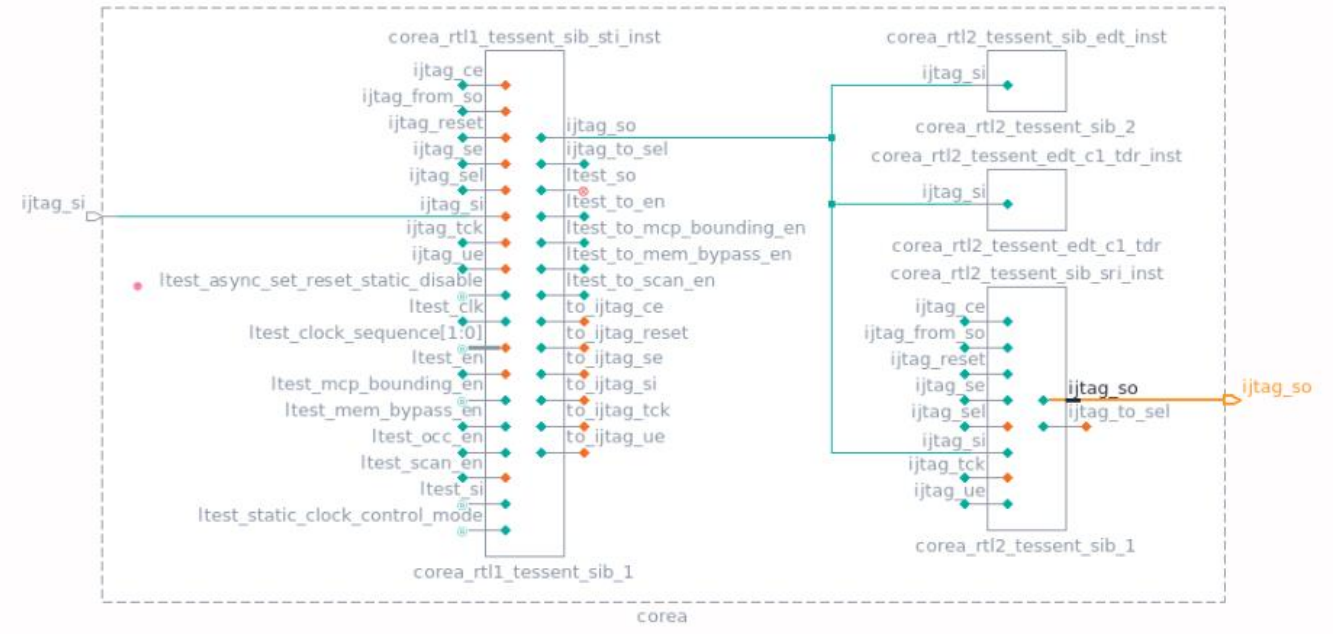


Coming from top level port but later it will be controlled by a SIB which is present inside the CoreB level.

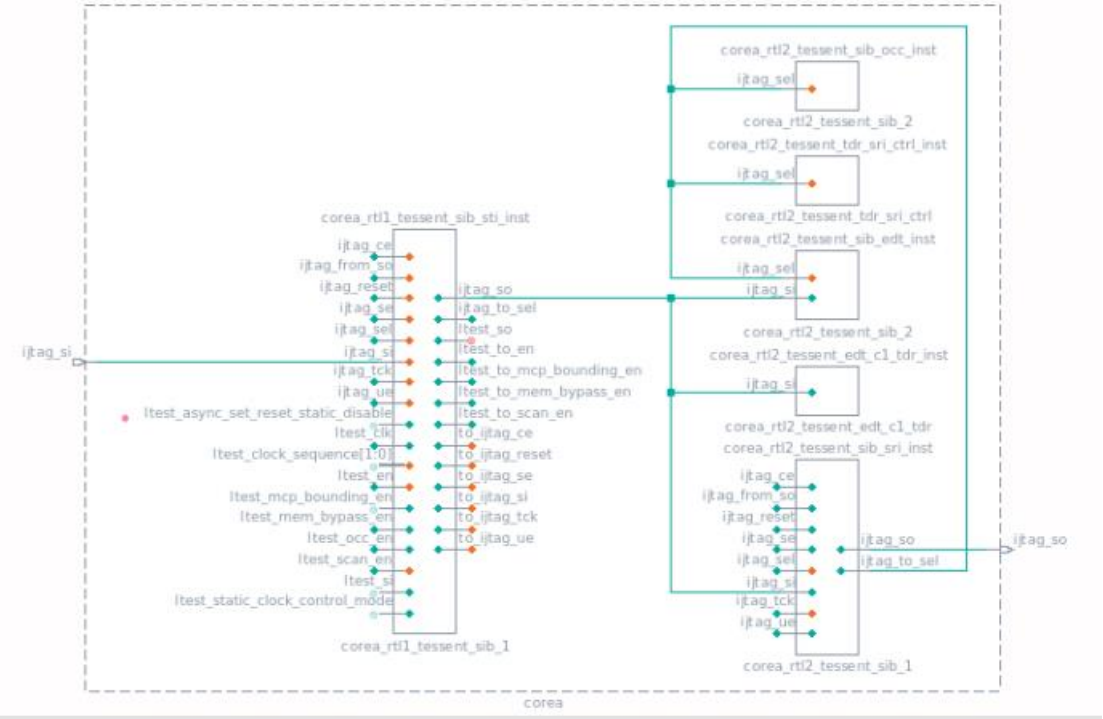
- The Ijtag_so will go to edt tdr, sib sri and sib edt

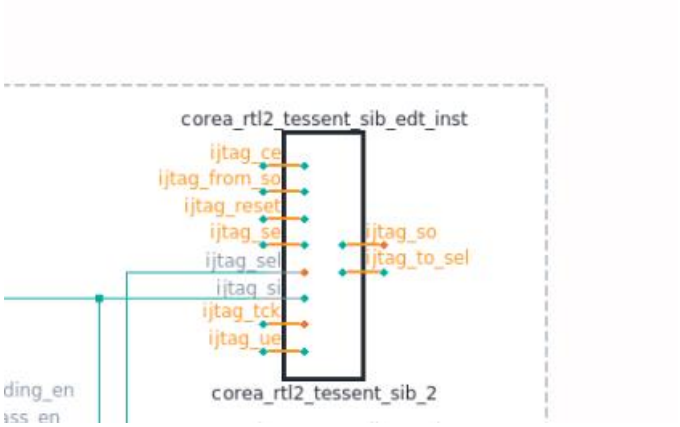




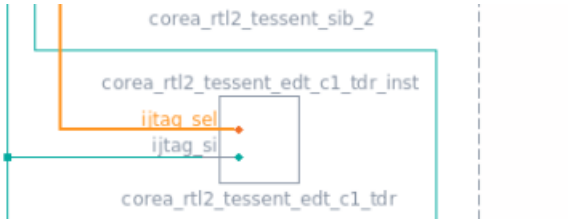


- Ijtag_to_sel will go to sib edt, sib occ and tdr sri ctrl

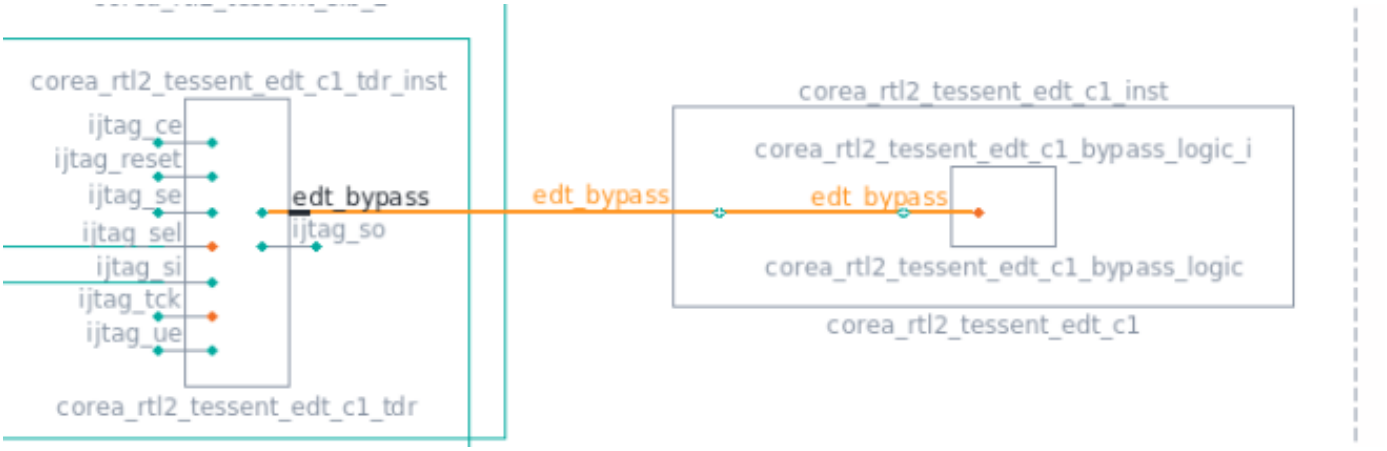




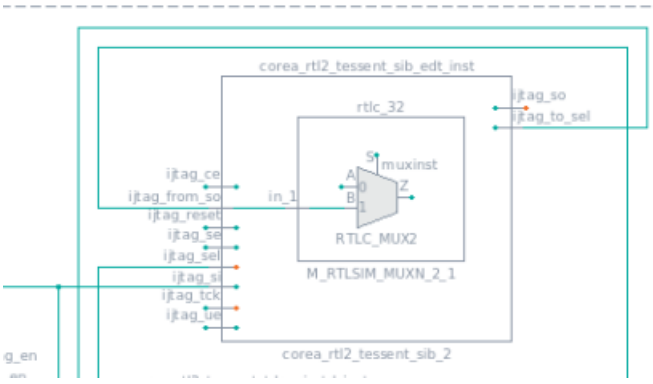
- Ijtag to sel it will go to edt tdr



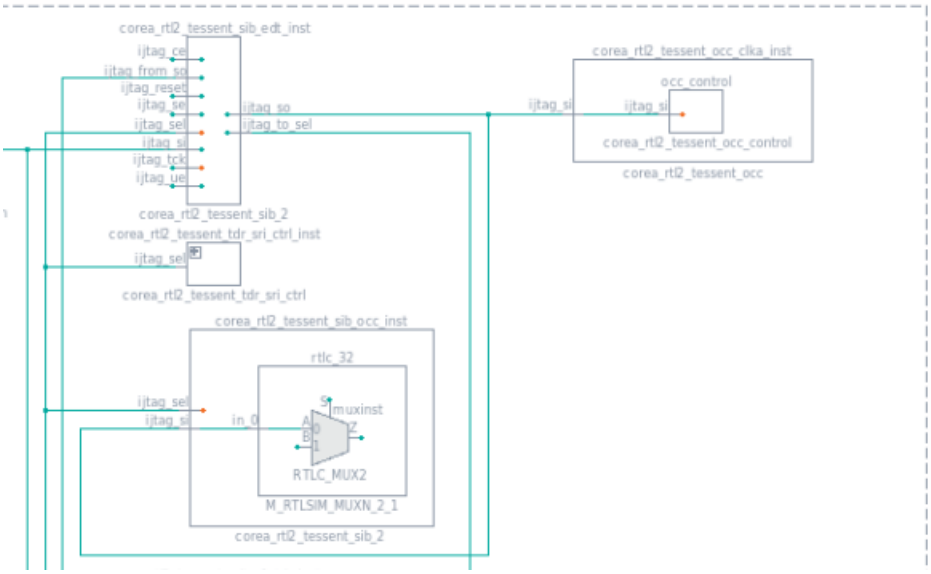
- What static signal will be controlling by this edt tdr



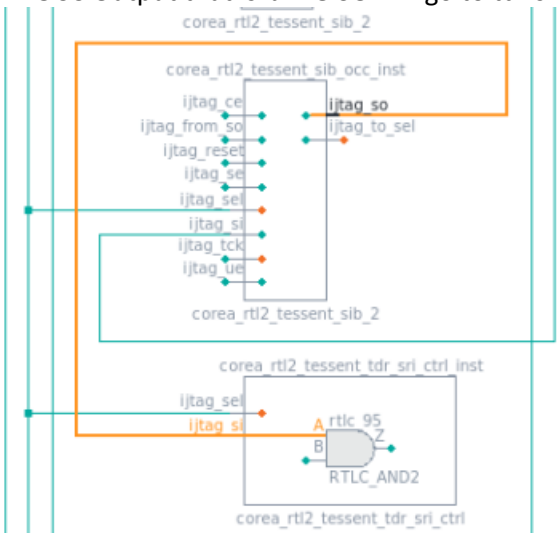
EDT TDR will be controlling the edt bypass signal and it will got edt bypass logic
IJTAG_SO will go to SIB EDT

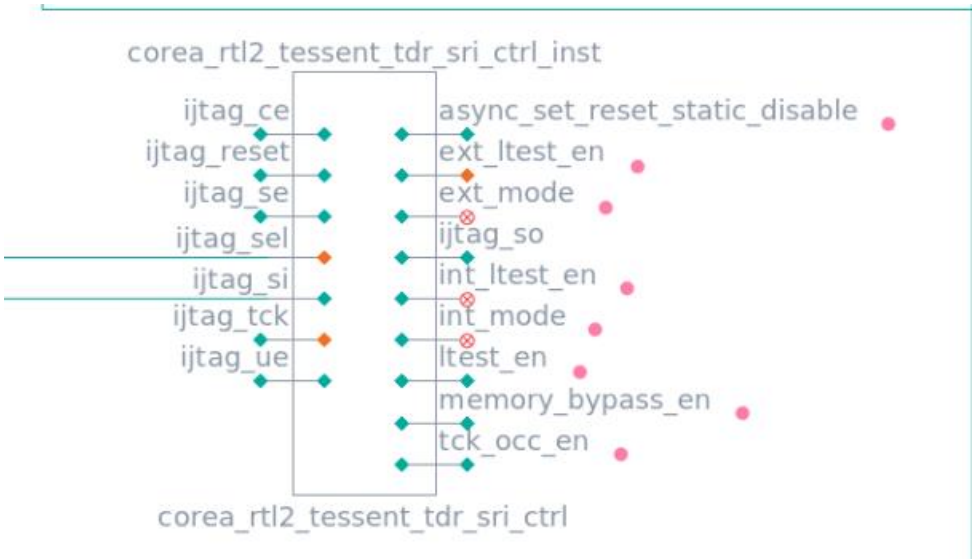


- One fanout of occ and another fanout will go to sib occ from sib edt



- SIB OCC Output that is IJTAG SO will go to tdr sri ctrl

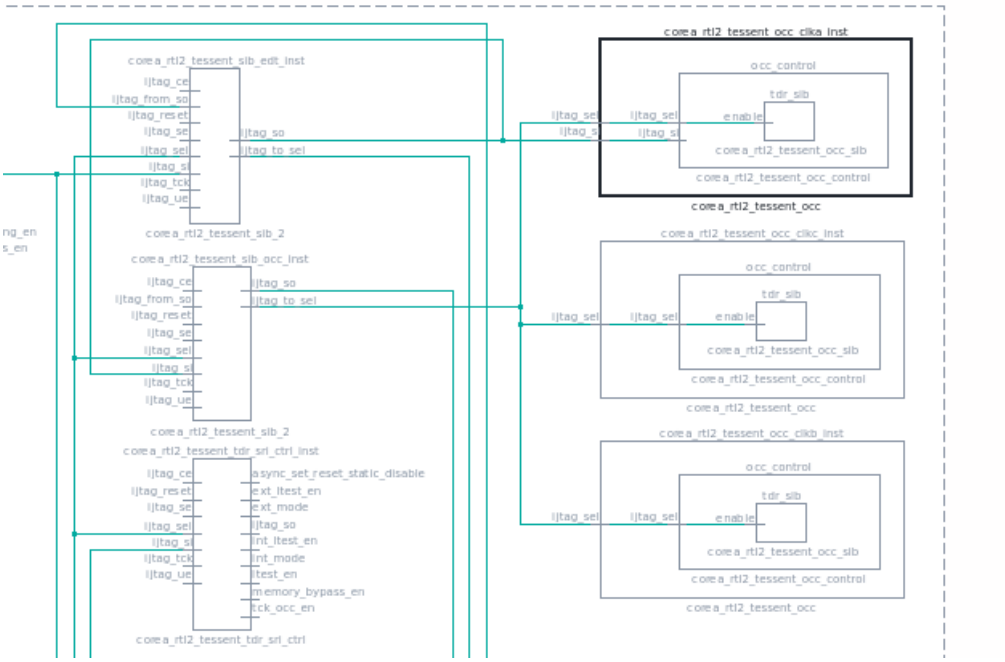




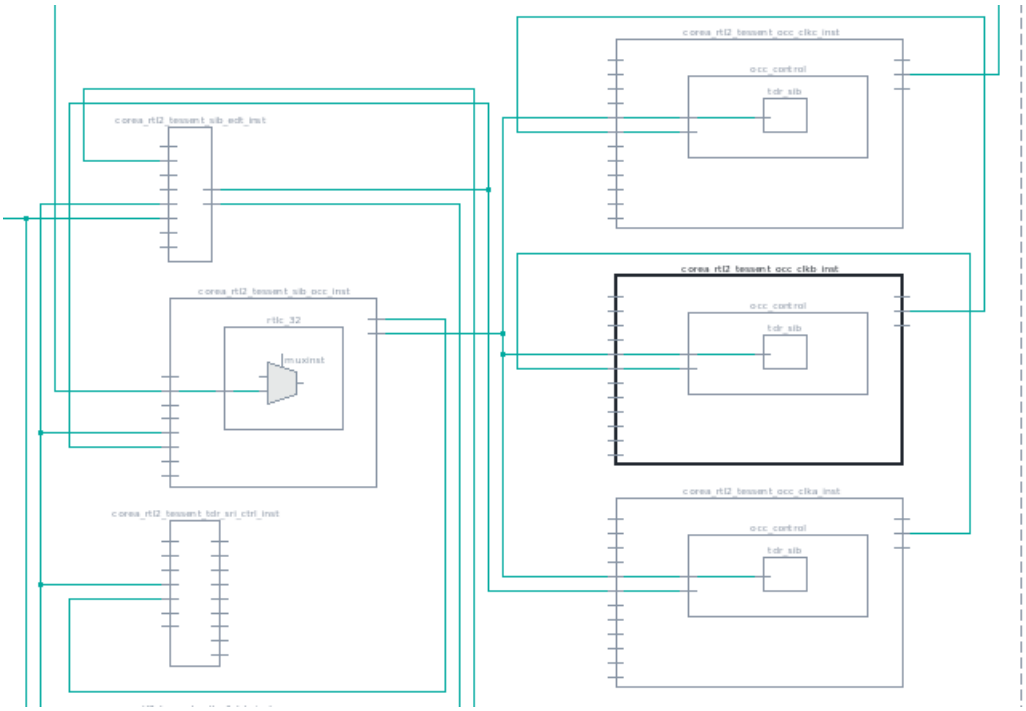
- Tdr sri ctrl will be controlling the all the signals int mode, ext mode and various other will be controlled by tdr sri control.

