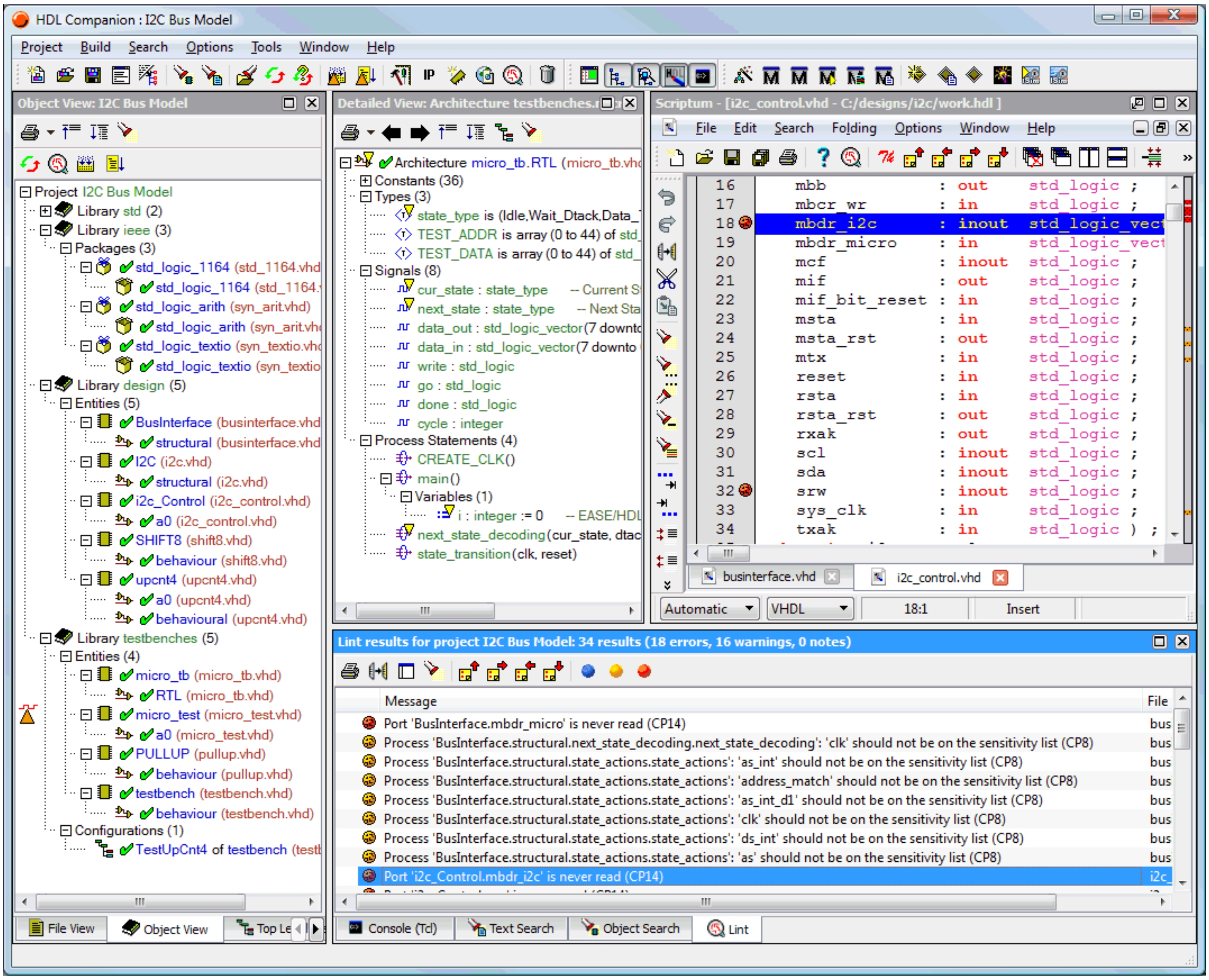
HDL Companion

The Swiss Army Knife for HDL Design Engineers

HDL Companion is like the Swiss Army Knife. It provides the HDL Design Engineer with 1001 different features like: Syntax checking, Design Exploration & Navigation, Signal Tracing, Linting, HDL Sensitive Editor, Global Search and Replace, HTML Generator, Source Revision Control and an EDA Tools configurator. It is easy to use, affordable and should be present in each designers toolbox.



```
mbb           : out    std_logic ;
mbcr_wr       : in     std_logic ;
mbdr_i2c      : inout  std_logic_vect
mbdr_micro    : in     std_logic_vect
mcf           : inout  std_logic ;
mif           : out    std_logic ;
mif_bit_reset : in     std_logic ;
msta          : in     std_logic ;
msta_rst      : out    std_logic ;
mtx           : in     std_logic ;
reset         : in     std_logic ;
rsta          : in     std_logic ;
rsta_rst      : out    std_logic ;
rxak          : out    std_logic ;
scl           : inout  std_logic ;
sda           : inout  std_logic ;
srw           : inout  std_logic ;
sys_clk       : in     std_logic ;
txak          : in     std_logic ) ;
```

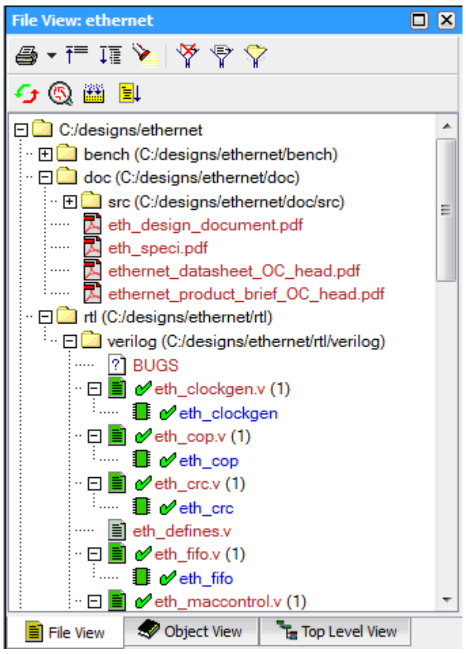
Lint results for project I2C Bus Model: 34 results (18 errors, 16 warnings, 0 notes)

Message	File
Port 'BusInterface.mbdr_micro' is never read (CP14)	bus
Process 'BusInterface.structural.next_state_decoding.next_state_decoding': 'clk' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'as_int' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'address_match' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'as_int_d1' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'clk' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'ds_int' should not be on the sensitivity list (CP8)	bus
Process 'BusInterface.structural.state_actions.state_actions': 'as' should not be on the sensitivity list (CP8)	bus
Port 'i2c_Control.mbdr_i2c' is never read (CP14)	i2c

In the past a designer had to use a simple text editor and Unix like ‘grep’ commands to find his way through many lines of HDL code or use a costly simulator license to understand the design structure. With HDL Companion a designer can drag and drop design files or directories into the file view and a complete design decomposition is performed in seconds, offering information regarding numerous aspects of the design.

HDL Companion

HDL Companion consists of four windows including a command console. Together they offer a complete overview of your design, from library level down to the details of your HDL source code. Each window can be enabled and disabled separately so you can easily configure the tool to show you exactly the information you are focusing on. HDL Companion is equipped with both fuzzy and fully compliant VHDL and Verilog parsers, freeing you from the burden of determining file dependencies while offering you full syntax checking. Just drag & drop your design folder in HDL Companion and get started.

