Design Profiler

Learn how to generate profiler report and how to read it

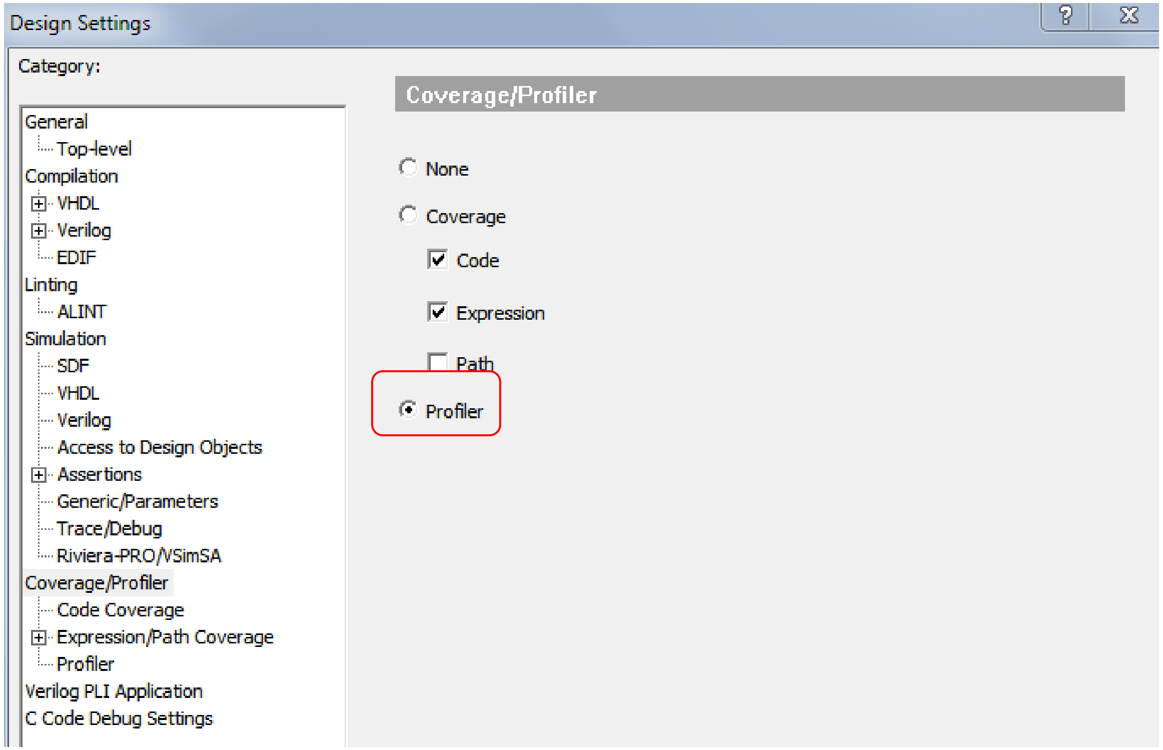
Design Profiler

- **Design Profiler** is a tool integrated within Active-HDL
- The **Design Profiler** provides insight into how the CPU is utilized during simulation. When the simulation is running, the profiling engine counts CPU ticks for each HDL statement. Profiling data is used to identify design units or code sections that put the greatest strain on the simulator

Enabling Design Profiler

- To enable Design Profiler you have to:

- Open the **Design Settings** window from the **Design** menu
- Select the **Coverage/Profiler** category
- Check the **Profiler** option



Design Settings

Category:

- General
  - Top-level
- Compilation
  - VHDL
  - Verilog
  - EDIF
- Linting
  - ALINT
- Simulation
  - SDF
  - VHDL
  - Verilog
  - Access to Design Objects
  - Assertions
  - Generic/Parameters
  - Trace/Debug
  - Riviera-PRO/VSimSA
- Coverage/Profiler
  - Code Coverage
  - Expression/Path Coverage
  - Profiler
- Verilog PLI Application
- C Code Debug Settings

Coverage/Profiler

○ None

○ Coverage

☑ Code

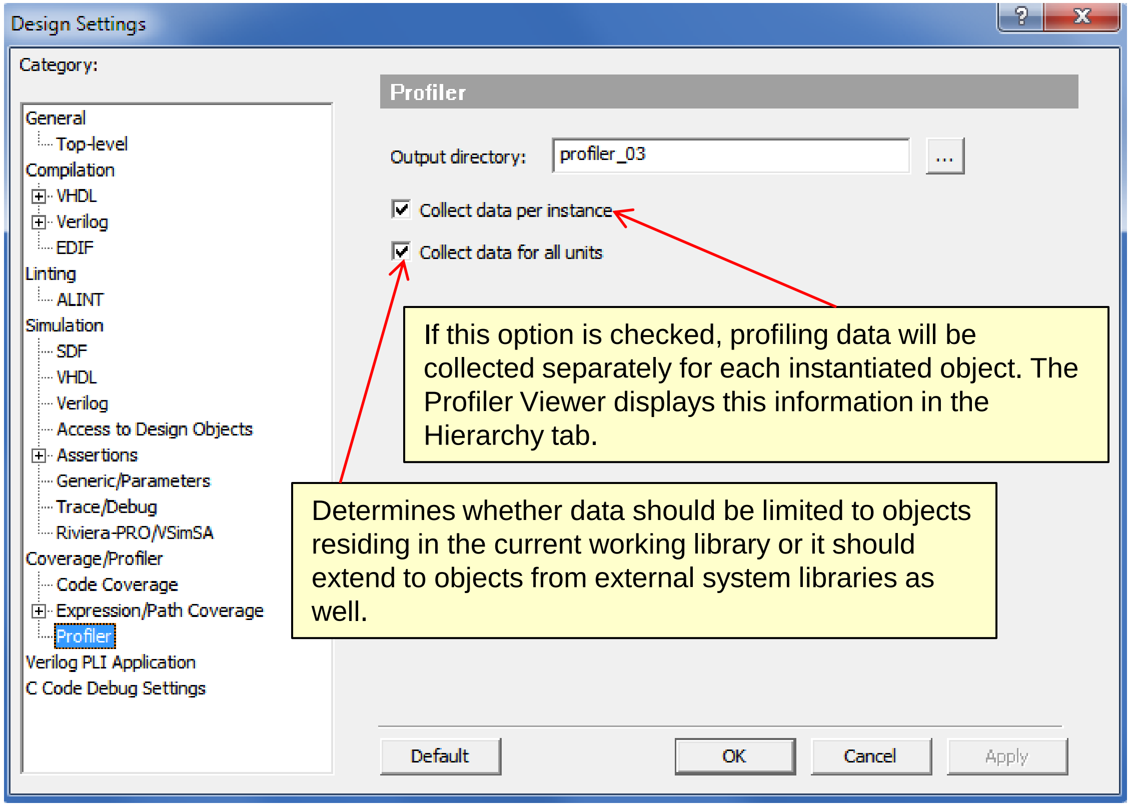
☑ Expression

□ Path

● Profiler

Enabling Design Profiler (Cont.)

- Now click on the **Profiler** category from the list
- Select the **output file folder** where you want profiler data to be saved
- Press OK button



Design Settings

Category:

Profiler

Output directory: profiler_03

☑ Collect data per instance

☑ Collect data for all units

If this option is checked, profiling data will be collected separately for each instantiated object. The Profiler Viewer displays this information in the Hierarchy tab.

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.

Enabling Design Profiler (Cont.)

- When a design simulation is initialized by a DO-macro file and you would like to use Design Profiler, you have to initialize simulation with following options:

```
asim -profiler -tbp_dest $DSN/profiler count
```

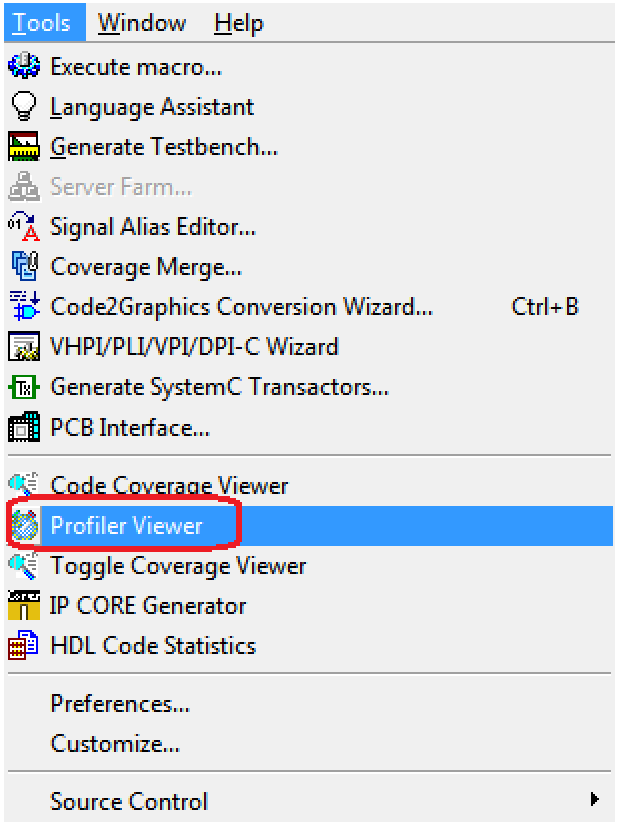
- This will enable Design Profiler data gathering in default mode i.e. information will be collected for each unit.

```
asim -profiler -profiler_hierarchy -tbp_dest $DSN/profiler count
```

Note: Refer to **Help** documentation for more details on **asim** command usage

Invoking Design Profiler Viewer

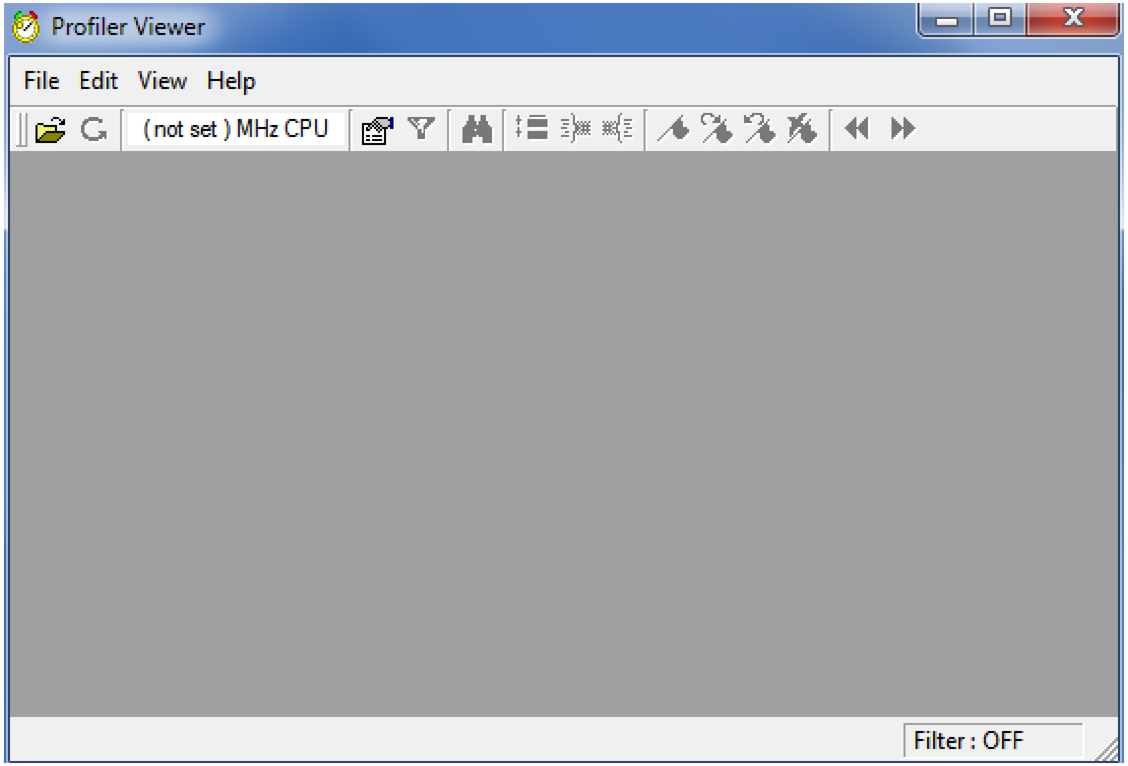
- When your simulation is finished, you can run the **Design Profiler** Viewer from the **Tools** menu
- All data gathered by Design Profiler can be presented in a graphical or textual form in the **Design Profiler Viewer** window



Design Profiler Viewer

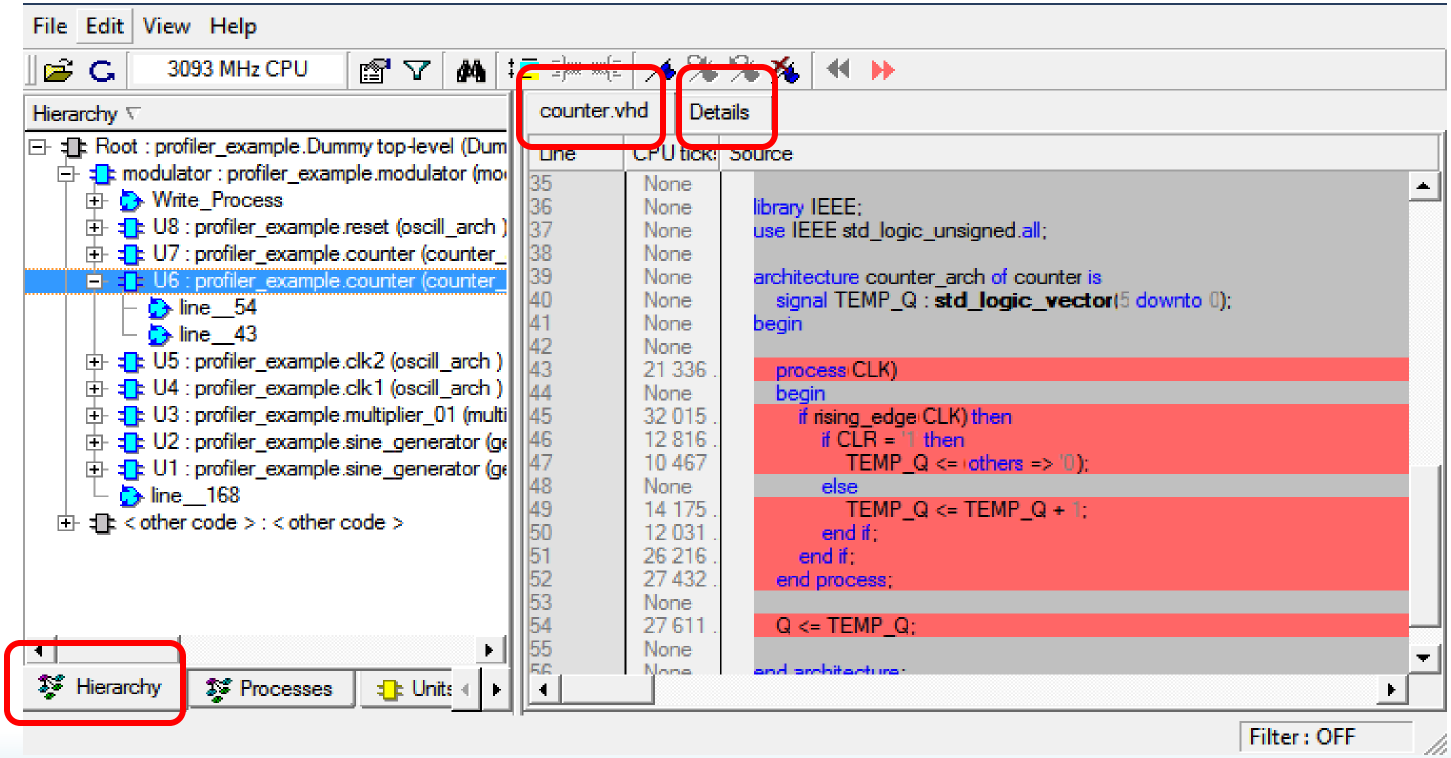
- To load the Design Profiler data collected during simulation run:

