

ALINT 2012.01

What's New Presentation

The ALDEC logo is positioned in the bottom right corner. It features the word "ALDEC" in a bold, blue, sans-serif font. The text is superimposed on a blue, semi-transparent sphere that has a gradient and a shadow, giving it a 3D effect. The sphere is partially obscured by the text. In the background, there are faint, light blue binary digits (0s and 1s) arranged in a grid-like pattern, suggesting a digital or data environment.

ALDEC

Agenda

- **Exclusion Management** *Justification Comments, AVDB Integration*
- Standalone Reports *Interactivity, Cross-probing*
- Windows 64 *Native Support*
- Productivity *Code Templates, Preferences, Console, Macro*

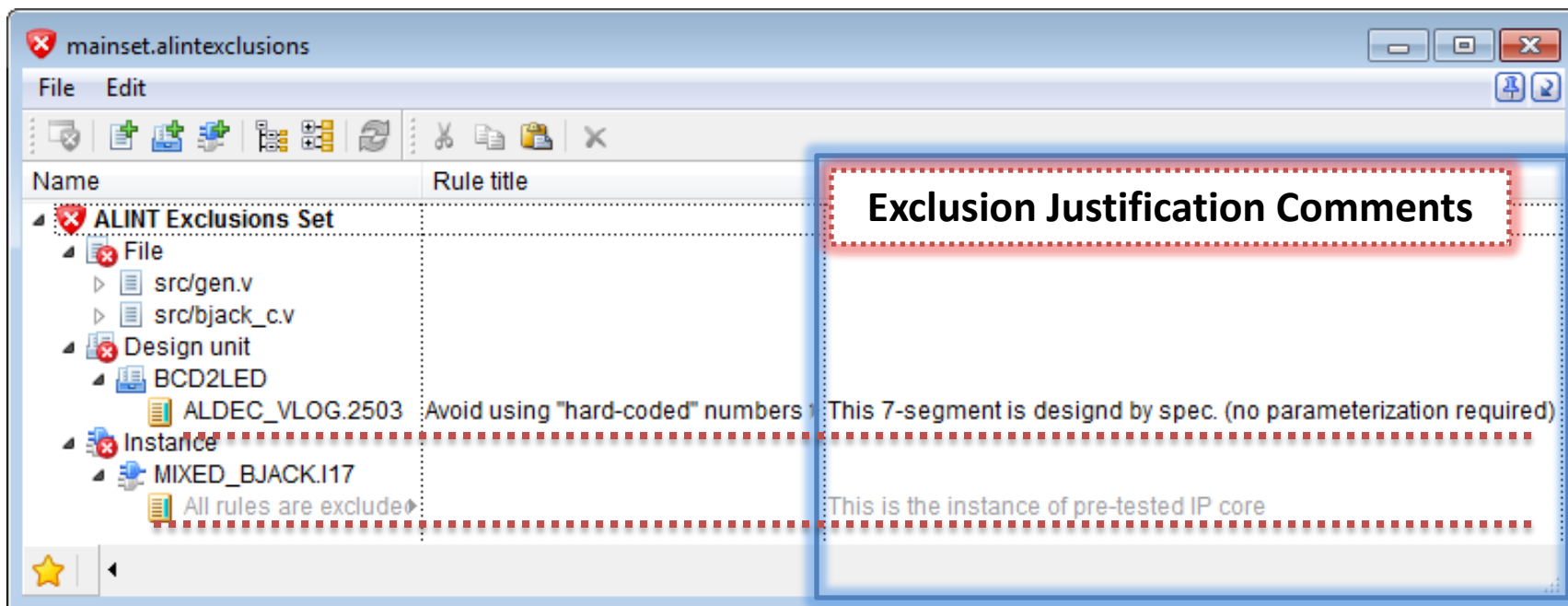
Exclusion Management Overview

- Exclusion management mechanism is redesigned to enable:
 - ◆ Justifying a reason for every rule violation marked as irrelevant
 - ◆ Viewing exclusions alongside with the justifications as a part of AVDB
 - ◆ Transitioning from the Batch to GUI-based flow seamlessly



Exclusion Management (Cont...)

- Every rule exclusion can be justified now:
 - ◆ A comment can be associated with every exclusion or group of exclusions



Example: ALDEC_VLOG.2503 is disabled/excluded for BCD2LED design unit since it was a requirement design a 7-segment LED (does not need to be parameterized).

Exclusion Management (Cont...)

- Irrelevant rule violations are stored in the violation database:
 - ◆ Excluded rule violations can be hidden or displayed in the Violation Viewer
 - ◆ The justification comments can be shown alongside with the exclusions

The screenshot shows the Violation Viewer interface. The top part is a tree view of design units. A red dashed box labeled '1' highlights the 'bcd2led.v' unit, which is grayed out. A red dashed box labeled '2' highlights the 'Unit: This 7-segment is designed by spec. (no parameterization required)' comment in the 'Comments' column of the table below.

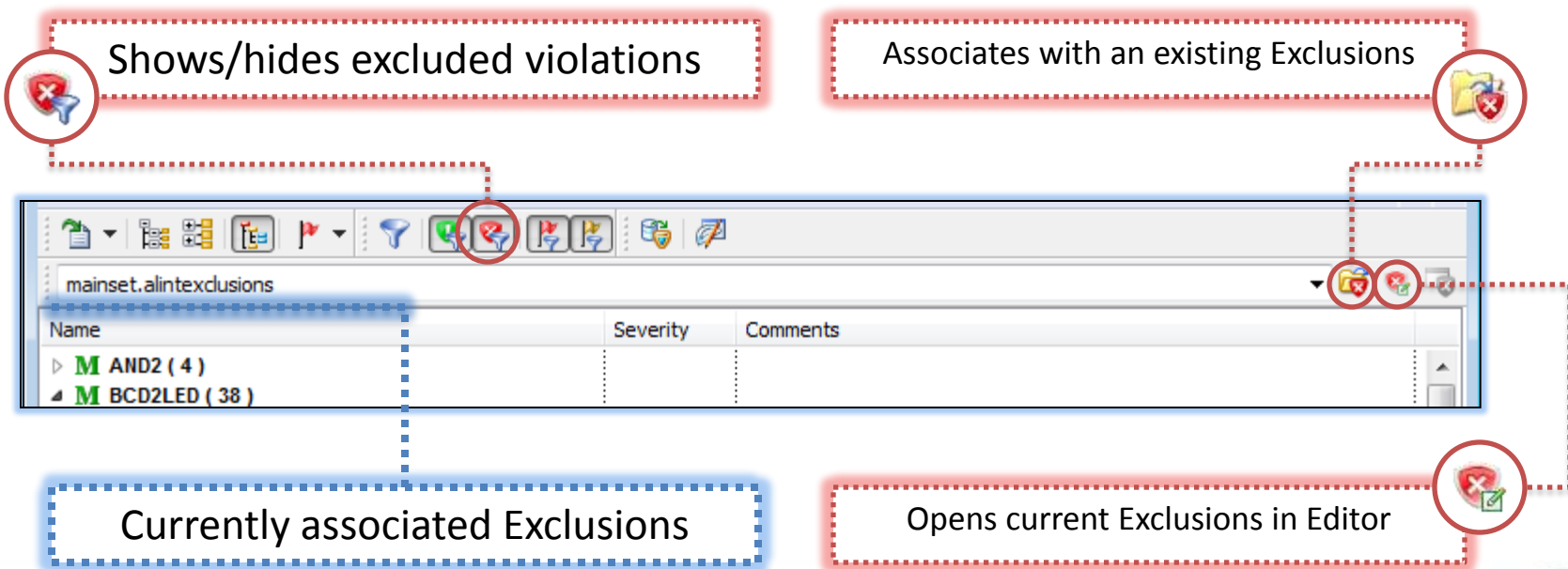
Name	Severity	Comments
▶ M AND2 (4)		
▶ M BCD2LED (38)		
▶ <all instances> (11)		
▶ Recommendation1		
▶ ALDEC_VLOG.2503		
bcd2led.v	Warning	Unit: This 7-segment is designed by spec. (no parameterization required)
▶ D0254_VLOG.2112 (1)		
▶ D0254_VLOG.2113 (2)		
▶ RMM.VLOG.5.2.1.4 (1)		
▶ Recommendation2 (6)		
▶ Recommendation3 (1)		

Line	Instance	Design U...	Severity	Description
10	<all instances>	BCD2LED	Warning	For better readability and portability it is recommended to use parameters instead of putting "hard-coded" values in the code.
38	<all instances>	BCD2LED	Warning	The "4'b0010" value should be replaced with parameter.
38	<all instances>	BCD2LED	Warning	The "7'b0000110" value should be replaced with parameter.
38	<all instances>	BCD2LED	Warning	The "7'b0001111" value should be replaced with parameter.
38	<all instances>	BCD2LED	Warning	The "7'b0010010" value should be replaced with parameter.

Summary Sources Design Units Instances Rulesets Rules Rule Levels Severity


Exclusion Management (Cont...)

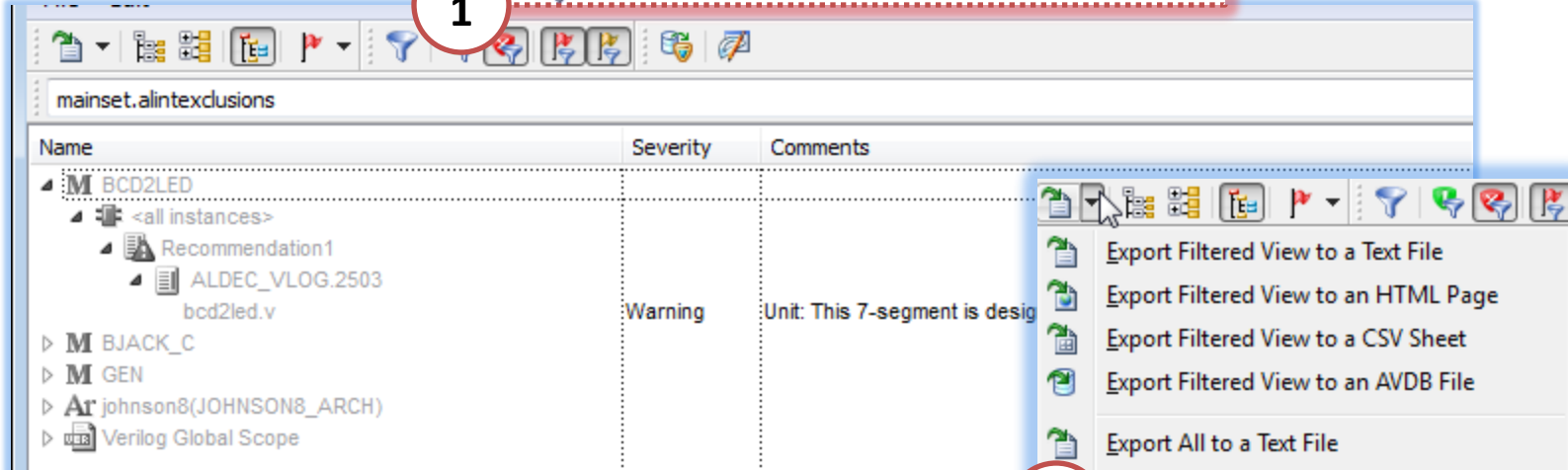
- Exclusions set and violation database are associated now:
 - ◆ Working directory based use flow is seamless
(In 2010.10 SR1 and earlier, exclusions had to be entered manually when no design was present)
 - ◆ One database can be associated with multiple exclusion sets
(Enables workspace-based flows that record multiple designs analysis results into a single database)



Exclusion Management (Cont...)

- Rule exclusions can be documented efficiently now:
 - ◆ Push-button interface for exporting to a spreadsheet or plain text

1  Display excluded violations only



Name	Severity	Comments
M BCD2LED <ul style="list-style-type: none"><all instances><ul style="list-style-type: none">Recommendation1<ul style="list-style-type: none">ALDEC_VLOG.2503<ul style="list-style-type: none">bcd2led.v	Warning	Unit: This 7-segment is design
M BJACK_C		
M GEN		
Ar johnson8(JOHNSON8_ARCH)		
VER Verilog Global Scope		

2

Export to HTML/CSV

- Export Filtered View to a Text File
- Export Filtered View to an HTML Page
- Export Filtered View to a CSV Sheet
- Export Filtered View to an AVDB File
- Export All to a Text File
- Export All to an HTML Page
- Export All to a CSV Sheet

Agenda

- Exclusion Management *Justification Comments, AVDB Integration*
- **Standalone Reports** *Interactivity, Cross-probing*
- Windows 64 *Native Support*
- Productivity *Code Templates, Preferences, Console, Macro*

Standalone Report Overview



- The all-new reporting mechanism enables:
 - ◆ Navigating design hierarchy by source files, design units, and instances
 - ◆ Analyzing violations associated with any file, unit, or instance
 - ◆ Cross-probing from violations to the source code
 - ◆ Viewing the reports in any common web browser (no license required)

Standalone Report (Cont...)

- Report is organized in two major panes:
 - ◆ **Hierarchy** pane with several tabs/viewpoints
 - ◆ **Violations** pane with linting session results

The screenshot displays the Aldec IDE interface. On the left, the 'Design Hierarchy' pane is active, showing a tree view of design units. On the right, the 'Violations' pane is active, displaying the source code for 'bcd2led.v' and a table of linting violations.

Design Hierarchy

- Sources
- Design Units
- Instances
- M AND2 (4)
- M BCD2LED (39)
- E bin2bcd (3)
- Ar bin2bcd(BIN2BCD_ARCH) (7)
- M BJACK_C (99)
- M GEN (20)
- M INV (8)
- E johnson8 (3)
- Ar johnson8(JOHNSON8_ARCH) (13)
- M MIXED_BJACK (51)
- M MUX (11)
- M OR2 (6)

Violations (41)

Name	Lir	Design	Instan	Description/Title
BCD2LEI11				Internally generated clock is detected.
BCD2LEI11				The global clock "MIXED_BJACK.GEN_C

Standalone Report (Cont...)

- The **Hierarchy** pane provides three alternative viewpoints:
 - ◆ **Sources** – violations grouped by source files
 - ◆ **Design Units** – violations grouped by design units
 - ◆ **Instances** – violations grouped by instances

Sources Design Units Instances

- VER bcd2led.v (41)
- VHD bin2bcd.vhd (11)
- VER bjack_c.v (100)
- VER blackjack.v (54)
- VER gates.v (21)
- VER gen.v (22)
- VHD johnson8.vhd (17)
- VER mux.v (12)

1. Sources

Sources Design Units Instances

- M AND2 (4)
- M BCD2LED (39)
- E bin2bcd (3)
- Ar bin2bcd(BIN2BCD_ARCH) (7)
- M BJACK_C (99)
- M GEN (20)
- M INV (8)
- E johnson8 (3)
- Ar johnson8(JOHNSON8_ARCH) (13)

2. Design Units

Sources Design Units Instances

- MIXED_BJACK (121)
- I1 (1)
- I10 (1)
- I11 (27)
- I12 (1)
- I13 (2)
- I14 (1)
- I17 (8)
- I5 (1)

3. Instances

Standalone Report (Cont...)

- Clicking any **Hierarchy** node displays violations associated with it:

The screenshot illustrates the workflow for viewing violations in the Aldec IDE:

- 1 Hierarchy Node:** A red dashed box highlights a node in the Design Hierarchy pane (e.g., `johnson8.vhd`).
- 2 Associated Violations:** A red dashed box highlights the violations pane, which displays a table of linting results. The table has columns for "Description/Title" and "Corr".
- 3 Cross-probing:** A red dashed box highlights the source code editor, which displays the code corresponding to the selected violation. A red arrow points from the violation table to the code editor.

The source code shown includes:

```

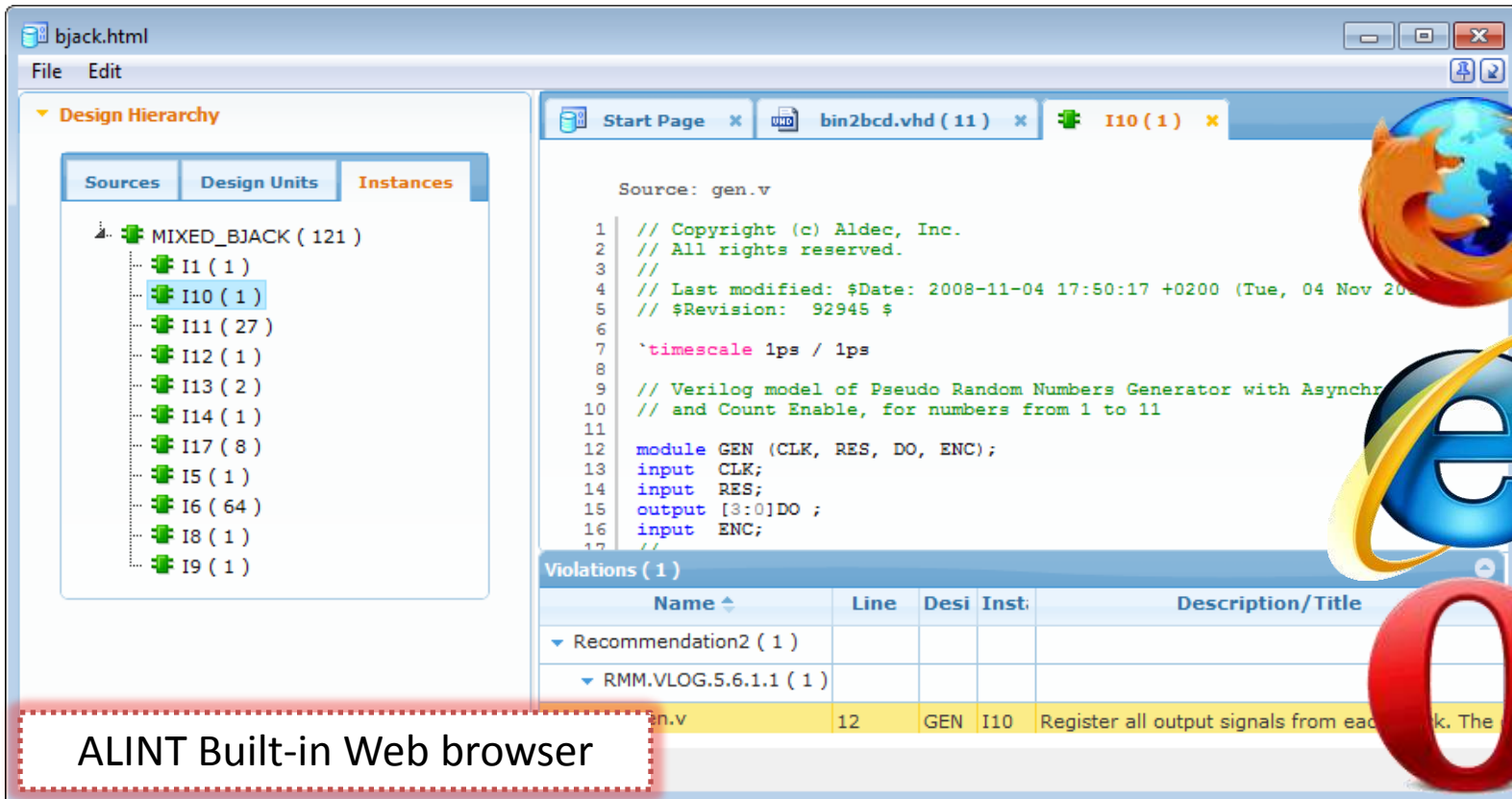
33 wire [4:0]HAND;
34 wire [4:0]VIS;
35
36 AND2 I1 (.IO(BUST), .I1(GEN_CLK), .O(OFF));
37 INV I13 (.I(START), .O(NEW_CARD));
38 bin2bcd I14 (.BIN(VIS));
39 INV I12 (.I(NEW_CARD));
40 INV I5 (.I(NEXT_CARD));
41 BJACK_C I6 (.CARD(DO), .NEW_C(NOT));
42 johnson8 I17 (.Q(LED_S), .CLK(CLK), .RESET(NEW_CARD));
43 OR2 I8 (.IO(HOLD), .I1(BUST), .O(S));
44 MUX I9 (.A(DO), .B(HAND), .Y(VIS), .S(S));
45 GEN I10 (.DO(DO), .CLK(GEN_CLK), .RES(GEN_RES), .ENC(NEW_CARD));
46 BCD2LED I11 (.DIGIT_H(D_H), .DIGIT_L(D_L), .LED_H(L_H), .LED_L(L_L), .OFF);
47
48 endmodule

```

- The **Violations** pane displays the results of linting session:
 - Violations associated with the currently selected Hierarchy node
 - Source code corresponding to the currently selected violation

Standalone Report (Cont...)

- The reports can be viewed in any common web browser:



Design Hierarchy

Sources Design Units Instances

MIXED_BJACK (121)

- I1 (1)
- I10 (1)
- I11 (27)
- I12 (1)
- I13 (2)
- I14 (1)
- I17 (8)
- I5 (1)
- I6 (64)
- I8 (1)
- I9 (1)

Source: gen.v

```

1 // Copyright (c) Aldec, Inc.
2 // All rights reserved.
3 //
4 // Last modified: $Date: 2008-11-04 17:50:17 +0200 (Tue, 04 Nov 2008)
5 // $Revision: 92945 $
6
7 `timescale 1ps / 1ps
8
9 // Verilog model of Pseudo Random Numbers Generator with Asynchronous
10 // and Count Enable, for numbers from 1 to 11
11
12 module GEN (CLK, RES, DO, ENC);
13 input CLK;
14 input RES;
15 output [3:0]DO ;
16 input ENC;
17 //

```

Violations (1)

Name	Line	Desi	Inst.	Description/Title
Recommendation2 (1)				
RMM.VLOG.5.6.1.1 (1)				
en.v	12	GEN	I10	Register all output signals from each block. The

ALINT Built-in Web browser

Agenda

- Exclusion Management *Justification Comments, AVDB Integration*
- Standalone Reports *Interactivity, Cross-probing*
- **Windows 64** *Native Support*
- Productivity *Code Templates, Preferences, Console, Macro*

Windows 64 Native Support

- 64-bit mode linting is now available on Windows® platform!
 - ◆ **Ability to analyze really large designs – no memory limitations imposed by 32-bit architecture**
 - ◆ This version was verified on the following operating systems:
 - Microsoft Windows® XP
 - Microsoft Windows Vista™
 - Microsoft Windows 7™
 - Microsoft Windows Server® 2008
 - ◆ Different installers are provided for 32- and 64-bit platforms
 - ALINT-<version>.exe (→ C:\Aldec\ALINT-<version>)
 - ALINT-<version>-x64.exe (→ C:\Aldec\ALINT-<version>-x64)

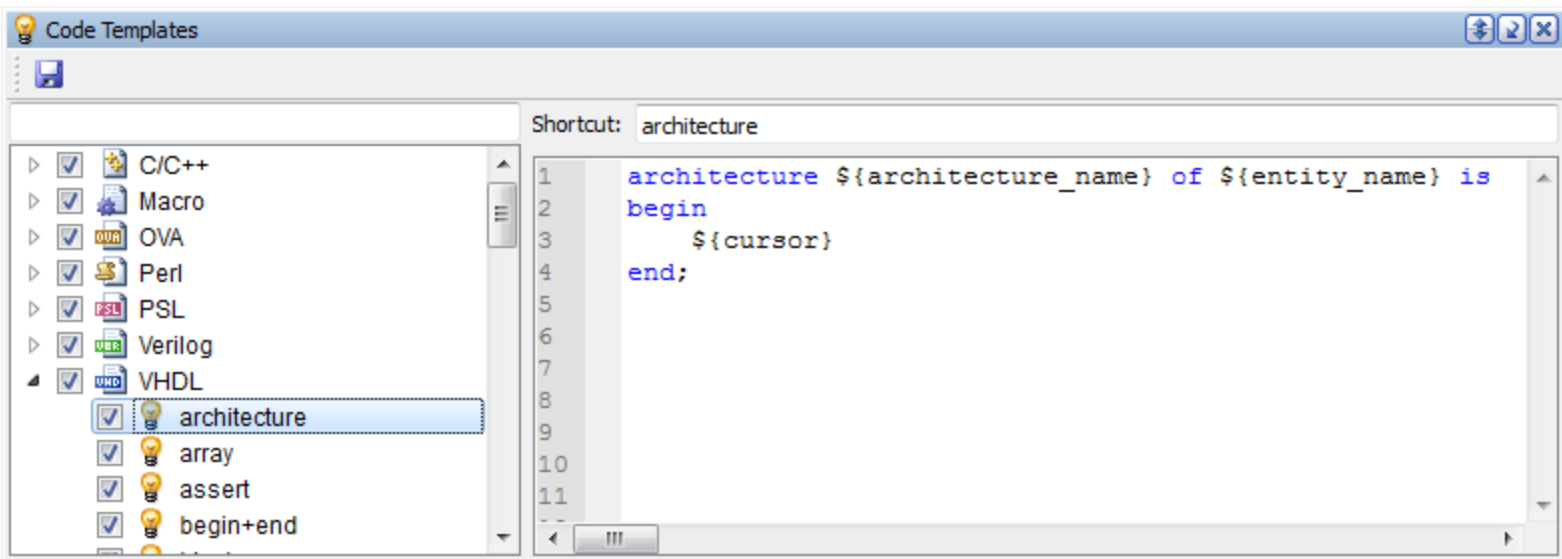


Agenda


- Exclusion Management *Justification Comments, AVDB Integration*
- Standalone Reports *Interactivity, Cross-probing*
- Windows 64 *Native Support*
- **Productivity** *Code Templates, Preferences, Console, Macro*

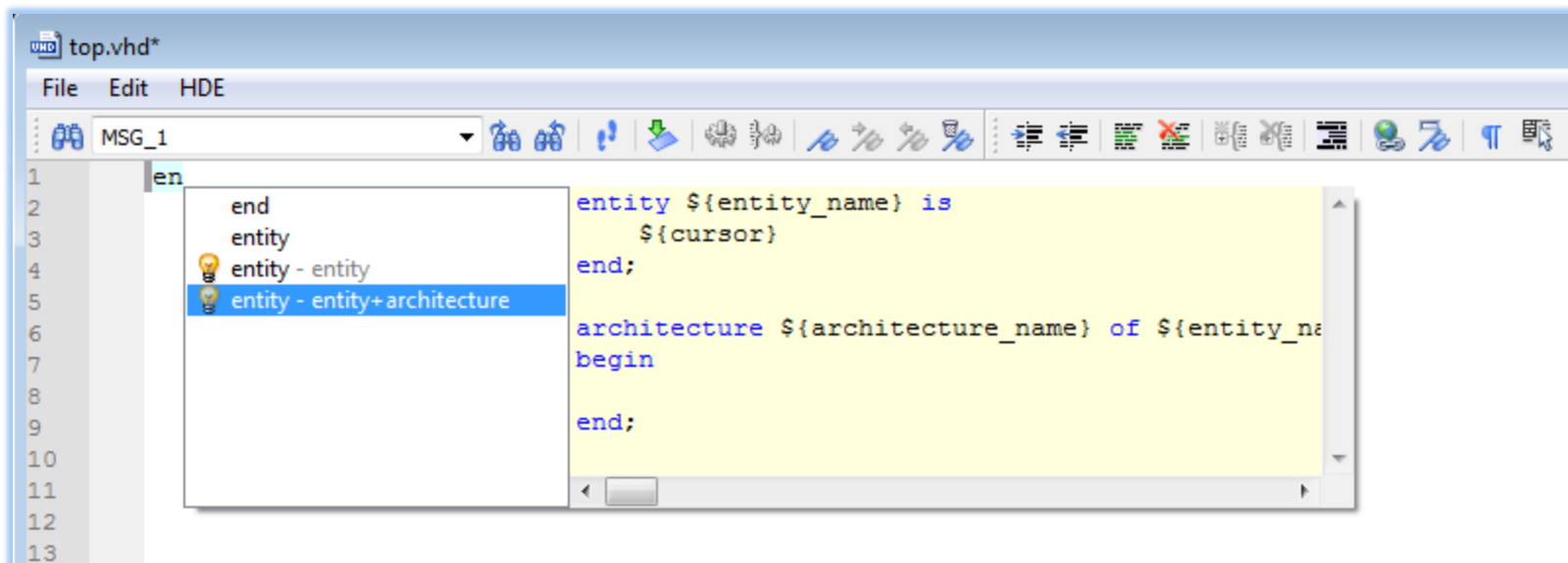
Code Templates in HDL Editor

- **Code Templates** is a new tool that facilitates code editing:
 - ◆ A variety of predefined templates for VHDL, Verilog, SystemC, Perl, Macro
 - ◆ The templates can be freely customized and extended
 - ◆ Integrated with the auto-complete feature of the HDL Editor
 - ◆ Also available as a separate window (view and edit, drag-n-drop)



Code Templates in HDL Editor (Cont.)

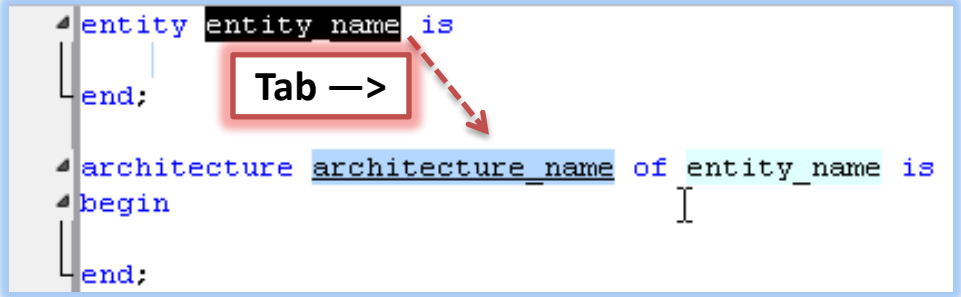
- Code Templates are integrated with auto-complete in **HDL Editor**:
 - ◆ Matching keywords and Code Templates pop up as first letters are typed
 - ◆ The code templates are marked with the  icon
 - ◆ Desired code template can be selected by pressing the **Enter** key



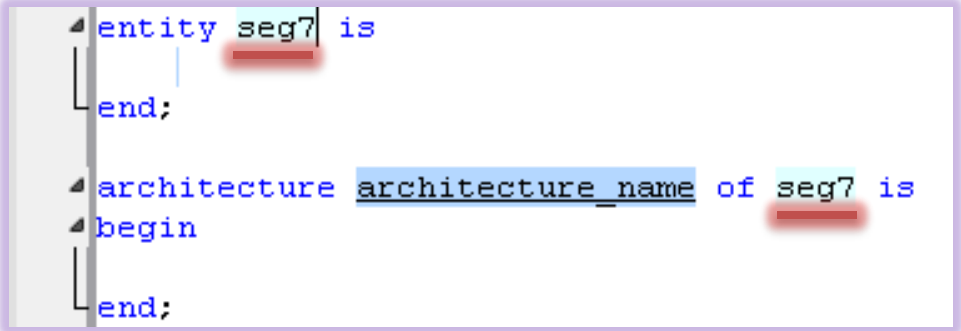
Code Templates in HDL Editor (Cont.)

- Variable-based templates enable quick change of special fields that occur multiple times within a code template:
 - ◆ **Tab** and **Shift+Tab** keys allow navigating between **special fields**
 - ◆ While editing the first occurrence of a special field, the other occurrences are updated automatically

```
entity entity_name is
end;
architecture architecture_name of entity_name is
begin
end;
```

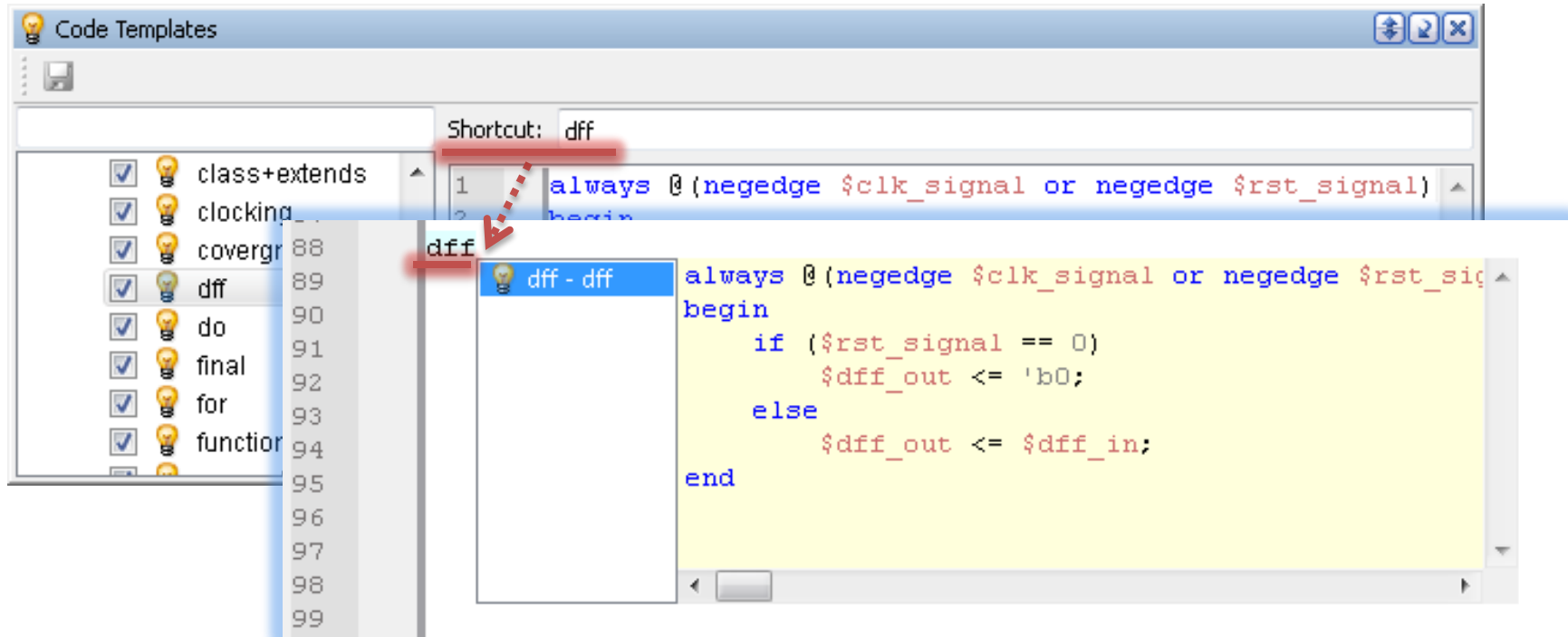


```
entity seg7 is
end;
architecture architecture_name of seg7 is
begin
end;
```



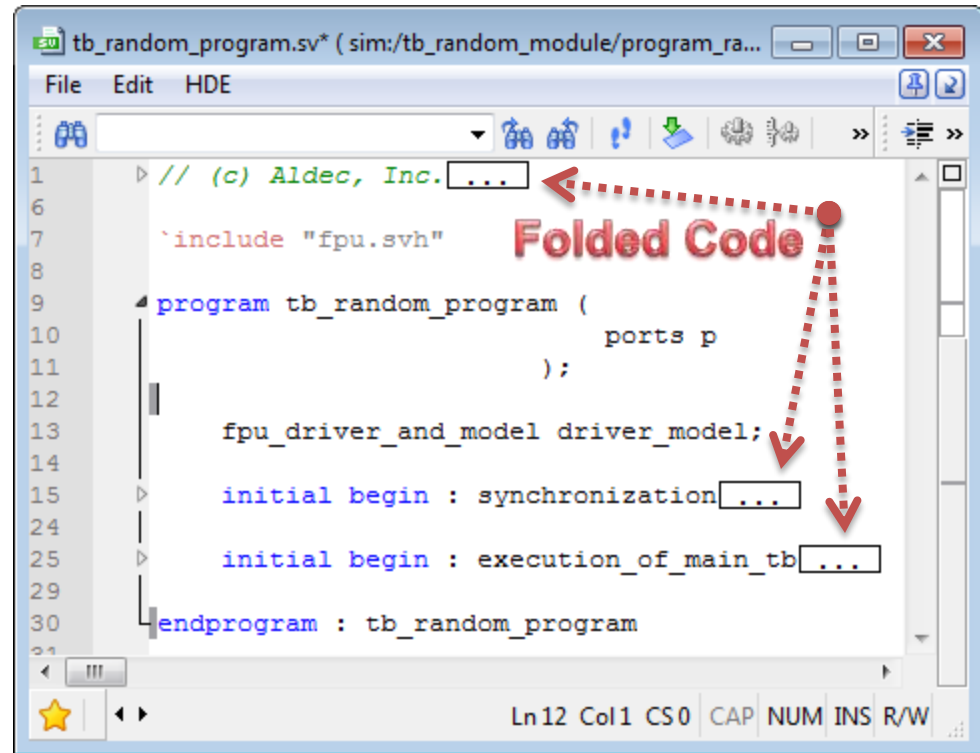
Code Templates in HDL Editor (Cont.)

- Code Templates enable defining and using custom templates:
 - ◆ **Shortcut** defines the auto-complete mnemonic
 - ◆ `#{special_field}` allows defining variables (special fields)



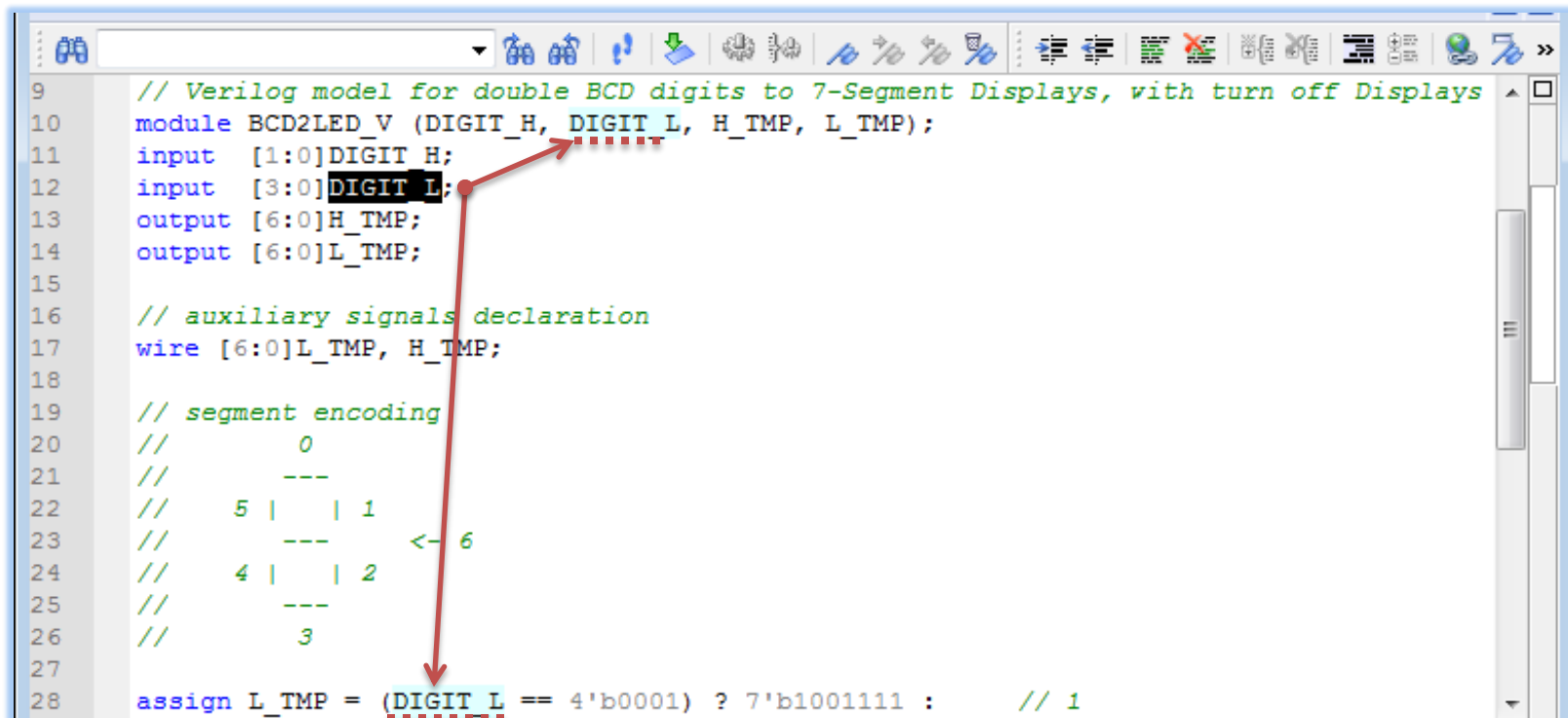
Code Folding in HDL Editor

- Code folding capabilities of the HDL Editor have been enhanced –
 - ◆ HDL code is *automatically* analyzed at the moment of opening a file
 - ◆ New code is analyzed on the fly, when being typed
 - ◆ Blocks of comments are recognized and treated as separate sections



Highlighting in HDL Editor

- A new option is available in the **Preferences**:
 - ◆ Highlights all other occurrences of currently selected fragment



```
9 // Verilog model for double BCD digits to 7-Segment Displays, with turn off Displays
10 module BCD2LED_V (DIGIT_H, DIGIT_L, H_TMP, L_TMP);
11 input [1:0]DIGIT_H;
12 input [3:0]DIGIT_L;
13 output [6:0]H_TMP;
14 output [6:0]L_TMP;
15
16 // auxiliary signals declaration
17 wire [6:0]L_TMP, H_TMP;
18
19 // segment encoding
20 // 0
21 // ---
22 // 5 | | 1
23 // --- <- 6
24 // 4 | | 2
25 // ---
26 // 3
27
28 assign L_TMP = (DIGIT_L == 4'b0001) ? 7'b10011111 : // 1
```

Cursor Position in HDL Editor

- The cursor position is now remembered when you jump to a source point during the violations analysis:
 - ◆ The **HDL Editor** saves the history of the opened source files
 - ◆ **Ctrl + [** and **Ctrl +]** shortcuts enable browsing by recent cursor locations

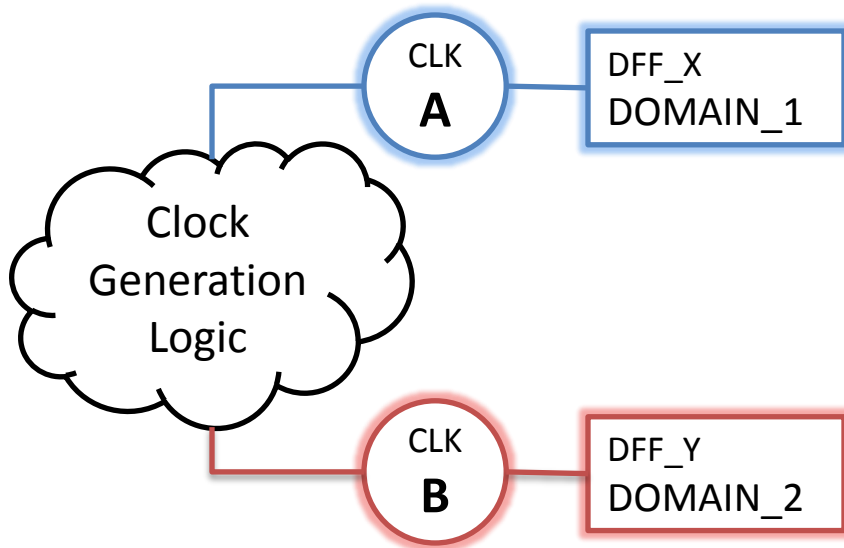
Instance	Design Unit	Description
8051_exp_tb.UUT	A8051_exp(A8051_exp)	The signal "PORT3(0)" has multiple drivers. Review the description for PORT3(0). Another driver is detected.
8051_exp_tb.UUT	A8051_exp(A8051_exp)	

```

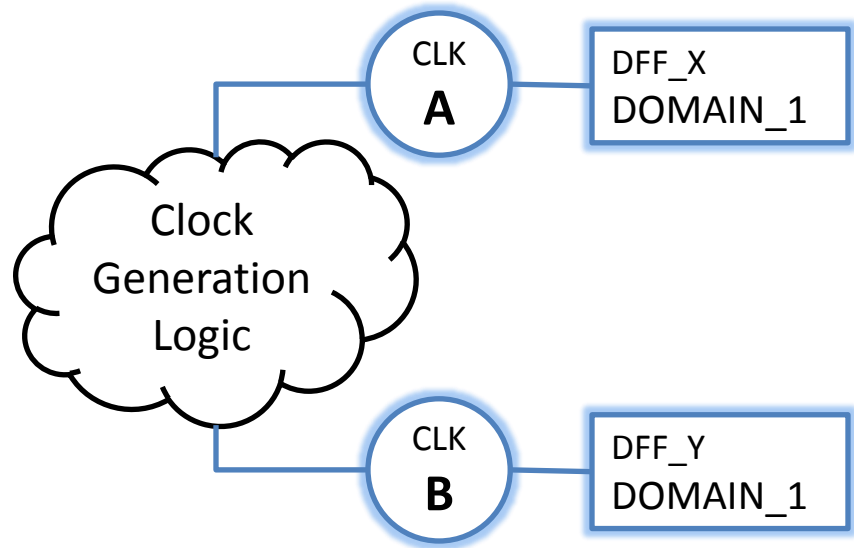
A8051_exp.vhd
File Edit HDE
PORT2
139 User Signal Assignments ----
140 PORT3_OUT(0);
141 PORT3_OUT(2 downto 1)
142 PORT3_IN(4 downto 0) <= PORT3(7 downto 4) &
143
144 Component instantiations ----
145
146
147
148 ALE => ALE,
149 CLK => CLK,
Ln140 Col1 CS0 CAP NUM INS R
  
```

Joining Clock Signals in Properties

- The **Join clock signals** option has been enabled in **Preferences**:
 - ◆ Allows specifying clock signals that belong to the same clock domain
 - ◆ FFs controlled by joined clocks are treated as members of same domain