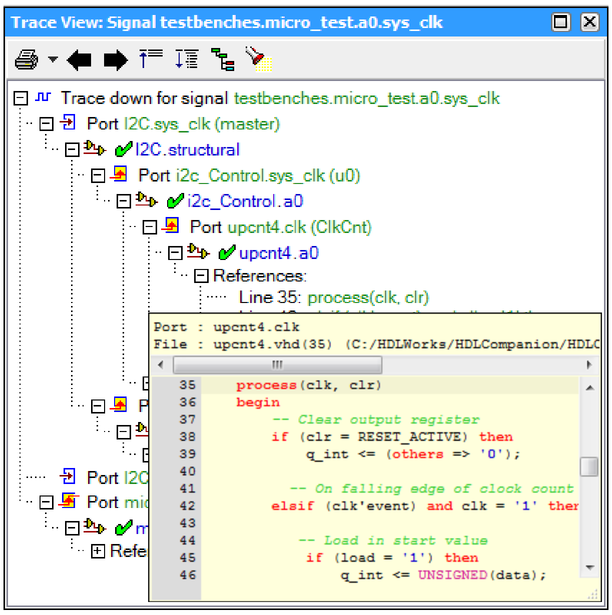


Global Window

The Global Window is separated into three tab pages offering you a file view, a library / object view and a hierarchical design view. The file view shows you the files present in your design in a browser like manner while also presenting the main design objects contained in each file. You can select any object to either edit it in the integrated HDL Editor (or your preferred editor) or have a more detailed description in the Detailed Window. The file view is not limited to HDL files, but shows all files in your project, providing easy access to all your design data, like PDF and other documents. It is also the place to perform version management actions when a version management system is enabled. File view filters allow you to focus on what you want to see. The object view shows all objects in the design (entities, architectures, configurations, modules, etc.) organized by HDL library. The hierarchical view shows the hierarchy of the selected design unit. The views are driven by fuzzy VHDL and Verilog parsers. These parsers take care of all the tedious work of placing objects in libraries and determine the correct file compilation order. They also allow you to work with incomplete or erroneous designs. All views are related and allow drag and drop for easy navigation through your project. A tooltip is available which shows you a (syntax directed and scrollable) view of the file contents without opening it.

Detailed Window

The Detailed Window offers you an in-depth view of the structure or hierarchy of a selected HDL object. For an entity or module the list of parameters, ports and architectures is shown, as well as the places where the entity or module is instantiated. This gives you detailed information about where and how signals are used throughout the whole hierarchy of your design. All labels in the tree view are click-able, providing easy access to the source code or other detailed view objects. The Detailed Window is also used to show the trace results of a port or signal. A trace shows how a signal is used through the hierarchy.

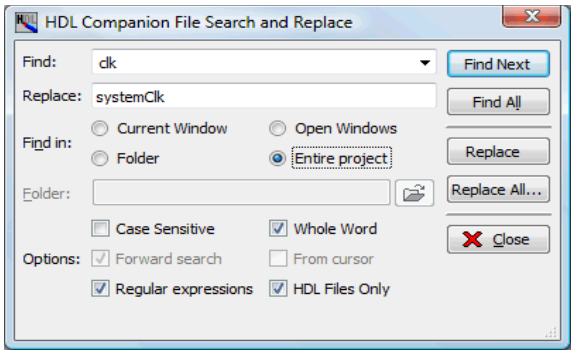


Trace View: Signal testbenches.micro_test.a0.sys_clk
Trace down for signal testbenches.micro_test.a0.sys_clk
Port I2C.sys_clk (master)
I2C.structural
Port i2c_Control.sys_clk (u0)
i2c_Control.a0
Port upcnt4.clk (ClkCnt)
upcnt4.a0
References:
Line 35: process(clk, clr)
Port I2C
Port mic
Refe

```
Port : upcnt4.clk
File : upcnt4.vhd(35)  (C:/HDLWorks/HDLCompanion/HDLC
```

```
35    process(clk, clr)
36    begin
37        -- Clear output register
38        if (clr = RESET_ACTIVE) then
39            q_int <= (others => '0');
40
41          -- On falling edge of clock count
42        elsif (clk'event) and clk = '1' ther
43
44          -- Load in start value
45            if (load = '1') then
46                q_int <= UNSIGNED(data);
```

HDL Editor



HDL Companion File Search and Replace

Find: clk

Replace: systemClk

Find in: Current Window, Open Windows, Folder, [✓ Entire project]

Folder:

Options: Case Sensitive, [✓ Whole Word], [✓ Forward search], From cursor, [✓ Regular expressions], [✓ HDL Files Only]