- Select **Open**…from the **File** menu or use button 📂 in the main toolbar.
- Find **profiler.tbp** file. It should have been created in previously specified path.



Design Profiler Viewer (Cont.)

- There are two panels in the Design Profiler Viewer window:
- **Hierarchy** pane - displays the hierarchical structure of the design
- **Source Code /Details** pane

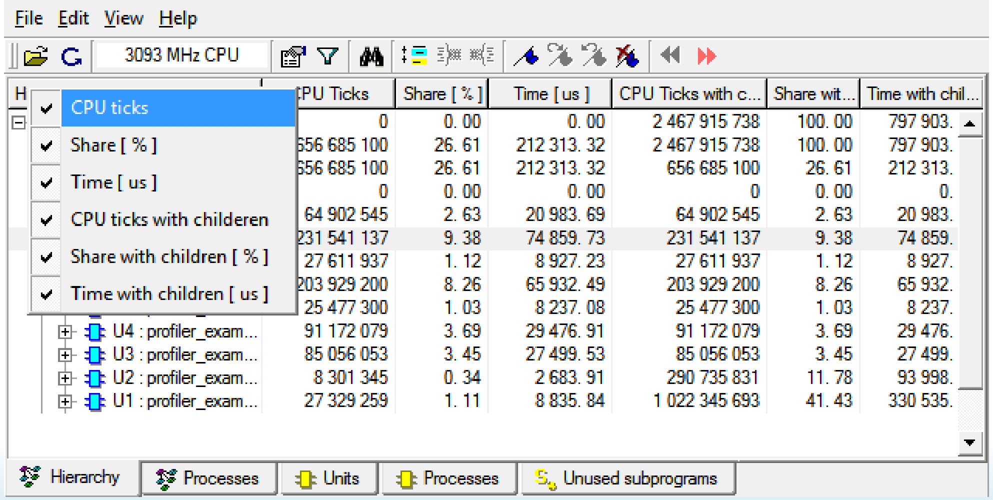


```
Root : profiler_example.Dummy top-level (Dum
  modulator : profiler_example.modulator (mo
    Write_Process
    U8 : profiler_example.reset (oscill_arch )
    U7 : profiler_example.counter (counter_
    U6 : profiler_example.counter (counter_
      line__54
      line__43
    U5 : profiler_example.clk2 (oscill_arch )
    U4 : profiler_example.clk1 (oscill_arch )
    U3 : profiler_example.multiplier_01 (multi
    U2 : profiler_example.sine_generator (ge
    U1 : profiler_example.sine_generator (ge
    line__168
  < other code > : < other code >
```

```
Line  CPU ticks  Source
35    None
36    None       library IEEE;
37    None       use IEEE.std_logic_unsigned.all;
38    None
39    None       architecture counter_arch of counter is
40    None          signal TEMP_Q : std_logic_vector(5 downto 0);
41    None       begin
42    None
43    21 336 .      process(CLK)
44    None          begin
45    32 015 .         if rising_edge(CLK) then
46    12 816 .            if CLR = '1' then
47    10 467                 TEMP_Q <= (others => '0');
48    None                else
49    14 175 .               TEMP_Q <= TEMP_Q + 1;
50    12 031 .            end if;
51    26 216 .         end if;
52    27 432 .      end process;
53    None
54    27 611 .      Q <= TEMP_Q;
55    None
56    None       end architecture;
```

Hierarchy Window

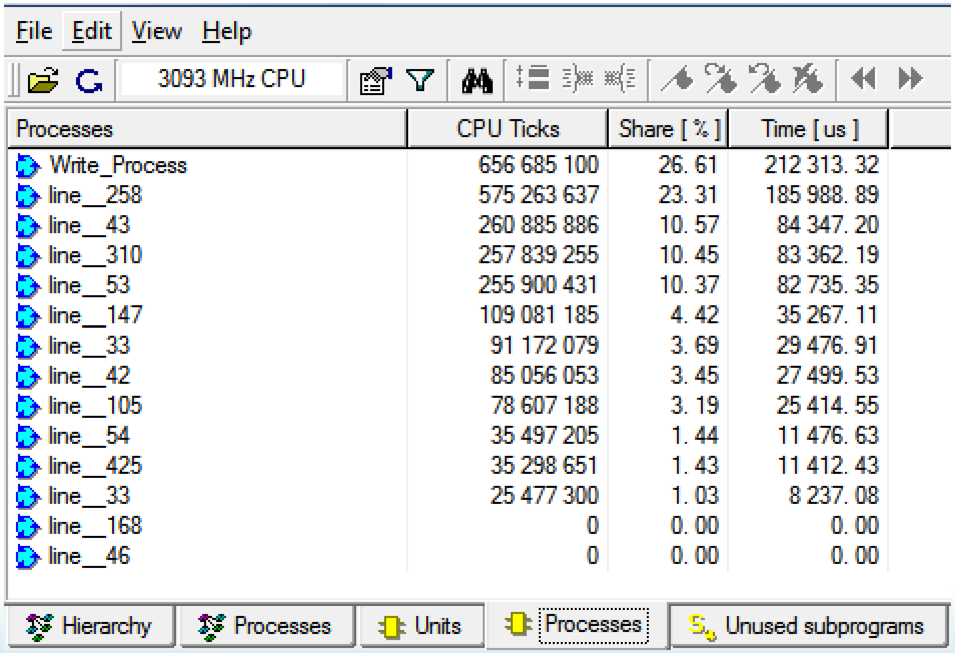
- The Hierarchy tab is divided into several columns:
  - **Hierarchy:** Shows an expandable tree with the design structure
  - **CPU Ticks:** Shows the number of CPU ticks that were required during simulation to execute the code for the object highlighted in the hierarchy tree
  - **Share [%]:** Shows the share in the total simulation time for the object highlighted in the hierarchy tree
  - **Time [us]:** Shows time in microseconds that was required during simulation to execute the code for the object highlighted in the hierarchy tree



H	PU Ticks	Share [%]	Time [us]	CPU Ticks with c...	Share wit...	Time with chil...
⊟	0	0. 00	0. 00	2 467 915 738	100. 00	797 903.
	656 685 100	26. 61	212 313. 32	2 467 915 738	100. 00	797 903.
	656 685 100	26. 61	212 313. 32	656 685 100	26. 61	212 313.
	0	0. 00	0. 00	0	0. 00	0.
	64 902 545	2. 63	20 983. 69	64 902 545	2. 63	20 983.
	231 541 137	9. 38	74 859. 73	231 541 137	9. 38	74 859.
	27 611 937	1. 12	8 927. 23	27 611 937	1. 12	8 927.
	203 929 200	8. 26	65 932. 49	203 929 200	8. 26	65 932.
	25 477 300	1. 03	8 237. 08	25 477 300	1. 03	8 237.
⊞ U4 : profiler_exam...	91 172 079	3. 69	29 476. 91	91 172 079	3. 69	29 476.
⊞ U3 : profiler_exam...	85 056 053	3. 45	27 499. 53	85 056 053	3. 45	27 499.
⊞ U2 : profiler_exam...	8 301 345	0. 34	2 683. 91	290 735 831	11. 78	93 998.
⊞ U1 : profiler_exam...	27 329 259	1. 11	8 835. 84	1 022 345 693	41. 43	330 535.

Hierarchy Window (Cont.)

- **Processes Tab (Hierarchical)**shows data for individual processes in the design hierarchy
- **Units tab** in the left pane shows profiling statistics for all units used in the design, irrespective of their position in the hierarchy tree. If a given unit is instantiated more than once in the design, profiling data is merged for all instances of that unit

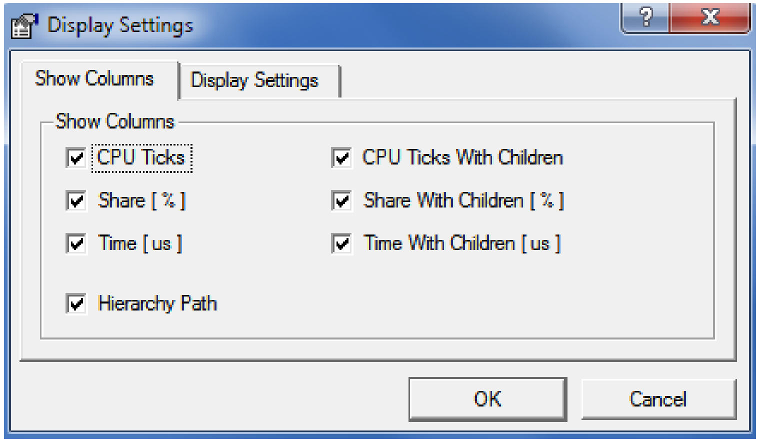


Processes	CPU Ticks	Share [%]	Time [us]
Write_Process	656 685 100	26. 61	212 313. 32
line__258	575 263 637	23. 31	185 988. 89
line__43	260 885 886	10. 57	84 347. 20
line__310	257 839 255	10. 45	83 362. 19
line__53	255 900 431	10. 37	82 735. 35
line__147	109 081 185	4. 42	35 267. 11
line__33	91 172 079	3. 69	29 476. 91
line__42	85 056 053	3. 45	27 499. 53
line__105	78 607 188	3. 19	25 414. 55
line__54	35 497 205	1. 44	11 476. 63
line__425	35 298 651	1. 43	11 412. 43
line__33	25 477 300	1. 03	8 237. 08
line__168	0	0. 00	0. 00
line__46	0	0. 00	0. 00

- **Processes Tab (Flat):**If the unit in which the process is defined is instantiated more than once in the design, profiling data is merged for all instances of the process
- The **Unused Subprograms tab** lists all subprograms that were not executed during the simulation process

Hierarchy Window (Cont.)

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



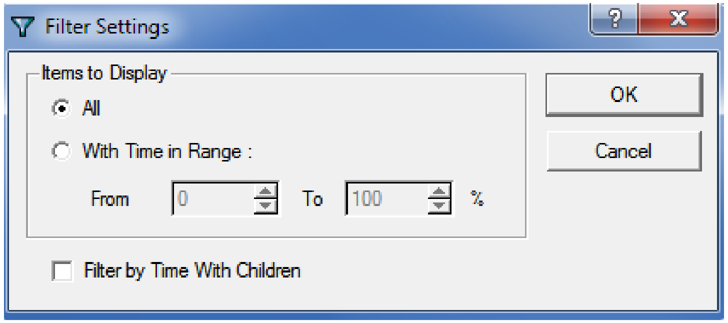
Display Settings

Show Columns | Display Settings

Show Columns

☑ CPU Ticks
☑ CPU Ticks With Children
☑ Share [%]
☑ Share With Children [%]
☑ Time [us]
☑ Time With Children [us]
☑ Hierarchy Path

- Display settings allows you to configure and customize view of gathered data by Design Profiler



Filter Settings

Items to Display

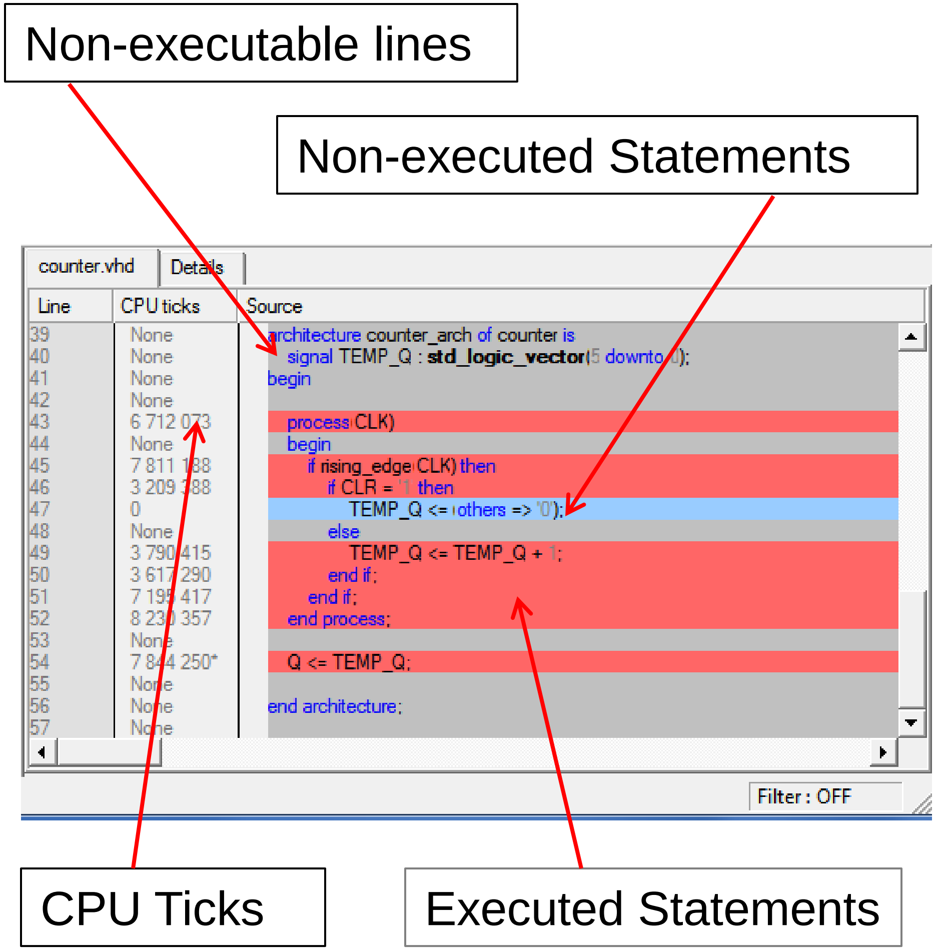
● All
○ With Time in Range :
From 0 To 100 %

□ Filter by Time With Children

- Display settings allows you to configure and customize view of gathered data by Design Profiler

Source Code Tab

- The source code of unit or process selected in **Hierarchy** pane is displayed in the **Source** tab
- Executed statements are displayed in red color. The number of CPU ticks is also shown to the left of corresponding line
- Statements that were not executed at all are shown in blue
- Non-executable lines or lines are displayed against a gray background



Non-executable lines

Non-executed Statements

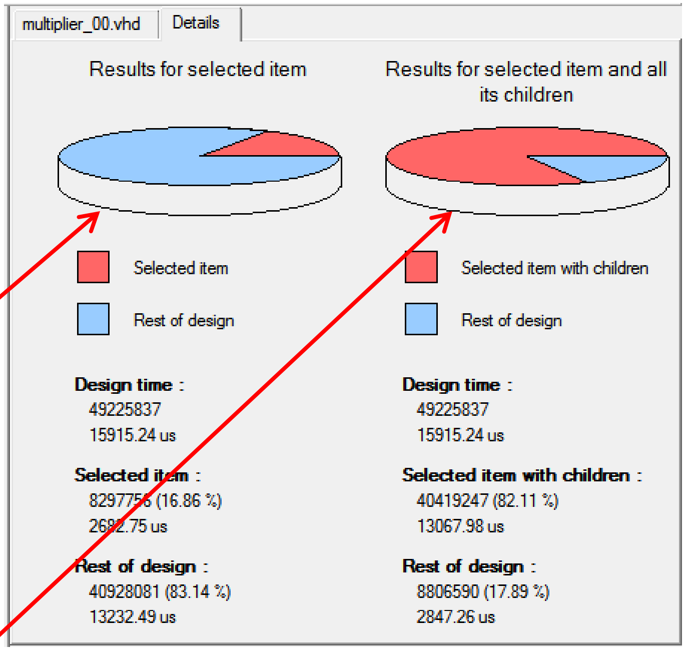
```
Line  CPU ticks   Source
39    None        architecture counter_arch of counter is
40    None          signal TEMP_Q : std_logic_vector(5 downto 0);
41    None        begin
42    None
43    6 712 073     process(CLK)
44    None          begin
45    7 811 188       if rising_edge(CLK) then
46    3 209 388         if CLR = '1' then
47    0                   TEMP_Q <= (others => '0');
48    None              else
49    3 790 415           TEMP_Q <= TEMP_Q + 1;
50    3 617 290         end if;
51    7 195 417       end if;
52    8 230 357     end process;
53    None
54    7 844 250*    Q <= TEMP_Q;
55    None
56    None        end architecture;
57    None
```

CPU Ticks

Executed Statements

Details Tab

- The **Details** tab shows pie charts with profiling statistics.
- The display on the **Details** tab is synchronized with the left pane of the Profiler Viewer. Selecting a unit or a process on the **Hierarchy, Processes**, or **Units** tab brings up the appropriate chart.
- The first chart (**Results for selected item**) shows statistics that do not account for objects nested further down in the hierarchy tree.
- The second chart (**Results for selected item and all its children**) is based on statistics that include nested objects. Accordingly, the time share for the top-level unit is 100%



Results for selected item

🟥 Selected item

🟦 Rest of design

Design time :
49225837
15915.24 us

Selected item :
8297756 (16.86 %)
2682.75 us

Rest of design :
40928081 (83.14 %)
13232.49 us

Results for selected item and all its children

🟥 Selected item with children

🟦 Rest of design

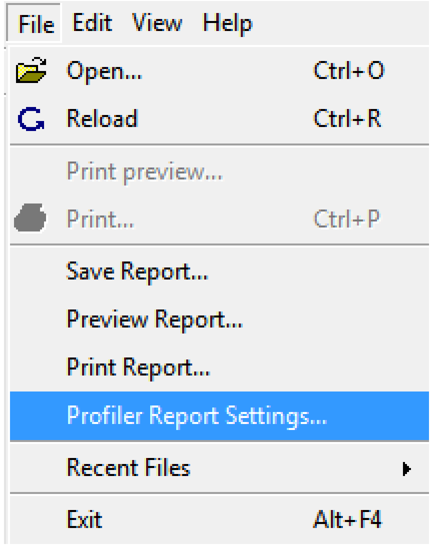
Design time :
49225837
15915.24 us

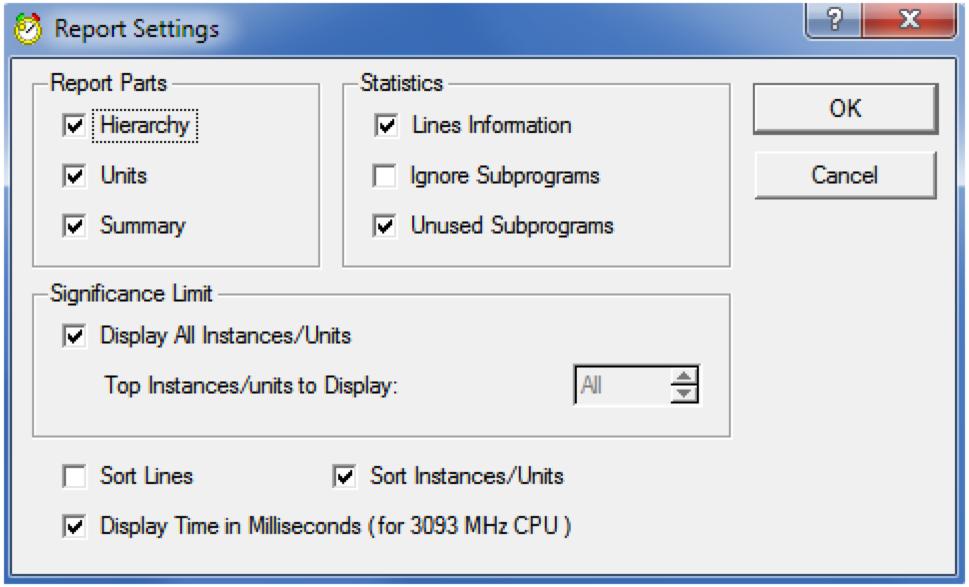
Selected item with children :
40419247 (82.11 %)
13067.98 us

Rest of design :
8806590 (17.89 %)
2847.26 us

Design Profiler Report

- The Design Profiler provides also possibility to generate a custom report file from gathered data.
- Using Profiler Report settings, you can easily choose interesting data which should be put into the report file.




