Code Coverage

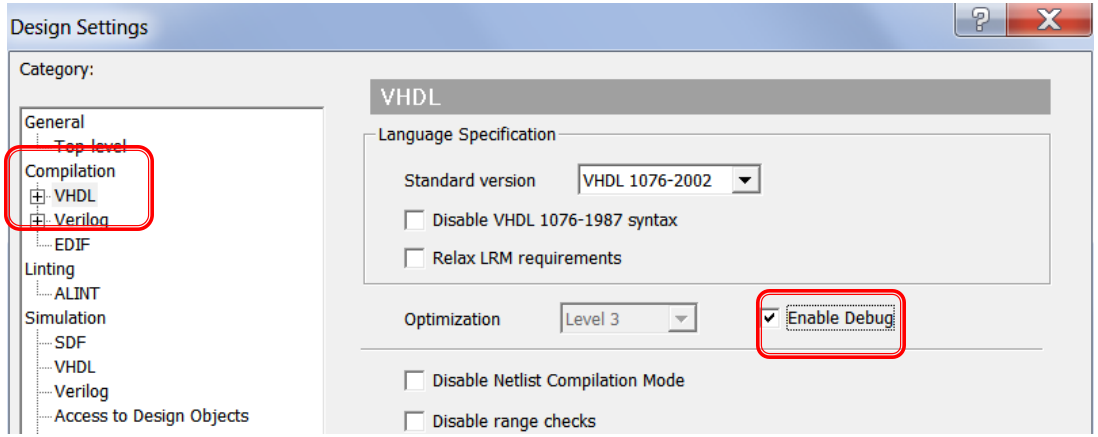
Learn how to generate different types of coverage data and how to view the results

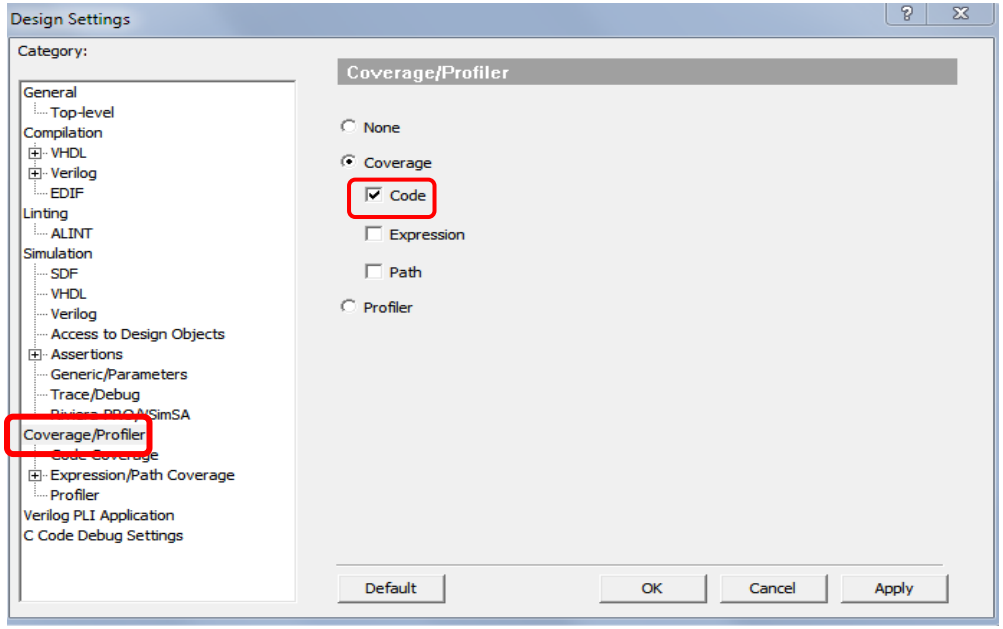
Code Coverage

- **Code Coverage** is an integrated tool within Active-HDL
- It is a debugging tool that allows you to check how well your testbench/test suit is checking design under test
- It can help you determine how much logic in the design is being actually exercised
- There are different types of coverage:
  - Statement Coverage
  - Branch Coverage
  - Toggle Coverage
  - Expression Coverage
  - Path Coverage

Enabling Code Coverage - GUI

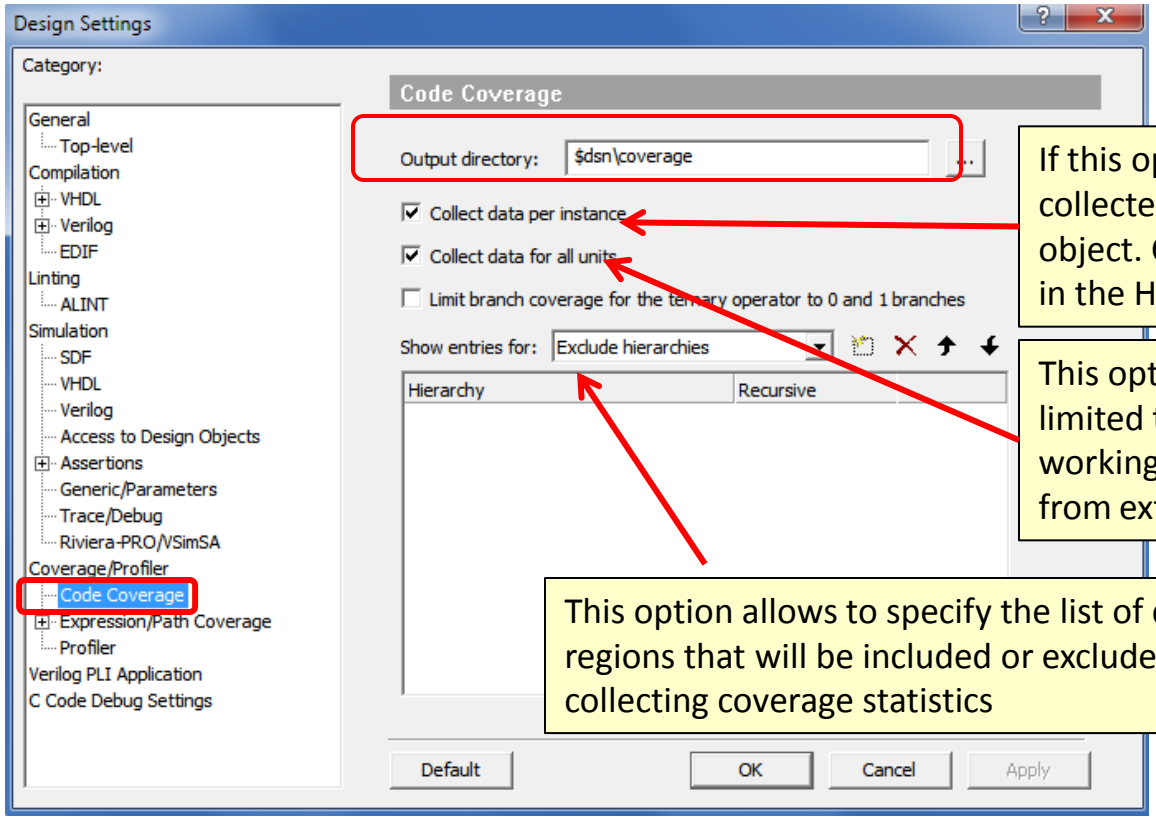
- To enable Code Coverage you have to:
  - Enable Debug from **Design Settings | Compilation | <Language>**
  - Open the **Design Settings** window from the **Design** menu
  - Select the **Coverage/Profiler** tab
  - Select Coverage radio button in the Enable section





Enabling Code Coverage - GUI

› To select **Output Directory** select **Code Coverage** category from the **Design Settings** menu



If this option is checked, coverage data will be collected separately for each instantiated object. Code Coverage displays this information in the Hierarchy tab.

This option determines whether data should be limited to objects residing in the current working library or it should extend to objects from external system libraries as well.

This option allows to specify the list of design regions that will be included or excluded from collecting coverage statistics

Enabling Code Coverage – Script

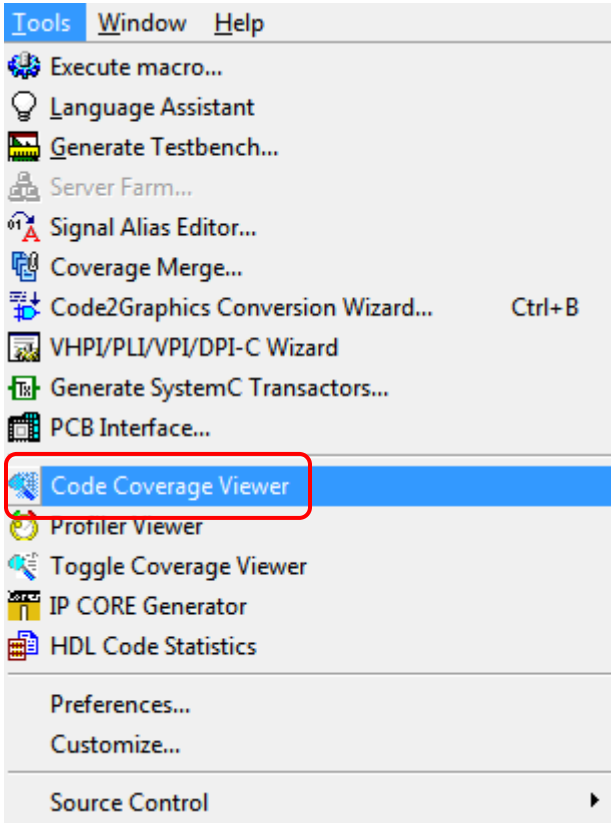
- When a design simulation is initialized by a DO-macro file and you would like to use Code Coverage, you have to initialize simulation with following options:
  - `asim -cc -cc_dest $DSN/coverage testbench`

- This will enable Code Coverage data gathering in default mode i.e. information will be collected for each unit. To distinguish each instance from the others, use syntax:
  - `asim -cc –cc_hierarchy -cc_dest$DSN/coverage testbench`

Note: Please refer to **Help** documentation for more details on **asim command usage.**

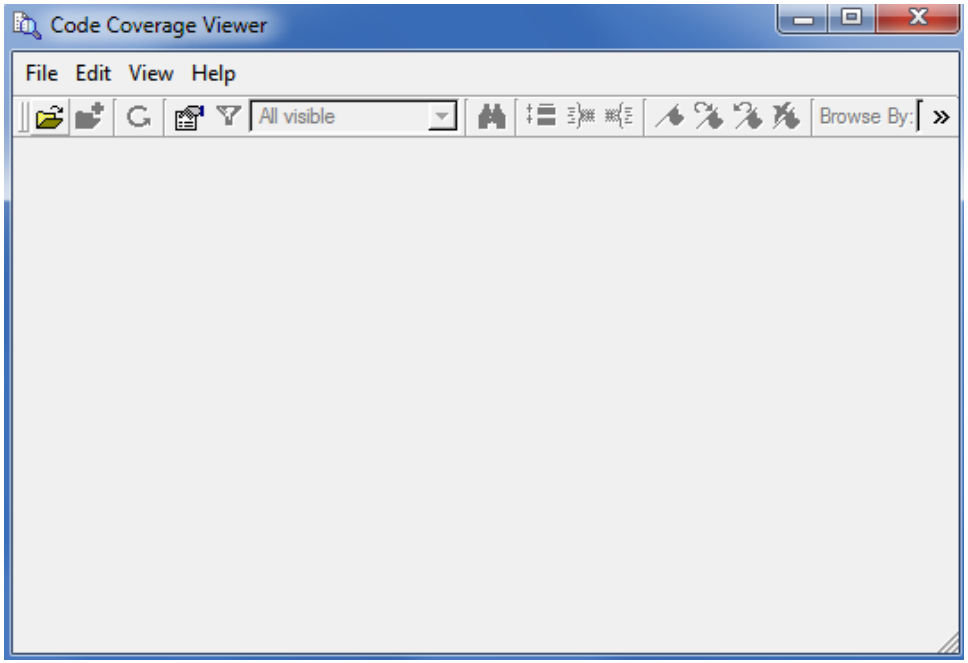
Invoking Coverage Viewer

- When your simulation is finished, you can open the **Code Coverage Viewer** from the **Tools** menu in.
- All data gathered by Code Coverage can be presented in a **graphical** and **textual** form in the **Code Coverage Viewer** window.



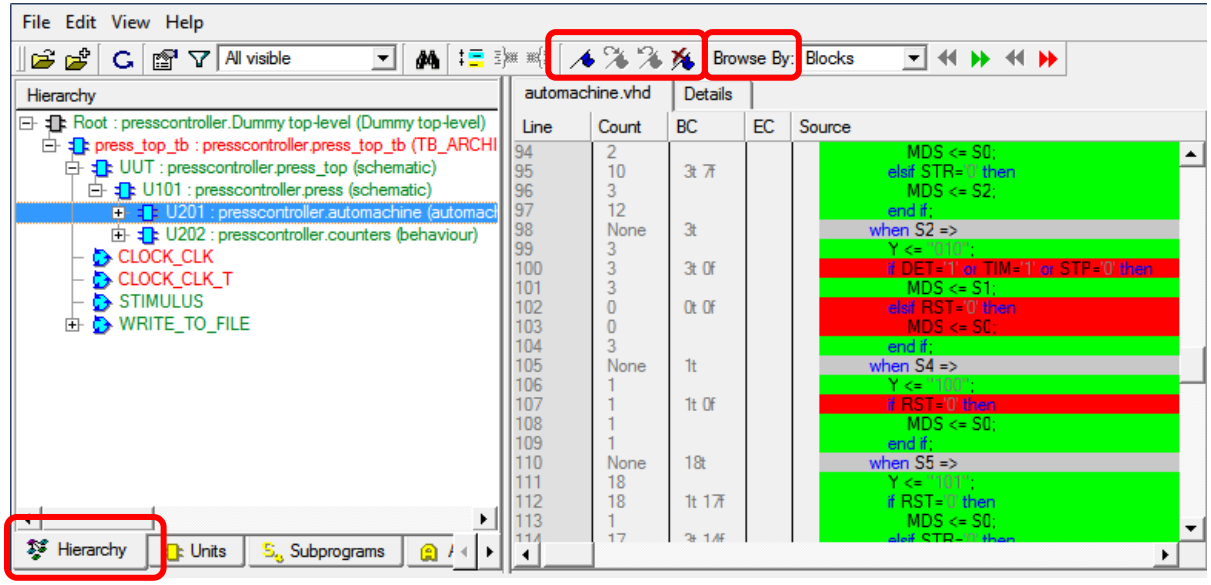
Opening Coverage File

- To load the Code Coverage data collected during simulation run:
  - Select **Open**…from the **File** menu or use button in the main toolbar.
  - Find **results.ccl** file. It should have been created in previously specified path.



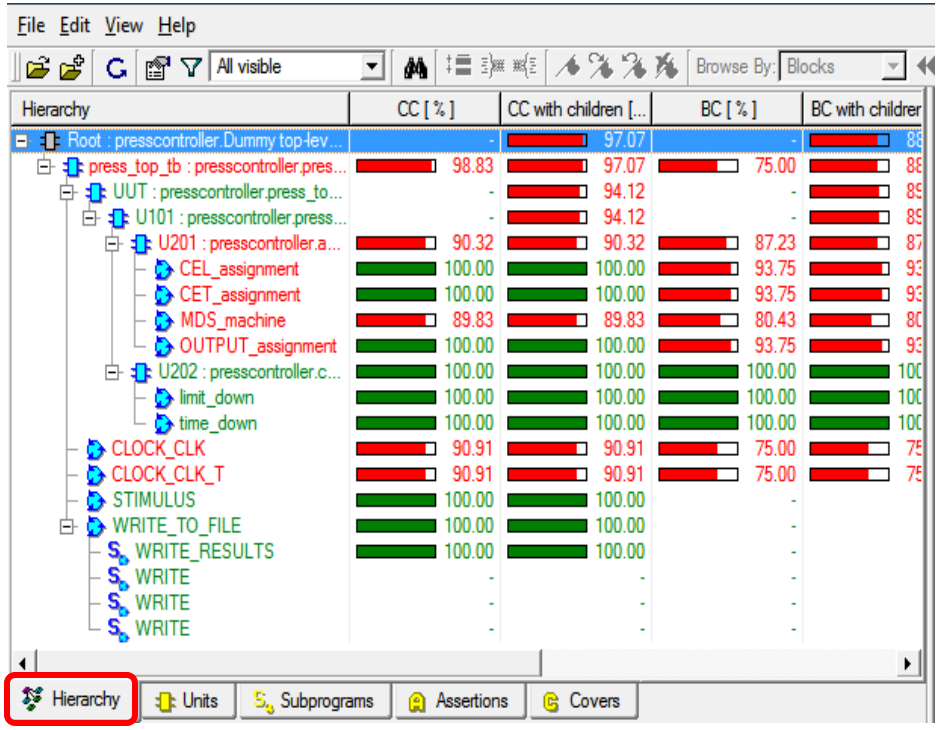
Coverage Viewer

- There are two panels in the Code Coverage Viewer window:
  - **Hierarchy pane** -displays the hierarchical structure of the design
  - **Source Code/Details** pane – displays HDL source code



Hierarchy Window

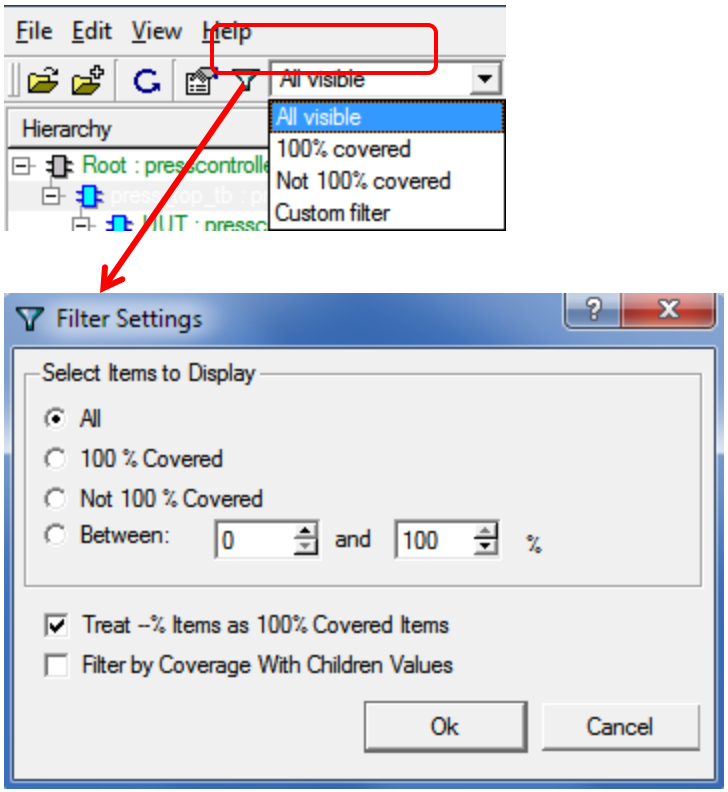
- The **Hierarchy** tab in the left pane shows the elaborated structure of the design and **Code Coverage** (CC) and **Branch Coverage** (BC) statistics for each item and its children in the design tree



- The icon captions in the design hierarchy tree can be either green (for 100% covered items) or red (for items that are not fully covered)
- The Units tab in the left pane shows Code Coverage statistics for all units used in the design, irrespective of their position in the hierarchy tree

Hierarchy Window

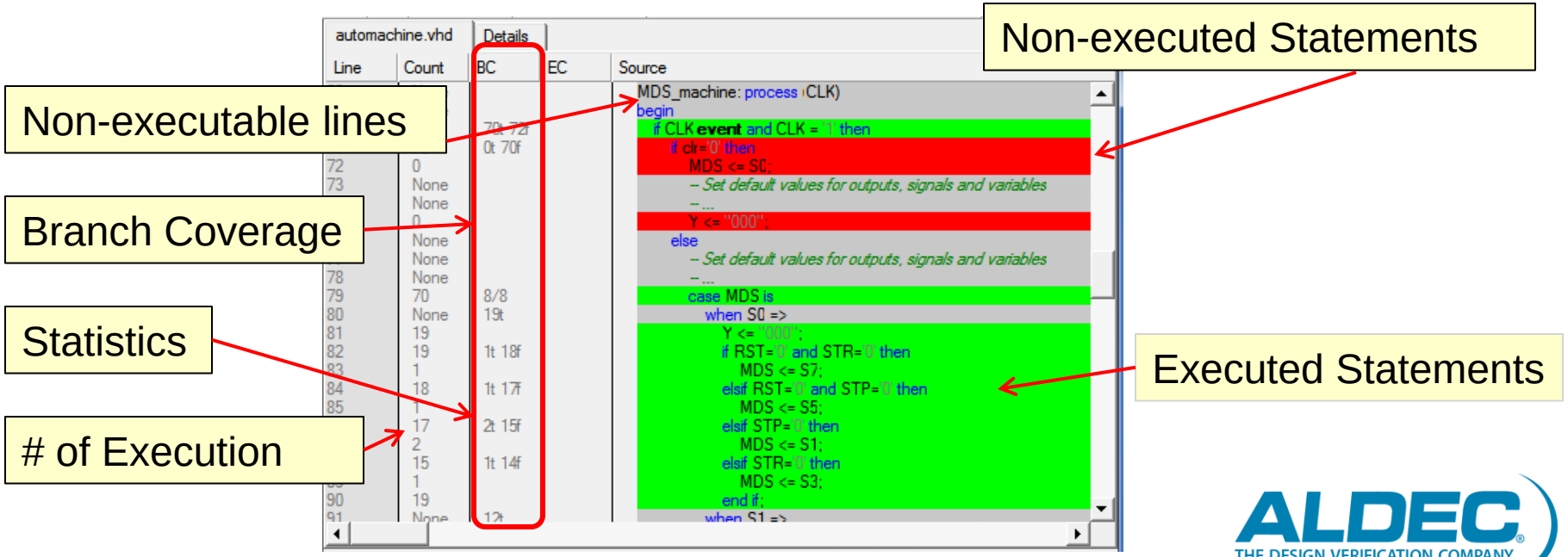
- You can select which instantiations should be displayed using list-box or button in the **Main Tool Bar**



- You can choose whether all instantiations should be displayed, instantiations with all statements executed, or instances with unexecuted statements or use customized view.
- To customize visibility, you should use the **Filter** dialog box.

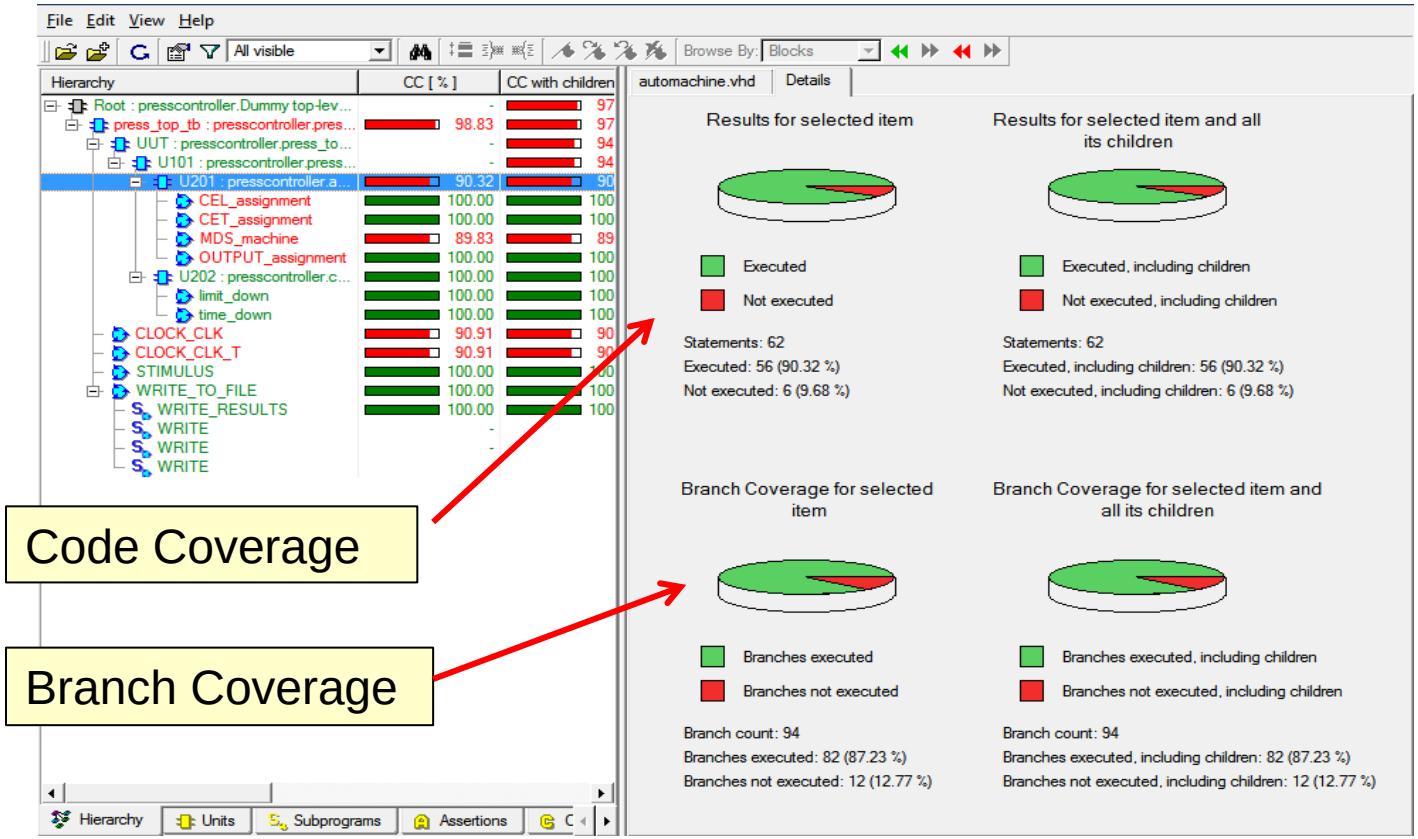
Source Code Tab

- The source code of module selected in **Hierarchy** pane is displayed in the **Source** tab
- Executed statements are displayed in green color. The number of executions is also shown to the left of corresponding line
- Statements that were not executed at all are in red



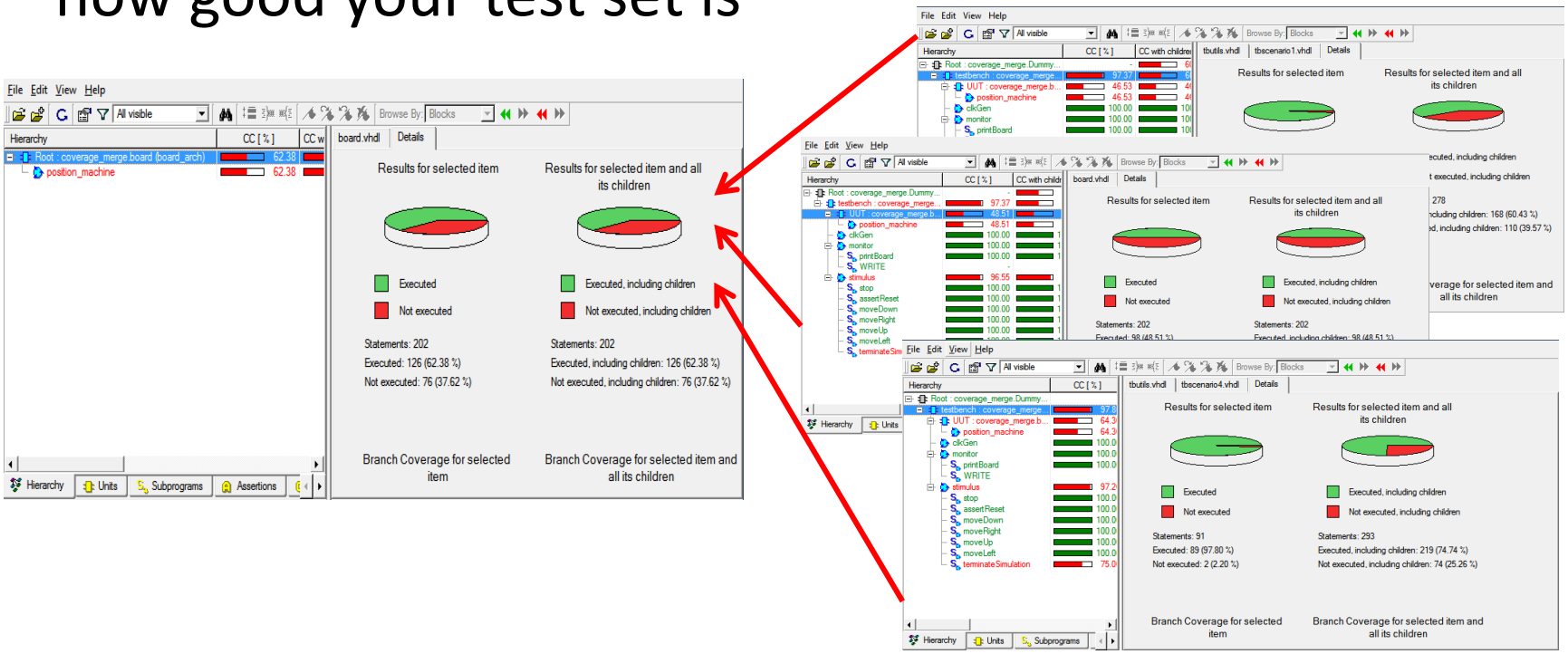
Details Tab

- The Details tab allows you to present the results of the coverage in graphical form. (Pie charts)



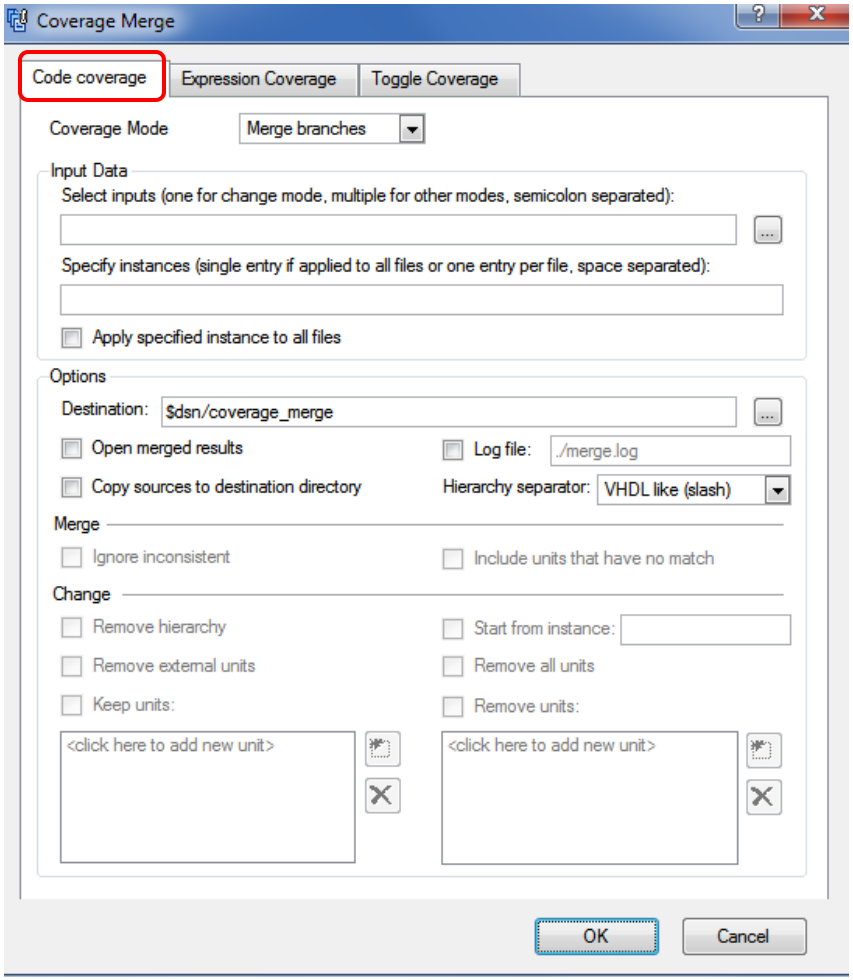
Merge Code Coverage Data

- It is possible to verify efficiency of your test suite using the Code Coverage Merge feature. You can combine the data gathered for individual simulation runs and see how good your test set is



Merge Coverage Data - GUI

- The merge process can also be started by using the **Coverage Merge** dialog box, available from the **Tools** menu
- Select the **Code Coverage** tab, choose the merge mode and configure options available for selected mode



Merge Coverage Data - Script

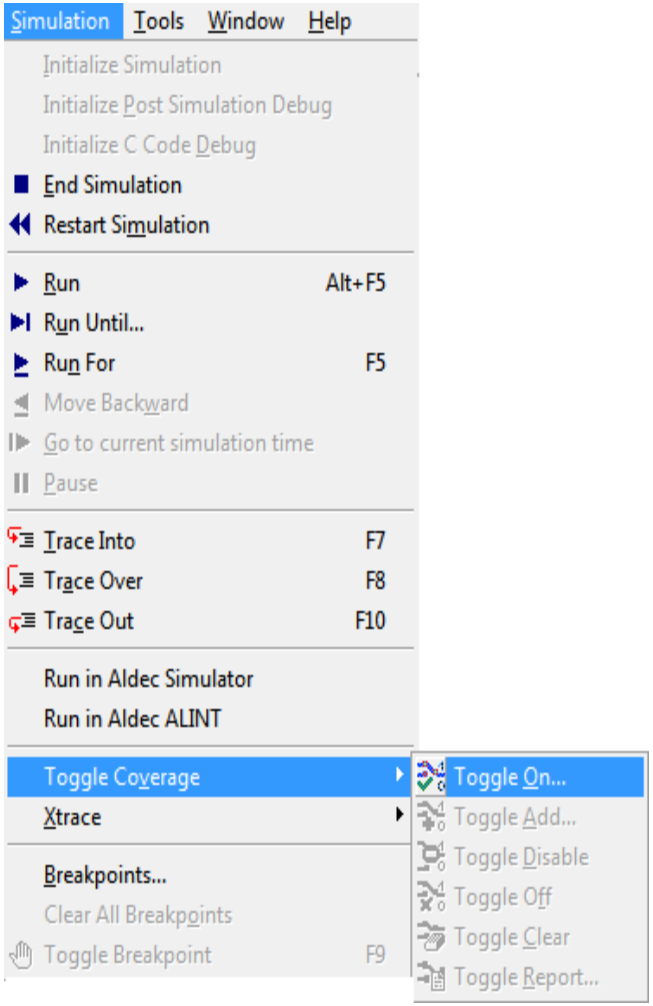
- To combine **Code Coverage** data obtained in two separate simulation sessions, you need to issue the following command in the **Console Window**
  - ```
    coverage merge -merge_branches UUT UUT -dir $DSN\coverage1
                -dir $DSN\coverage2  -dest $DSN\coverage_merged
    ```
- This will merge data starting from the specified instances (UUT). This is particularly useful when combining data from simulation of two different top-level units containing the same tested units. Alternatively, the –merge_hierarchies switch will combine whole trees, if top-level units have same name and structure
  - ```
    coverage merge -merge_hierarchies -dir $DSN\coverage1 -dir
    $DSN\coverage2 –dest $DSN\coverage_merged
    ```

Toggle Coverage

- **Toggle Coverage** measures design activity in terms of changes of signal logic values
- It efficiently helps to verify the quality of the stimulus and locate "dead" structures of the design. Signals that were not initialized during simulation or not exercised properly by the testbench can be easily identified

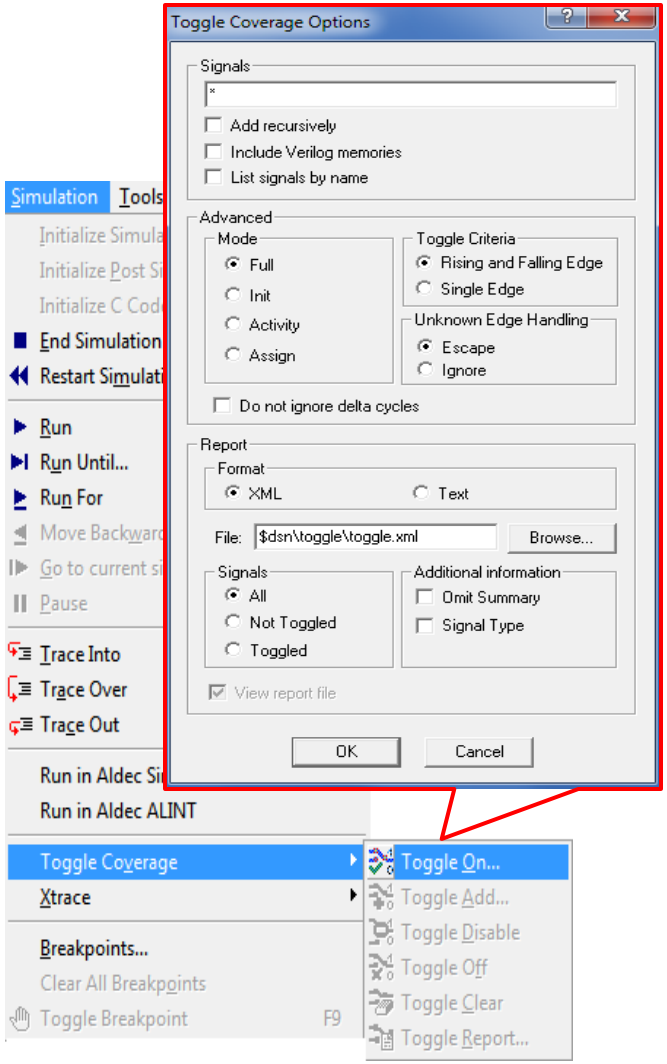
Enabling Toggle Coverage

- To enable **Toggle Coverage**:
  - **Initialize simulation** session
  - On the Structure tab **select** desired **signals/unit** and
  - Use the **Toggle Coverage|Toggle On**... option from the **Simulation** menu.
- You can also start Toggle Coverage session using toggle command:
  - `toggle -toggle_type full /UUT/I17/*`



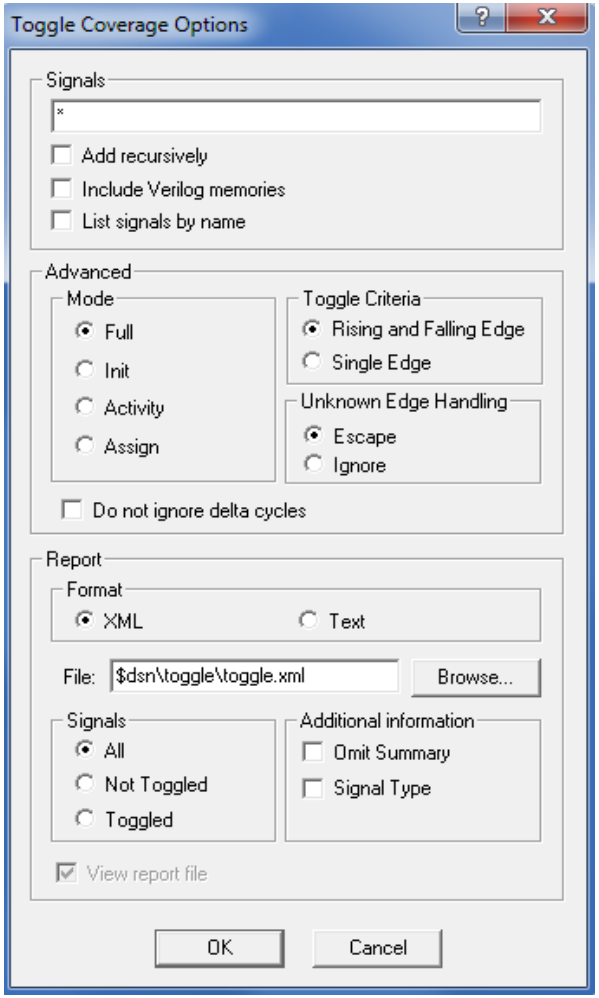
Enabling Toggle Coverage

- In the **Toggle Coverage Options** window you can specify the **settings** of the Toggle Coverage session.
- A **report** is written automatically to the **subfolder specified** in the Toggle Coverage **Options** window when the **endsim** command is used or when the Toggle Coverage engine is switched off.



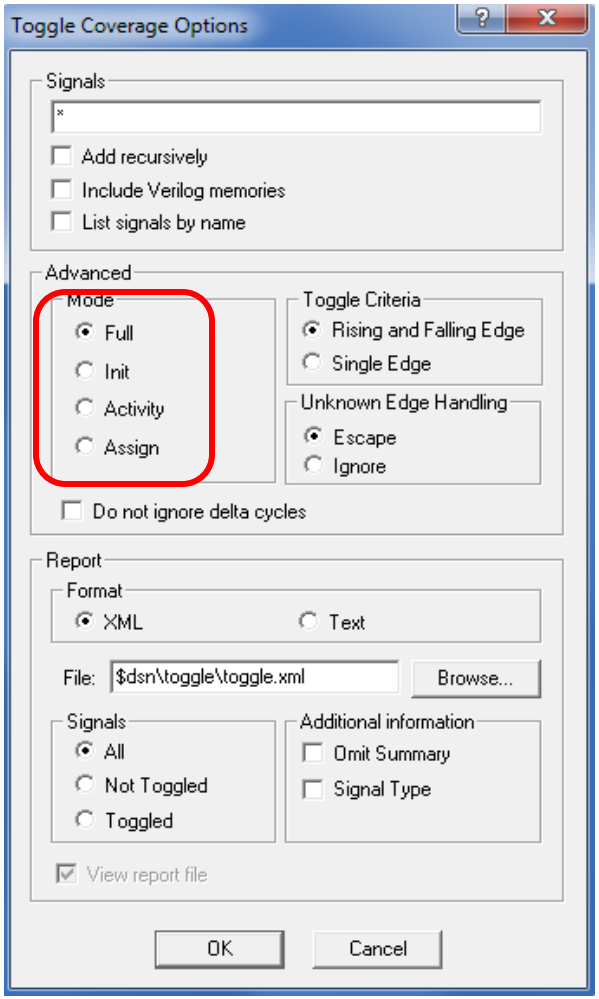
Toggle Coverage Options

- The **Toggle Coverage Options** dialog box allows you to specify the settings of the Toggle Coverage session. In this window, you can specify:
  - Type of the working mode
  - When a signal should be considered as toggled
  - How unknown values should be treated
  - The name and settings of the report file



Toggle Coverage Modes

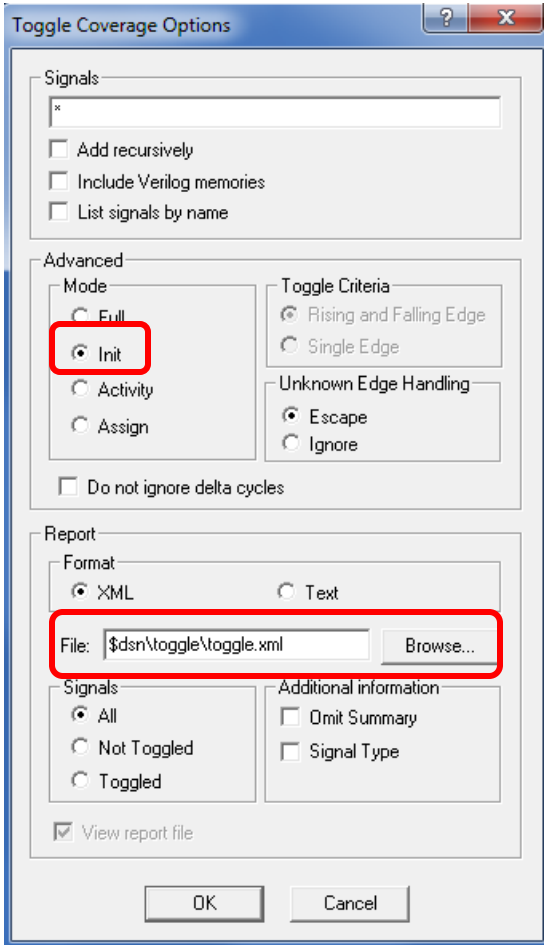
- The Toggle Coverage engine can generate four different types of reports:
  - Full mode
  - Init mode
  - Activity mode
  - Assign mode
- The mode can be specified in the **Toggle Coverage Options** window or by using the `toggle -toggle_type` command



Toggle Coverage – Init Mode

- The **Init mode** checks if selected signals have been at least once set to ‘0’or ‘1’value.
- This mode is useful to verify whether selected signals were initialized or not.
- The macro below invokes toggle coverage with proper switches listed below:

```
toggle -toggle_type init -
rec/*
```



Toggle Coverage – Init Mode

- After execution of **run_init_mode.do** macro appropriate toggle coverage report file is generated.
- Report file can be opened using **Toggle Coverage Viewer** from **Tools** menu
- You can check that some signals, e.g. D_L and D_H have not been properly initialized.

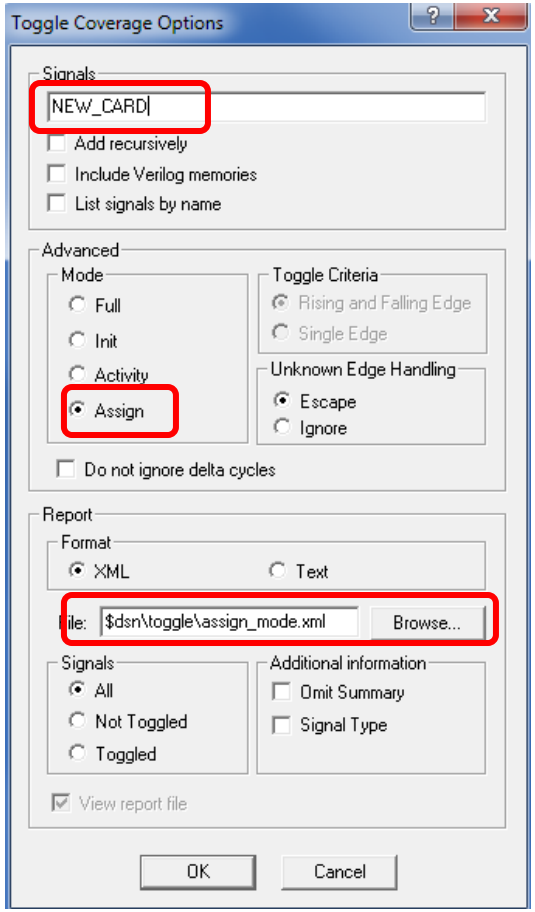
Signals | Summary

Signal Name	Covered
/V_BJACK_tb/BUST	YES
/V_BJACK_tb/D_H[1:0]	NO
/V_BJACK_tb/D_L[3:0]	NO
/V_BJACK_tb/END_SIM	YES
/V_BJACK_tb/GEN_CLK	YES
/V_BJACK_tb/GEN_RES	YES
/V_BJACK_tb/HOLD	YES
/V_BJACK_tb/L_H[6:0]	NO
/V_BJACK_tb/L_L[6:0]	NO
/V_BJACK_tb/LEDS[7:0]	YES
/V_BJACK_tb/NEW_CARD	YES
/V_BJACK_tb/res_file[31:0]	YES
/V_BJACK_tb/START	YES
/V_BJACK_tb/SYS_CLK	YES
/V_BJACK_tb/UUT/BUST	YES
/V_BJACK_tb/UUT/CLK	YES
/V_BJACK_tb/UUT/D_H[1:0]	NO
/V_BJACK_tb/UUT/D_L[3:0]	NO
/V_BJACK_tb/UUT/DO[3:0]	NO
/V_BJACK_tb/UUT/GEN_CLK	YES
/V_BJACK_tb/UUT/GEN_RES	YES
/V_BJACK_tb/UUT/HAND[4:0]	YES
/V_BJACK_tb/UUT/HOLD	YES
/V_BJACK_tb/UUT/I1/I0	YES

Toggle Coverage – Assign Mode

- The **Assign** mode provides information on how many pulses happened on a signal while the signal was being monitored.
- In Verilog designs, the Toggle Coverage counts **0** and **1** pulses. For VHDL the **L** and **H** values of the **std_ulogic** type are also included in the statistics.
- To invoke gathering assign data for the **NEW_CARD** signal, please execute **run_assign_mode.do** macro:

```
toggle -toggle_type assign
NEW_CARD
```



Toggle Coverage – Assign Mode

- After execution of **run_assign_mode.do** macro, a new toggle coverage report file is generated.
- This mode, for example, can be used to count how many times action “new card”is executed

Signals | Summary

Signal Name	Covered	0 assigns	1 assigns
/V_BJACK_tb/NEW_CARD	YES	15	15

Toggle Coverage – Full Mode

- In the **Full** mode signals are checked to detect if both rising and falling edges occurred. Additionally, the edge definition can be customized with the -posedge/-negedge switch.
- Macro run_full_mode.do gathers the full data for all ports of the “UUT”unit:
  - `toggle -toggle_type full -posedge"01 L1 Z1" /UUT/*`

Toggle Coverage – Full Mode

- Using **Full** mode you can verify whether proper edge has ever occurred on selected signals.
- In our example, the posedge has not occurred on LEDS(4), but the negedge did.

Signals | Summary

Signal Name	Covered	Negedges	Posedges
DO	[NO] 0/4	0	0
GEN_CLK	YES	1	1
GEN_RES	NO	0	0
HAND	[NO] 0/5	0	0
HOLD	NO	0	0
L_H	[NO] 0/7	0	0
L_L	[NO] 0/7	0	0
LEDS	[MIXED] 3/8		
LEDS[7]	NO	0	0
LEDS[6]	NO	0	0
LEDS[5]	NO	0	0
LEDS[4]	NO	1	0
LEDS[3]	NO	1	0
LEDS[2]	YES	1	1
LEDS[1]	YES	1	1
LEDS[0]	YES	1	1

Toggle Coverage – Activity Mode

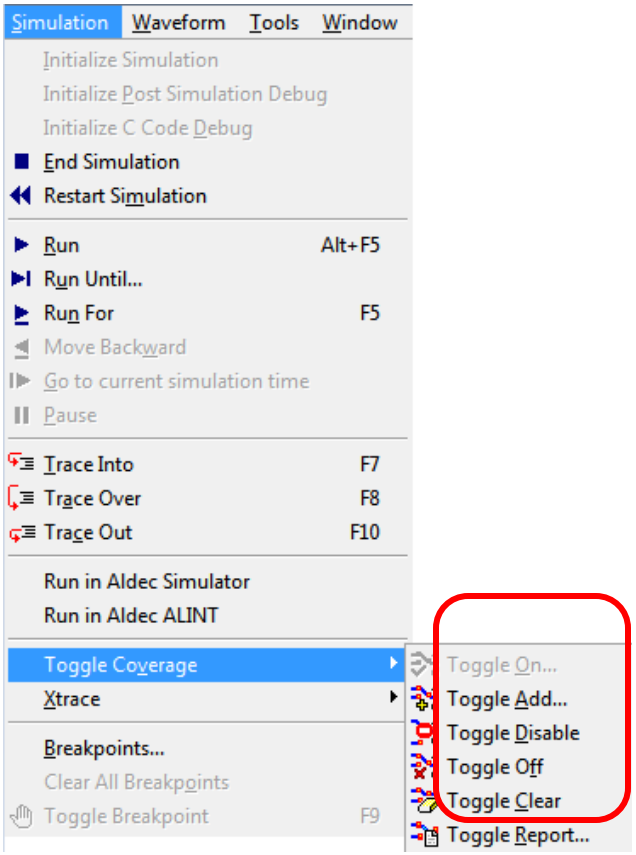
- The **Activity** mode provides information on how many rising and falling edges happened on a signal while the signal was monitored.
- The last example **run_activity_mode.do** gathers the activity data for all output ports.

Signals | Summary

Signal Name	Covered	Negedges	Posedges
/V_BJACK_tb/UUT/I17/I	[MIXED] 1/32	3	2
/V_BJACK_tb/UUT/I17/Q	[MIXED] 3/8	11	6
/V_BJACK_tb/UUT/I17/Q_I	[MIXED] 3/8	11	6
/V_BJACK_tb/UUT/I17/RESET	YES	3	2
/V_BJACK_tb/UUT/I17/tmp	NO	0	0
/V_BJACK_tb/UUT/I5/I	YES	13	14
/V_BJACK_tb/UUT/I5/O	YES	14	13
/V_BJACK_tb/UUT/I6/Ace	NO	0	0
/V_BJACK_tb/UUT/I6/BlackJack	[MIXED] 4/8	38	38
/V_BJACK_tb/UUT/I6/BUST	NO	0	0
/V_BJACK_tb/UUT/I6/CARD	[NO] 0/4	0	0
/V_BJACK_tb/UUT/I6/CLOCK	YES	192	193
/V_BJACK_tb/UUT/I6/HAND	[NO] 0/5	0	0
/V_BJACK_tb/UUT/I6/HOLD	NO	0	0
/V_BJACK_tb/UUT/I6/NEW_C	YES	15	14
/V_BJACK_tb/UUT/I6/NEW_G	YES	3	2
/V_BJACK_tb/UUT/I6/NEXT_C	YES	13	14
/V_BJACK_tb/UUT/I6/Total	[NO] 0/5	0	0
/V_BJACK_tb/UUT/I8/I0	NO	0	0
/V_BJACK_tb/UUT/I8/I1	NO	0	0
/V_BJACK_tb/UUT/I8/O	NO	0	0
/V_BJACK_tb/UUT/I9/A	[NO] 0/4	0	0
/V_BJACK_tb/UUT/I9/B	[NO] 0/5	0	0
/V_BJACK_tb/UUT/I9/S	NO	0	0

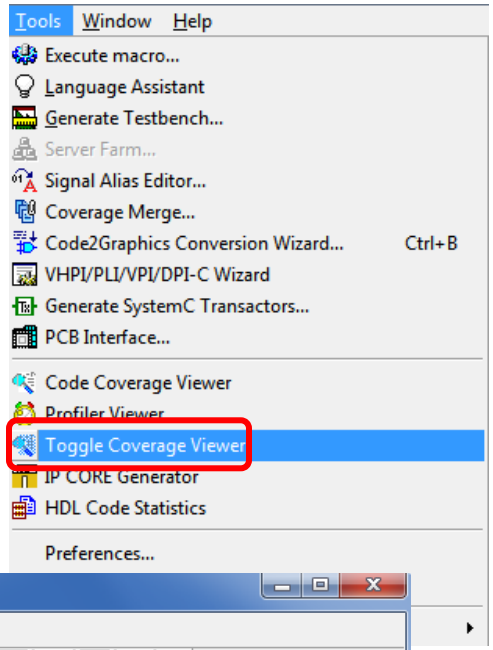
Toggle Coverage Options

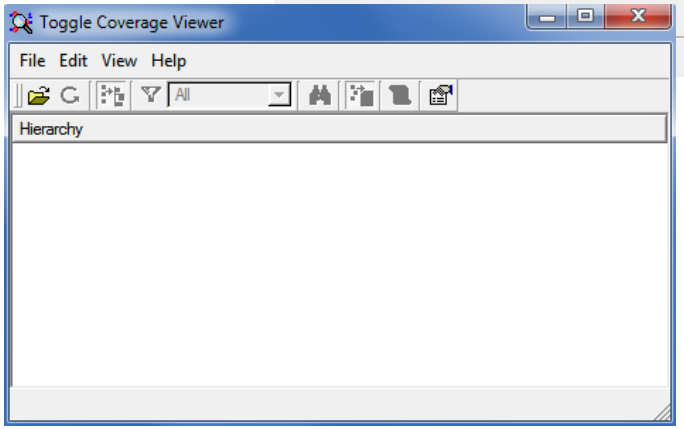
- The toggle report is written automatically when the simulation is finished, when the Toggle Coverage engine is switched off by selecting the **Toggle Off** option , or when you choose the **Toggle Report** option
  - **Toggle Add** option allows adding more signals to Toggle Coverage
  - **Toggle Disable** option temporarily disable collecting toggle data.
  - **Toggle Clear** option clears toggle data collected so far without saving it to a file.



Toggle Coverage Viewer

- **Toggle Coverage Viewer** can be opened from **Tools** menu
- To load the Toggle Coverage data collected during simulation run:
  - Select **Open**…from the **File** menu in the main toolbar.
  - Find toggle.xml file. It should have been created in the $DSN/toggle folder by default.

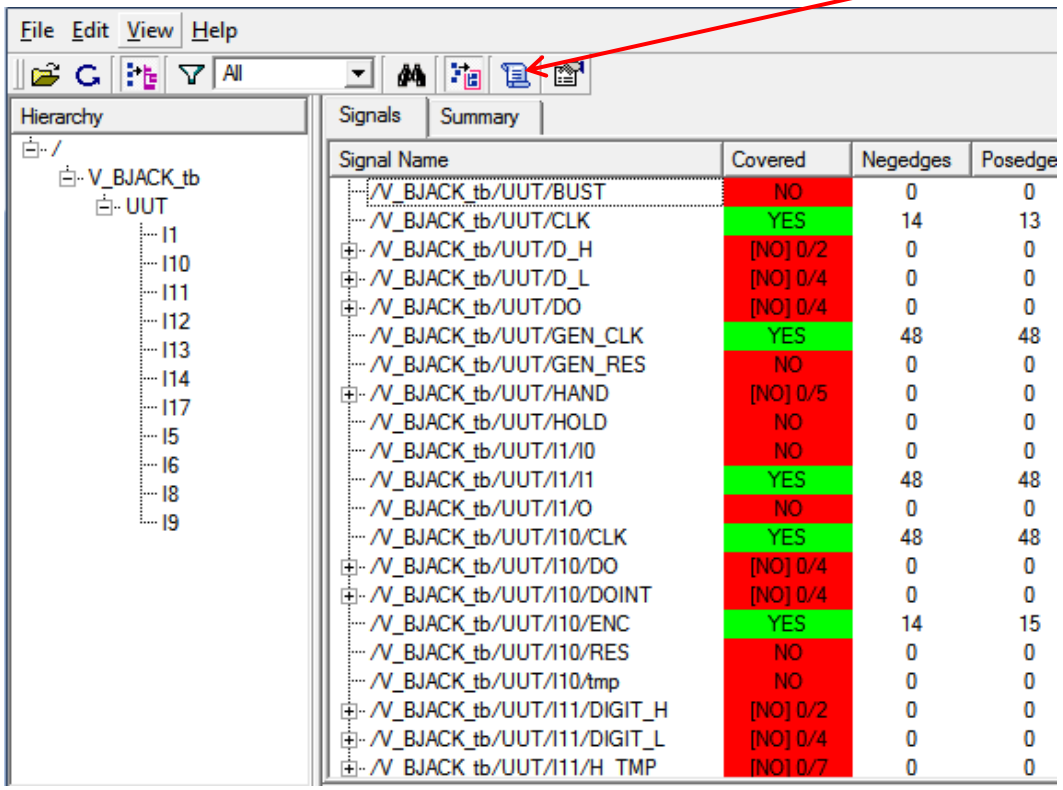




Toggle Coverage Viewer

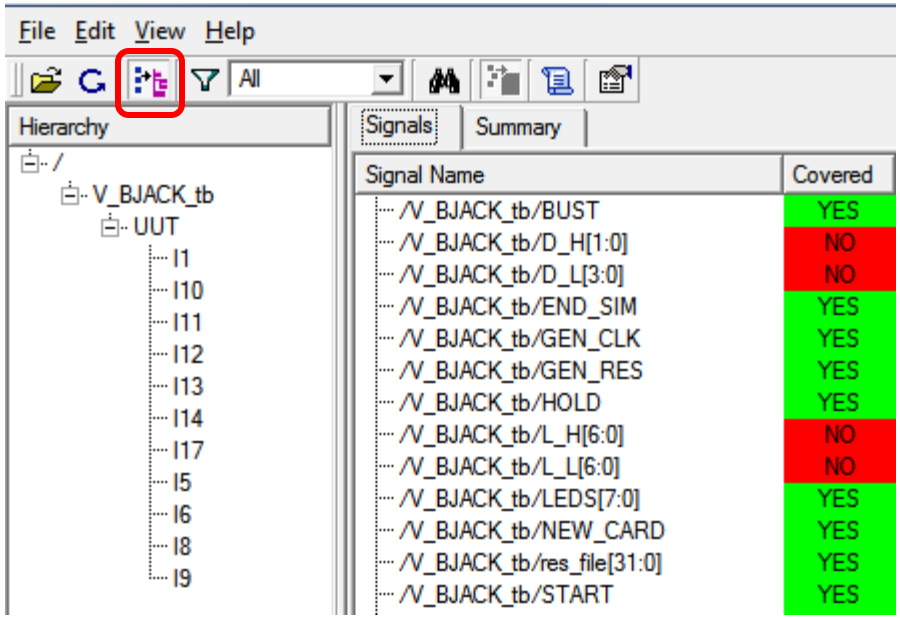
- By default the results are displayed in form of a flat list showing all signals and their hierarchical path
- Signals that have not been properly toggled are displayed in **red** and the others in **green**

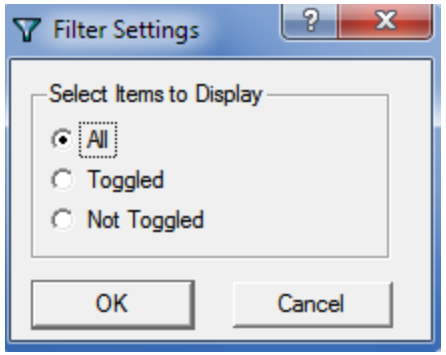
Generates text report

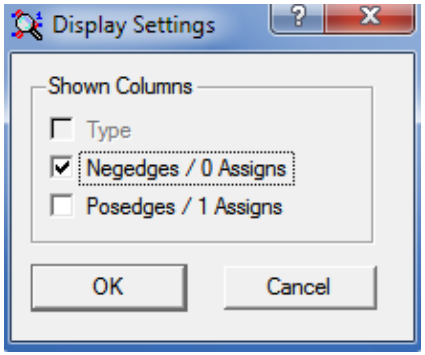


Toggle Coverage Viewer

- Using the **Show/Hide hierarchy** browser button hierarchical mode can be enabled
- **Filter Settings:** Allows to display either all the signals, ones that have been **covered** and **not covered**.
- **Display Settings**: Allows displaying the number of Posedges, Negedges and signal type (if such option has been used)

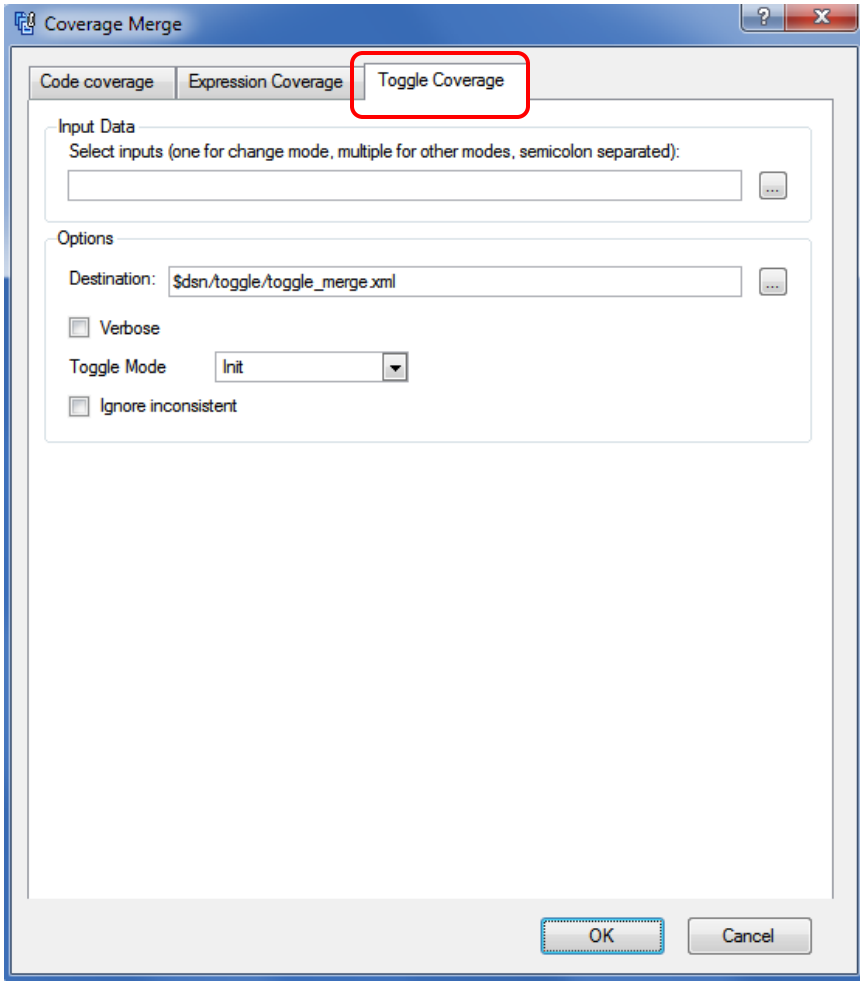






Merge Coverage Data

- The merge process can also be started by using the **Coverage Merge** dialog box, available from the **Tools** menu
- Select the **Toggle Coverage** tab, and configure the available options

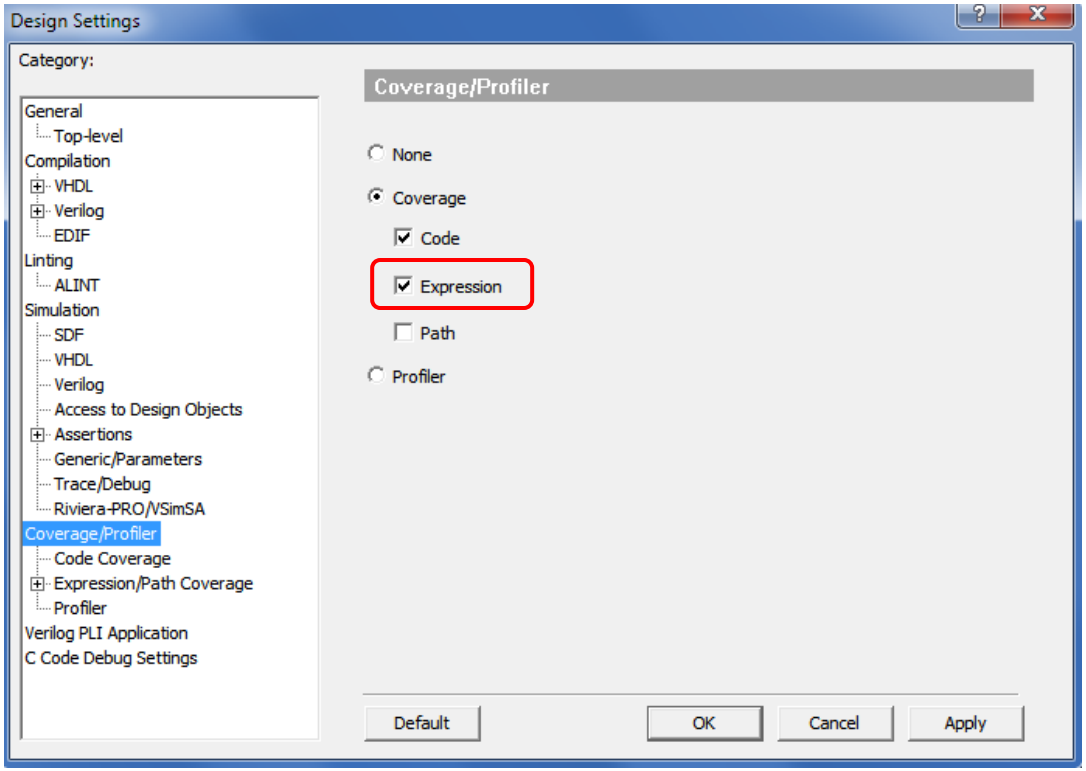


Expression/Condition Coverage

- **Expression Coverage** is a debugging tool that factorizes **logical expressions** and monitors them during simulation
- **Condition Coverage** is a part of the Expression Coverage engine that monitors and factorizes logical expressions used in **conditional statements**

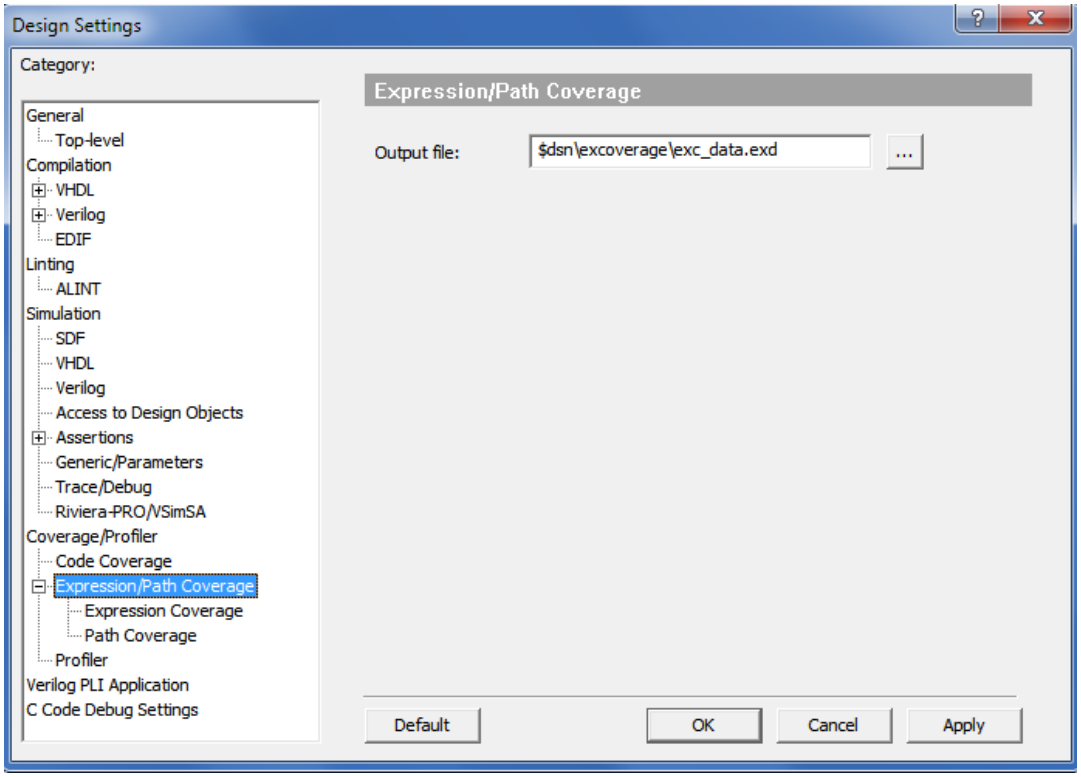
Enabling Expression Coverage - GUI

- To enable Expression Coverage you have to:
  - Open the **Design Settings** window from the **Design** menu
  - Select the **Coverage/Profiler** tab
  - Select Coverage radio button in the Enable section



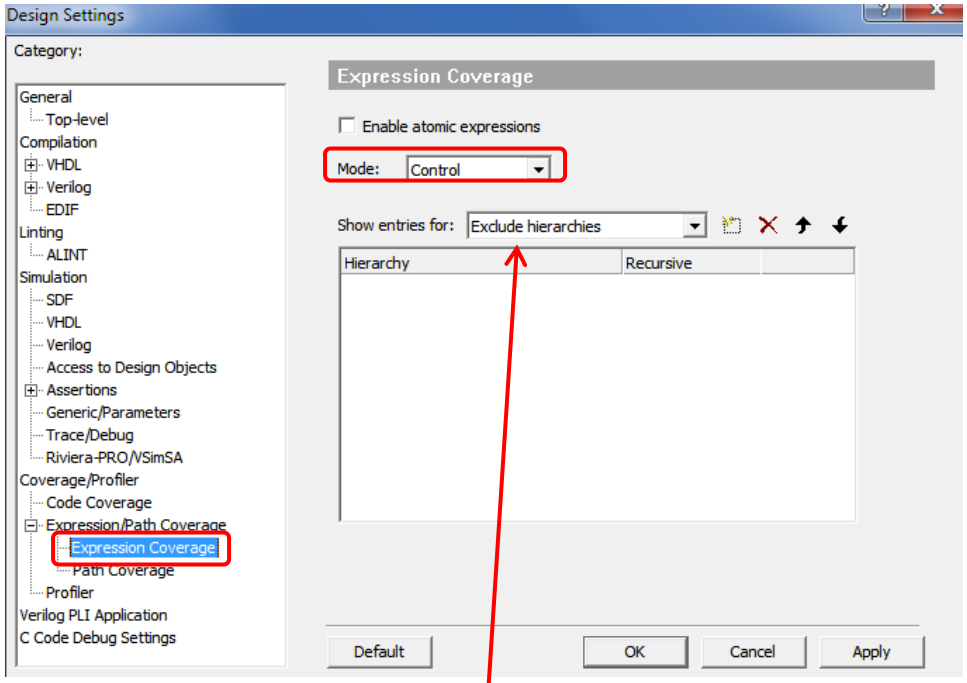
Select Output Directory- GUI

- To select output directory you have to:
  - Open the **Design Settings** window from the **Design** menu
  - Select the **Expression/ Path Coverage** tab
  - Select the path where you would like to save the coverage data



Expression Coverage modes - GUI

- There are two modes to evaluate the design.
  - **Control**: This mode controls expressions of single bit signals and checks whether each input of an expression has contributed to an expression result during simulation
  - **Vector**: This mode is an extension of the Control mode and can be used in case of expressions employing vectors
- Modes can be selected from **Expression Coverage** tab



Allows you to specify the list of design regions that will be included or excluded from collecting coverage statistics

Expression Coverage – Using Scripts

- All of the action described in previous slides can also be performed using commands as well
- Enabling Expression Coverage:

```
#use –exc argument in acom/alog command
acom -exc ./vhdl/uart.vhd
#use –exc and control/vector in asim command
asim -t 100ps -exc control transmit_tb
```

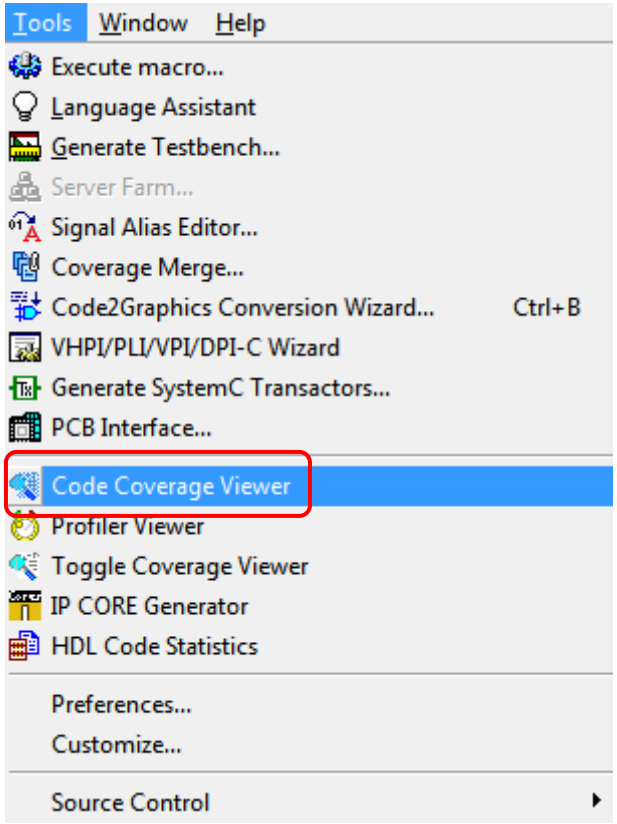
- Selecting and output directory and writing results in a file

```
#use excoverage write command to write to .exd file
excoverage write ./expcov/transmit.exd
#use excoverage report command to write to .txt/.html file
excoverage report ./expcov/transmit.exd ./expcov/transmit.txt
```

- Results also can be viewed using **Coverage Viewer**

Invoking Coverage Viewer

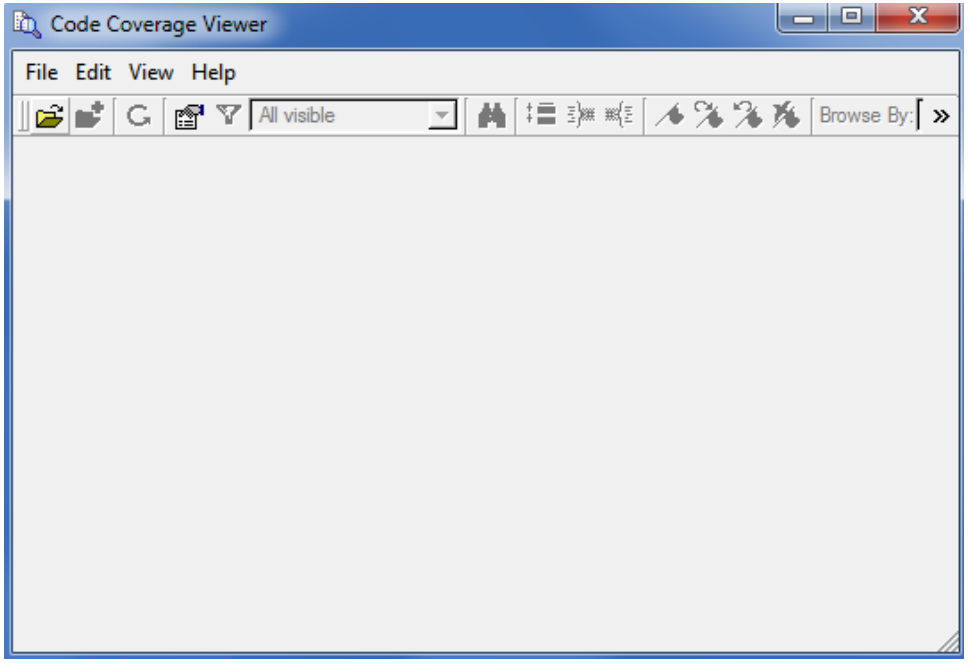
- When your simulation is finished, you can open the **Code Coverage Viewer** from the **Tools** menu.
- All data gathered by **Expression Coverage** can be presented in a **graphical** and **textual** form in the **Code Coverage Viewer** window.



Opening Expression Coverage File

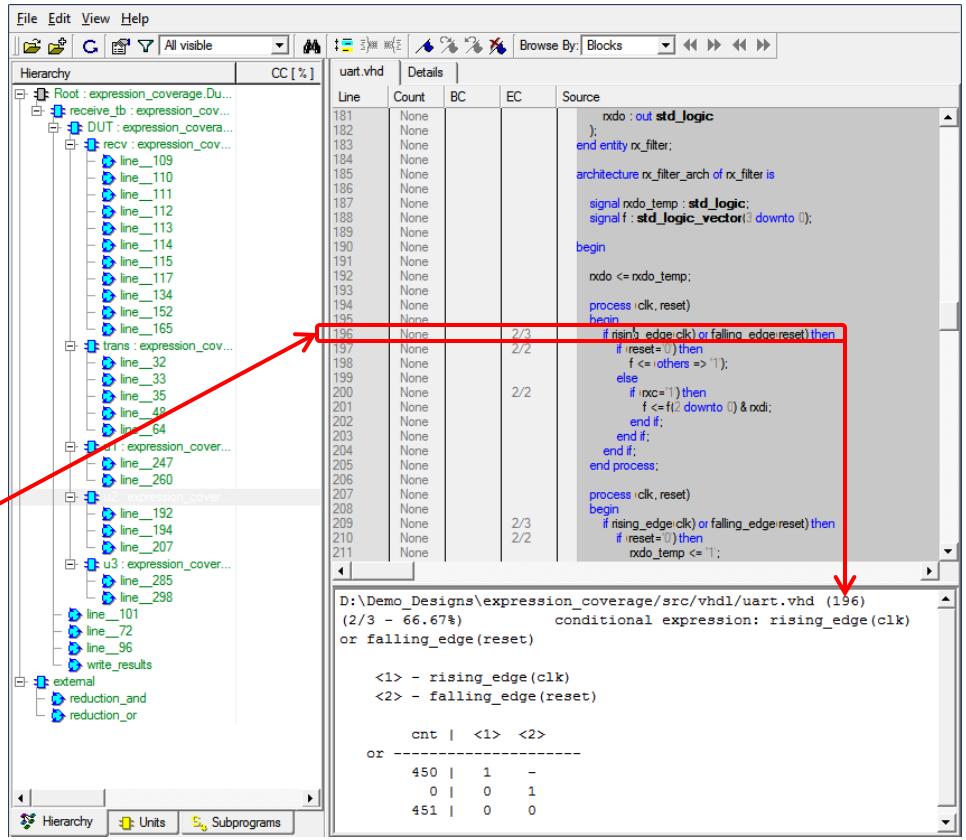
- To load the Expression Coverage data collected during simulation run:

- Select **Open**…from the **File** menu or use button in the main toolbar 
- Find **<results>.exd** file. It should have been created in previously specified path



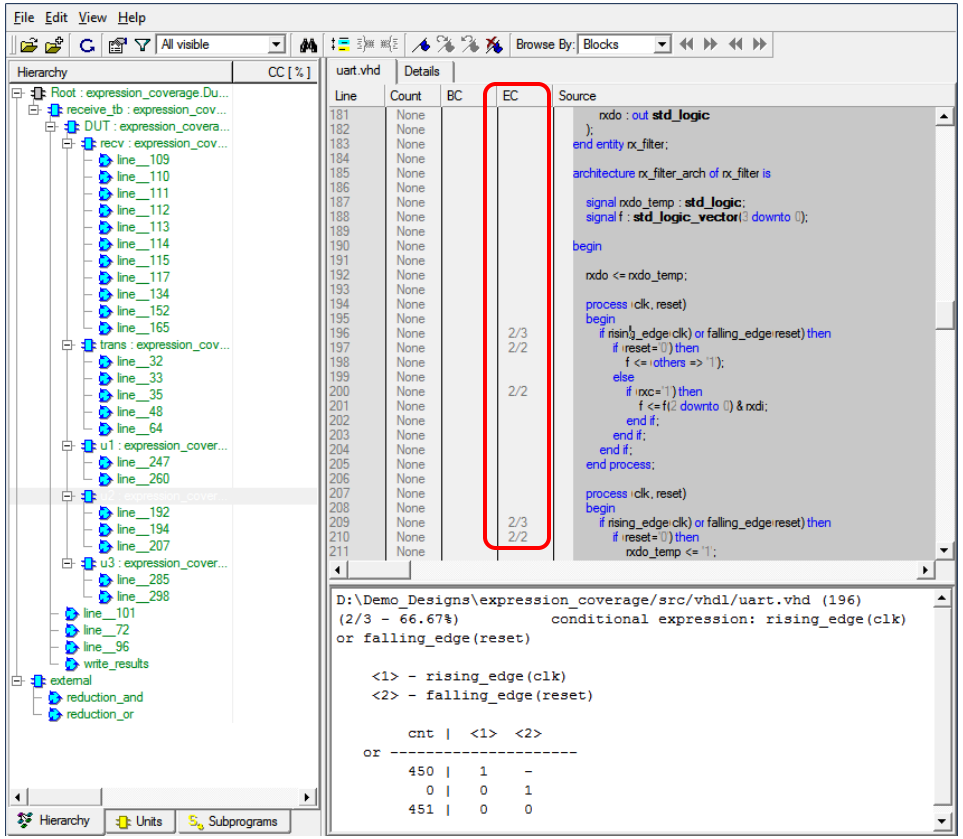
Expression Coverage Viewer

- There are three panels in the Code Coverage Viewer window:
  - **Hierarchy pane** - displays the hierarchical structure of the design
  - **Source Code/Details** pane – displays HDL source code
- Truth Table for particular line can be opened by clicking on HDL code



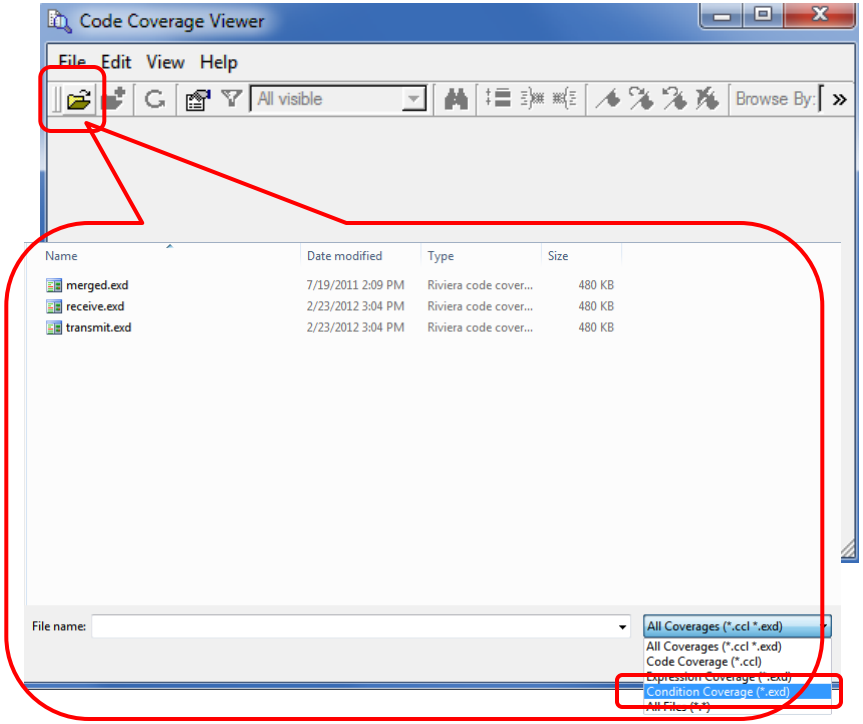
Expression Coverage Viewer

- Expression Coverage(EC) column displays coverage data for logical expressions.
- This information is expressed as the ratio of the number of exercised expression cases to all possible to cover expression cases for the given type of the expression



Condition Coverage Results

- When the simulation session is finished, condition coverage statistics are processed and saved to a coverage database (**.exd*)
- **Condition Coverage** data is a subset of statistics produced by the Expression Coverage engine
- **Condition Coverage** share the database with Expression Coverage, i.e. both coverage statistics are collected simultaneously during the same simulation session