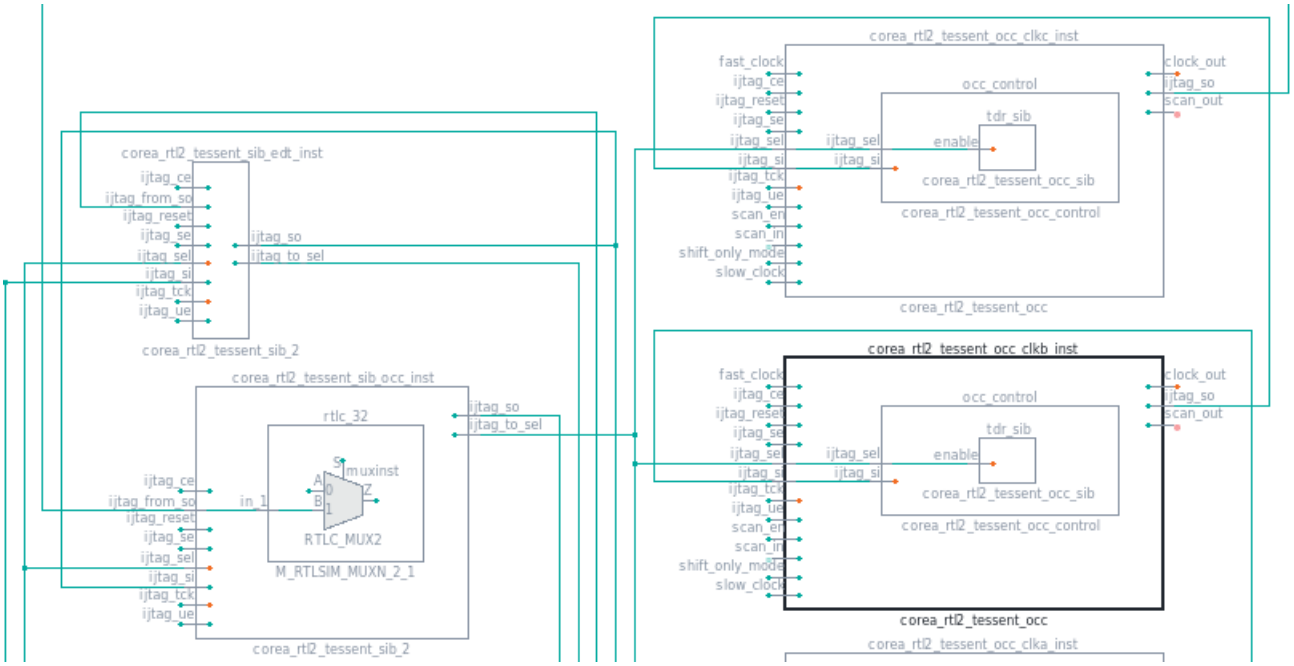
- SIB OCC - Ijtag to sel will be going to as a enable signal for the tdr signal which is present inside the occ logic, so total 3 OCCs we have inserted so for all the three OCCs signals will be going as a enable signal.

The TDR will be controlling the FCM

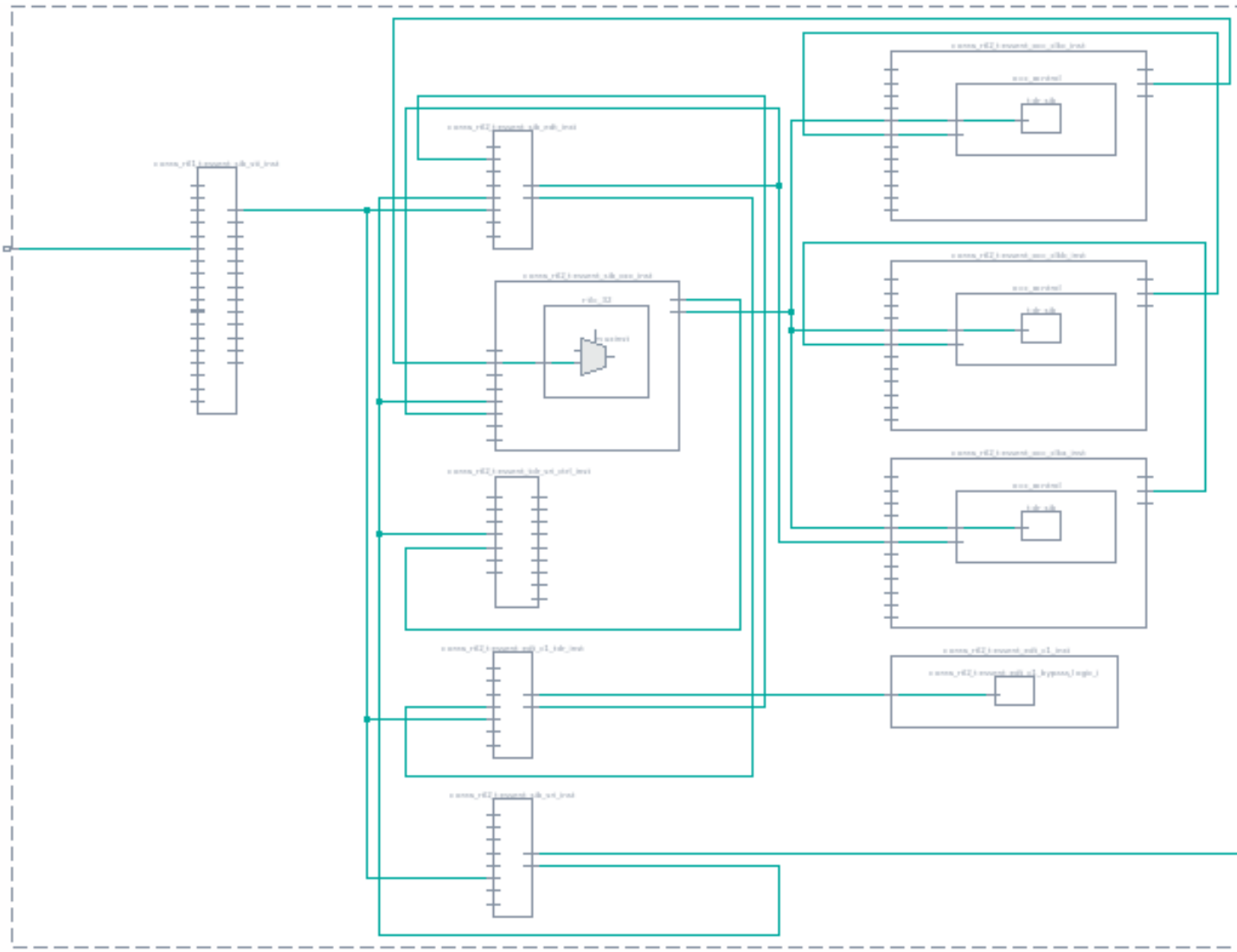


- Clka OCC will go to CLkb OCC and Clkb OCC will go to CLKc OCC
  Clkc OCC will go to from SO to SIB OCC



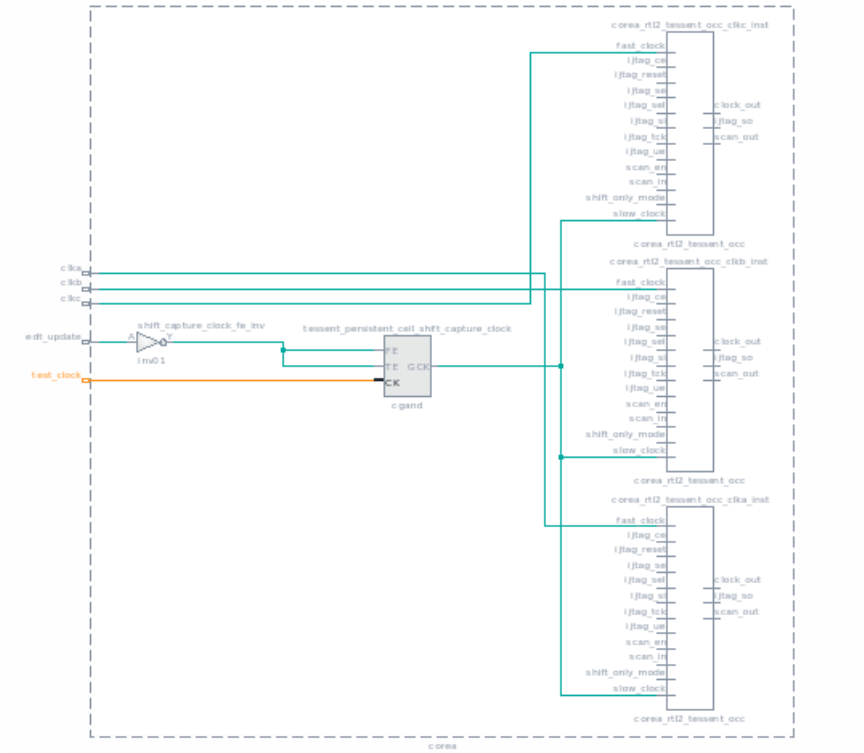


- The whole hierarchical schematic of corea IJTAG Network



- OCC of clka, clkb and clkc

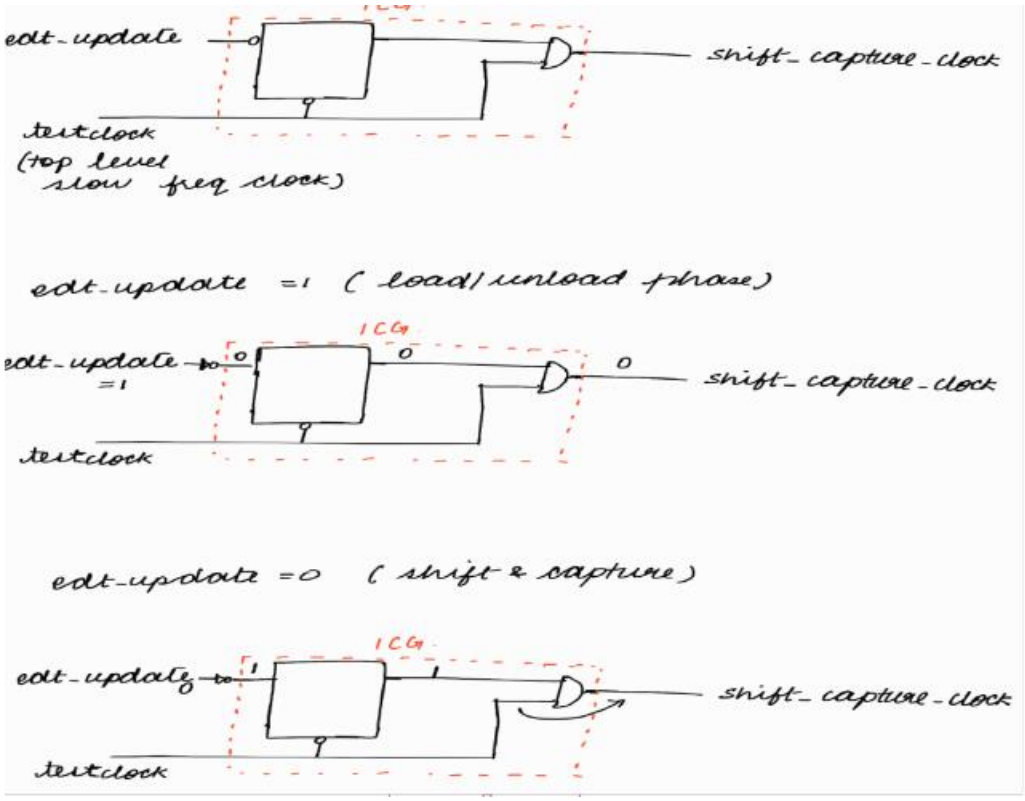
```
ANALYSIS> get_instance *occ*
{corea_rtl2_tessent_occ_clka_inst corea_rtl2_tessent_occ_clkb_inst corea_rtl2_tessent_occ_clkc_inst corea_rt
essent_sib_occ_inst}
ANALYSIS> a_s_o corea_rtl2_tessent_occ_clka_inst -disp hier
ANALYSIS> a_s_o corea_rtl2_tessent_occ_clkb_inst -disp hier
ANALYSIS> a_s_o corea_rtl2_tessent_occ_clkc_inst -disp hier
```

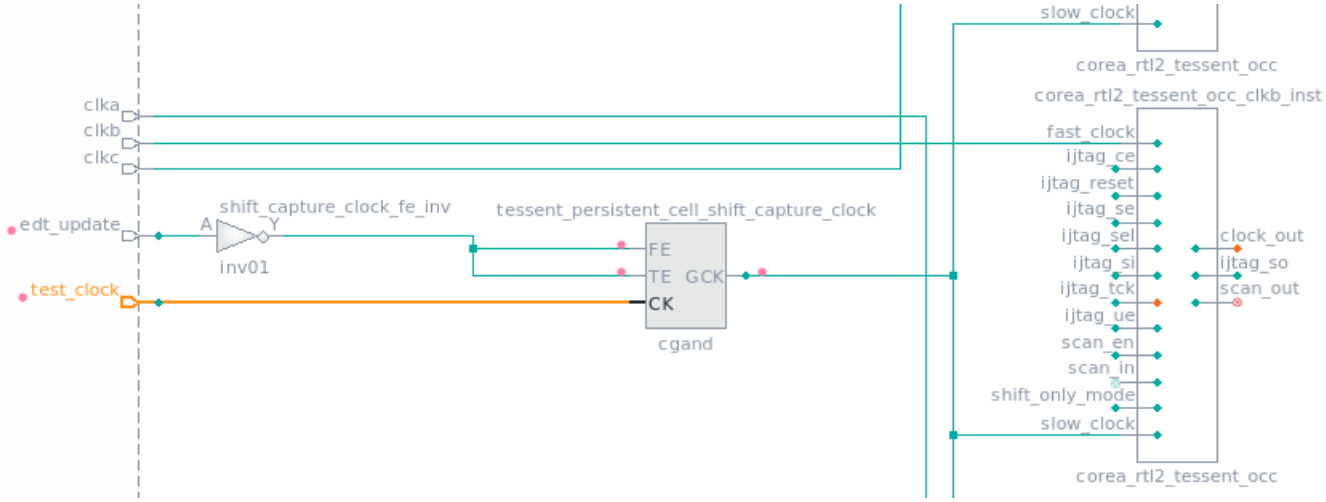


Slow clk is coming from ICG logic

- FE and TE will be coming from EDT update signal, that is not of edt update signal.
- Input of the clk will be test_clock

From 2nd diagram of below so only shift capture clock will go to the Input of the OCC





- ICG output will be going to slow clock

Why it is creating this ICG logic ?

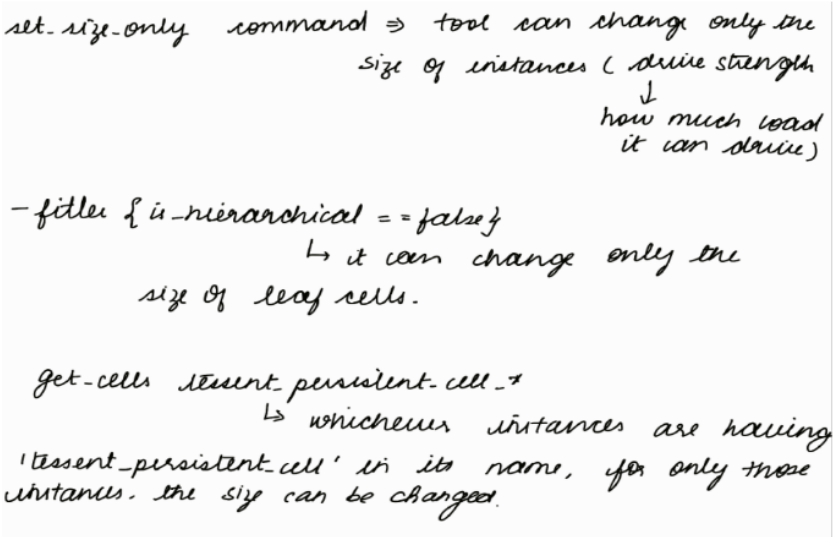
- Because the edt clock and shift capture clock from the top level test that's why the tool has added this ICG.

STEP 3 - SYNTHESIS

```
1
2  sh mkdir -p -v outputs
3
4  set design_name corea
5
6
7  ## Defind the library files
8  set target_library "../../../library/adk.db"
9  set link_library "../../../library/adk.db $target_library"
10 read_db $target_library
11
12 # Bus naming style for Verilog
13 set bus_naming_style {%s[%d]}
14
15 # Read input design files
16 source ../2.insert_edt_occ/${design_name}.dc_shell_import_script
17
18 # Synthesize the top module
19 elaborate ${design_name}
20 set_size_only [get_cells tessent_persistent_cell_* -hier -filter {is_hierarchical==false}] -all_instances
21 link
22
23 # Check design for inconsistencies
24 check_design
25
26 # Timing specification
27 create_clock -period 10 -waveform {0 5} clka
28 create_clock -period 10 -waveform {0 5} clkb
29
30
31 # Avoid assign statements in the synthesized netlist.
32 set_fix_multiple_port_nets -feedthroughs -outputs -buffer_constants
33
34 # Compile design
35 uniquify
36 compile -map_effort medium
37
38 # Report design results for TOP design
39 report_area > outputs/${design_name}_dc_script_report.out
40 report_constraint -all_violators -verbose >> outputs/${design_name}_dc_script_report.out
41 report_timing -path full -delay max >> outputs/${design_name}_dc_script_report.out
42 report_reference >> outputs/${design_name}_dc_script_report.out
43
44 write -f verilog -hierarchy -o outputs/${design_name}_top_gate.v
45
46 sh rm *.syn *.pvl *.mr
47
48 exit
```

Line 20: Explanation

SYNTHESIS SCRIPT FILE EXPLANATION



STEP 4: SCAN INSERTION

```
set_context dft -scan   -design_id gate -hier
set_tsdb_output_directory ../tsdb_outdir

read_cell_library ../../../library/adk.tcelllib
read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary

read_verilog ../3.synthesis/outputs/corea_top_gate.v

read_design corea -design_id rtl2 -icl_only -verbose
set_current_design corea
set_design_level physical_block
check_design_rules
set_wrapper_analysis_options -exclude_ports [get_ports *edt_channel*]
set_dedicated_wrapper_cell_options on -ports rst
analyze_wrapper_cells

add_scan_mode int_mode -edt_instances corea_rtl2_tessent_edt_c1_inst
add_scan_mode ext_mode -chain_length 32
analyze_scan_chains
insert_test_logic


set_context patterns -scan

foreach_in_collection mode_wrapper [get_config_elements Core(corea)/Scan/Mode -part tcd -silent] {
  set mode_name [get_config_value $mode_wrapper -id <0>]
  set mode_type [get_config_value type -in $mode_wrapper]
  import_scan_mode $mode_name
  #In setup mode
  report_clocks
  check_design_rules
  #In Analysis mode
  report_clocks
  if {$mode_type eq "external"} {
    report_scan_cells
    set_attribute_value [get_ports *edt_channel*] -name ignore_for_graybox
    analyze_graybox
    write_design -tsdb -graybox -verbose
  }
  set_system_mode setup
}
```

Line1: The design Id we are giving so in the tsdb output directory another sub directory will form in this name. -hier means that we are hierarchical scan.
Line 2: In order to change the path of the tsdb OP directory, we are using this command.
Line 4 and 5: Reading the library file and memory related file.
Line 7: Reading the scan inserted netlist.
Line 9: rtl2 was edt and occ insertion design id, from the outputs directory of edt occ insertion so in tsdb output directory the sub directory will be created that is rtl2.
Line 10: doing the elaboration
Line 11:
Line 12: checking the drcs

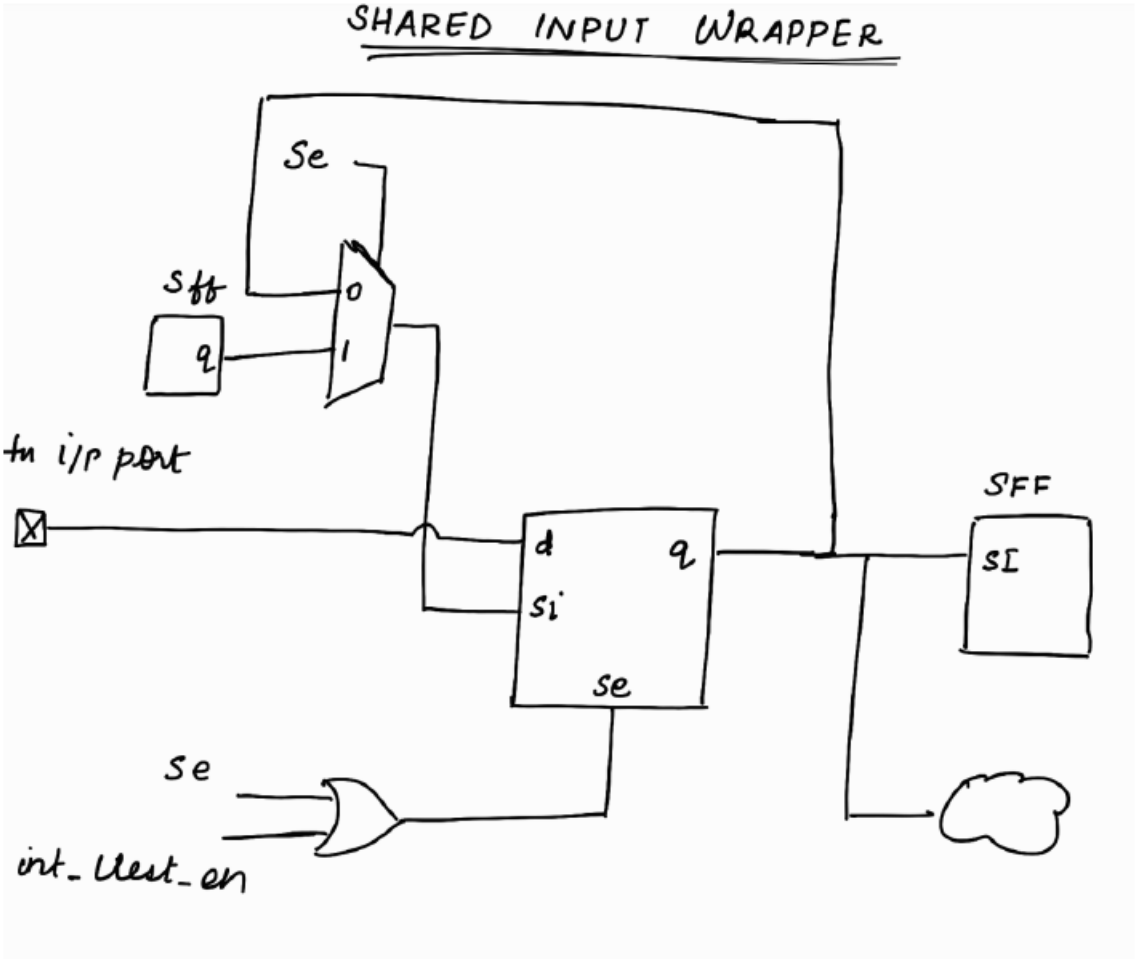
```
/home/vinayakp/aug23/level3/Lab3/corea/4.scan_insertion>>tessent -shell -dofile corea_dft_gate.tcl
//  Warning: Tessent user documentation not found
//  Tessent Shell  2021.1    Fri Feb 26 20:45:56 GMT 2021
//                Copyright 2011-2021 Mentor Graphics Corporation
//
//                           All Rights Reserved.
//
//    THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION WHICH
//  IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS AND IS
//                         SUBJECT TO LICENSE TERMS.
//
//  Mentor Graphics software executing under x86-64 Linux on Tue Jan 02 09:53:32 IST 2024.
//  64 bit version
//  Host: vlsiguru (64168 MB RAM, 32191 MB Swap)
//
//  command: set_context dft -scan   -design_id gate -hier
//  command: set_tsdb_output_directory ../tsdb_outdir
//  command: read_cell_library ../../../library/adk.tcelllib
//  Reading DFT Library file ../../../library/adk.tcelllib
//  Finished reading file ../../../library/adk.tcelllib
//  command: read_verilog ../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary
//  Warning: File: ../../../library/mems/SYNC_1R1W_16x8.v, Line: 2:  Module 'SYNC_1R1W_16x8' includes RTL co
ucts that are discarded since the tool is expecting a gate-level design. This module therefore cannot be mod
d, and will be omitted when the design is written out.
//  command: read_verilog ../3.synthesis/outputs/corea_top_gate.v
//  command: read_design corea -design_id rtl2 -icl_only -verbose
//  sub-command: read_icl ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.icl -skip
ld_blocks  -no_notes
//  sub-command: source ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/corea.pdl
//  sub-command: read_core_descriptions ../tsdb_outdir/dft_inserted_designs/corea_rtl2.dft_inserted_design/c
.tcd
//  command: set_current_design corea
//  Warning: 34 cases: Undriven net in netlist module
```

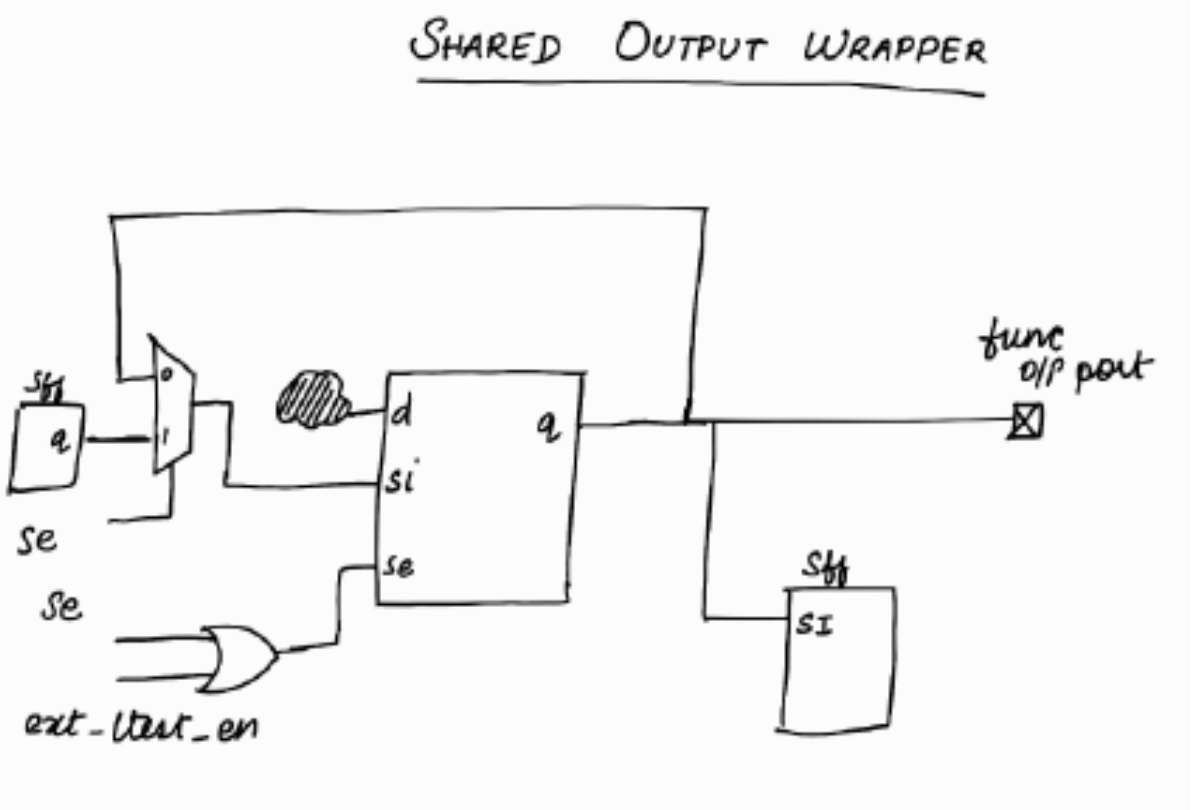
```
//  Warning: 186 cases: Net in netlist not connected
//  Note: Issue set_current_design with the -show_elaboration_warnings option to see more details about previ
 warnings
//  ---------------------------------------------------------------------
//  Begin ICL elaboration and checking.
//  -----------------------------------
//  ICL elaboration completed, CPU time=0.12 sec.
//  ---------------------------------------------------------------------
//  Warning: Primary input 'tessent_persistent_cell_shift_capture_clock/GCK_pport' is added at pin '/tessent_
sistent_cell_shift_capture_clock/GCK'
//  Note: Specified test clock connection point tessent_persistent_cell_shift_capture_clock/GCK will be mappe
o internal clock tessent_persistent_cell_shift_capture_clock/GCK_pport
//  command: set_design_level physical_block
//  command: check_design_rules
//  Warning: Rule FN1 violation occurs 228 times
//  Warning: Rule FN4 violation occurs 831 times
//  Flattening process completed, cell instances=6671, gates=11475, PIs=80+5(pseudo ports), POs=67, CPU time=
7 sec.
//  ---------------------------------------------------------------------
//  Begin circuit learning analyses.
//  --------------------------------
//  Learning completed, CPU time=0.05 sec.
//  ---------------------------------------------------------------------
//  Begin scan chain identification process, memory elements = 831,
//  sequential library cells = 831.
//  ---------------------------------------------------------------------
//  Begin simulation of test_setup procedure with 139 cycles.
//  Simulation of test_setup procedure completed, CPU time=0.0 sec.
//  Begin simulation of auto-generated load_unload procedure.
//  Simulation of load_unload procedure completed, CPU time=0.0 sec.
//  Scan segment = /corea_rtl1_tessent_sib_sti_inst/ltest_so successfully traced with scan_cells = 7.
//  Scan segment = /corea_rtl2_tessent_occ_clkc_inst/scan_out successfully traced with scan_cells = 3.
```

```
//  Scan segment = /corea_rtl2_tessent_occ_clkb_inst/scan_out successfully traced with scan_cells = 3.
//  Scan segment = /corea_rtl2_tessent_occ_clka_inst/scan_out successfully traced with scan_cells = 3.
//  3 external shadows that use shift clocking have been identified.
//  16 scan cells have been identified in 4 scan segments.
//  Longest scan segment has 7 scan cells.
//  Warning: 4 edge-triggered clock ports set to stable high. (D7)
//  Warning: Model 'nlatch' has no muxscan scan equivalent and is treated as nonscan model
//  ---------------------------------------------------------------------------
//  167 sequential library cells are treated as non-scan.
//  ---------------------------------------------------------------------------
//    10 sequential library cells missing mux-scan equivalent.
//    71 sequential library cells below hard module.
//    86 sequential library cells defined non-scan.
//  ---------------------------------------------------------------------------
//  Begin scannability rules checking for 642 sequential library cells
//  and 4 scan segments. The scan segments contain 22 additional cells.
//  ---------------------------------------------------------------------------
//  Note: There were 3 S7 violations (Potentially scannable cell that is not in the clock path is driven by
nstant value).
//  642 sequential library cells and 4 scan segments identified as scannable.
//  ---------------------------------------------------------------------------
//  Begin transparent latch checking for 21 latches.
//  ---------------------------------------------------------------------------
//  Warning: 1 latches not transparent due to unobservable. (D6)
//  Number transparent latches = 20.
//  ---------------------------------------------------------------------------
//  Begin scan clock rules checking.
//  ---------------------------------------------------------------------------
//  10 scan clock/set/reset lines have been identified.
//  All scan clocks successfully passed off-state check.
//  85 sequential cells passed clock stability checking.
//  There were 38 clock rule C3 fails (clock may capture data affected by its captured data).
//  Note: Trailing edge triggered device can capture data affected by leading edge.
//  ---------------------------------------------------------------------------
//  166 non-scan memory elements are identified.
//  ---------------------------------------------------------------------------
//  53 non-scan memory elements are identified as TIE-0.  (D5)
//  27 non-scan memory elements are identified as TIE-1.  (D5)
//   1 non-scan memory element is   identified as TIE-X.  (D5)
//  65 non-scan memory elements are identified as INIT-X. (D5)
//  20 non-scan memory elements are identified as TLA.    (D5)
//  ---------------------------------------------------------------------------
//  ---------------------------------------------------------------------------
//  Begin shift register identification for 642 sequential library cells.
//  ---------------------------------------------------------------------------
//  Number of shift register flops recorded for scan insertion: 36 (4.33%)
//  Number of shift registers recorded for scan insertion: 17
//  Longest shift register has 3 flops.
//  Shortest shift register has 2 flops.
//  Potential number of nonscan flops to be converted to scan cells: 17
//  Potential number of scan cells to be converted to nonscan flops: 0
//  Number of targeted sequential library cells = 642
//  command: stop
//  Error: Command 'stop' is unknown
// 'DOFile_corea_dft_gate.tcl' aborted at line 13
```

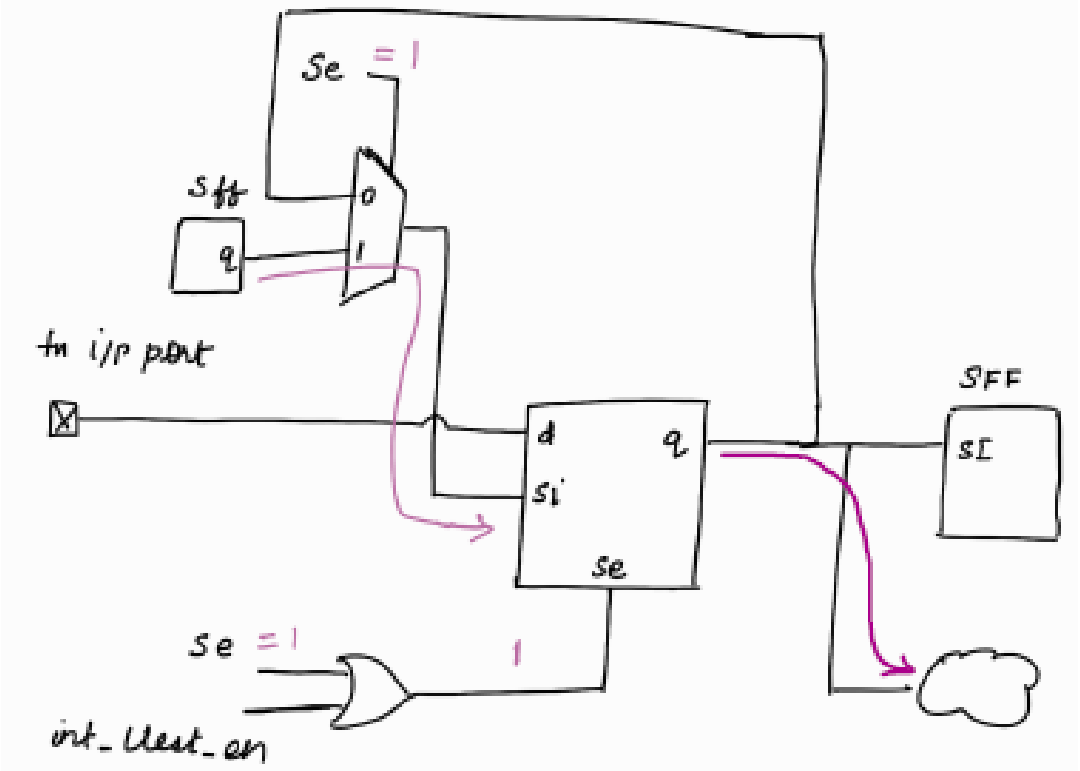
SHARED WRAPPERS

- **Shared input wrapper**

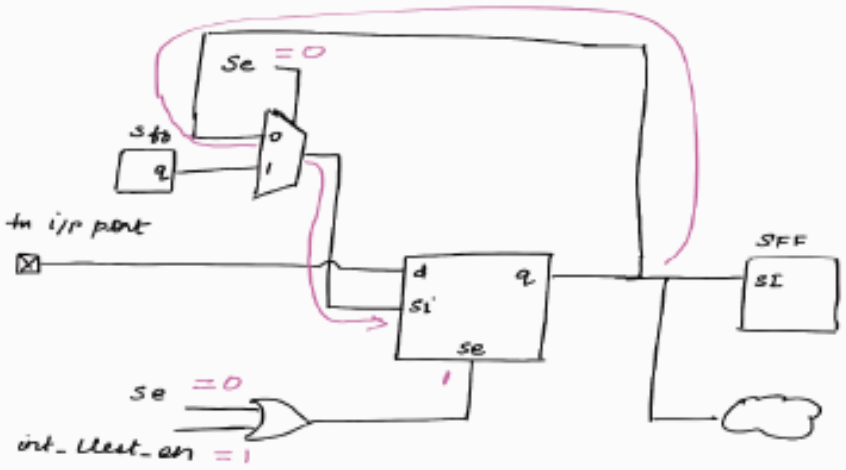




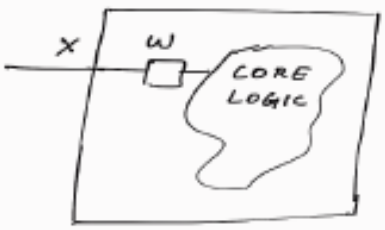
SHIFT :- Se = 1



CAPTURE :- Se = 0

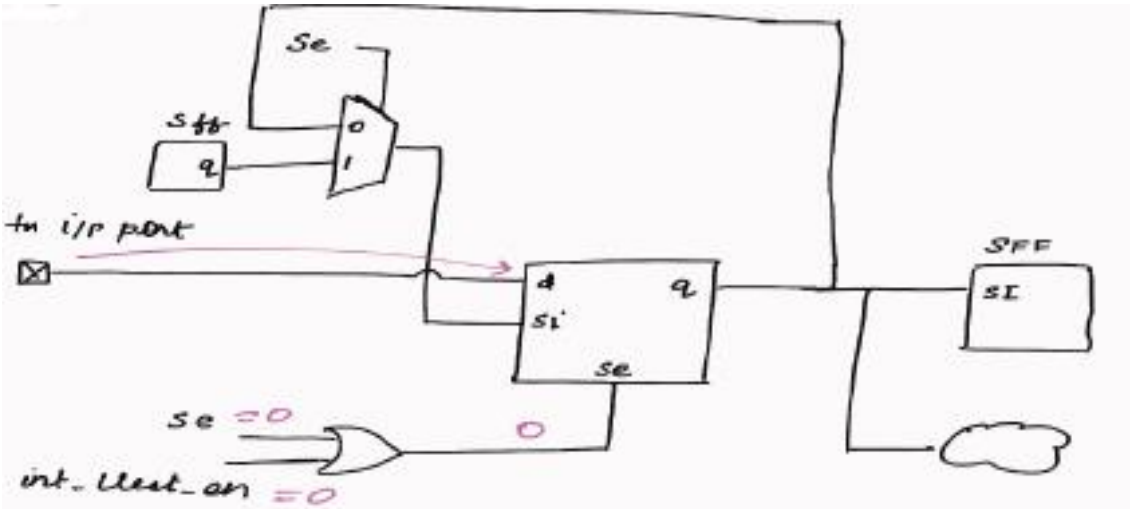


(i) int mode

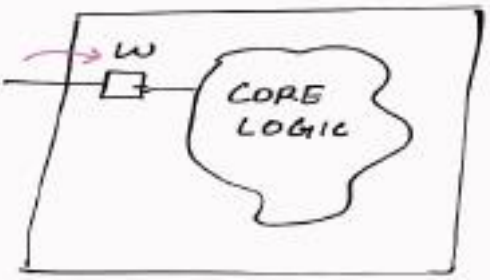


wrapper flop should not capture this X.

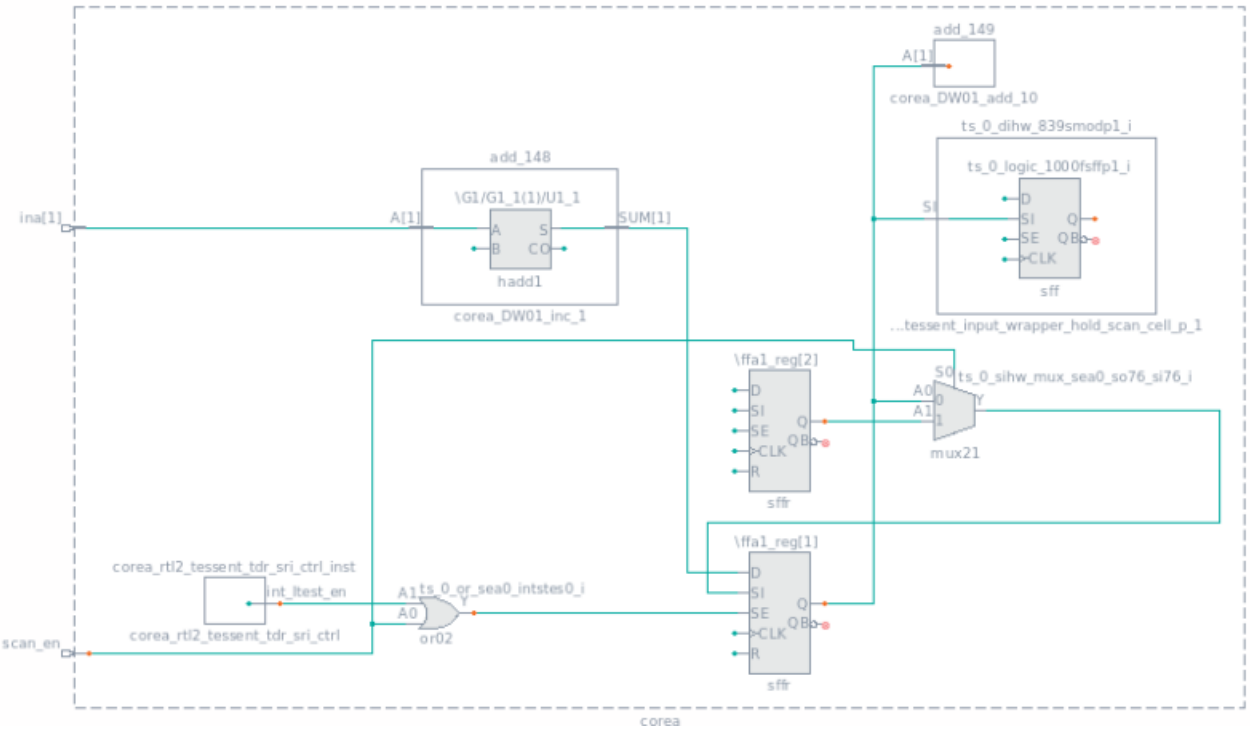
int_test_en = 1



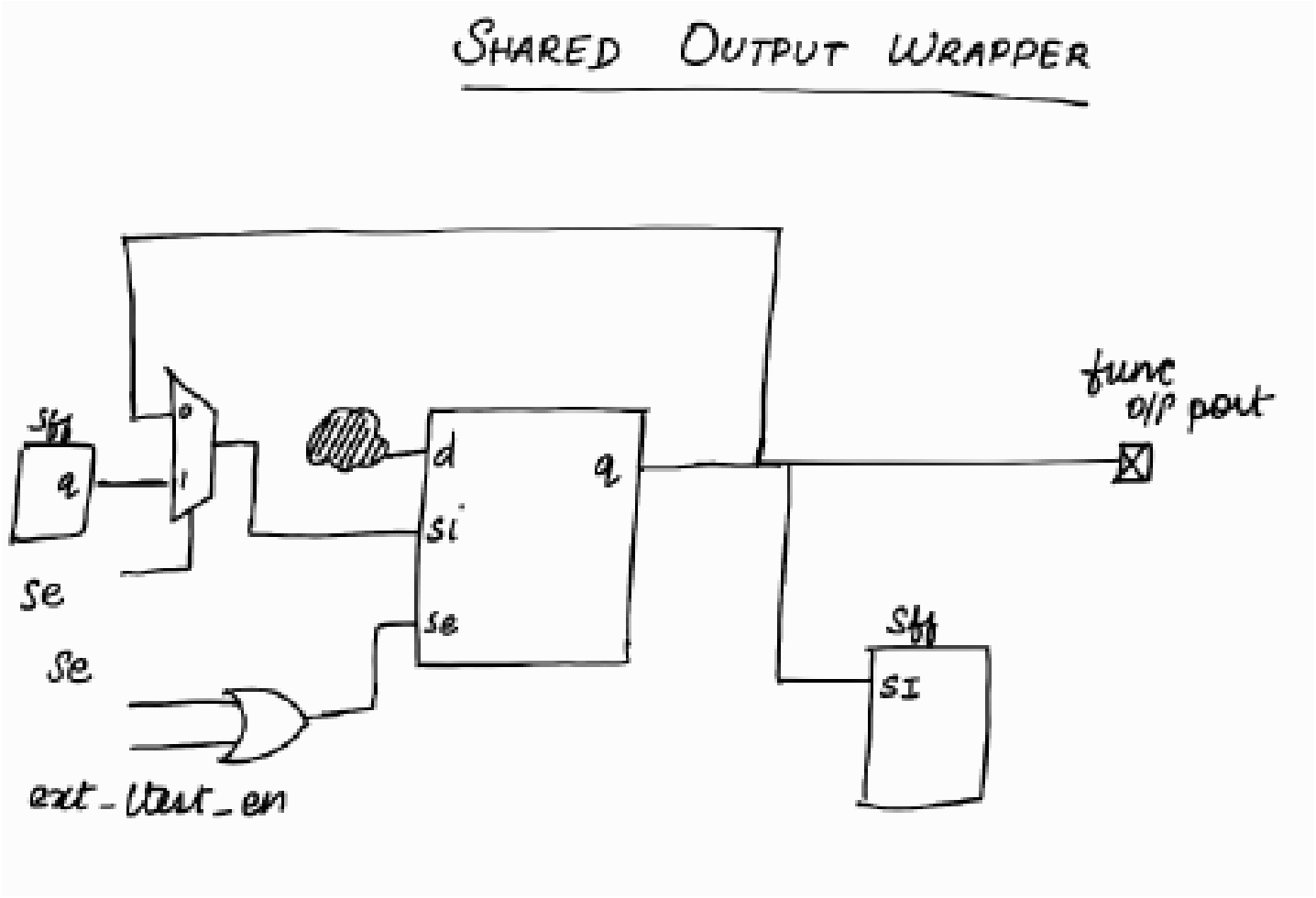
ii) ext mode



int_ltest_en = 0

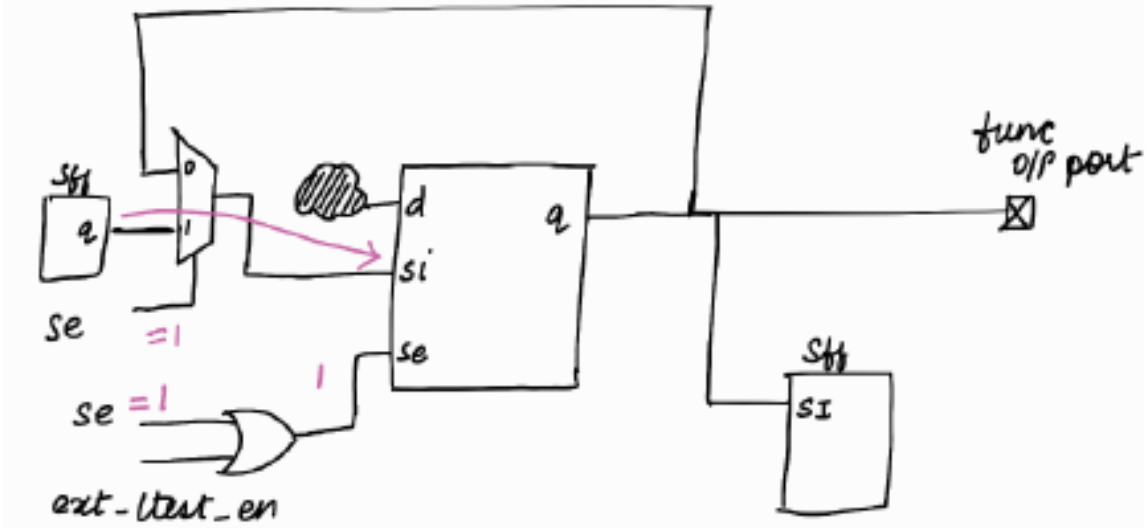


- **Shared output wrapper**

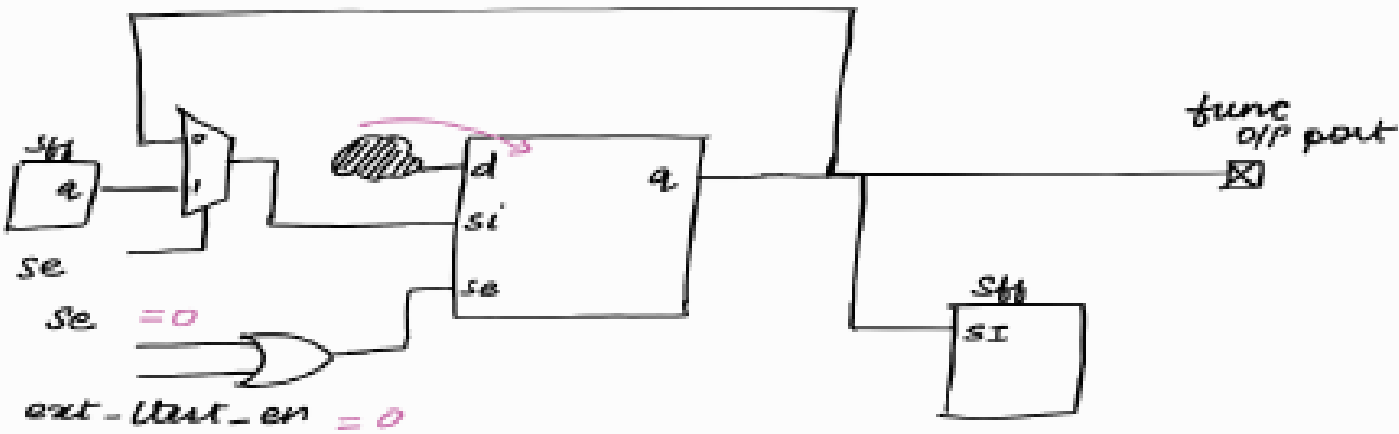


SHIFT :-

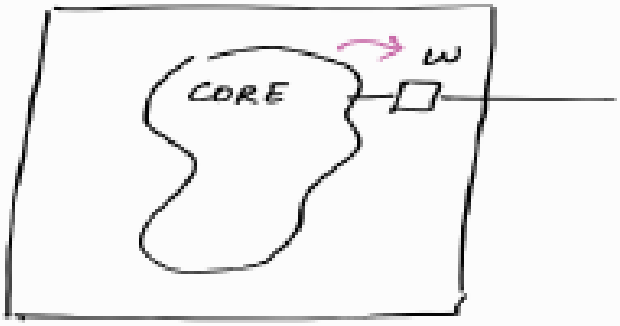
Se = 1

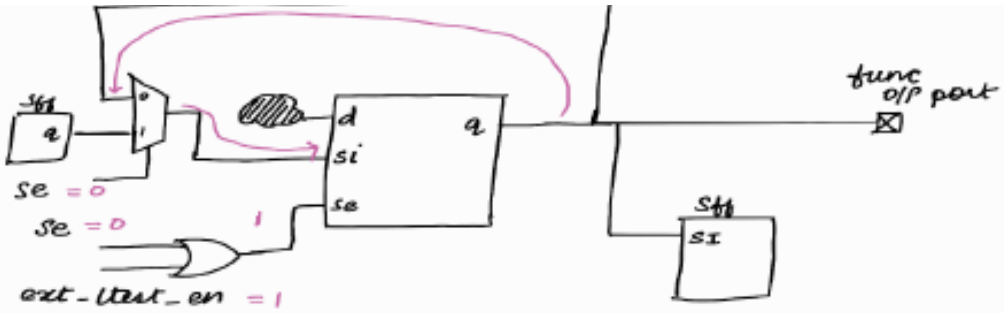


CAPTURE : Se = 0



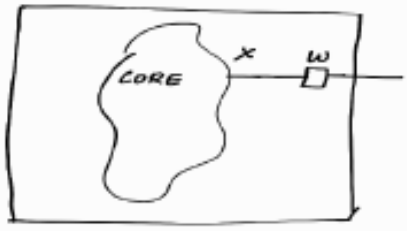
int mode :- ext_test_en = 0



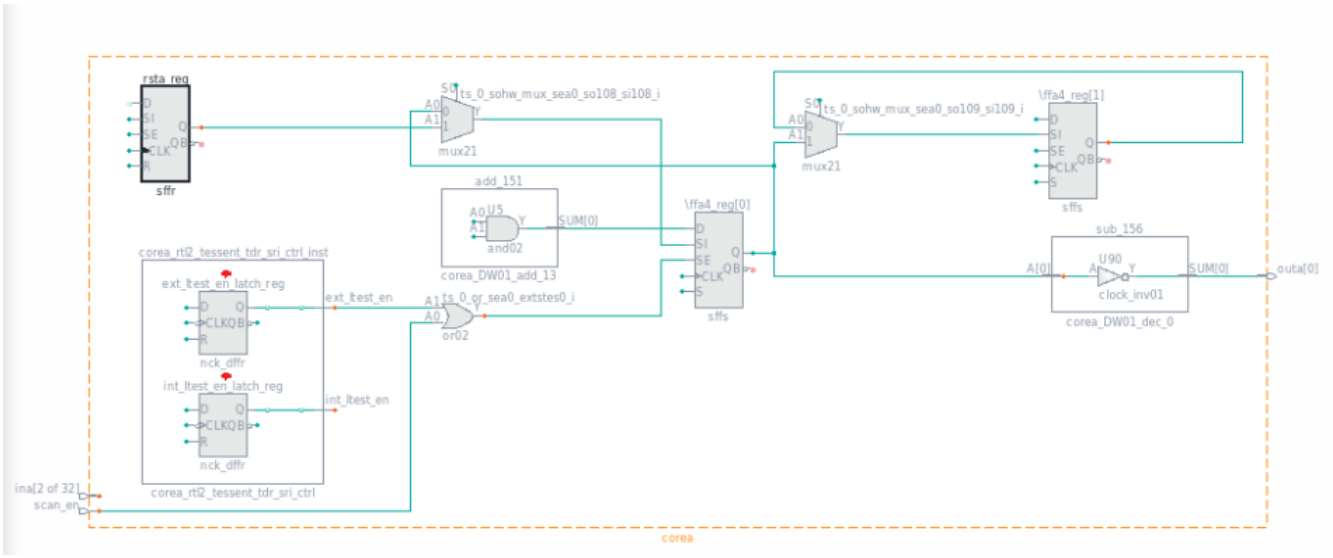


(ii) ext_mode :-

ext_test_en =1

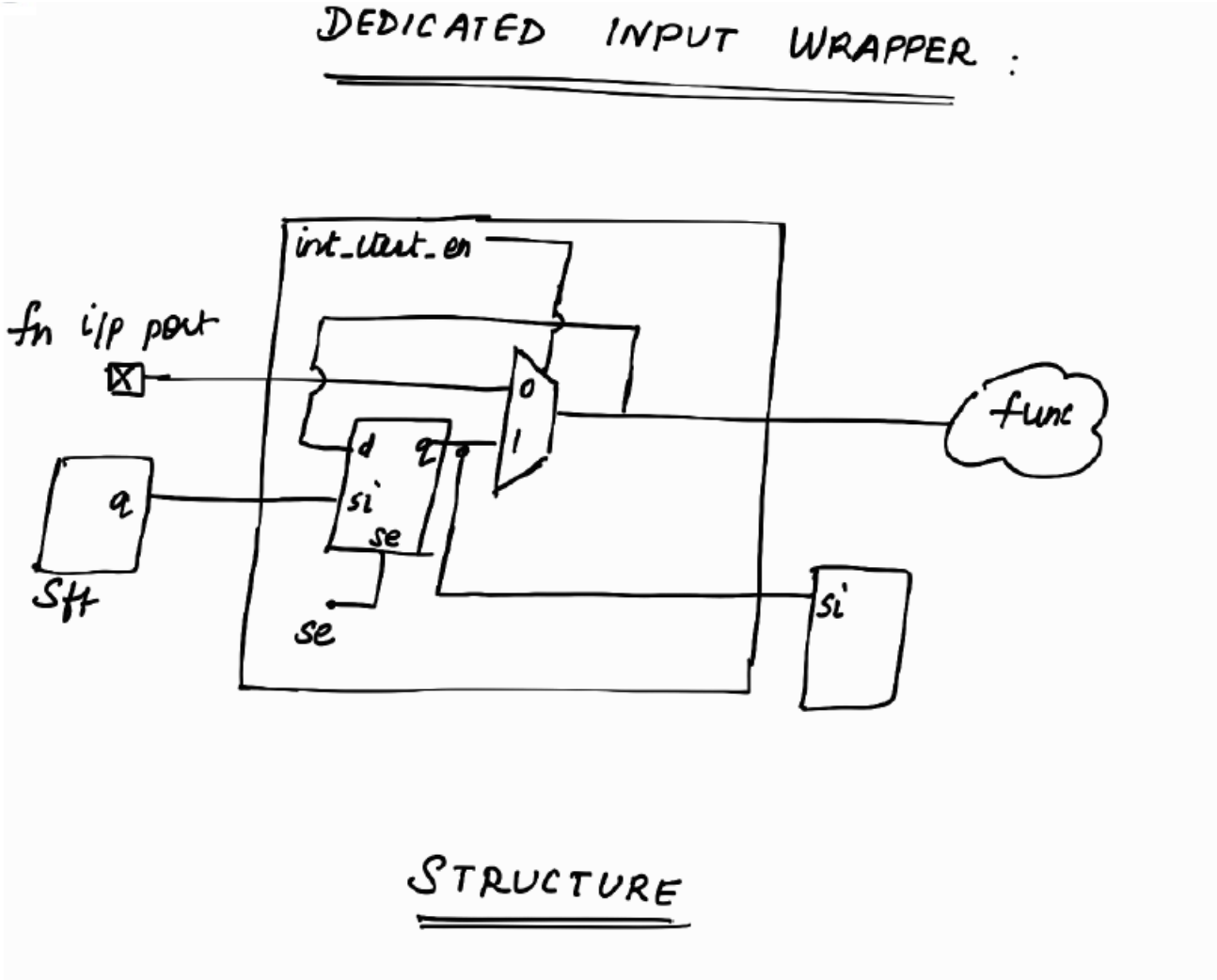


wrapper flop should not capture x.

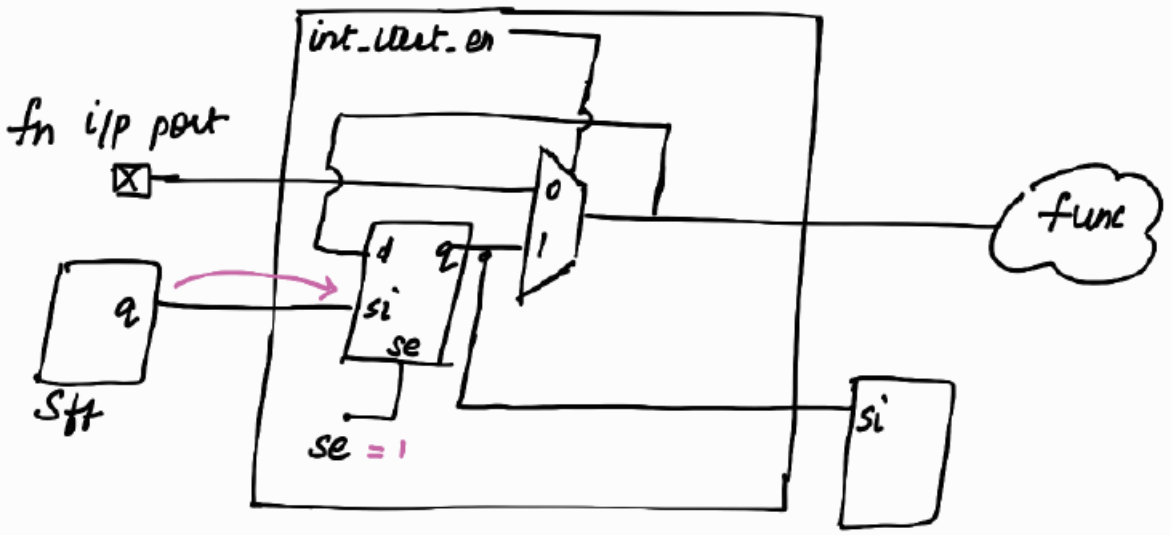

rsta_reg
sffr
ts_0_sohw_mux_sea0_so108_si108_i
mux21
add_151
corea_DW01_add_13
and02
SUM[0]
ts_0_sohw_mux_sea0_so109_si109_i
mux21
\ffa4_reg[1]
sffs
\ffa4_reg[0]
sffs
sub_156
U90
clock_inv01
corea_DW01_dec_0
A[0]
SUM[0]
outa[0]
corea_rtl2_tessent_tdr_sri_ctrl_inst
ext_ltest_en_latch_reg
nck_dffr
ext_ltest_en
int_ltest_en_latch_reg
nck_dffr
int_ltest_en
corea_rtl2_tessent_tdr_sri_ctrl
ts_0_or_sea0_extstes0_i
or02
ina[2 of 32]
scan_en
corea

DEDICATED WRAPPERS

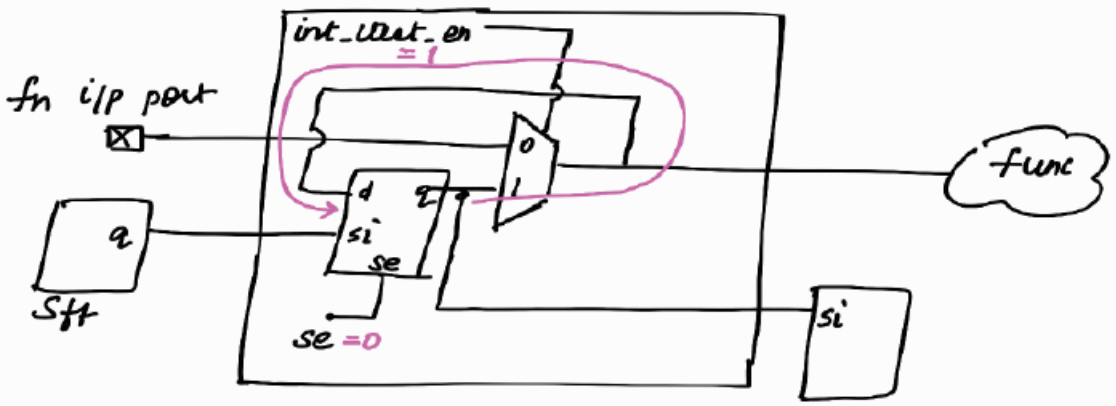
- **Dedicated Input Wrapper**



SHIFT :- Se = 1

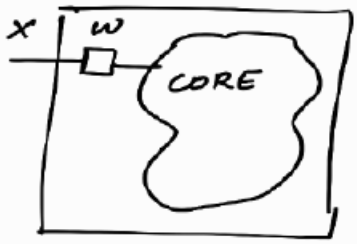


CAPTURE :- se = 0

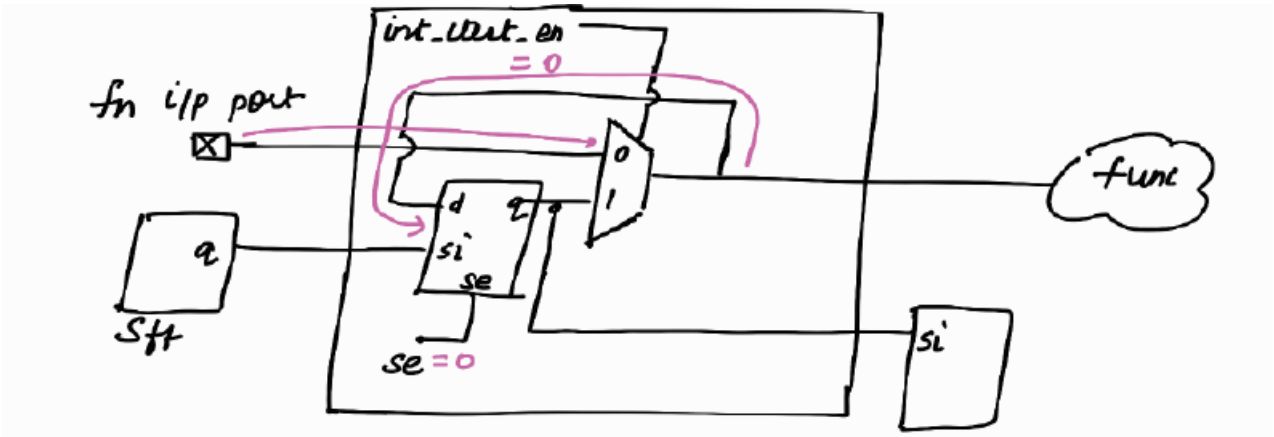


(i) int_mode

int_test_en = 1

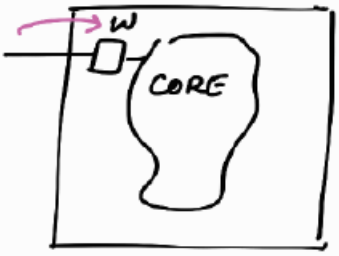


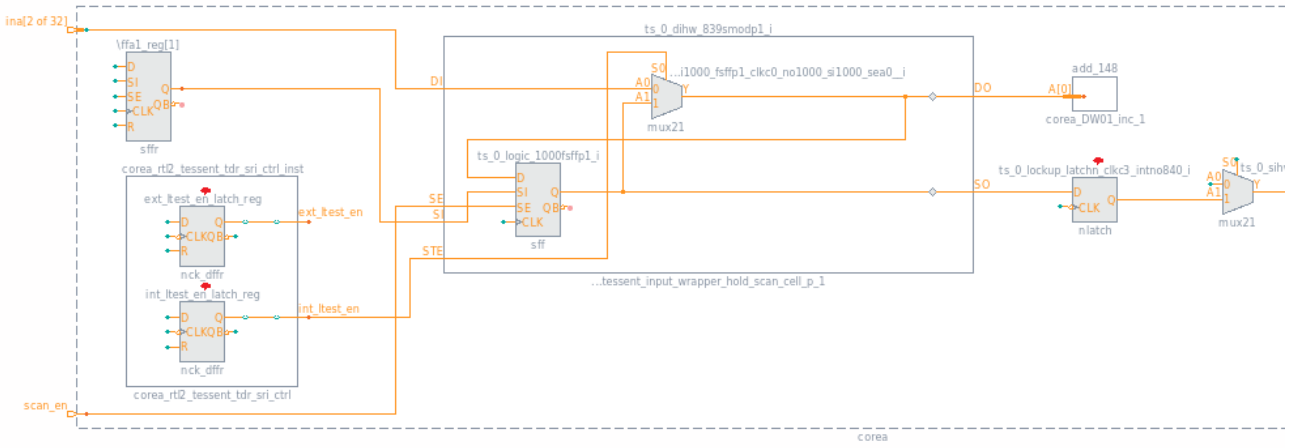
wrapper should not capture x.

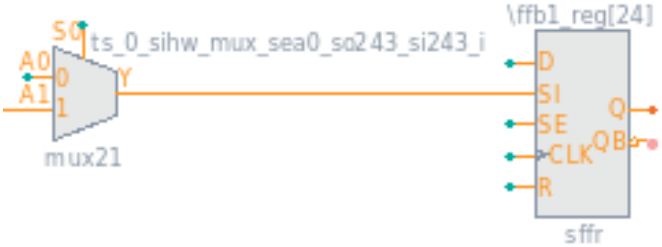


(ii) ext_mode :

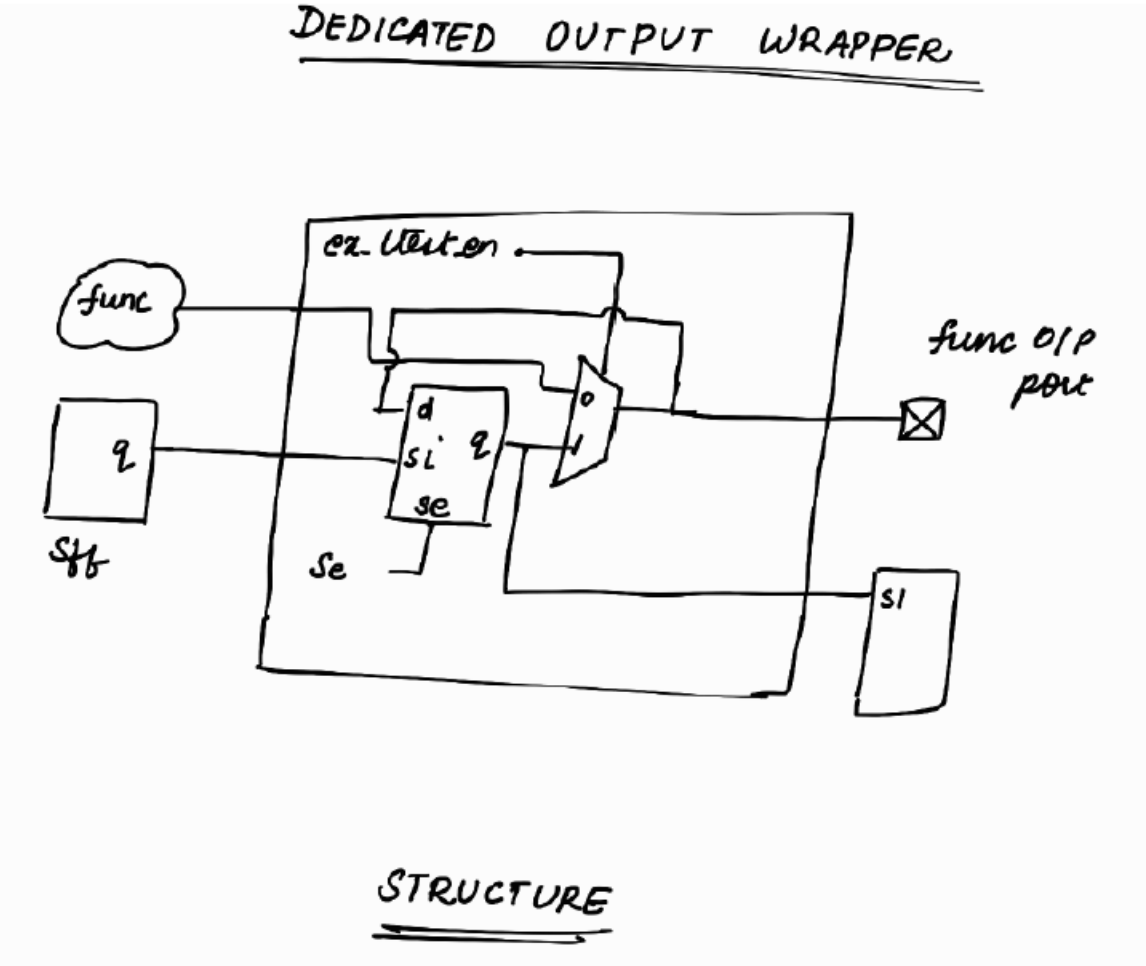
int_ ltest_ en = 0

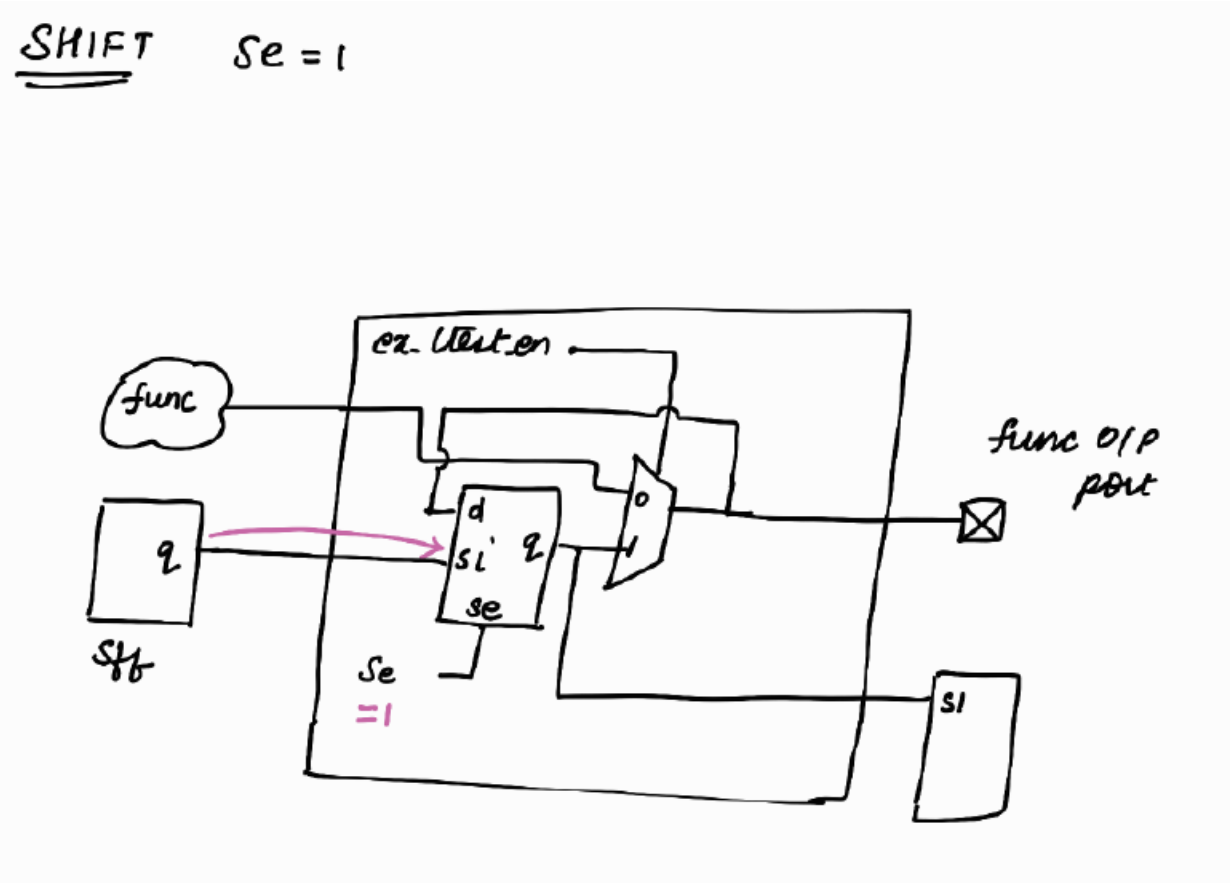




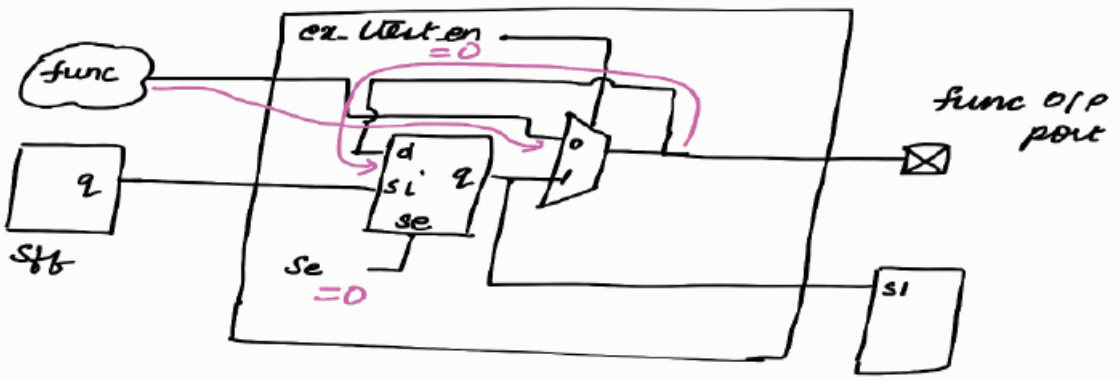


- **Dedicated Output Wrapper**

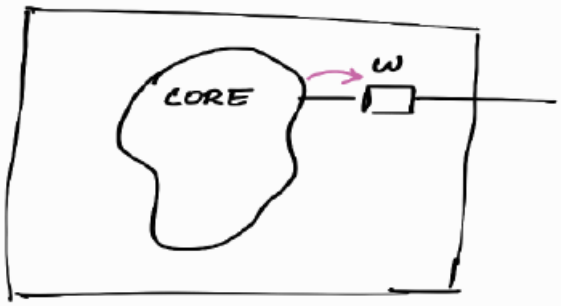


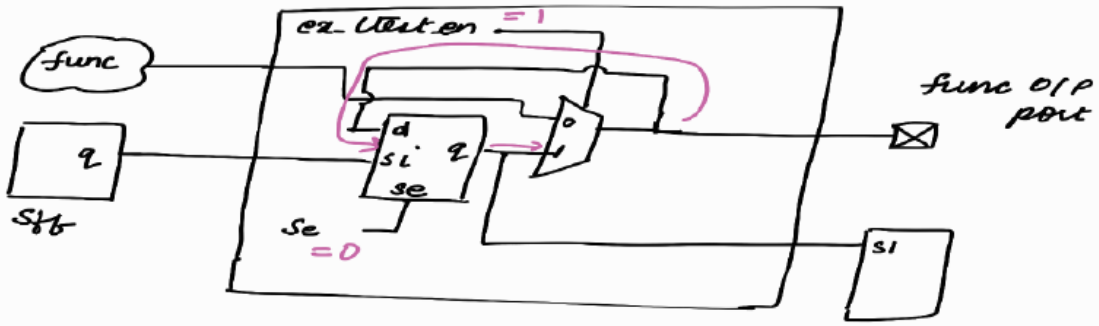


CAPTURE Se = 0

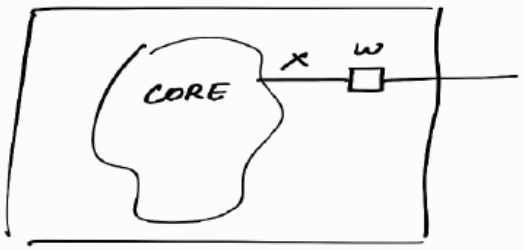


(i) int_mode ex_ctest_en = 0

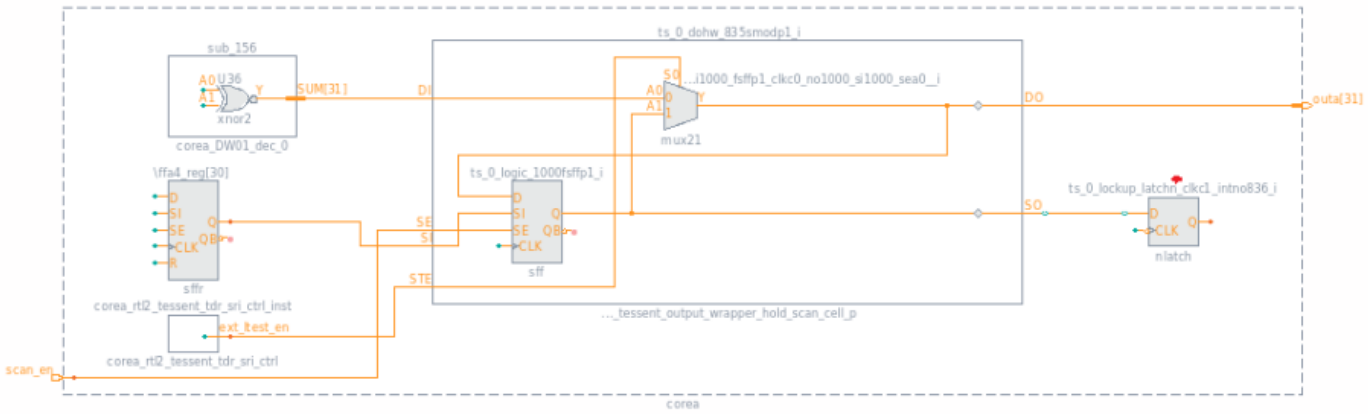




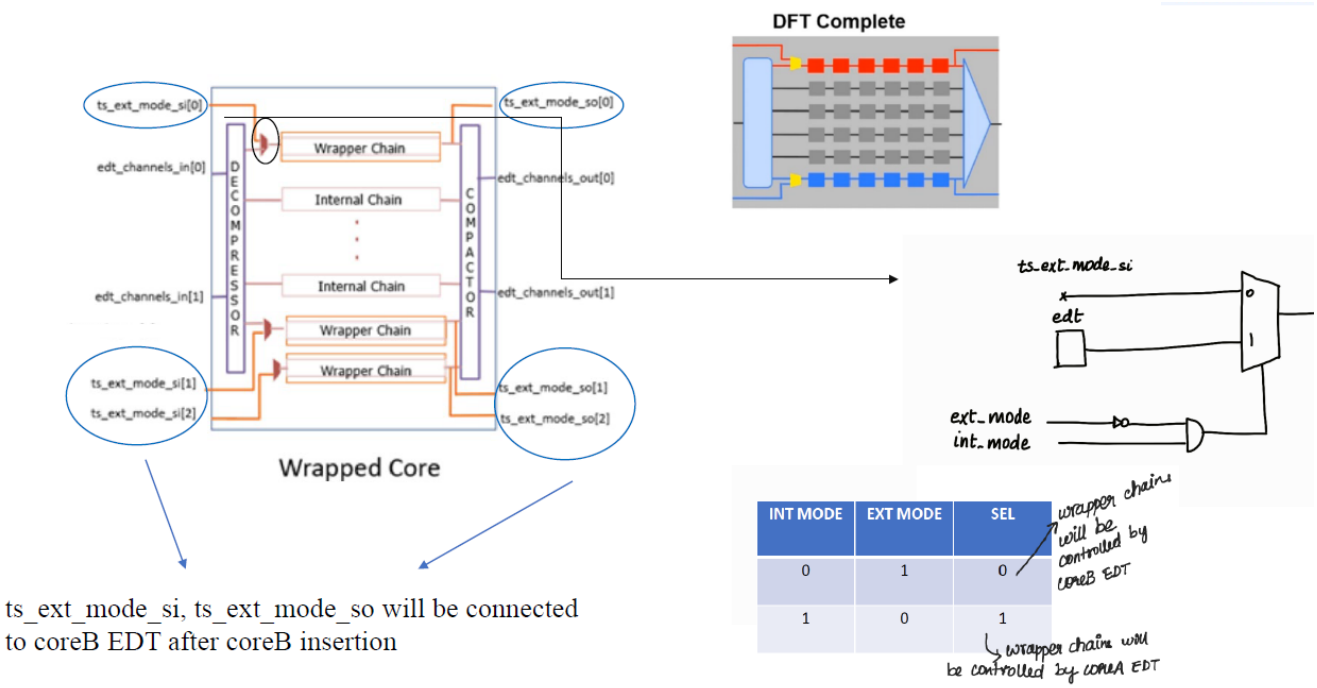
ext_mode :- ex_ctest_en = 1



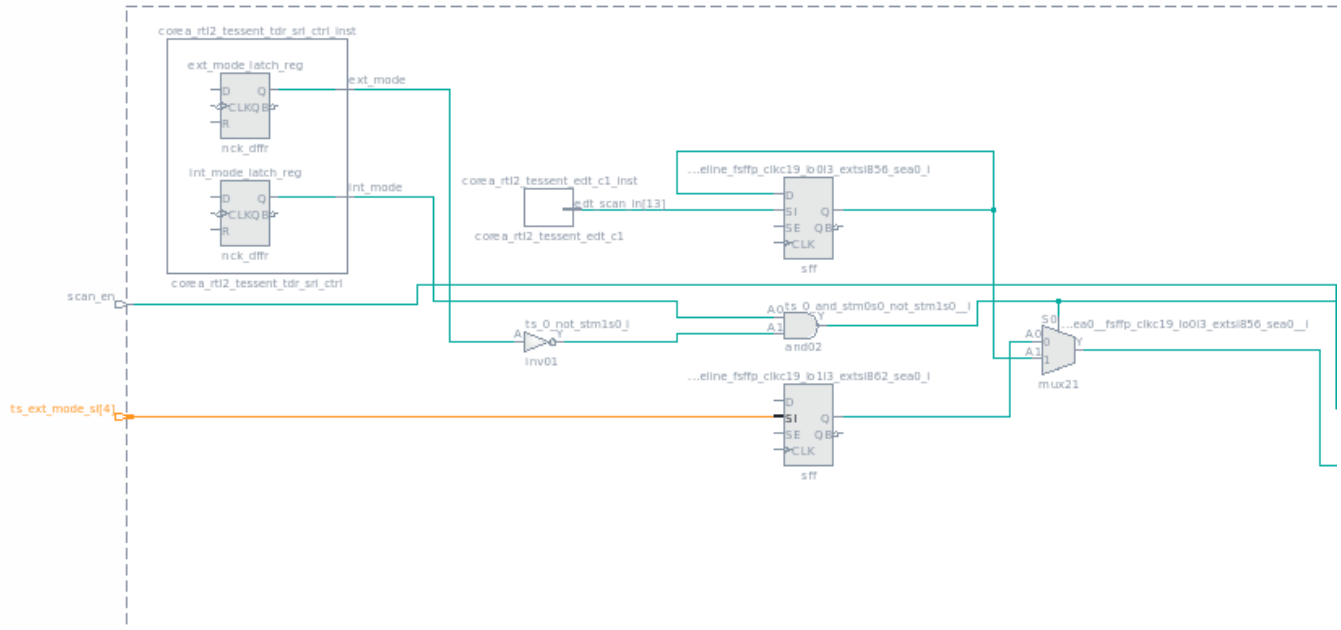
wrapper should not capture this x.



- **Wrapper Chain Schematic**

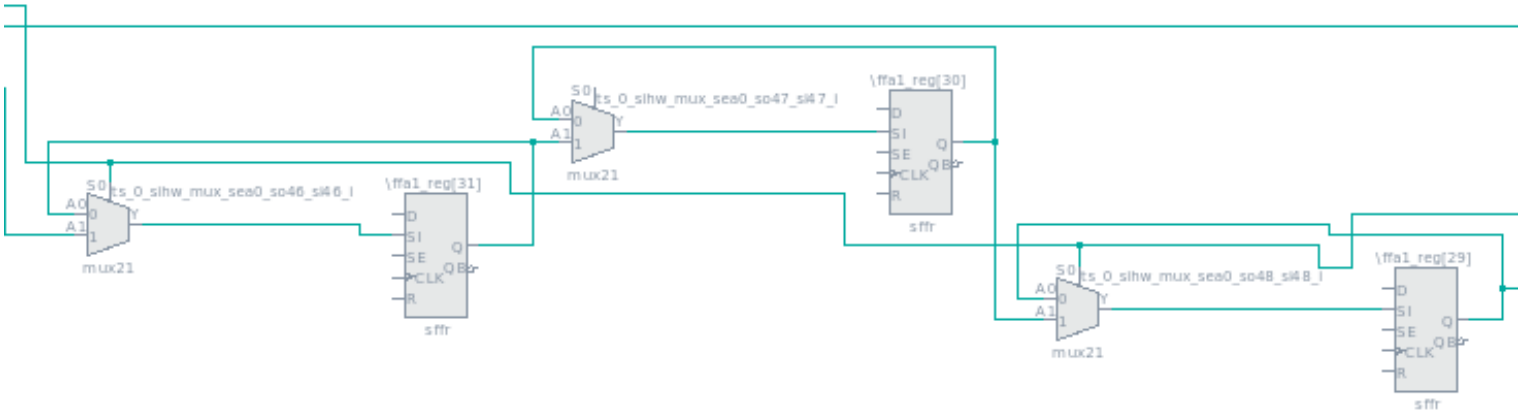


ts_ext_mode_si, ts_ext_mode_so will be connected to coreB EDT after coreB insertion

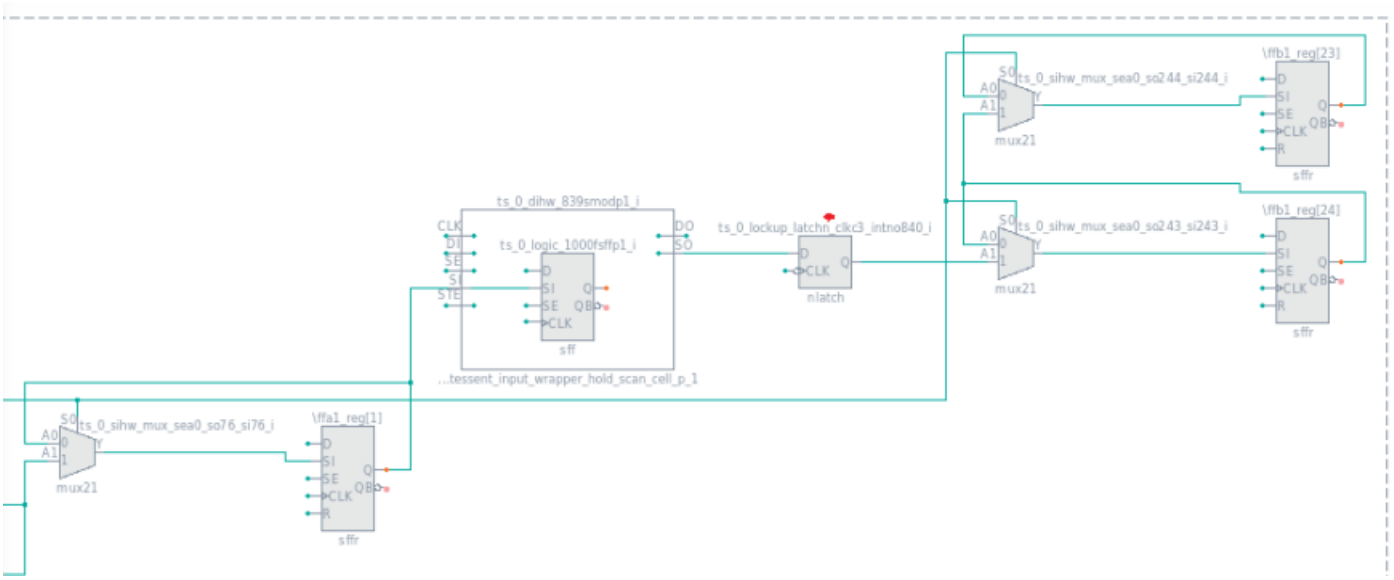


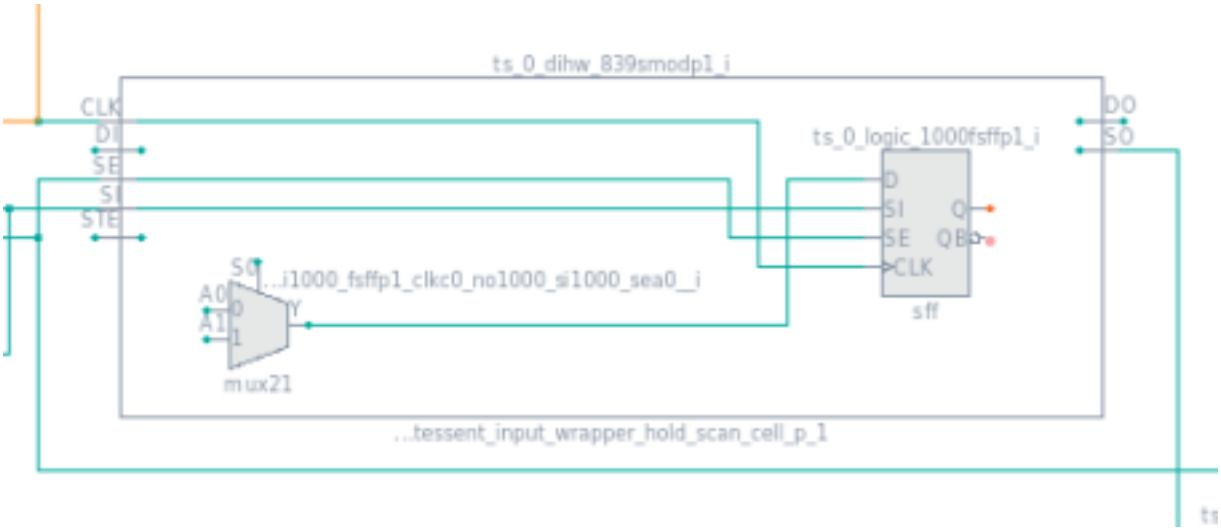
1st mux21 will be there in the starting of the wrapper chain, the select line of the mux will be connected to and gate where one input of and gate is connected to int_mode and

another one connected to OR'ed followed by ext_mode, 0th input of mux is connected to pipelined flop and 1st input of the mux through one pipelined logic will be connected to the CoreA EDT logic.
Output of the mux will be going to the input of the wrapper chains.



The above one is shared input wrapper, 0th input of the mux will be connected to the wrapper flop output
And 1st input of the mux will be coming from previous wrapper flop output and output of the mux will be going as SI to wrapper flop.