Report Settings

Report Parts
- ☑ Hierarchy
- ☑ Units
- ☑ Summary

Statistics
- ☑ Lines Information
- □ Ignore Subprograms
- ☑ Unused Subprograms

Significance Limit
- ☑ Display All Instances/Units
- Top Instances/units to Display: All

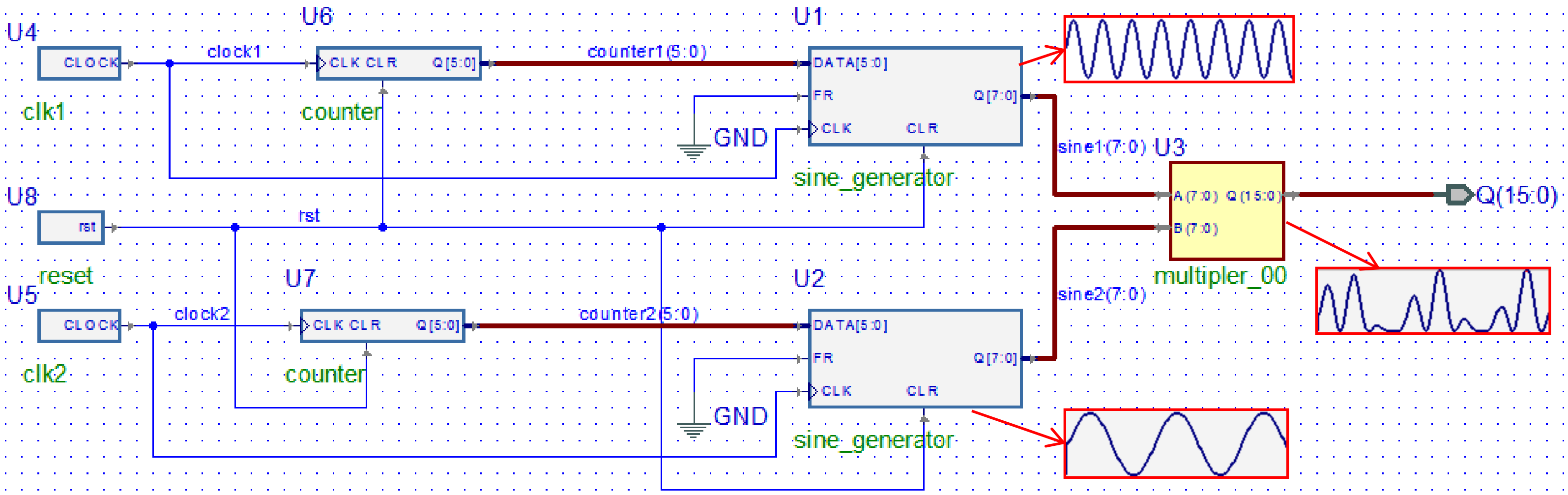
- □ Sort Lines
- ☑ Sort Instances/Units
- ☑ Display Time in Milliseconds (for 3093 MHz CPU)

Design Profiler Example

Design Name: Profiler_example

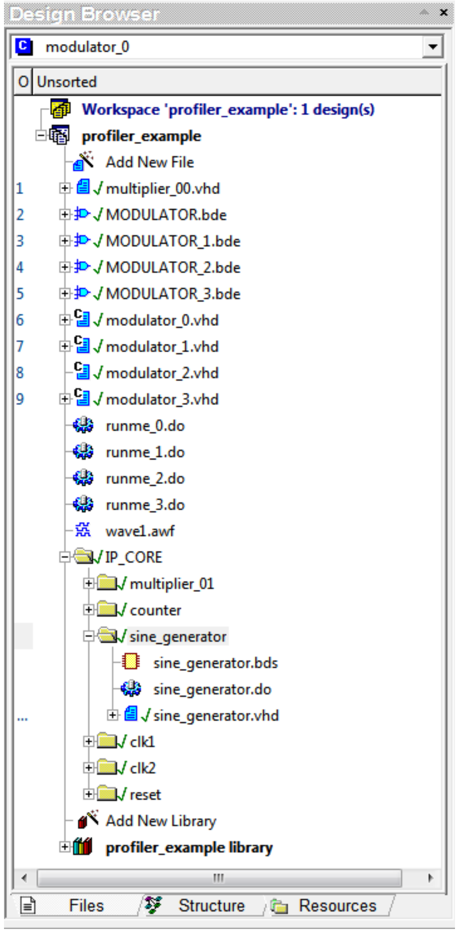
Preview

- Project functionality is based on modulator sample installed with Active-HDL. The top level –modulator contains two sine generators (oscillator + counter + main sine generator) and multiplier module generated by IP Core Generator.



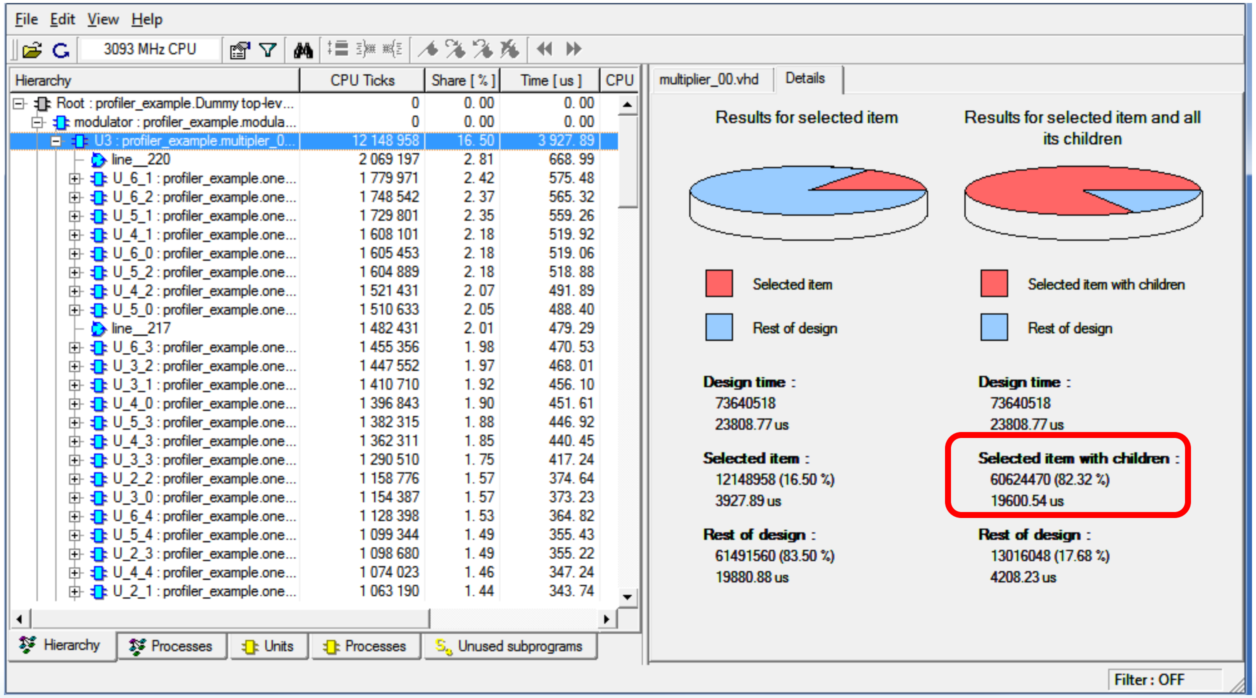
Design Contents

- Design includes four architectures of Modulator entity top-level and four configurations to invoke simulation with proper architecture. The IP_CORE subfolder contains units generated by **IP Core Generator**.
- Additionally, for each example a macro file has been provided. This way, to verify each case you have to invoke the proper macro.



Modulator Architecture

- In this case as a multiplier module a fully synthesizable unit containing muxesand adders has been used. After execution “runme_0.do”macro you can see that this component and its subcomponents takes over 82% of simulation execution time



Hierarchy	CPU Ticks	Share [%]	Time [us]	CPU
Root : profiler_example.Dummy top-lev...	0	0. 00	0. 00	
modulator : profiler_example.modula...	0	0. 00	0. 00	
U3 : profiler_example.multipler_0...	12 148 958	16. 50	3 927. 89	
line__220	2 069 197	2. 81	668. 99	
U_6_1 : profiler_example.one...	1 779 971	2. 42	575. 48	
U_6_2 : profiler_example.one...	1 748 542	2. 37	565. 32	
U_5_1 : profiler_example.one...	1 729 801	2. 35	559. 26	
U_4_1 : profiler_example.one...	1 608 101	2. 18	519. 92	
U_6_0 : profiler_example.one...	1 605 453	2. 18	519. 06	
U_5_2 : profiler_example.one...	1 604 889	2. 18	518. 88	
U_4_2 : profiler_example.one...	1 521 431	2. 07	491. 89	
U_5_0 : profiler_example.one...	1 510 633	2. 05	488. 40	
line__217	1 482 431	2. 01	479. 29	
U_6_3 : profiler_example.one...	1 455 356	1. 98	470. 53	
U_3_2 : profiler_example.one...	1 447 552	1. 97	468. 01	
U_3_1 : profiler_example.one...	1 410 710	1. 92	456. 10	
U_4_0 : profiler_example.one...	1 396 843	1. 90	451. 61	
U_5_3 : profiler_example.one...	1 382 315	1. 88	446. 92	
U_4_3 : profiler_example.one...	1 362 311	1. 85	440. 45	
U_3_3 : profiler_example.one...	1 290 510	1. 75	417. 24	
U_2_2 : profiler_example.one...	1 158 776	1. 57	374. 64	
U_3_0 : profiler_example.one...	1 154 387	1. 57	373. 23	
U_6_4 : profiler_example.one...	1 128 398	1. 53	364. 82	
U_5_4 : profiler_example.one...	1 099 344	1. 49	355. 43	
U_2_3 : profiler_example.one...	1 098 680	1. 49	355. 22	
U_4_4 : profiler_example.one...	1 074 023	1. 46	347. 24	
U_2_1 : profiler_example.one...	1 063 190	1. 44	343. 74	

Results for selected item

🟥 Selected item

🟦 Rest of design

Design time :
73640518
23808.77 us

Selected item :
12148958 (16.50 %)
3927.89 us

Rest of design :
61491560 (83.50 %)
19880.88 us

Results for selected item and all its children

🟥 Selected item with children

🟦 Rest of design

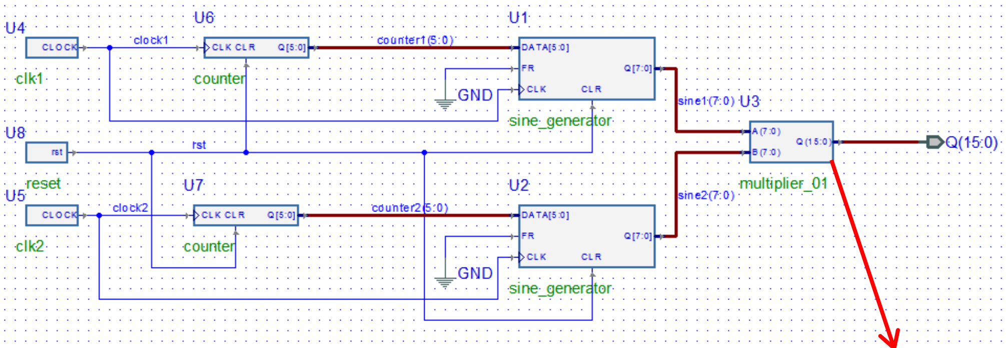
Design time :
73640518
23808.77 us

Selected item with children :
60624470 (82.32 %)
19600.54 us

Rest of design :
13016048 (17.68 %)
4208.23 us

Modulator_1 Architecture

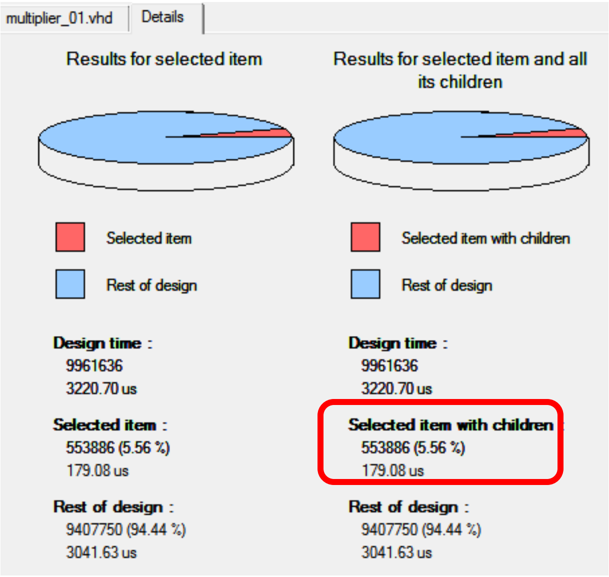
- To provide better performance the multiplier unit has been replaced by multiplier_01. The new unit has the same functionality but its architecture is much simpler: ”Q<=A*B”.
- If you execute the “runme_1.do”macroyou can easily find difference: the CPU usage for multiplier unit has been drastically decreased -about 16 times!



```
architecture multi_arch of multiplier_01 is
begin

    Q <= A * B;

end multi_arch;
```