Conditional Coverage Report

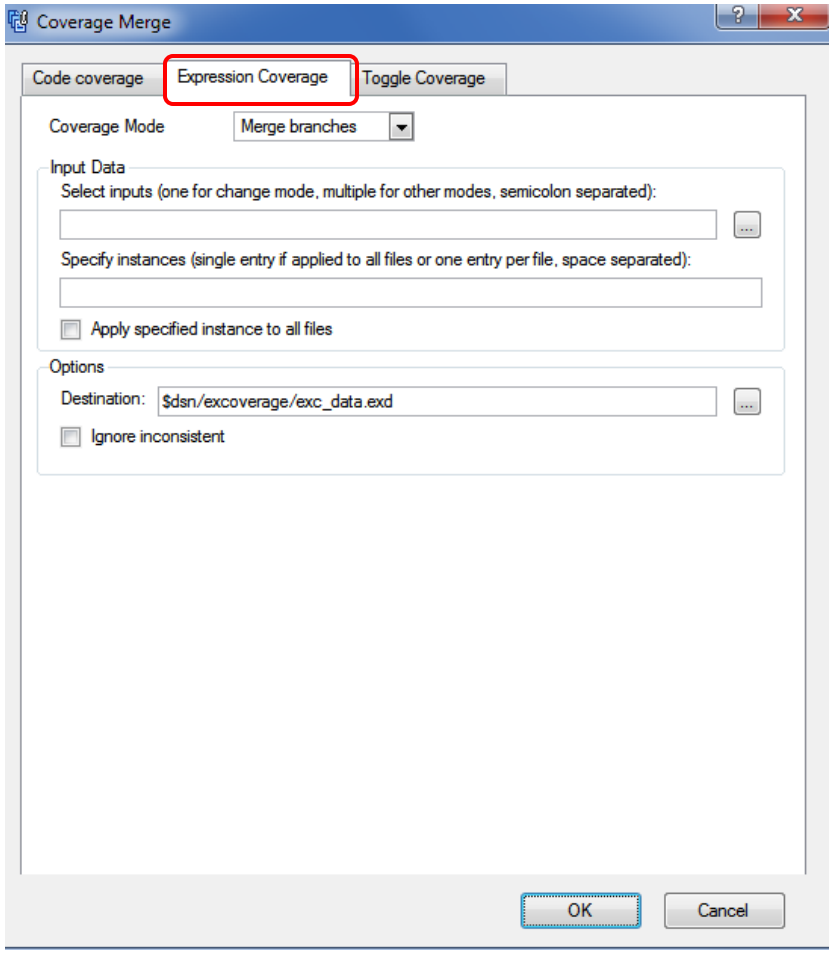
- Expression of logical equality in line 155 can evaluate to either true or false, the truth table has two rows. The total number of expression cases possible to cover is 2 for this particular type of expression. In this case, the expression evaluated 450 times during simulation: 5 times to Logical Equality and 445 times to Logical Inequality, which means that all possible expression cases (2) for this type of the expression have been covered (denoted in the report by: 2/2 - 100.00%)

```
D:\Demo_Designs\expression_coverage/src/vhdl/uart.vhd (155)
(2/2 - 100%)          conditional expression: (reset = '0')

    <1> - reset
    <2> - '0'

        cnt |  <1>     <2>
  = ---------------------
          5 |  lhs   =  rhs
        445 |  lhs  /=  rhs
```

```
154  None        2/3   if (rising_edge(clk) or falling_edge(reset)) then
155  None        2/2     if (reset='0') then
156  None                  shf_reg <= (others => '0');
157  None                else
158  None        5/6       if (sample and rxc and frame) = '1' then
159  None                    shf_reg <= rxd & shf_reg(7 downto 1);
```

Merge Coverage Data

- The merge process can also be started by using the **Coverage Merge** dialog box, available from the **Tools** menu
- Select the **Expression Coverage** tab, choose the merge mode and configure options available for selected mode

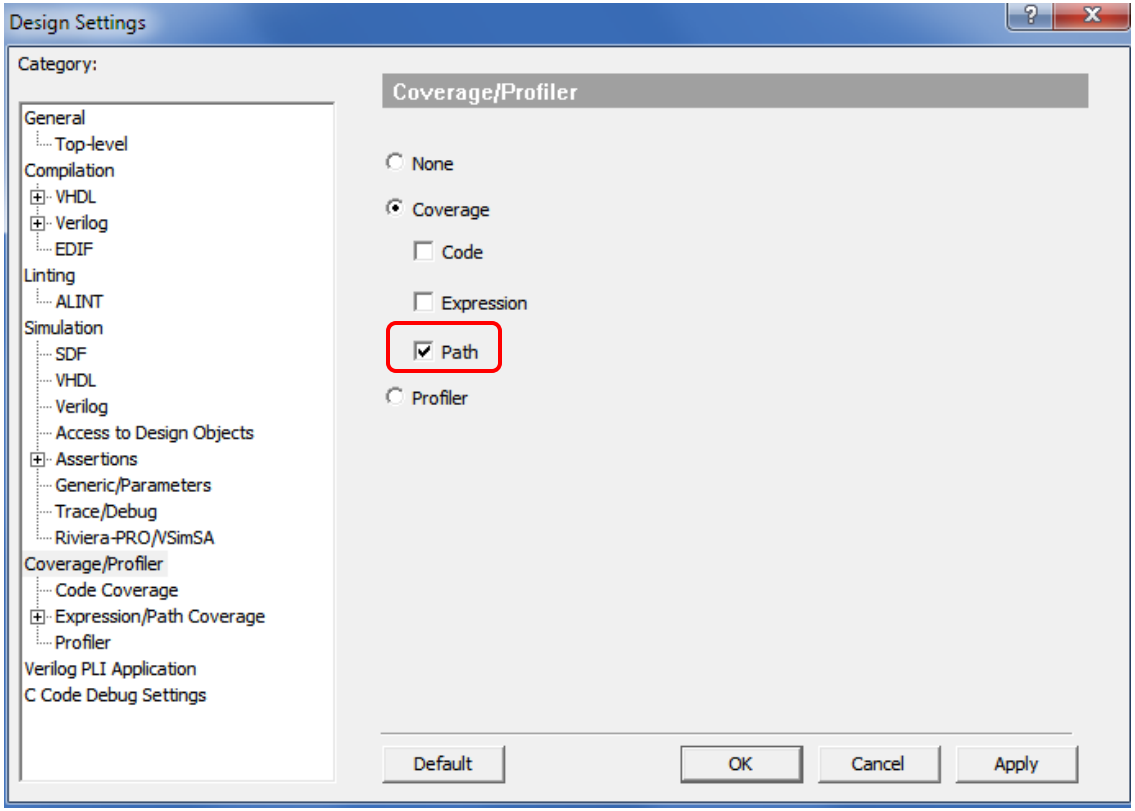


Path Coverage (VHDL Only)

- **Path Coverage** collects information about the execution of program paths and analyzes whether all possible sequences of program execution were verified by a testbench.
- A program path is a sequence of conditional statement executions performed in a particular order. In this type of analysis, except checking whether a logical condition was met (and in consequence a particular statement was executed), the tool also collects information about the order the consecutive statements are executed, the branches that are examined, and how logical conditions evaluated during simulation.

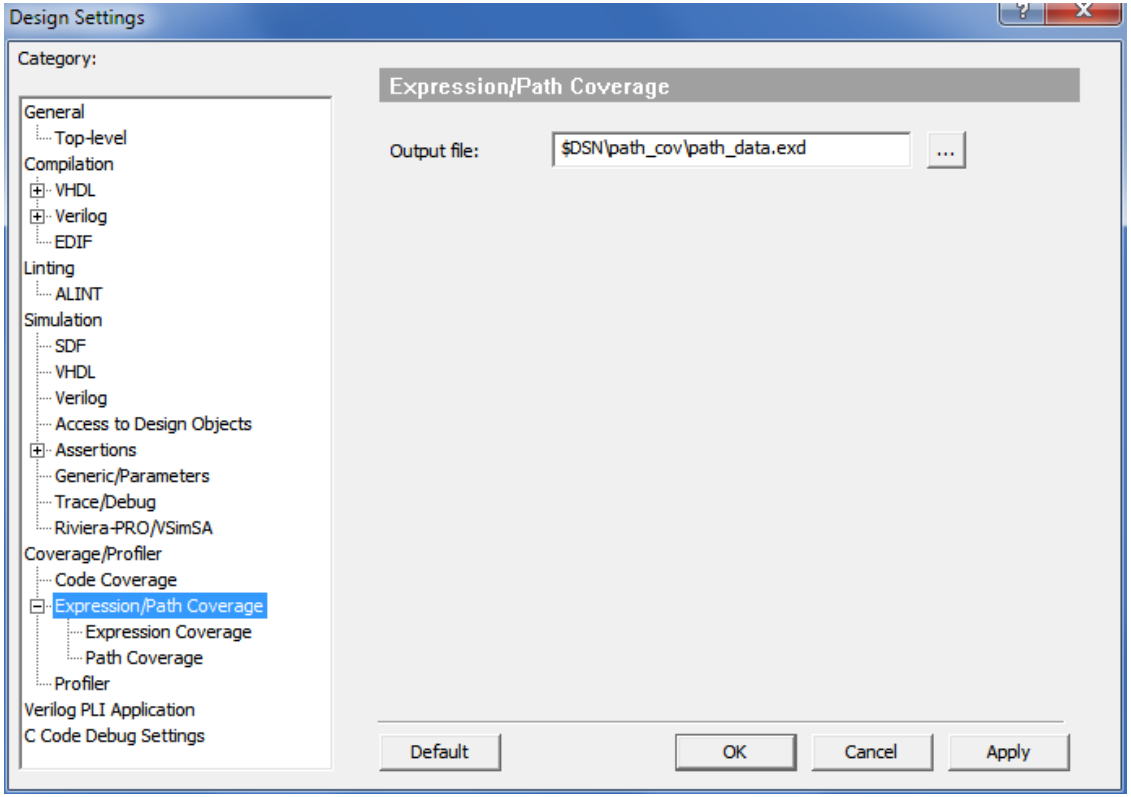
Enabling Path Coverage - GUI

- To enable Path Coverage you have to:
  - Open the **Design Settings** window from the **Design** menu
  - Select the **Coverage/Profiler** tab
  - Select Coverage radio button in the Enable section



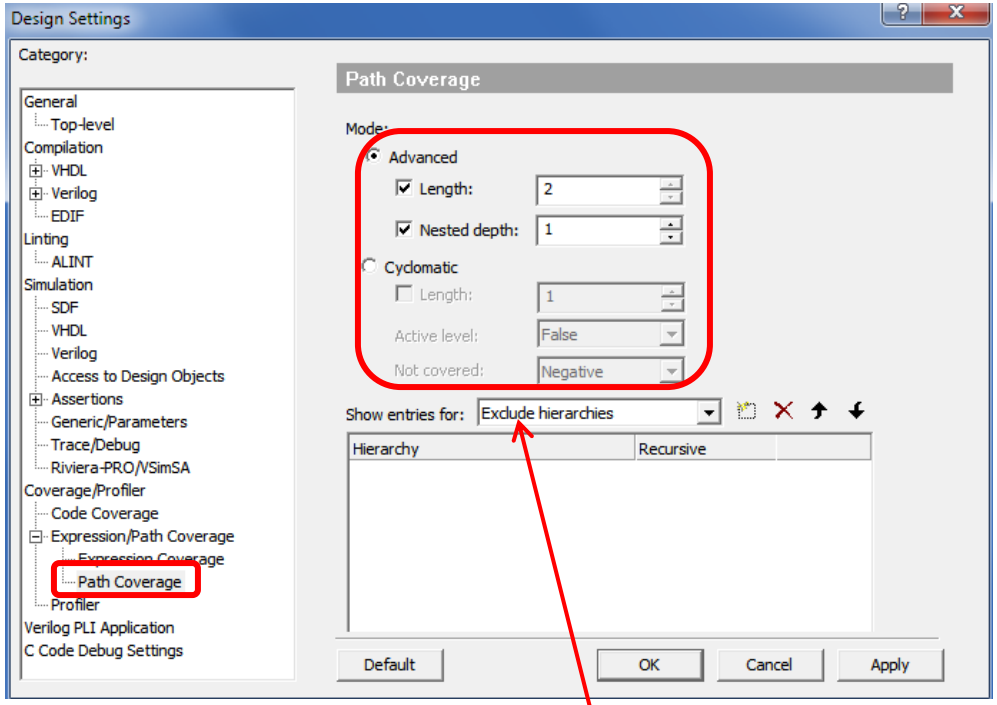
Select Output Directory- GUI

- To select output directory you have to:
  - Open the **Design Settings** window from the **Design** menu
  - Select the **Expression/ Path Coverage** tab
  - Select the path where you would like to save the coverage data



Path Coverage modes - GUI

- Path Coverage statistics can be collected in one of two available working modes.
  - **Cyclomatic Mode**
  - **Advanced Mode**
- Modes can be selected from **Path Coverage** tab



Allows you to specify the list of design regions that will be included or excluded from collecting coverage statistics

Note: Please refer to HELP documentation for more details on two modes

Path Coverage – Using Scripts

- All of the action described in previous slides can also be performed using commands as well
- Enabling Path Coverage:

```
#use –pac argument in acom command
acom -pac ./vhdl/uart.vhd
#use –pac advanced/cyclomatic in asim command
asim -t 100ps -pac advanced transmit_tb
```

- Selecting and output directory and writing results in a file

```
#use pathcoverage write command to write to .exd file
pathcoverage write ./pathcov/transmit.exd
#use pathcoverage report command to write to .txt file
pathoverage report ./pathc/transmit.exd ./pathc/transmit.txt
```

Path Coverage – Viewing Results

- When simulation is finished, coverage statistics are calculated and results are automatically saved to a coverage database
- These data needs to be exported to a textual report file
- The Path Coverage report is based on data extracted from a binary database (*.exd) common for both Path Coverage and Expression/Condition Coverage
- You can use scripts or GUI to create textual report from path coverage results.

Path Coverage – Viewing Results

- Using Scripts:

```
#use pathcoverage report command to write to .txt file
pathoverage report ./pathc/transmit.exd ./pathc/transmit.txt
```

- Using GUI:
  - Open the **Code Coverage Viewer** from the **Tools** menu
  - Select **Open**…from the **File** menu or use button in the main toolbar and open **<results>.exd** file
  - Go to menu **File** and select **Expression Coverage Report** and save the report in textual format