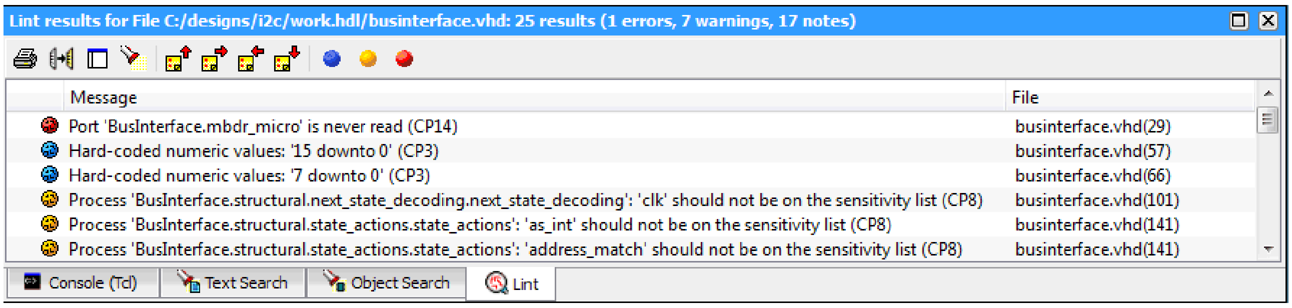
The HDL Editor offers a fully featured language sensitive text editor with a multiple document interface. You can avoid typing errors and improve your productivity by using language templates, identifier repeat and one-touch line and column manipulation. Syntax highlighting keeps your text highly readable and well structured. Functions like 'Search and Replace', 'Copy Entity', 'Copy Instance' and 'Create Testbench' help you to speed-up development tasks. The marker system allows you to drop markers for fast navigation between various file and text positions. Syntax errors or warnings are shown by coloured dots at the beginning of a line. An overview of all reported issues is shown at the right side of each edit window. Tight integration with the data model allows you to directly jump to object definitions and their use. If you want to use your own favourite text editor, you can configure HDL Companion to use the integrated editor for viewing only.

Linting

Linting is an additional verification effort to find potential design problems (like range mismatches in assignments of vectors, or read-only signals) and optimize the design by identifying unused signals and definitions. HDL Companion also supports a number of the DO 254 guidelines which helps you to improve the design quality. Many lint checks are available, including the following:

DO 254 Coding Practice:	Miscellaneous:
Avoid duplicate signal assignments	Avoid processes with multiple clocks
Avoid hard-coded numeric values	Avoid latches
Avoid mismatching ranges	Avoid positional port map in instantiations
Ensure complete sensitivity list	Avoid reset with different value than initial value
Avoid unused declarations	Avoid non standard clock expressions

The severity level can be set for each check separately or the check can be disabled altogether. Errors, warnings and notes are reported in the verification pane. The messages are hot-linked to the text editor so you can quickly navigate to the offending code.



Lint results for File C:/designs/i2c/work.hdl/businterface.vhd: 25 results (1 errors, 7 warnings, 17 notes)

	Message	File
●	Port 'BusInterface.mbdr_micro' is never read (CP14)	businterface.vhd(29)
●	Hard-coded numeric values: '15 downto 0' (CP3)	businterface.vhd(57)
●	Hard-coded numeric values: '7 downto 0' (CP3)	businterface.vhd(66)
●	Process 'BusInterface.structural.next_state_decoding.next_state_decoding': 'clk' should not be on the sensitivity list (CP8)	businterface.vhd(101)
●	Process 'BusInterface.structural.state_actions.state_actions': 'as_int' should not be on the sensitivity list (CP8)	businterface.vhd(141)
●	Process 'BusInterface.structural.state_actions.state_actions': 'address_match' should not be on the sensitivity list (CP8)	businterface.vhd(141)

Naming Conventions

The name convention checking allow you to check any name conventions you specify using regular expressions.

TCL integration

The console window also functions as a TCL interpreter, allowing you to execute any TCL or shell command. The TCL interface offers access to HDL Companion's internal data structures, allowing you to write your own TCL scripts to generate reports, perform specific checks or add your own functionality.

HTML

The HTML documentation function allows you to export the HDL Companion views into HTML documents with the same cross reference links as in HDL Companion. HTML files are created for each HDL file with syntax highlighting and links to units, signals and types. Any view can be printed or placed on the clipboard for documentation purposes.

Design Flow

The tool flow wizard allows you to configure a complete flow of other tools (like simulation, synthesis and FPGA place and route) you want to use in your project. After running the wizard, buttons are present in the toolbar to start your tools. In combination with the top-level marker HDL Companion can compile the HDL files with the selected simulator or generate synthesis scripts for the synthesis tool. HDL Companion projects can also created using FPGA Vendor projects files from Altera, Lattice and Xilinx.

Features and benefits	
• VHDL, