Lab 6: Formal verification

Objective

Learn how to use Formality to detect unexpected differences that may have been introduced into a design during development.

Reports

Please include the formality results and any problems and findings in the report, especially:

1. Screenshot of design after each step
2. Generated scripts & reports

Introduction

The purpose of Formality is to detect unexpected differences that may have been introduced into a design during development.

In Formality the following concepts are used:

Reference design: This design is the golden design, the standard against which Formality tests for equivalence.

Implementation design: This design is the changed design. It is the design whose accuracy you want to prove. For example, a newly synthesized design is an implementation of the source RTL design.

To start Formality, enter the following command at the terminal:

```
% fm_shell
fm_shell (setup)>
```

The fm_shell command starts the Formality shell environment. From here, start the graphical user interface (GUI) as follows:

```
fm_shell (setup)> start_gui
```

When Formality is invoked, begin in the **setup** mode. The **setup** indicates the mode that currently in when using commands. The modes are **setup**, **match**, and **verify**.

This laboratory work includes the following sections:

0. Guidance (Load Automated Setup File)

1. Reference (Specify the Reference Design)

2. Implementation (Specify the Implementation Design)

3. Set up (Set Up the Design)

4. Match (Match Compare Points)

5. Verify (Verify the Designs)

6. Debug.

Laboratory Tasks

Preparation

Copy the lab files and change the directory to lab06

```
$ cp -a /home/tools/synopsys/m3/lab06 $HOME/icdesign/m3
$ cd $HOME/icdesign/m3/lab06
$ source /home/tools/synopsys/env.sh
```

Rerun the synthesis and pnr steps

1. Run the synthesis with the following command

The following command will run the synthesis that we did from the previous lab.

```
$ cd $HOME/icdesign/m3/lab06/syn
$ dc_shell -f scripts/compile.tcl 2>&1 | tee run.log
```

2. Run the floorplan, placement and route with the following command

The following command will run the synthesis that we did from the previous lab.

```
$ cd $HOME/icdesign/m3/lab06/pnr
$ icc2_shell -f scripts/flow.tcl 2>&1 | tee run.log
```

3. Open the cell after Place & Route and write out the necessary file

```
$ icc2_shell -gui
```

Click on “open existing blocks”. Select the library and open the block “11_i2c_master_top_route_finish”

Run the following command to write out necessary files for primetime:

```
icc2_shell> write_verilog ../results/i2c_master_top.pnr-nopg.v -exclude pg_objects
```

Guidance (Load Automated Setup File)

Before specifying the reference and implementation designs, an automated setup file (.svf) can be optionally loaded into Formality. The automated setup file helps Formality process design changes caused by other tools used in the design flow. Formality uses this file to assist the compare point matching and verification process. For each automated setup file that is loaded, Formality processes the content and stores the information for use during the name-based compare point matching period.

If Formality is given the result of DC, then work in the **results** directory, and if Formality is given the result of ICC, then work in the **results** directory with the suffix pnr. This lab shows an example where Formality works with the result of ICC. To run Formality, which works with the result of DC is similar to this lab steps.

Start Formality graphical user interface (GUI) from fm directory. To start it, run:

```
% fm_shell -gui
```

This opens the Formality top-level GUI window (Fig. 1).

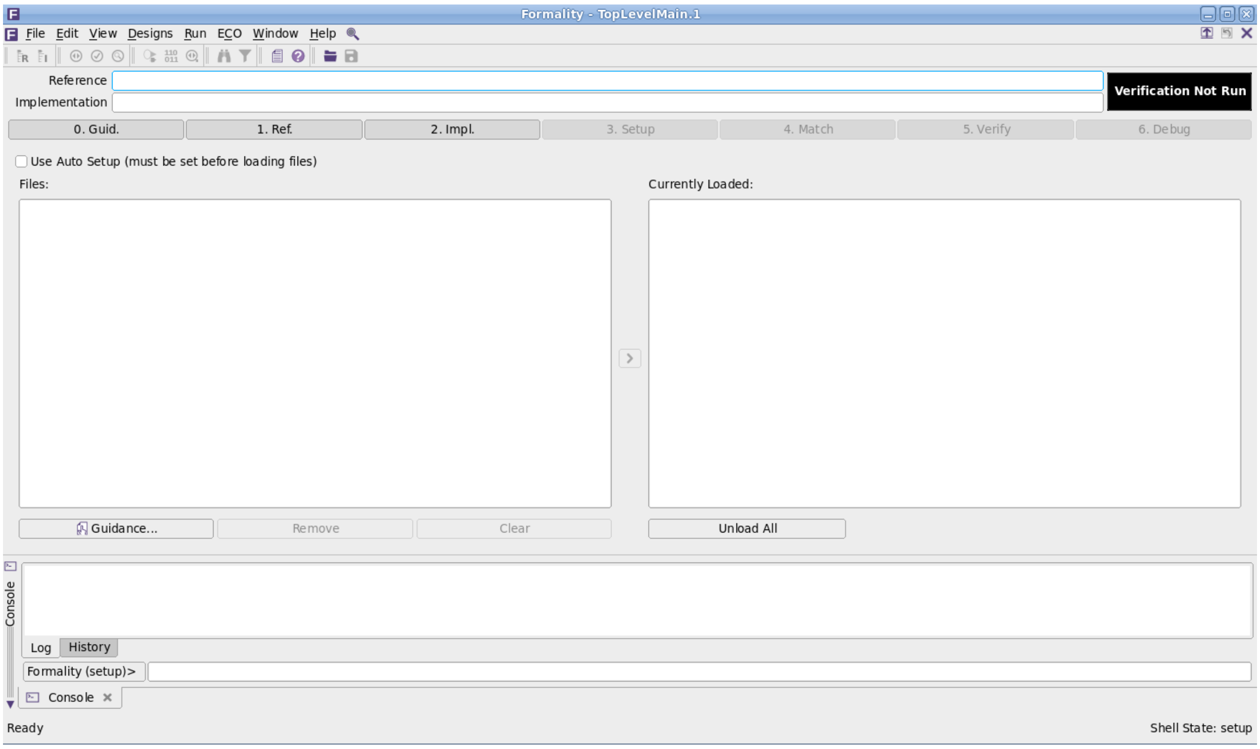


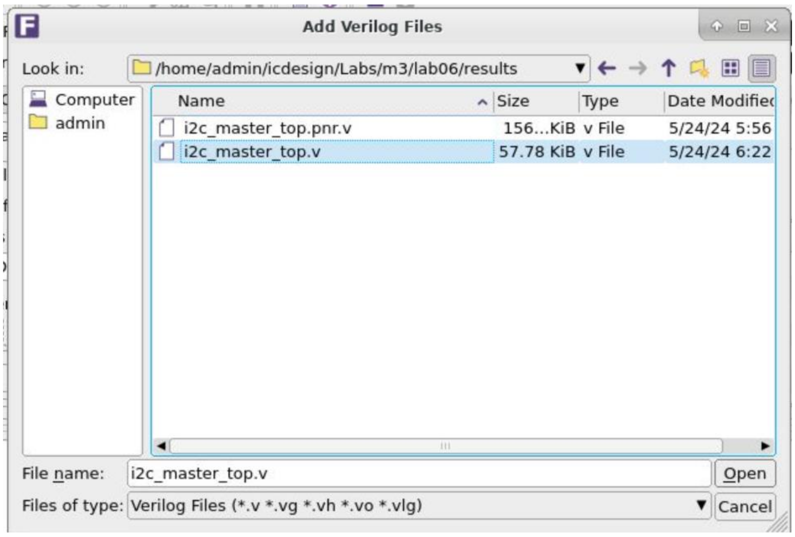
Fig.1. The Main Window of Formality GUI

1. Reference (Specify the Reference Design)

Specifying the reference design involves reading in design files, optionally reading in technology libraries, and setting the top-level design.

Click on **1. Ref. >> 1. Read Design File >> Verilog.** Then click on the **verilog** button under **files.** A dialog will appear as in Fig. 2.

The reference design is the design against which the transformed (implementation) design is compared. The reference design is the RTL source file named i2c_master_top.v file in the **lab06/results** folder. Click Open (Fig 2).



Look in: /home/admin/icdesign/Labs/m3/lab06/results

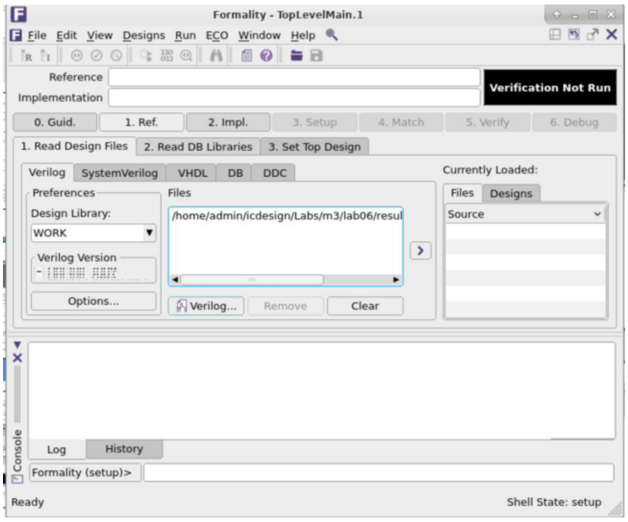
Name	Size	Type	Date Modified
i2c_master_top.pnr.v	156...KiB	v File	5/24/24 5:56
i2c_master_top.v	57.78 KiB	v File	5/24/24 6:22

File name: i2c_master_top.v

Files of type: Verilog Files (*.v *.vg *.vh *.vo *.vlg)

Fig. 2. Set up DC gate level Source

Click on the arrow to load the file as in Fig. 3.



WORK

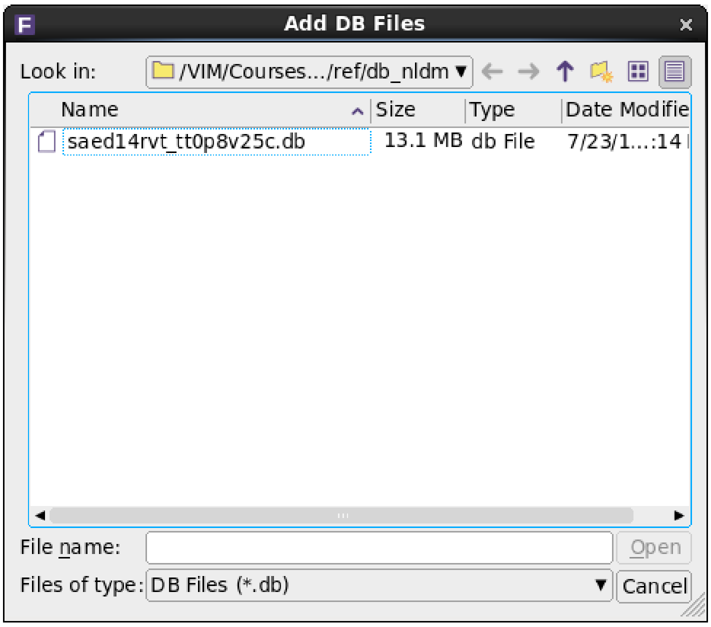
/home/admin/icdesign/Labs/m3/lab06/resul

Source

Formality (setup)>

Fig.3. Load the synthesized design

Source the DB file by clicking on **2. Read DB Libraries** >> **Select /home/dkits/SAED14nm/stdcell_rvt/db_nldm/saed14rvt_tt0p8v25c.db** file click **Open** as in Fig. 4.



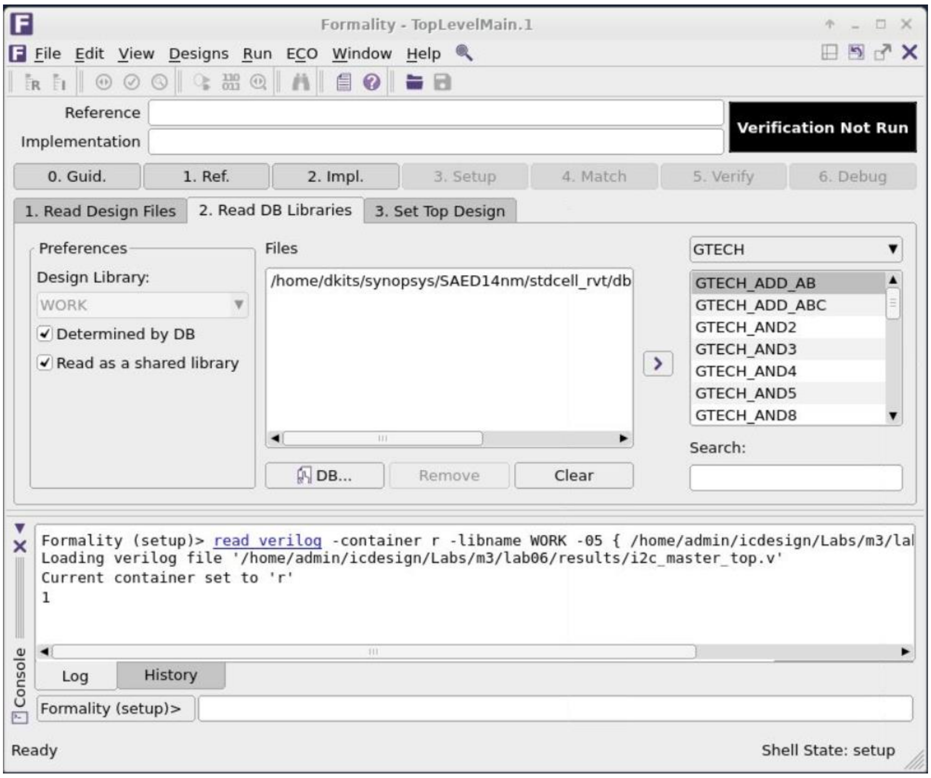
/VIM/Courses.../ref/db_nldm

Name	Size	Type	Date Modifie
saed14rvt_tt0p8v25c.db	13.1 MB	db File	7/23/1...:14

DB Files (*.db)

Fig. 4. Set up .db file.

Click on the arrow to load the db file as in Fig. 5.



- Design Library: WORK
- ☑ Determined by DB
- ☑ Read as a shared library

Files: /home/dkits/synopsys/SAED14nm/stdcell_rvt/db

GTECH

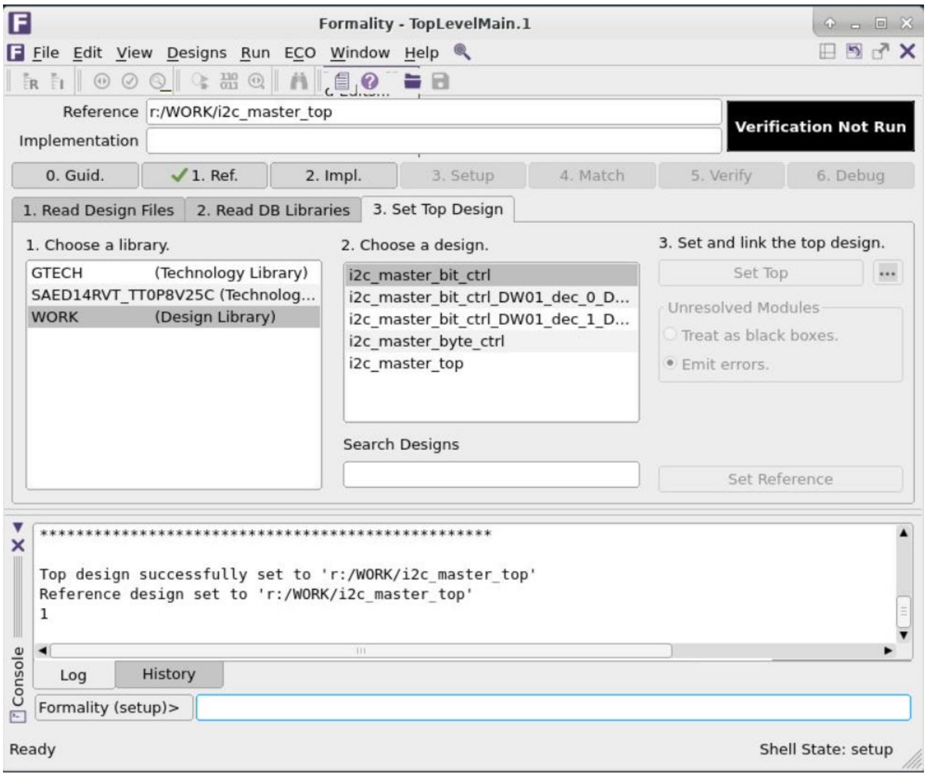
GTECH_ADD_AB
GTECH_ADD_ABC
GTECH_AND2
GTECH_AND3
GTECH_AND4
GTECH_AND5
GTECH_AND8

```
Formality (setup)> read_verilog -container r -libname WORK -05 { /home/admin/icdesign/Labs/m3/lal
Loading verilog file '/home/admin/icdesign/Labs/m3/lab06/results/i2c_master_top.v'
Current container set to 'r'
1
```

Formality (setup)>

Fig. 5. load .db file.

Set Top Design of reference. Select Library WORK, choose a design i2c_master_top and click Set Top button (Fig. 6).



Reference: r:/WORK/i2c_master_top

1. Choose a library.

GTECH (Technology Library)
SAED14RVT_TT0P8V25C (Technolog...
WORK (Design Library)

2. Choose a design.

i2c_master_bit_ctrl
i2c_master_bit_ctrl_DW01_dec_0_D...
i2c_master_bit_ctrl_DW01_dec_1_D...
i2c_master_byte_ctrl
i2c_master_top

3. Set and link the top design.

Unresolved Modules
○ Treat as black boxes.
● Emit errors.

```
**************************************************

Top design successfully set to 'r:/WORK/i2c_master_top'
Reference design set to 'r:/WORK/i2c_master_top'
1
```

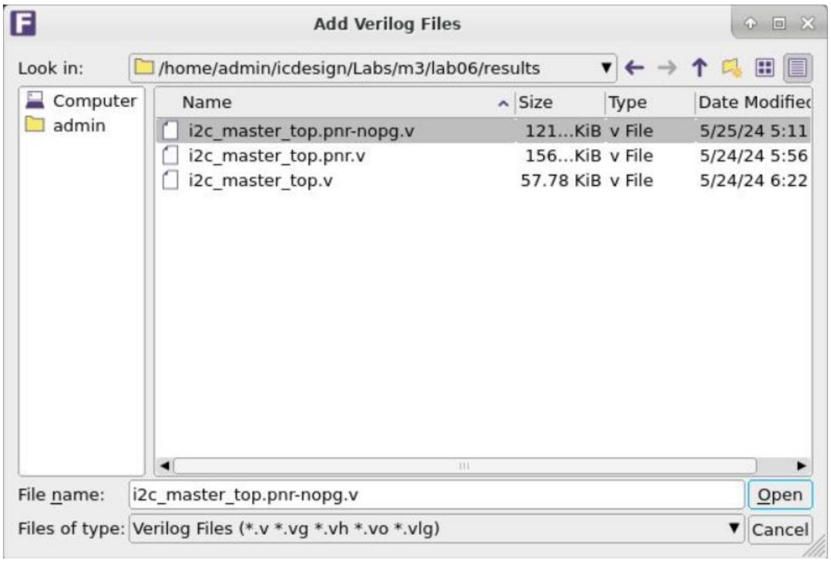
Formality (setup)>

Fig.6. Set Top

2. Implementation (Specify the Implementation Design)

The procedure for specifying the implementation design is identical to that for specifying the reference design. The implementation design is the gate level netlist after IC Compiler II named this source file i2c_master_top.pnr-nopg.v file.

Click on 2. Impl. >> 1. Read Design Files >> Verilog. After that, select the verilog button in the files section. Click Open as in Fig. 7.



Look in: /home/admin/icdesign/Labs/m3/lab06/results

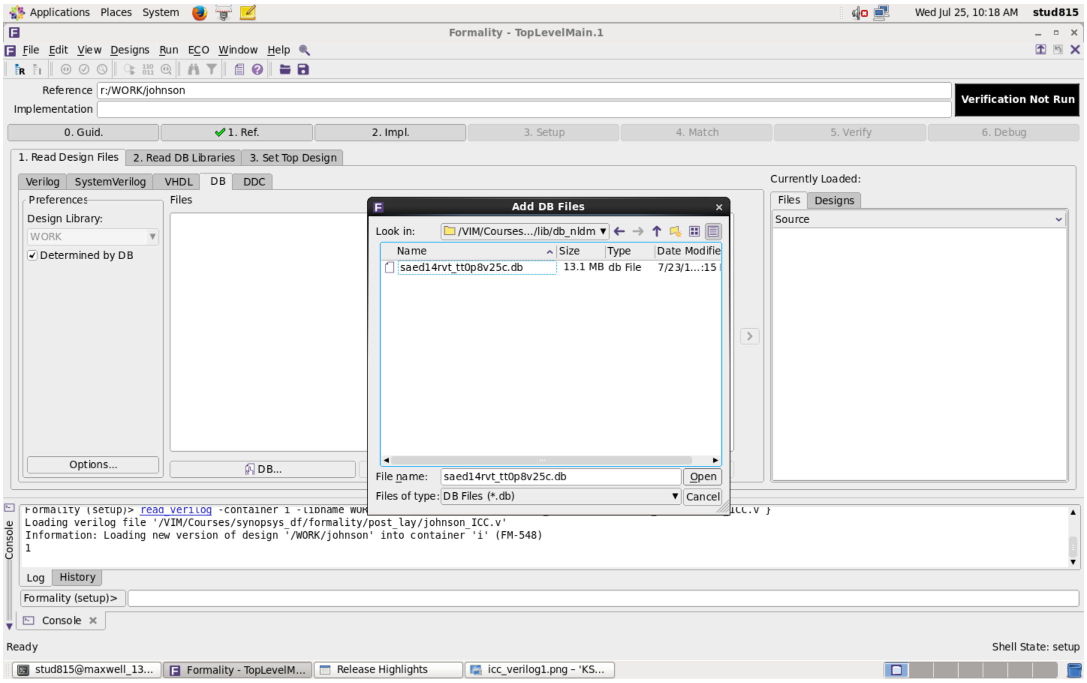
Name	Size	Type	Date Modified
i2c_master_top.pnr-nopg.v	121...KiB	v File	5/25/24 5:11
i2c_master_top.pnr.v	156...KiB	v File	5/24/24 5:56
i2c_master_top.v	57.78 KiB	v File	5/24/24 6:22

File name: i2c_master_top.pnr-nopg.v

Files of type: Verilog Files (*.v *.vg *.vh *.vo *.vlg)

Fig.5. Set up ICC gate level Verilog

Source the DB file by clicking on **2. Read DB Libraries** >> **Select /home/dkits/SAED14nm/stdcell_rvt/db_nldm/saed14rvt_tt0p8v25c.db** file click **Open** as in Fig. 6.



Reference: r:/WORK/johnson

Implementation:

Design Library: WORK

☑ Determined by DB

Source

Look in: /VIM/Courses.../lib/db_nldm

Name	Size	Type	Date Modifie
saed14rvt_tt0p8v25c.db	13.1 MB	db File	7/23/1...:15

File name: saed14rvt_tt0p8v25c.db

Files of type: DB Files (*.db)

```
Formality (setup)> read_verilog -container i -libname WOR                                      ICC.v }
Loading verilog file '/VIM/Courses/synopsys_df/formality/post_lay/johnson_ICC.v'
Information: Loading new version of design '/WORK/johnson' into container 'i' (FM-548)
1
```

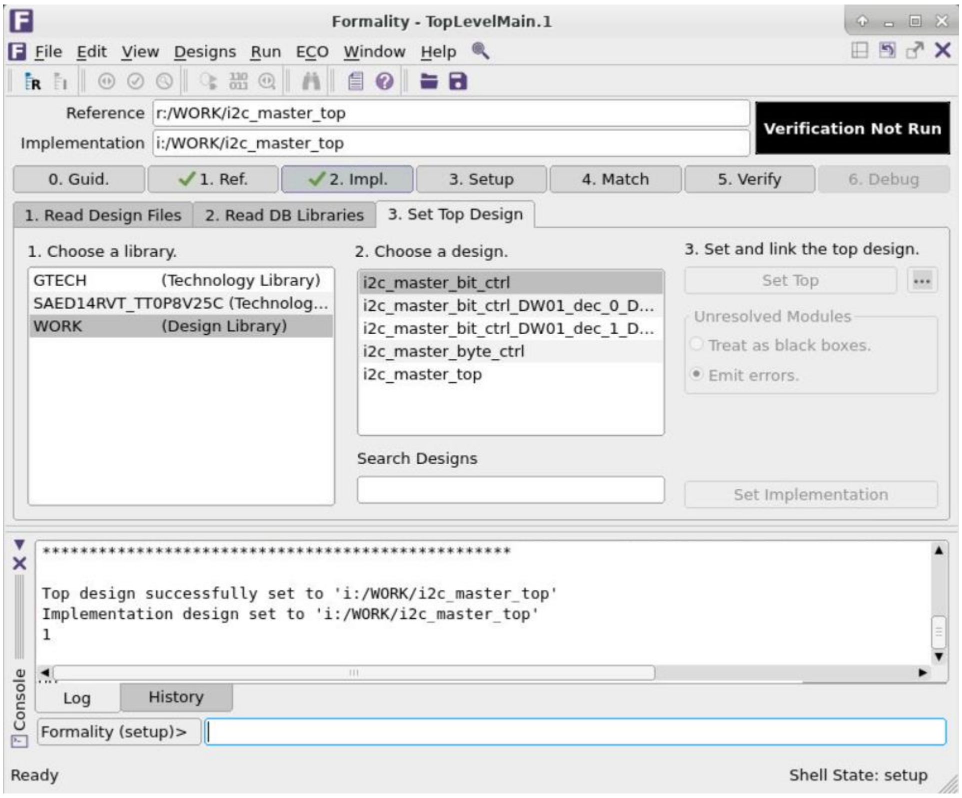
Formality (setup)>

Ready

Shell State: setup

Fig.6. Set gate .db file

Set Top Design of implementation. Select Library WORK, choose a design i2c_master_top and click Set Top button.



Reference r:/WORK/i2c_master_top

Implementation i:/WORK/i2c_master_top

1. Choose a library.

GTECH	(Technology Library)
SAED14RVT_TT0P8V25C (Technolog...	
WORK	(Design Library)

2. Choose a design.

- i2c_master_bit_ctrl
- i2c_master_bit_ctrl_DW01_dec_0_D...
- i2c_master_bit_ctrl_DW01_dec_1_D...
- i2c_master_byte_ctrl
- i2c_master_top

Search Designs

3. Set and link the top design.

Unresolved Modules

○ Treat as black boxes.

● Emit errors.

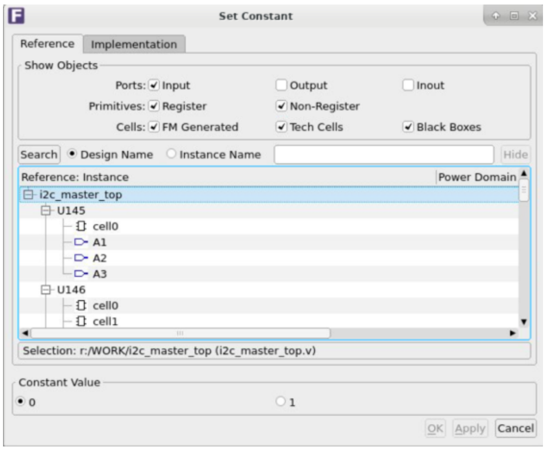
```
****************************************************

Top design successfully set to 'i:/WORK/i2c_master_top'
Implementation design set to 'i:/WORK/i2c_master_top'
1
```

Formality (setup)>

3. Set up (Set Up the Design)

To Set up the design, click 3.set up (Fig.7 and Fig.8).



Show Objects

Ports: ☑ Input ☐ Output ☐ Inout

Primitives: ☑ Register ☑ Non-Register

Cells: ☑ FM Generated ☑ Tech Cells ☑ Black Boxes

● Design Name ○ Instance Name

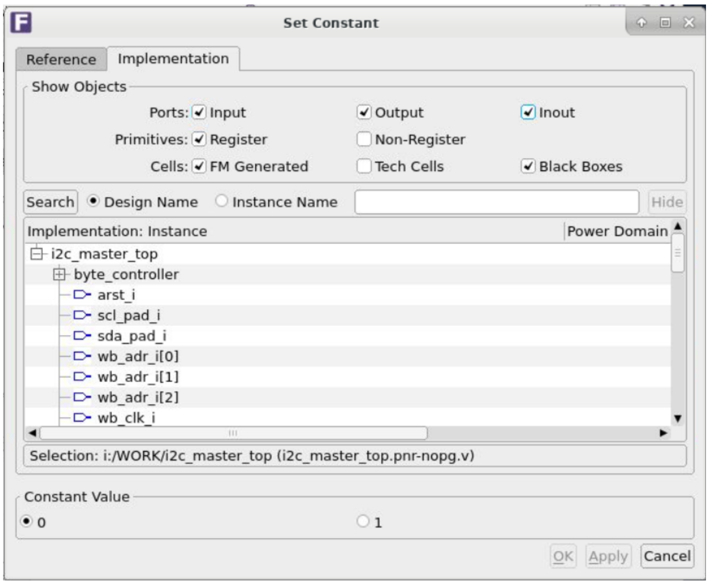
Reference: Instance	Power Domain
i2c_master_top	
U145	
cell0	
A1	
A2	
A3	
U146	
cell0	
cell1	

Selection: r:/WORK/i2c_master_top (i2c_master_top.v)

Constant Value

● 0 ○ 1

Fig.7. Set Up the Design window



Set Constant

Reference | Implementation

Show Objects

Ports: ☑ Input ☑ Output ☑ Inout

Primitives: ☑ Register □ Non-Register

Cells: ☑ FM Generated □ Tech Cells ☑ Black Boxes

Search ● Design Name ○ Instance Name Hide

Implementation: Instance	Power Domain
i2c_master_top	
byte_controller	
arst_i	
scl_pad_i	
sda_pad_i	
wb_adr_i[0]	
wb_adr_i[1]	
wb_adr_i[2]	
wb_clk_i	

Selection: i:/WORK/i2c_master_top (i2c_master_top.pnr-nopg.v)

Constant Value

● 0 ○ 1

OK Apply Cancel

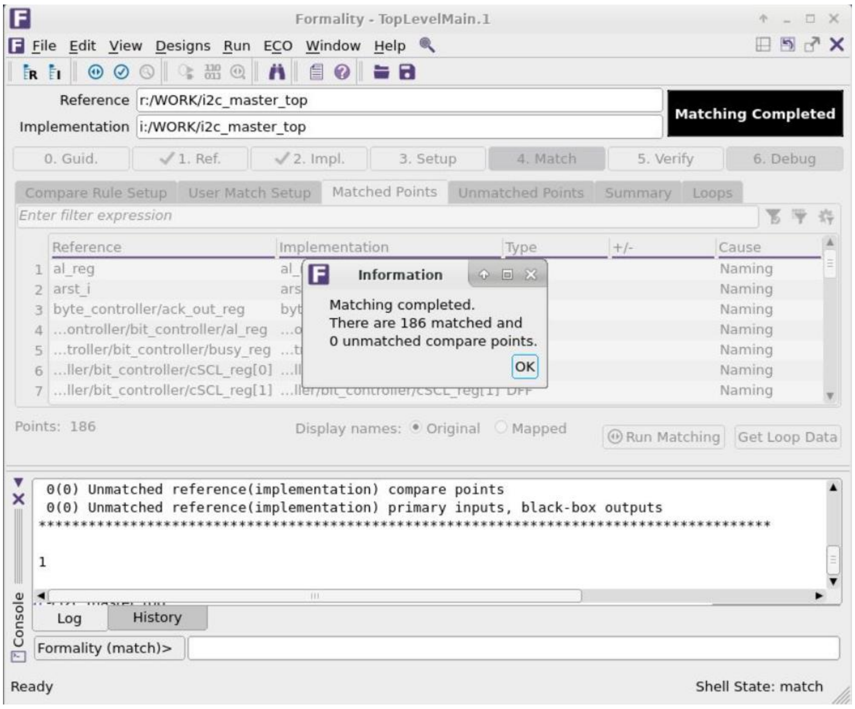
Fig.8. After Set Up the Design

4. Match (Match Compare Points)

Match compare points is the process by which Formality segments the reference and implementation designs into logical units, called logic cones.

To match compare points between i2c_master_top.v and i2c_master_top.pnr-nopg.v (after ICC), do the following:

In Main Window click 4. Match >> Match Points for match compare points. After that, click on **Run Matching** button



Formality - TopLevelMain.1

Reference r:/WORK/i2c_master_top

Implementation i:/WORK/i2c_master_top

Matching Completed

	Reference	Implementation	Type	+/-	Cause
1	al_reg	al_			Naming
2	arst_i	ars			Naming
3	byte_controller/ack_out_reg	byt			Naming
4	...ontroller/bit_controller/al_reg	...o			Naming
5	...troller/bit_controller/busy_reg	...t			Naming
6	...ller/bit_controller/cSCL_reg[0]	...ll			Naming
7	...ller/bit_controller/cSCL_reg[1]	...ller/bit_controller/cSCL_reg[1]	DFF		Naming

Information

Matching completed.
There are 186 matched and
0 unmatched compare points.

OK

Points: 186

Display names: ● Original ○ Mapped

```
 0(0) Unmatched reference(implementation) compare points
 0(0) Unmatched reference(implementation) primary inputs, black-box outputs
*******************************************************************************

1
```

Formality (match)>

Ready

Shell State: match

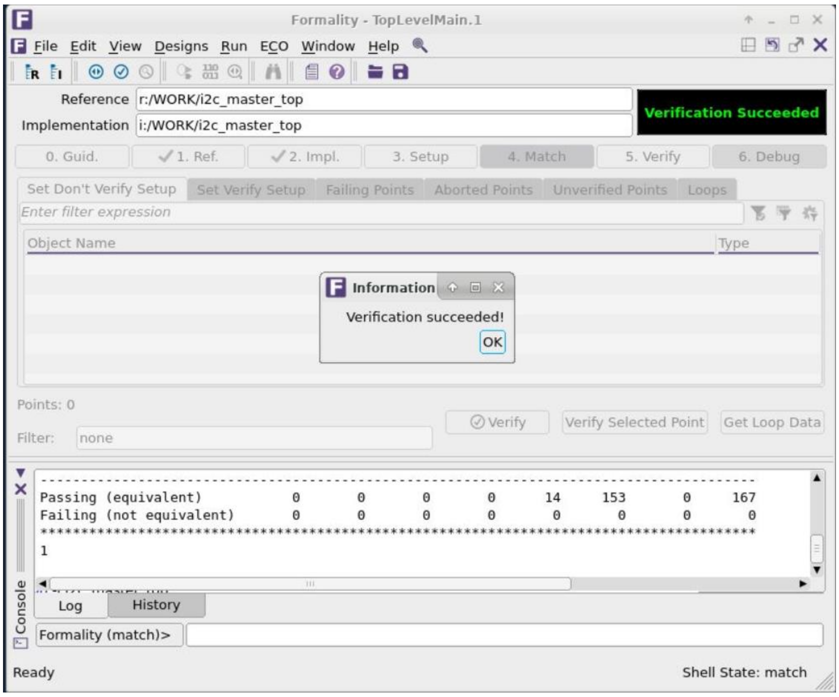
Fig.9. Match compare points

As seen from the massage shown in Fig.9, there are 0 unmatched compare points.

5. Verify (Verify the Designs)

When verify command is used, Formality attempts to prove design equivalence between an implementation design and a reference design. This section describes how to verify a design or a single compare point as well as how to perform traditional hierarchical verification.

On the main toolbar, click the **5. Verify** tab, then click **Verify**.



Reference r:/WORK/i2c_master_top

Implementation i:/WORK/i2c_master_top

Verification succeeded!

Points: 0

Filter: none

```
--------------------------------------------------------------------------------
Passing (equivalent)        0       0       0       0      14     153       0     167
Failing (not equivalent)    0       0       0       0       0       0       0       0
********************************************************************************
1
```