Results for selected item

Results for selected item and all its children

🟥 Selected item

🟦 Rest of design

🟥 Selected item with children

🟦 Rest of design

Design time :
9961636
3220.70 us

Selected item :
553886 (5.56 %)
179.08 us

Rest of design :
9407750 (94.44 %)
3041.63 us

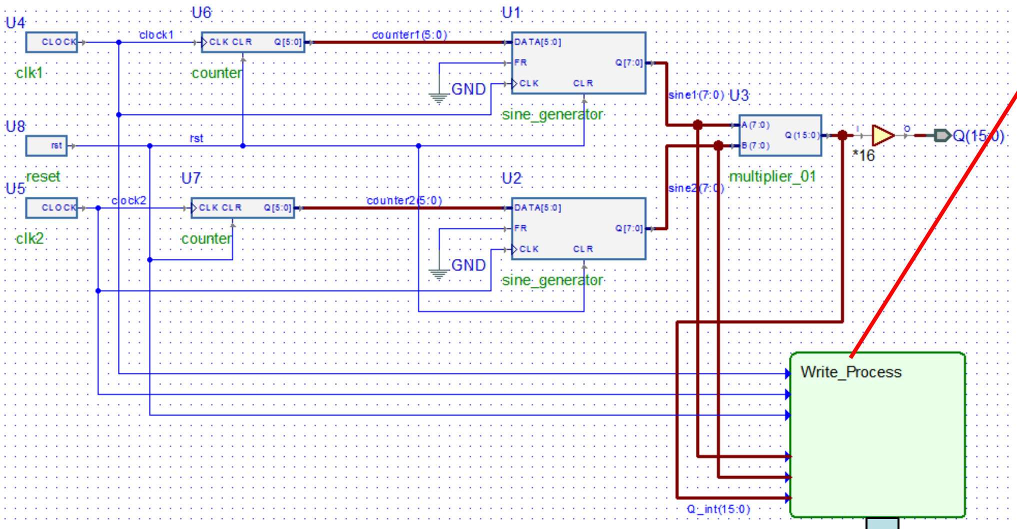
Design time :
9961636
3220.70 us

Selected item with children
553886 (5.56 %)
179.08 us

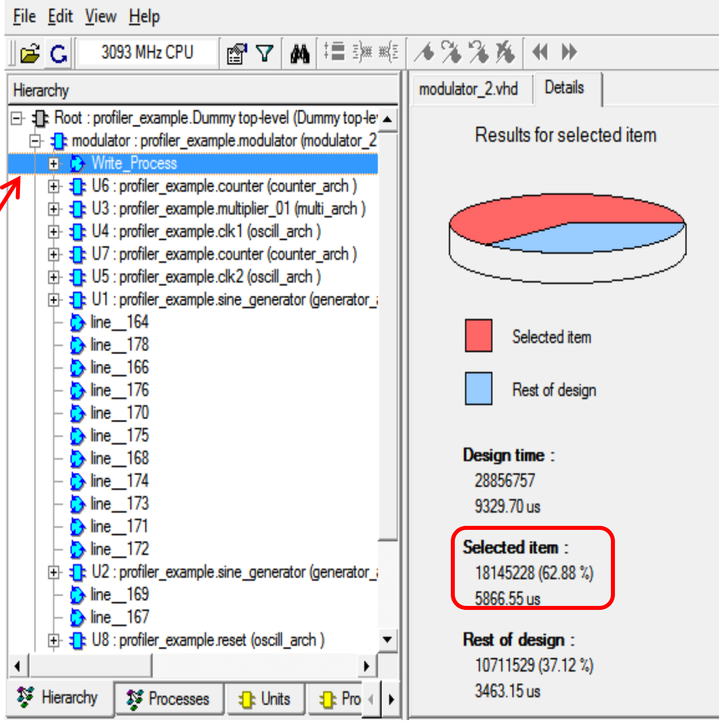
Rest of design :
9407750 (94.44 %)
3041.63 us

Modulator_2 Architecture

- The “Modulator_2”architecture includes additional process. This process writes all simulation results to text file. After execution the “runme_2.do”macro you can find that new process takes 62% of entire CPU time.



```
Write_Process :
process (clock1, clock2, sine1, sine2, Q_int, rst)
file results: TEXT open WRITE_MODE is "results.txt";
variable L_OUT : LINE;        -- pointer to string
begin
```



Root : profiler_example.Dummy top-level (Dummy top-le
modulator : profiler_example.modulator (modulator_2
Write_Process
U6 : profiler_example.counter (counter_arch)
U3 : profiler_example.multiplier_01 (multi_arch)
U4 : profiler_example.clk1 (oscill_arch)
U7 : profiler_example.counter (counter_arch)
U5 : profiler_example.clk2 (oscill_arch)
U1 : profiler_example.sine_generator (generator_
line__164
line__178
line__166
line__176
line__170
line__175
line__168
line__174
line__173
line__171
line__172
U2 : profiler_example.sine_generator (generator_
line__169
line__167
U8 : profiler_example.reset (oscill_arch)

Results for selected item

🟥 Selected item

🟦 Rest of design

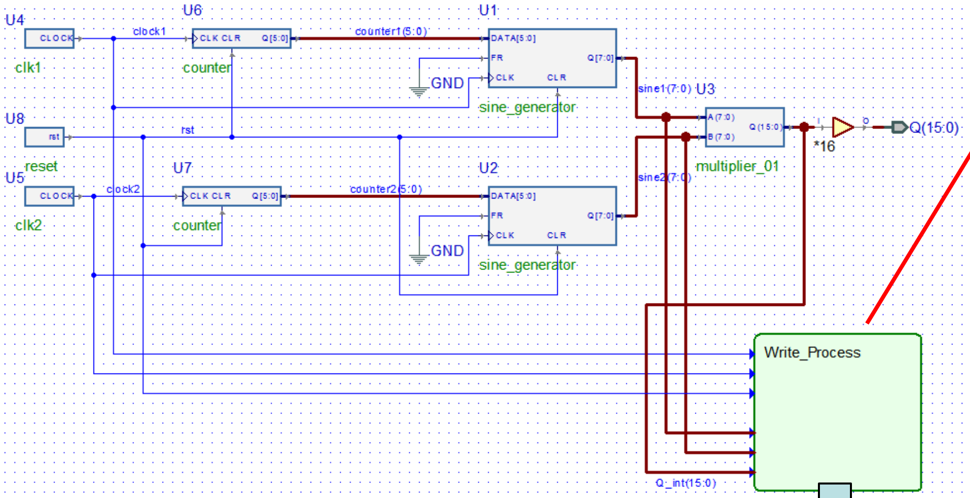
Design time :
28856757
9329.70 us

Selected item :
18145228 (62.88 %)
5866.55 us

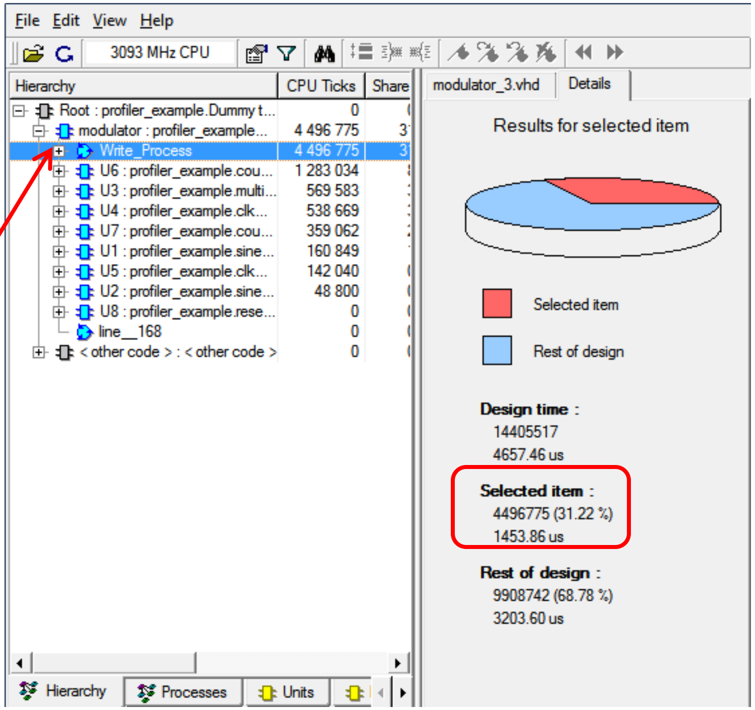
Rest of design :
10711529 (37.12 %)
3463.15 us

Modulator_3 Architecture

- The Write_Process has been optimized on “modulator_3” architecture. The sensitivity list has been reduced to the only one signal: “Q_int”. After execution “runme_3.do”you can verify the optimization effect.



```
Write_Process :
process (Q_int)
file results: TEXT open WRITE_MODE is "results.txt";
variable L_OUT : LINE;        -- pointer to string
begin
```



Hierarchy	CPU Ticks	Share
Root : profiler_example.Dummy t...	0	
modulator : profiler_example...	4 496 775	3
Write_Process	4 496 775	3
U6 : profiler_example.cou...	1 283 034	
U3 : profiler_example.multi...	569 583	
U4 : profiler_example.clk...	538 669	
U7 : profiler_example.cou...	359 062	
U1 : profiler_example.sine...	160 849	
U5 : profiler_example.clk...	142 040	
U2 : profiler_example.sine...	48 800	
U8 : profiler_example.rese...	0	
line__168	0	
< other code > : < other code >	0	

Results for selected item

🟥 Selected item

🟦 Rest of design

Design time :
14405517
4657.46 us

Selected item :
4496775 (31.22 %)
1453.86 us

Rest of design :
9908742 (68.78 %)
3203.60 us