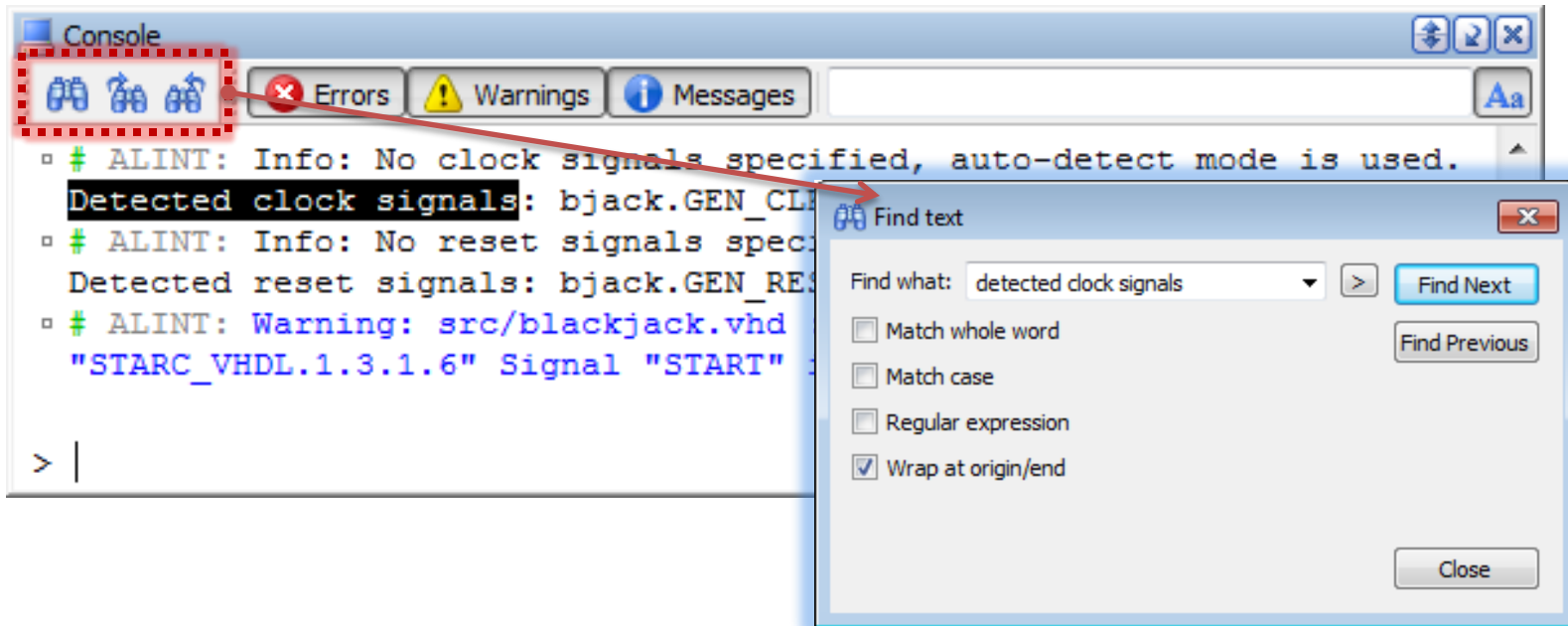
CLKA, CLKB – Regular Clocks



CLKA, CLKB – Joined Clocks

Console Find Dialog

- The **Find text** dialog box was introduced:
 - ◆ Allows searching the **Console** window for a specified text string
 - ◆ The search pattern can be specified as a regular expression



Macro Command Highlights

- The new **avdbreport** command is available:
 - ◆ Generates a detailed and well-structured text report based on .avdb
 - ◆ The report contains the header, optional summary, and violation sections

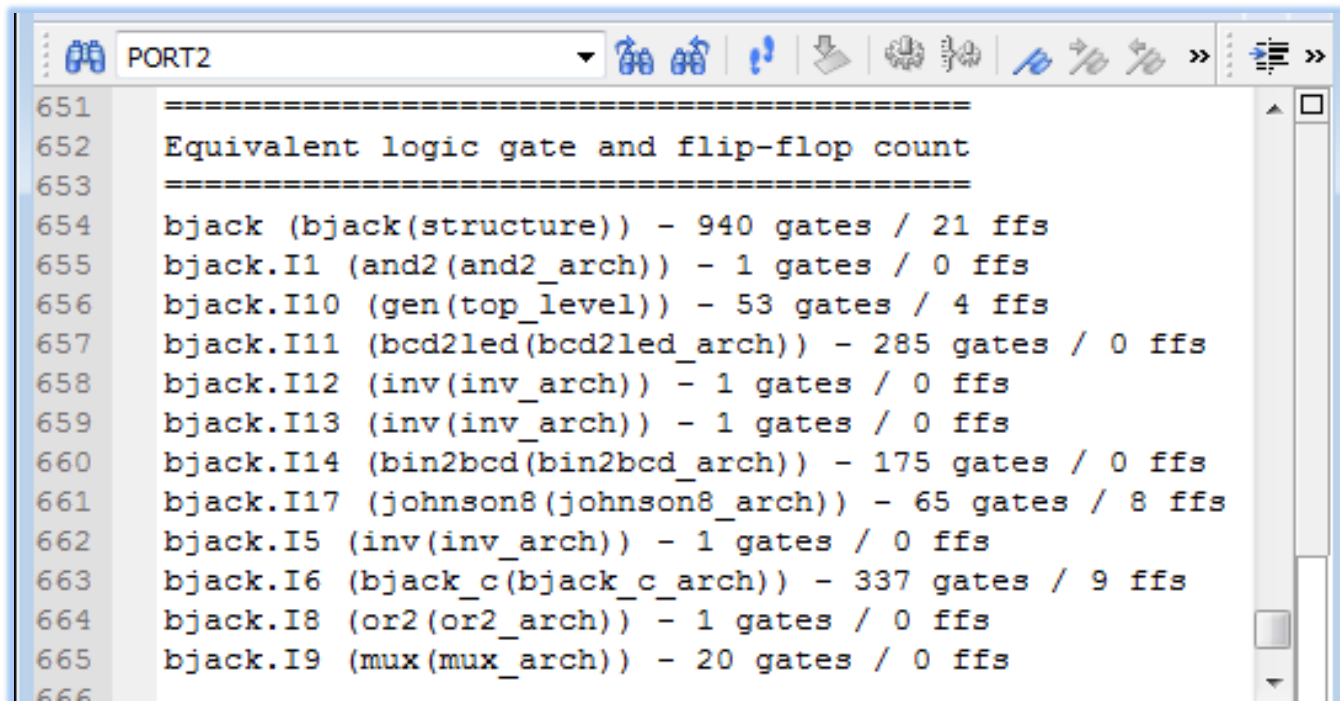
```

17 =====
18 ALDEC_VHDL.2051 - Name objects clearly to enforce
19 =====
20 Help: | "c:/Aldec/ALINT_2012.01.RTM/do
21 =====
22 |
23 Rule: | ALDEC_VHDL.2051
24 Violation: | Architecture "bjack_c_arch" co
25 File: | "c:/Aldec/ALINT_2012.01.RTM/ex
26 Line number: | 25
27 Design unit: | Unit "bjack_c(bjack_c_arch)"
28 Violation type: | Relevant
29 Severity: | Warning
30 Rule level: | Recommendation 2
31 -----
32 Details:

```

Macro Command Highlights (Cont.)

- The **alint_synthesislog** was extended with the information about the inferred flip-flops count:
 - ◆ Enables the early estimation of the design size



```
PORT2
=====
Equivalent logic gate and flip-flop count
=====
651
652
653
654 bjack (bjack(structure)) - 940 gates / 21 ffs
655 bjack.I1 (and2(and2_arch)) - 1 gates / 0 ffs
656 bjack.I10 (gen(top_level)) - 53 gates / 4 ffs
657 bjack.I11 (bcd2led(bcd2led_arch)) - 285 gates / 0 ffs
658 bjack.I12 (inv(inv_arch)) - 1 gates / 0 ffs
659 bjack.I13 (inv(inv_arch)) - 1 gates / 0 ffs
660 bjack.I14 (bin2bcd(bin2bcd_arch)) - 175 gates / 0 ffs
661 bjack.I17 (johnson8(johnson8_arch)) - 65 gates / 8 ffs
662 bjack.I5 (inv(inv_arch)) - 1 gates / 0 ffs
663 bjack.I6 (bjack_c(bjack_c_arch)) - 337 gates / 9 ffs
664 bjack.I8 (or2(or2_arch)) - 1 gates / 0 ffs
665 bjack.I9 (mux(mux_arch)) - 20 gates / 0 ffs
666
```

Conclusions – Top Reasons to Upgrade

- Exclusions management mechanism
- Advanced standalone reports
- Windows 64-bit native support
- Numerous productivity features