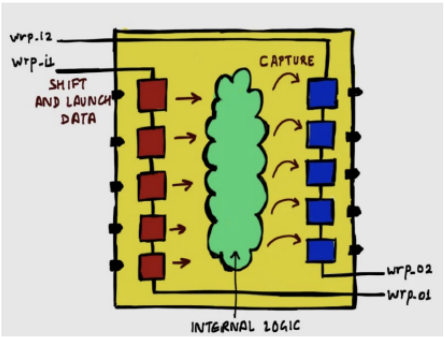


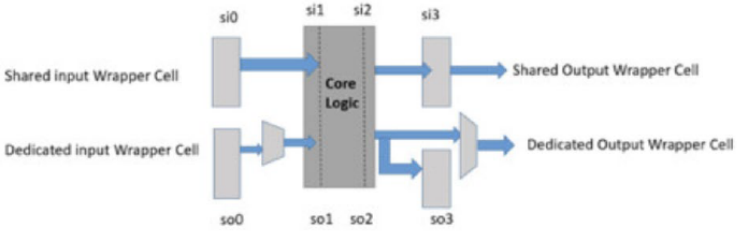


The above schematic is dedicated wrapper flop, The SO will be connected to lock up latch and again it will go to shared wrapper

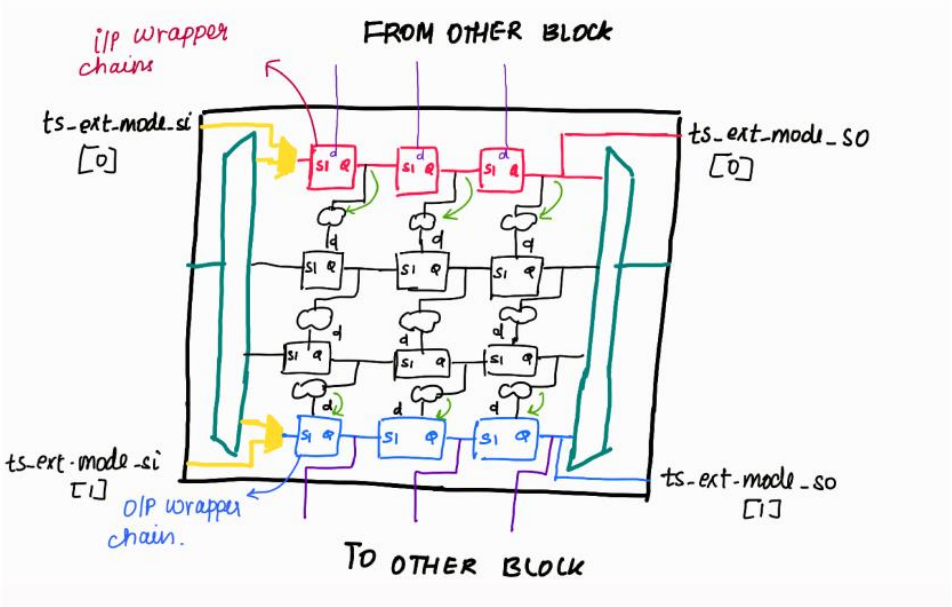
In INTEST mode, all inputs to submodules are controllable using the Input wrapper scan chains and all outputs are observable through the Output wrapper scan chains.

Input wrapper chains launch data into the inside logic and output wrapper chains capture data from inside logic.





(Figure [2]: Inward Facing (Intest) Mode)



ts_ext_mode_si, ts_ext_mode_so will be connected to coreB EDT after coreB insertion

Line 15: The tool will see if a PI if it is directly connected to a flop or if PI for some small combo logic if it is connected to a flop then the tool will consider/reuse that flop as a wrapper flop so it is assigning shared wrapper flop for the PI ports and even for PO ports. After this tool will see which are all ports have not been assigned with shared wrapper, so for the ports which is not been assigned with shared wrapper the tool will add a dedicated wrapper.

```
ANALYSIS> set_wrapper_analysis_options -exclude_ports [get_ports *edt_channel*]
ANALYSIS> set_dedicated_wrapper_cell_options on -ports rst
ANALYSIS> analyze_wrapper_cells
//
//      Port information and user constraints:
//      --------------------------------------
//          Input ports:    80 total,    13 ignored,     2 excluded,     0 off,     1 on,    64 auto
//         Output ports:    67 total,     1 ignored,     2 excluded,     0 off,     0 on,    64 auto
//
//
//      Wrapper analysis summary:
//      -------------------------
//         1 output port required a dedicated wrapper cell.
//         3 input ports required a dedicated wrapper cell.
//         65 flip-flops were converted into output shared wrapper cells.
//         62 flip-flops were converted into input shared wrapper cells.
//         Use report_wrapper_cells for more details.
```

Input Port: Tool will not be adding the edt channel for (2 excluded) for input ports and tool will be ignoring the IJTAG and Clk ports so the wrappers will not be added.
Output Port: (1 ignored) is TDO that is IJTAG output and the edt channel for (2 excluded).

Line 17,18: giving the instance name of this edt logic and from where we have got this? From the modified rtl of the edt insertion run

- **while doing edt insertion we have given that what is the longest scan chain and what is the number of scan chain and how many scan channels we want**

- **So all these we have already given in edt insertion run. So giving instance name (corea_rtl2_tessent_edt_c1_inst) the tool will automatically get the information from the previous run**
- **Int_mode scan chain will tell all the flops which is having in coreA logic**
- **Ext_mode scan chain will be having only wrapper flops**

Line 20: analyze_scan_chain (It will distribute the flops into different scan chains)

Line 21: insert_test_logic (It will replace the normal flops to scan flops and it will do scan chain stitching)

- **The wrapper flops will be having int mode chain and ext mode chain.**
- **The normal flop will be only present in int_mode_chain.**

Graybox:

GRAYBOX

- Graybox is a simplified representation of the core module describing its periphery logic.
- The graybox model is used in place of full core netlist in any situation in which only the logic at the boundary of the core is needed.
- The inserted wrapper chains of the block will be included in the graybox.

- In graybox netlist we have only information about the wrapper chain
- While we are doing coreb level insertion we do not require the complete logic of corea we only require wrapper chains information of corea,
- so while doing the coreb level insertion we will be reading the graybox netlist of corea

GRAYBOX GENERATION COMMANDS

- **get_config_elements Core(corea)/Scan/Mode -part tcd –silent**
  get_config_elements : Returns a collection of configuration elements. We will be getting it from our TCD file of scan.
- **get_config_value :** Returns a value associated with a configuration element based on the specified option.
- **import_scan_mode :** Imports configuration settings from tcd_scan.
- **set_attribute_value [get_ports *edt_channel*] -name ignore_for_graybox :**
  Sets an attribute's value.
  ignore_for_graybox : Setting this attribute on any pin of a module has no effect on graybox analysis.
- **analyze_graybox :** Identifies the instances and nets to be included in the graybox netlist.
- **write_design -tsdb -graybox –verbose :** write the graybox netlist into tsdb output directory.

mode_name : int_mode, ext_mode
mode_type : internal, external

```
23 set_context patterns -scan
24
25 foreach_in_collection mode_wrapper [get_config_elements Core(corea)/Scan/Mode -part tcd -silent] {
26   set mode_name [get_config_value $mode_wrapper -id <0>]
27   set mode_type [get_config_value type -in $mode_wrapper]
28   import_scan_mode $mode_name
29   #In setup mode
30   report_clocks
31   check_design_rules
32   #In Analysis mode
33   report_clocks
34   if {$mode_type eq "external"} {
35     report_scan_cells
36     set_attribute_value [get_ports *edt_channel*] -name ignore_for_graybox
37     analyze_graybox
38     write_design -tsdb -graybox -verbose
39   }
40   set_system_mode setup
41 }
```

Line 25: inside the Core(corea) we have to give the current design that is corea.
Line 26: what is config value? Its a configuration element based on the specified option. What is there in mode_name? that is int_mode and ext_mode
Line 27: mode_type will be holding internal and external
Line 36: it will not be a part of graybox netlist
Line 37: edt channel will not included in the graybox netlist, it will do analysis for the grayblox netlist which are all the nets and instances have to be included as the graybox netlist.
Line 38: the graybox will be generated and it will automatically written out into the tsdb output directory4

STEP 5: ATPG

Intest:

```
1 set_context patterns -scan
2 set_tsdb_output_directory ../../../tsdb_outdir
3
4 read_cell_library ../../../../../library/adk.tcelllib
5 read_verilog ../../../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary
6
7 read_design corea -design_id gate -verbose
8 set_current_design corea
9
10 set_current_mode edt_stuck_intest -type internal
11 import_scan_mode int_mode -fast_capture_mode off
12
13 add_black_boxes -auto
14 add_output_masks -all
15 check_design_rules
16
17 set_fault_type stuck
18
19 create_patterns
20 write_tsdb_data -replace
21
22 system mkdir -p -v simulation
23
24 write_patterns /hdd2/home/vidishar/aug23/level3/Lab3/corea/5.atpg/sa/intest/simulation/corea_int_mode_parall
   el.v   -verilog -parallel -replace -parameter_list {SIM_TOP_NAME parallel_TB SIM_KEEP_PATH 1 SIM_STATUS_MSG
   1}
```

Intest is to test the internal logic of the block
And extest is to test the logic outside the block

Line 7: scan insertion design id
Line 8: giving top module name for doing the elaboration
Line 10:
Line 11: FCM in the OCC while theoretical concepts of OCC, if it is stuck it will be zero
Line 14:

```
Fault Sub-classes
  -------------------------------
  AU (atpg_untestable)
    UDN   (undriven)                  64 ( 0.13%)      same ( 0.14%)
    EDT   (edt_blocks)               199 ( 0.40%)        deleted
    PC*   (pin_constraints)          623 ( 1.26%)      same ( 1.33%)
    TC*   (tied_cells)               293 ( 0.59%)      same ( 0.62%)
    MPO   (mask_po)                  266 ( 0.54%)      same ( 0.57%)
    SEQ   (sequential_depth)          20 ( 0.04%)      same ( 0.04%)
    OCC   (on_chip_clock_control)   1707 ( 3.45%)        deleted
    IJTAG (ijtag)                    718 ( 1.45%)        deleted
    Unclassified                     169 ( 0.34%)      same ( 0.36%)
  UC+UO
    AAB   (atpg_abort)                 2 ( 0.00%)      same ( 0.00%)
    UNS   (unsuccess)                  9 ( 0.02%)      same ( 0.02%)
    EAB   (edt_abort)                 19 ( 0.04%)      same ( 0.04%)
  *Use "report_statistics -detailed_analysis" for details.
----------------------------------------------------------------------
Coverage
  -------------------------------
  test_coverage                      91.35%            96.72%
  fault_coverage                     87.25%            92.13%
  atpg_effectiveness                 99.94%            99.94%
----------------------------------------------------------------------
#test_patterns                                          312
  #basic_patterns                                       121
  #clock_sequential_patterns                            191
#simulated_patterns                                     325
CPU_time (secs)                                        11.1
----------------------------------------------------------------------
```

For simulation

```
1 rm -rf work
2
3 vlib work
4
5 vlog corea_int_mode_parallel.v -f verilog_files.list +nospecify -override_timescale 1ns/1ps -work work -l co
  mpile.log
6
7 vsim  -voptargs=+acc parallel_TB -do "run -all" -c +nospecify -l simulation.log
```

Line 5 is TB file name and line 7 is top module name

```
# Simulated         308 patterns
#
# Simulated         309 patterns
#
# Simulated         310 patterns
#
# Simulated         311 patterns
#
# Simulated         312 patterns
#
# No error between simulated and expected patterns
#
# ** Note: $finish    : corea_int_mode_parallel.v(5201)
#    Time: 49522 ns  Iteration: 0  Instance: /parallel_TB
# End time: 15:36:28 on Jan 22,2024, Elapsed time: 0:00:02
# Errors: 0, Warnings: 12
```

Extest:

During the extest, the TB will write the value at the Primary input to make it controllable
The actual testing of interconnection will happen at CoreB level

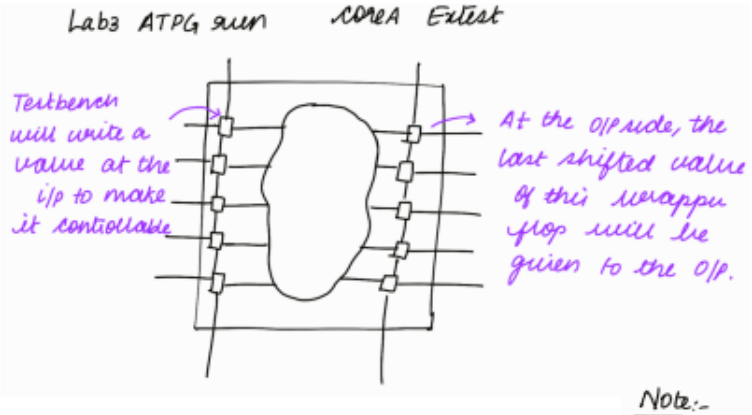
```
1 set_context patterns -scan
2 set_tsdb_output_directory ../../../tsdb_outdir
3
4 read_cell_library ../../../../../library/adk.tcelllib
5 read_verilog ../../../../../library/mems/SYNC_1R1W_16x8.v -exclude_from_file_dictionary
6
7 read_design corea -design_id gate -verbose
8 set_current_design corea
9
10 set_current_mode edt_stuck_extest -type external
11 import_scan_mode ext_mode -fast_capture_mode off
12
13 add_black_boxes -auto
14 add_output_masks -all
15
16 check_design_rules
17 report drc rules
18
19 set_fault_type stuck
20
21 read_faults /hdd2/home/vidishar/aug23/level3/Lab3/corea/tsdb_outdir/logic_test_cores/corea_gate.logic_test_core/corea
   g_mode_edt_stuck_intest/corea_edt_stuck_intest_stuck.faults.gz -protect -retain
22
23 create_patterns
24 stop
25 write_tsdb_data -replace
26
27 system mkdir -p -v simulations
28
29 write_patterns /hdd2/home/vidishar/aug23/level3/Lab3/corea/5.atpg/sa/extest/simulations/corea_ext_mode_parallel.v
   log -parallel -replace -parameter_list {SIM_TOP_NAME parallel_TB SIM_KEEP_PATH 1 SIM_STATUS_MSG 1}
```

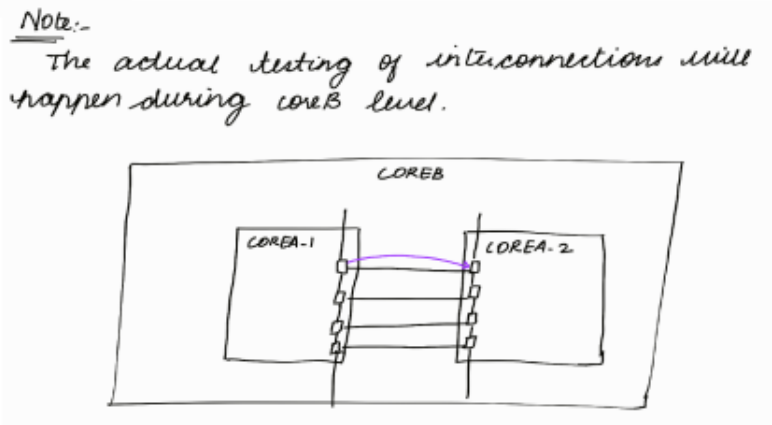
Only mode name will be changed (edt stuck extest)
Line 25: tsdb_outdir - It will write out the fault list, pat db pattern, flat model everything into the tsdb

```
                        Statistics Report
                         Stuck-at Faults
-------------------------------------------------------------------------
Fault Classes                        #faults          #faults
                                     (total)      (total relevant)
-----------------------------  ---------------- --------------------
  FU (full)                          49506             46960
  ---------------------------  ---------------- --------------------
  DS (det_simulation)              1036  ( 2.09%)       same  ( 2.21%)
  DI (det_implication)                8  ( 0.02%)       same  ( 0.02%)
     (protected)                  43192  (87.25%)       same  (91.98%)
  PU (posdet_untestable)              4  ( 0.01%)       same  ( 0.01%)
  PT (posdet_testable)               15  ( 0.03%)       same  ( 0.03%)
  UU (unused)                      1376  ( 2.78%)       same  ( 2.93%)
  TI (tied)                         285  ( 0.58%)       same  ( 0.61%)
  BL (blocked)                      205  ( 0.41%)       same  ( 0.44%)
  RE (redundant)                    355  ( 0.72%)       same  ( 0.76%)
  AU (atpg_untestable)             3030  ( 6.12%)        484  ( 1.03%)
-------------------------------------------------------------------------
Fault Sub-classes
  ---------------------------
  AU (atpg_untestable)
    UDN   (undriven)                 64  ( 0.13%)       same  ( 0.14%)
    PC*   (pin_constraints)           4  ( 0.01%)       same  ( 0.01%)
    TC*   (tied_cells)              106  ( 0.21%)       same  ( 0.23%)
    MPO   (mask_po)                 266  ( 0.54%)       same  ( 0.57%)
    SEQ   (sequential_depth)         22  ( 0.04%)       same  ( 0.05%)
    IJTAG (ijtag)                  2546  ( 5.14%)         deleted
    Unclassified                     22  ( 0.04%)       same  ( 0.05%)
  *Use "report_statistics -detailed_analysis" for details.
-------------------------------------------------------------------------
Coverage
  ---------------------------
  test_coverage                      93.57%              98.90%
  fault_coverage                     89.37%              94.22%
  atpg_effectiveness                 99.98%              99.98%
-----------------------------
  Protected Faults alone:
  test_coverage                      91.34%              96.54%
  fault_coverage                     87.25%              91.98%
-------------------------------------------------------------------------
#test_patterns                                              22
  #basic_patterns                                           12
  #clock_sequential_patterns                                10
#simulated_patterns                                         48
CPU_time (secs)                                            5.0
-------------------------------------------------------------------------
```

So, here intest and extest will give the complete coverage …. So here we are reading the corea intest faults so this will stored into tsdb output directory(line 21)

- Corea stuckat intest faults have been read.
- And because TC and MPO, we are getting the coverage drop

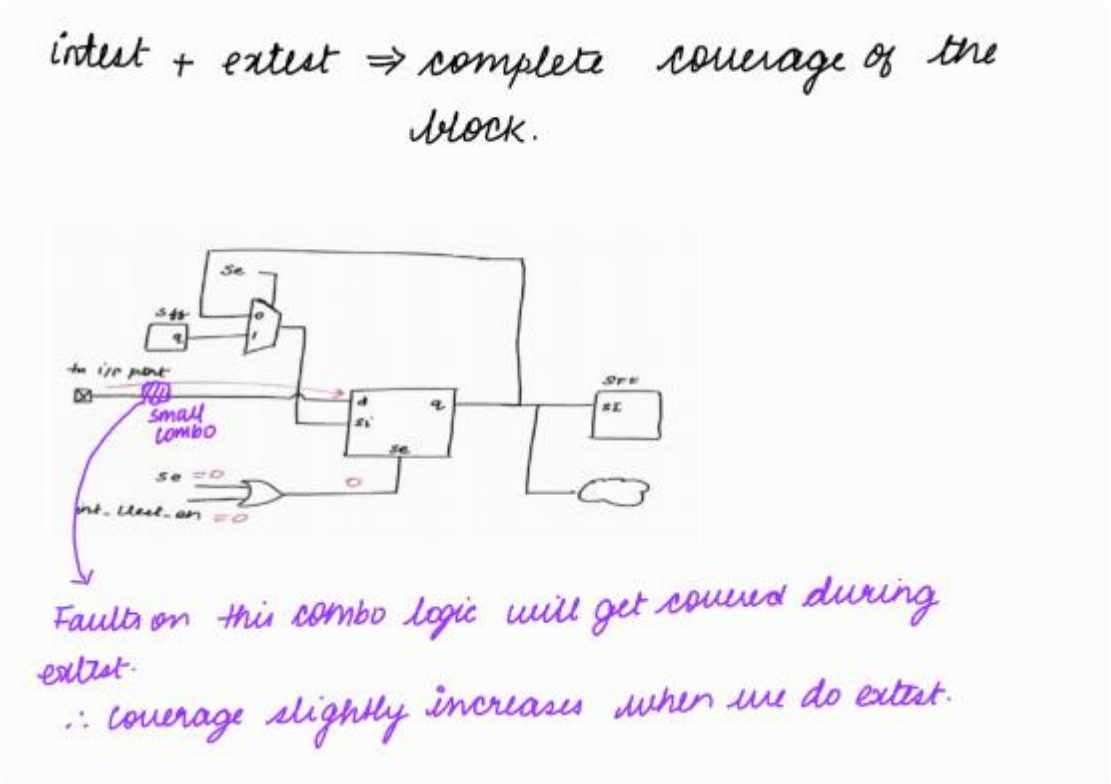




Here the Lab3 ATPG CoreA Extest run, just the TB is driving the value at the input to make in controllable and at the output side

Output of the wrapper flop will be given to the output port but actual testing of the interconnection will be happening in CoreB level..

- During CoreB level, the last shifted value of CoreA-1 will be captured in CoreA-2 level and during the shift out phase of the flop it will be compared with the expected value\
- SO actual interconnection of the testing will be happening in the CoreB level



Shared Input wrapper through functional input of a small combo logic if it gets connected to a flop then tool will reuse the same flop as a wrapper flop.

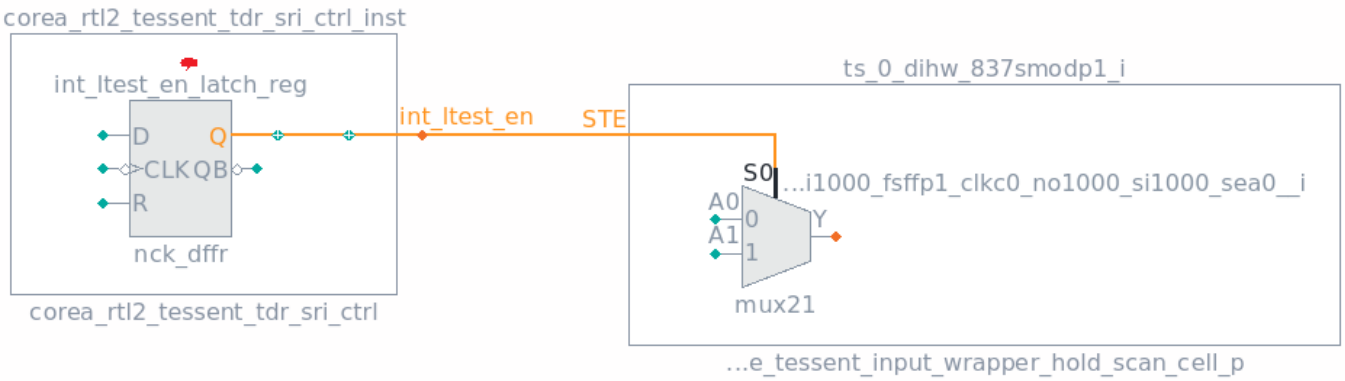
So while running extest, the tool will write the value at the input to make it controllable and it will be captured at the wrapper flop, so that is why the faults on this combo logic is getting covered by the while we do the extest run.

- So during extest, the tool that is TB will write the value into input to make in controllable, so this value through the combo logic it is getting captured into the wrapper flop, so that is why faults onto this combo logic it is getting covered while we do the extest runs, so that is why extest coverage is getting increased.

```
ANALYSIS> write_faults AU_TC.txt -class AU.TC -rep
ANALYSIS> system gvim AU_TC.txt
Gtk-Message: 12:33:11.782: Failed to load module "canberra-gtk-module"
0
```

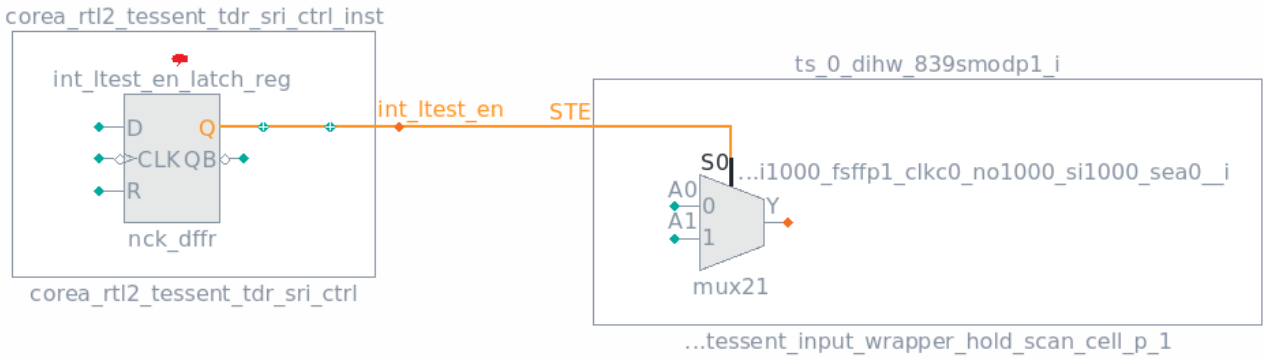
```
FaultInformation {
 version : 1;
 FaultType (Stuck) {
  FaultList {
   FaultCollapsing : FALSE;
   Format : Identifier, Class, Location;
   Instance ("") {
      1,  AU.TC,     "/tessent_persistent_cell_edt_clock/FE";
      1,  EQ,        "/tessent_persistent_cell_edt_clock/TE";
      0,  AU.TC,     "/tessent_persistent_cell_edt_clock/CK";
      1,  AU.TC,     "/tessent_persistent_cell_edt_clock/CK";
      0,  AU.TC,     "/U64/Y";
      1,  EQ,        "/U64/A";
      0,  AU.TC,     "/ts_0_dihw_837smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_dihw_839smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_dihw_841smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_cgfix_or_sea0_extstes0_i/A1";
      0,  AU.TC,     "/ts_0_or_sea0_intstes0_i/A1";
      1,  AU.TC,     "/add_187/U2/A2";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/U16/Y";
      1,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U16/A";
      1,  EQ,        "/corea_rtl1_tessent_sib_sti_inst/U26/Y";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/U14/Y";
      0,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U14/A";
      0,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U15/Y";
      1,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U15/A";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/S0";
      0,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/A0";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/A0";
      1,  AU.TC,     "/ram2_interface_inst/U30/Y";
      1,  EQ,        "/ram2_interface_inst/U30/A";
      0,  AU.TC,     "/ram2_interface_inst/U5/Y";
      1,  EQ,        "/ram2_interface_inst/U5/A";
```

The TDRs will be loaded with the suitable value in test setup phase and TDRs will be controlled during the static signals.
The static signals will be coming in AU.TC



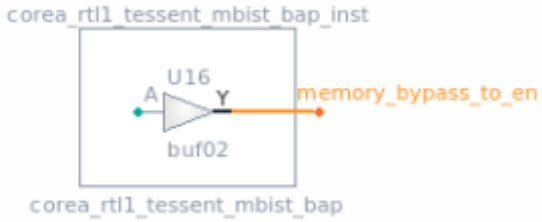
Int_ltest_en it will be zero during the ext mode and it will always be 0 because it is a static signal. So the tool will not be able to detect the stuckat zero on this particular net.

- Because in order to detect the stuckat 0 the tool will try to make it as one which it cannot make becausee this is stuckat zero.



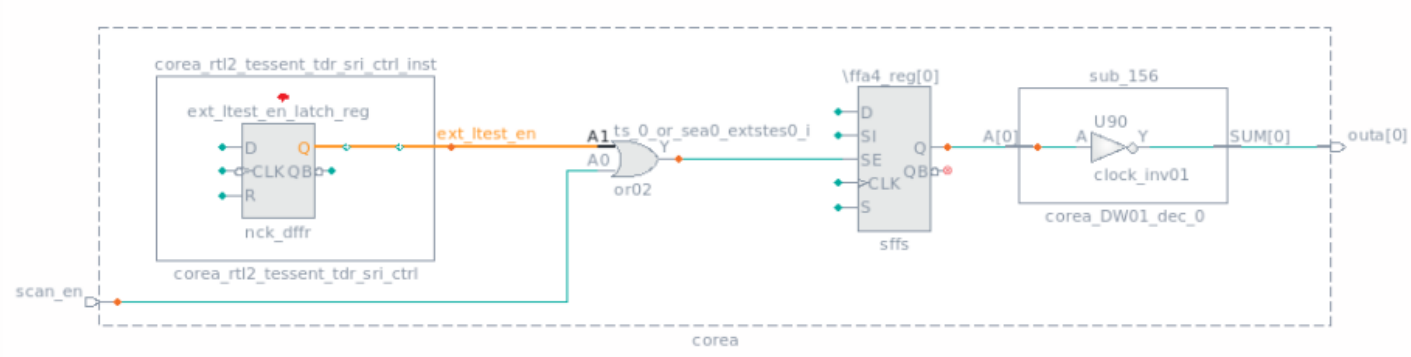

```
FaultInformation {
 version : 1;
 FaultType (Stuck) {
  FaultList {
   FaultCollapsing : FALSE;
   Format : Identifier, Class, Location;
   Instance ("") {
      1,  AU.TC,     "/tessent_persistent_cell_edt_clock/FE";
      1,  EQ,        "/tessent_persistent_cell_edt_clock/TE";
      0,  AU.TC,     "/tessent_persistent_cell_edt_clock/CK";
      1,  AU.TC,     "/tessent_persistent_cell_edt_clock/CK";
      0,  AU.TC,     "/U64/Y";
      1,  EQ,        "/U64/A";
      0,  AU.TC,     "/ts_0_dihw_837smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_dihw_839smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_dihw_841smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0";
      0,  AU.TC,     "/ts_0_cgfix_or_sea0_extstes0_i/A1";
      0,  AU.TC,     "/ts_0_or_sea0_intstes0_i/A1";
      1,  AU.TC,     "/add_187/U2/A2";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/U16/Y";
      1,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U16/A";
      1,  EQ,        "/corea_rtl1_tessent_sib_sti_inst/U26/Y";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/U14/Y";
      0,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U14/A";
      0,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U15/Y";
      1,  EQ,        "/corea_rtl1_tessent_mbist_bap_inst/U15/A";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/S0";
      0,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/A0";
      1,  AU.TC,     "/corea_rtl1_tessent_mbist_bap_inst/tdr_inst/tessent_persistent_cell_BIST_ASYNC_RESET_mux/A0";
      1,  AU.TC,     "/ram2_interface_inst/U30/Y";
      1,  EQ,        "/ram2_interface_inst/U30/A";
      0,  AU.TC,     "/ram2_interface_inst/U5/Y";
      1,  EQ,        "/ram2_interface_inst/U5/A";
```

Stuck at 0 is not being detected -
(ts_0_dihw_837smodp1_i/ts_0_logic_1000mux_stes0_di1000_fsffp1_clkc0_no1000_si1000_sea0__i/S0)

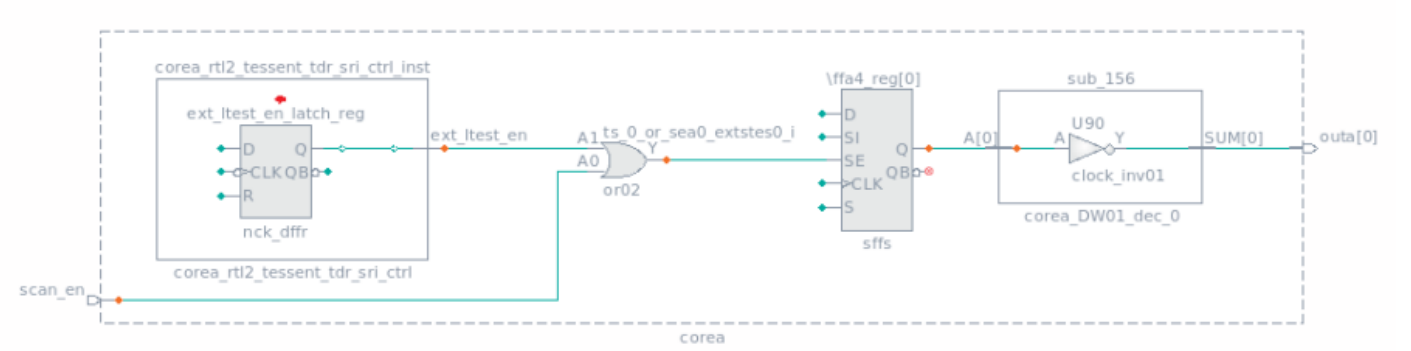


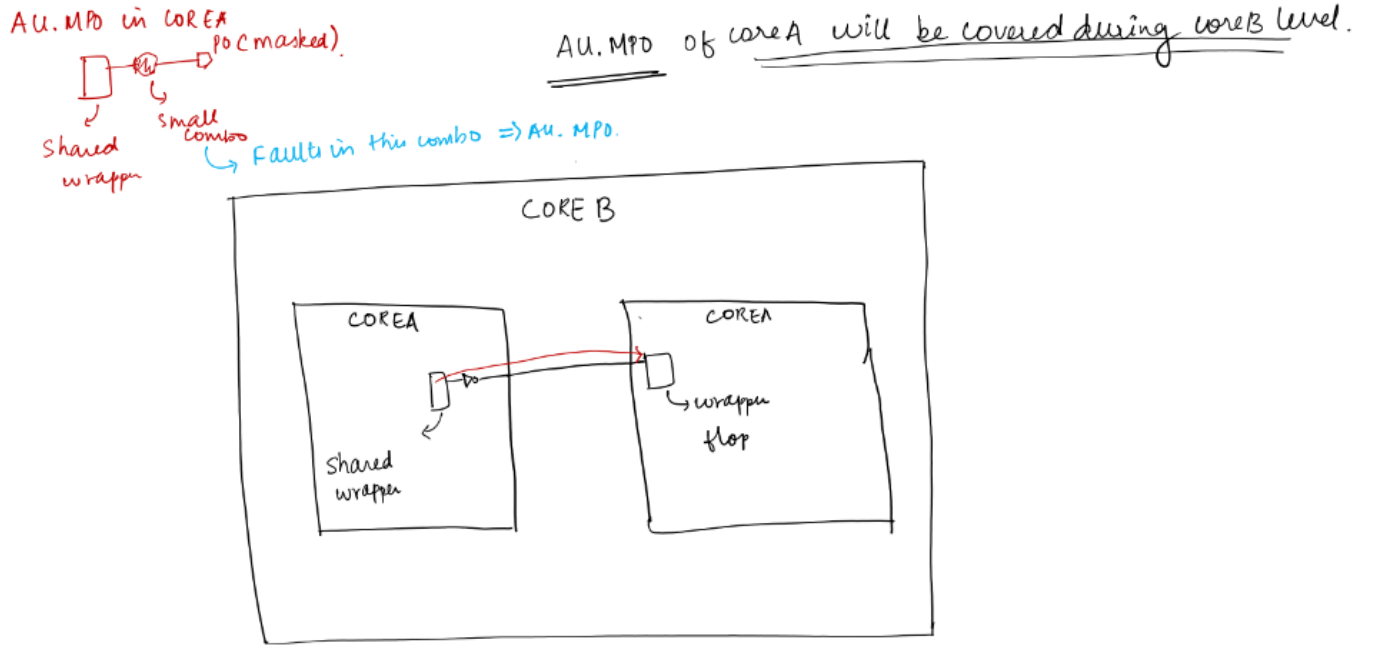
Memory bypass should be one during ATPG time, so that's why stuck at one cannot be detected, so static signals will be coming in AU.TC and we have to accept the coverage drop, it has to be one during the ATPG time
So it has to be one during atpg time and we cannot make it zero so stuck at one during memory bypass enable. So we cannot change the value of static signals.
If we make Memory bypass as zero, then memory won't be by passed.

Shared output wrapper flop:-



AU_MPO.txt





- **The last shifted value of the wrapper through the combo logic is being captured into the wrapper flop and later during shift out phase the wrapper flop will be compared with the expected value.**
- **0.57% will be covered in the CoreB level**

Simulations:-

```
# Begin chain test
#
# Simulated           1 patterns
#
# End chain test
#
# No error between simulated and expected patterns
#
# ** Note: $finish    : corea_ext_mode_parallel.v(2233)
#    Time: 1402 ns  Iteration: 0  Instance: /parallel_TB
# End time: 15:26:27 on Jan 28,2024, Elapsed time: 0:00:02
# Errors: 0, Warnings: 5
```

STEP 6: MBIST patterns on netlist

While doing mbist insertion we would have written the patterns and those patterns would have been validated but this is done in RTL stage.

- **So here ICL Pattern, MBIST patterns, and parallel retention patterns. (All this three has been validated during MBIST type)**

SO why we have to validate the MBIST patterns.??

- **So we have validate the patterns on the netlist.**

```
1 set_context patterns
2 set_tsdb_output_directory ../tsdb_outdir
3
4 read_cell_library ../../../library/adk.tcelllib
5
6 read_design corea -design_id gate -view interface -verbose
7 set_current_design corea
8
9 set spec [create_patterns_spec]
10 report_config_data $spec
11 process_patterns_specification
12
13 set_simulation_library_sources -v ../../../library/adk.v -y ../../../library/mems/ -extension v
14 run_testbench_simulations
15
```

The same set of commands from line 9 to line line 14 only thing is that, the 1.insert_mbist will be validating at RTL level and on Mbist patterns on netlist will be validating it on the netlist.

```
Starting 3 simulations for ./simulation_outdir/corea_gate.simulation_signoff
//  Waiting for the simulation(s) to complete

unscheduled 0 queued 0 running 0 pass 3 fail 0
```