Formality (match)>

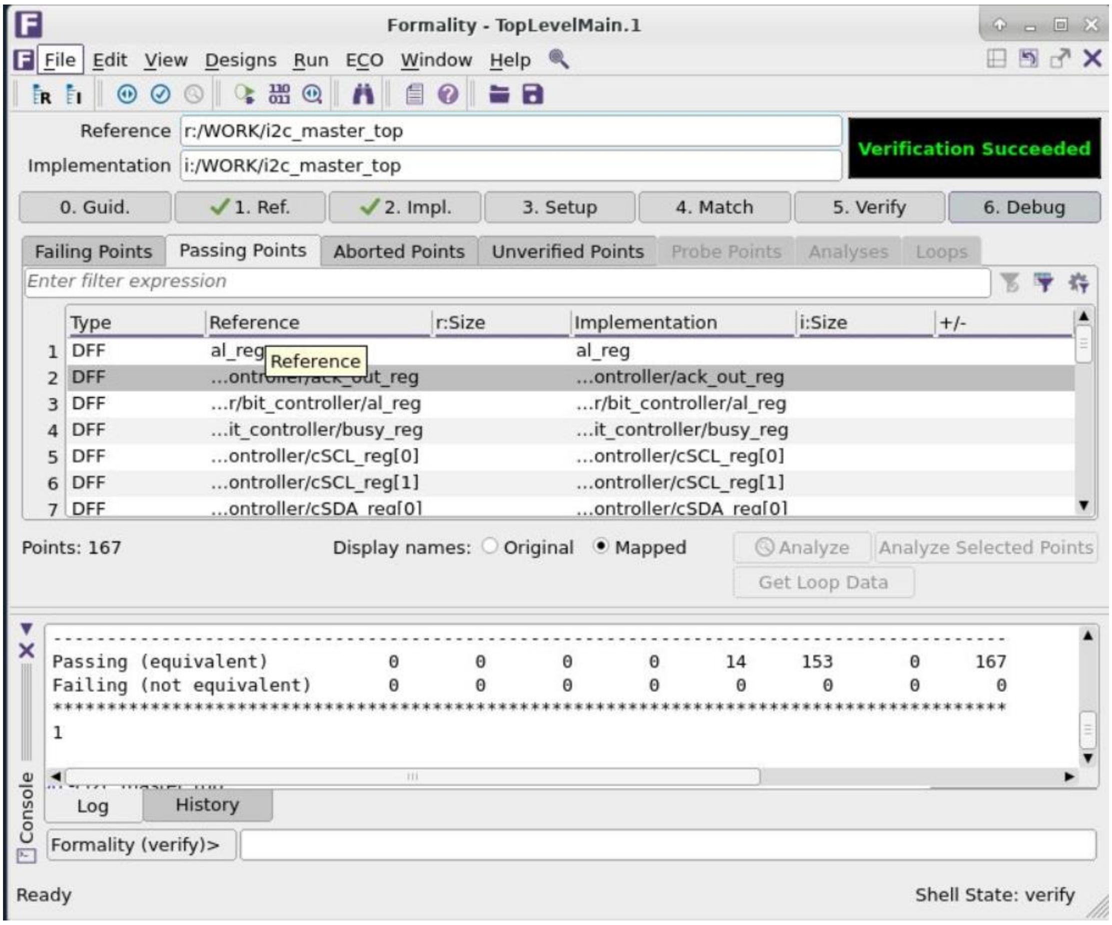
Shell State: match

Fig.10. Verify the design

Verifying process completed successfully.

6. Debug

During debugging, the exact points in the designs that exhibit the difference in functionality and then fix them must be found (Fig. 11).



Reference: r:/WORK/i2c_master_top

Implementation: i:/WORK/i2c_master_top

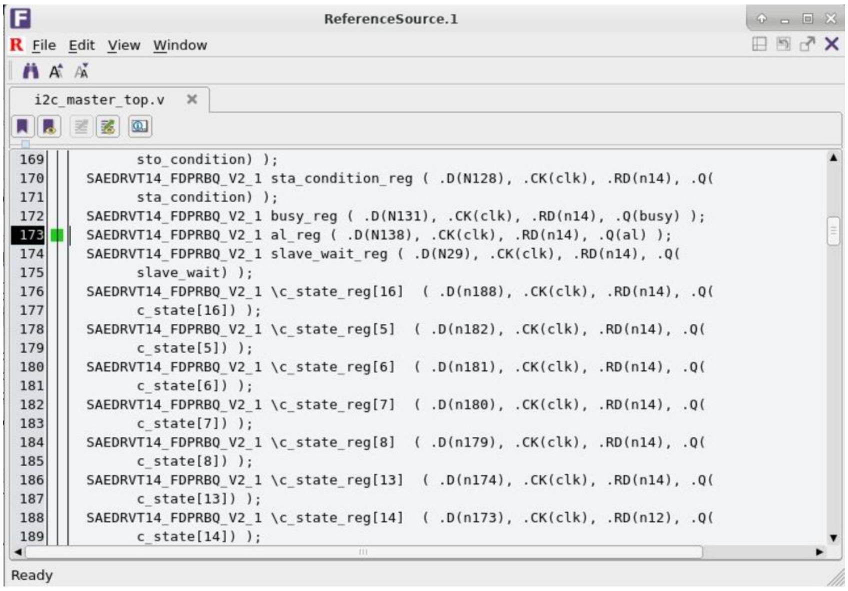
	Type	Reference	r:Size	Implementation	i:Size	+/-
1	DFF	al_reg		al_reg		
2	DFF	...ontroller/ack_out_reg		...ontroller/ack_out_reg		
3	DFF	...r/bit_controller/al_reg		...r/bit_controller/al_reg		
4	DFF	...it_controller/busy_reg		...it_controller/busy_reg		
5	DFF	...ontroller/cSCL_reg[0]		...ontroller/cSCL_reg[0]		
6	DFF	...ontroller/cSCL_reg[1]		...ontroller/cSCL_reg[1]		
7	DFF	...ontroller/cSDA_reg[0]		...ontroller/cSDA_reg[0]		

Points: 167 Display names: ○ Original ● Mapped

```
 --------------------------------------------------------------------------------
 Passing (equivalent)        0      0      0      0     14    153      0    167
 Failing (not equivalent)    0      0      0      0      0      0      0      0
 ********************************************************************************
 1
```

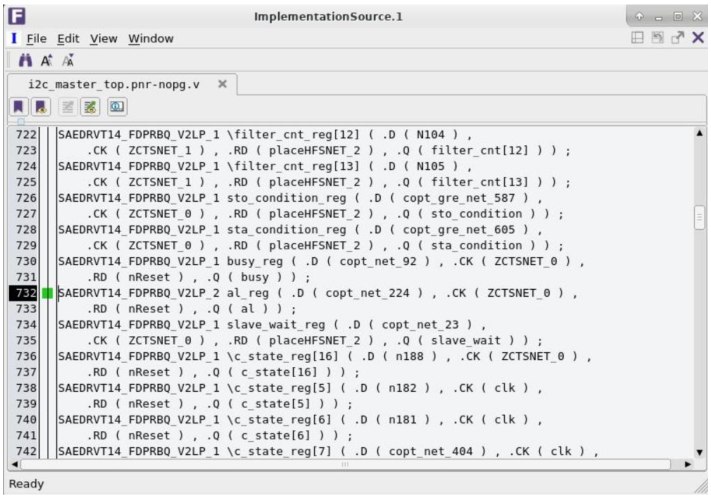
Fig.11. Debugging the design

Formality is able to simultaneously display reference and implementation Verilog views and mark differences and /or similarities (Fig. 12).



```
        sto_condition) );
  SAEDRVT14_FDPRBQ_V2_1 sta_condition_reg ( .D(N128), .CK(clk), .RD(n14), .Q(
        sta_condition) );
  SAEDRVT14_FDPRBQ_V2_1 busy_reg ( .D(N131), .CK(clk), .RD(n14), .Q(busy) );
  SAEDRVT14_FDPRBQ_V2_1 al_reg ( .D(N138), .CK(clk), .RD(n14), .Q(al) );
  SAEDRVT14_FDPRBQ_V2_1 slave_wait_reg ( .D(N29), .CK(clk), .RD(n14), .Q(
        slave_wait) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[16]  ( .D(n188), .CK(clk), .RD(n14), .Q(
        c_state[16]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[5]  ( .D(n182), .CK(clk), .RD(n14), .Q(
        c_state[5]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[6]  ( .D(n181), .CK(clk), .RD(n14), .Q(
        c_state[6]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[7]  ( .D(n180), .CK(clk), .RD(n14), .Q(
        c_state[7]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[8]  ( .D(n179), .CK(clk), .RD(n14), .Q(
        c_state[8]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[13]  ( .D(n174), .CK(clk), .RD(n14), .Q(
        c_state[13]) );
  SAEDRVT14_FDPRBQ_V2_1 \c_state_reg[14]  ( .D(n173), .CK(clk), .RD(n12), .Q(
        c_state[14]) );
```

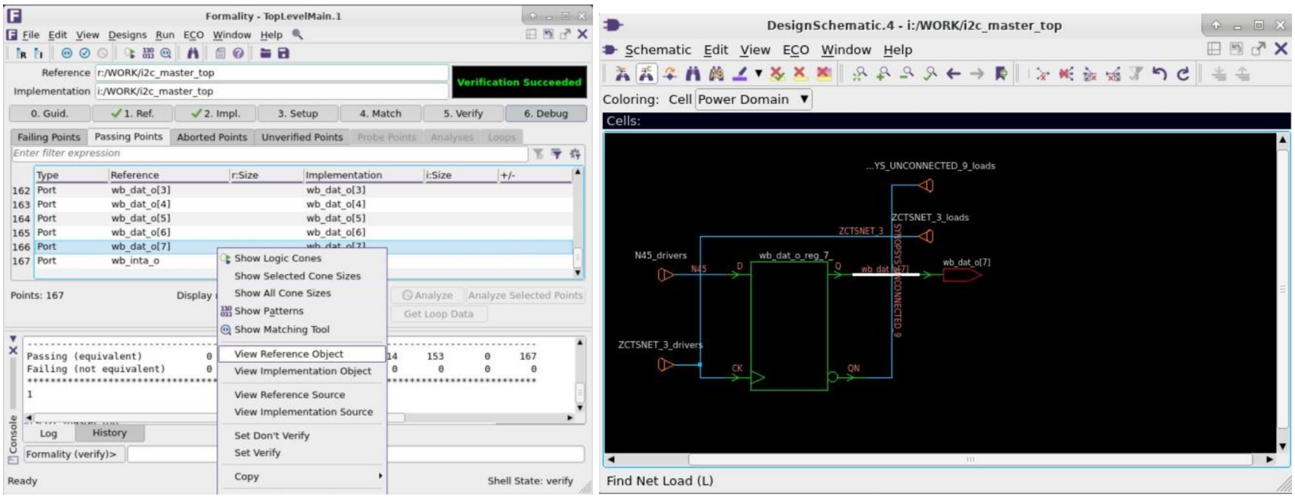
Fig.12(a). Implementation Verilog



```
SAEDRVT14_FDPRBQ_V2LP_1 \filter_cnt_reg[12] ( .D ( N104 ) ,
    .CK ( ZCTSNET_1 ) , .RD ( placeHFSNET_2 ) , .Q ( filter_cnt[12] ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 \filter_cnt_reg[13] ( .D ( N105 ) ,
    .CK ( ZCTSNET_1 ) , .RD ( placeHFSNET_2 ) , .Q ( filter_cnt[13] ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 sto_condition_reg ( .D ( copt_gre_net_587 ) ,
    .CK ( ZCTSNET_0 ) , .RD ( placeHFSNET_2 ) , .Q ( sto_condition ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 sta_condition_reg ( .D ( copt_gre_net_605 ) ,
    .CK ( ZCTSNET_0 ) , .RD ( placeHFSNET_2 ) , .Q ( sta_condition ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 busy_reg ( .D ( copt_net_92 ) , .CK ( ZCTSNET_0 ) ,
    .RD ( nReset ) , .Q ( busy ) ) ;
SAEDRVT14_FDPRBQ_V2LP_2 al_reg ( .D ( copt_net_224 ) , .CK ( ZCTSNET_0 ) ,
    .RD ( nReset ) , .Q ( al ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 slave_wait_reg ( .D ( copt_net_23 ) ,
    .CK ( ZCTSNET_0 ) , .RD ( placeHFSNET_2 ) , .Q ( slave_wait ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 \c_state_reg[16] ( .D ( n188 ) , .CK ( ZCTSNET_0 ) ,
    .RD ( nReset ) , .Q ( c_state[16] ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 \c_state_reg[5] ( .D ( n182 ) , .CK ( clk ) ,
    .RD ( nReset ) , .Q ( c_state[5] ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 \c_state_reg[6] ( .D ( n181 ) , .CK ( clk ) ,
    .RD ( nReset ) , .Q ( c_state[6] ) ) ;
SAEDRVT14_FDPRBQ_V2LP_1 \c_state_reg[7] ( .D ( copt_net_404 ) , .CK ( clk ) ,
```

Fig. 12(b). Reference Verilog

Besides, Formality can display schematic view and highlight reference object on it (Fig. 13).



Reference r:/WORK/i2c_master_top
Implementation i:/WORK/i2c_master_top
Verification Succeeded

	Type	Reference	r:Size	Implementation	i:Size	+/-
162	Port	wb_dat_o[3]		wb_dat_o[3]		
163	Port	wb_dat_o[4]		wb_dat_o[4]		
164	Port	wb_dat_o[5]		wb_dat_o[5]		
165	Port	wb_dat_o[6]		wb_dat_o[6]		
166	Port	wb_dat_o[7]		wb_dat_o[7]		
167	Port	wb_inta_o				

Points: 167

```
Passing (equivalent)         0    14    153    0    167
Failing (not equivalent)     0     0      0    0      0
****************************************************
1
```

Formality (verify)>

Ready   Shell State: verify

Coloring: Cell Power Domain

Fig.13. View Reference Object in schematic

To exit Formality, write exit in the command line.

```
Formality (verify)> exit
```

7. Run formal check to compare the synthesis netlist and the RTL netlist

Run the following commands in fm_shell:

```
set_svf -append { ../syn/i2c_master_top.svf }
read_verilog -container r -libname WORK -01 \
        { ../src/rtl/verilog/i2c_master_defines.v \
          ../src/rtl/verilog/i2c_master_bit_ctrl.v \
          ../src/rtl/verilog/i2c_master_byte_ctrl.v \
          ../src/rtl/verilog/i2c_master_top.v}
set_top r:/WORK/i2c_master_top
read_db -container i { /home/dkits/synopsys/SAED14nm/stdcell_rvt/db_nldm/saed14rvt_tt0p8v25c.db }
read_verilog -container i -libname WORK -01 { ../results/i2c_master_top.v }
set_top i:/WORK/i2c_master_top
match
verify
file mkdir results
```

```
save_session -replace results/fm_post_syn
exit
```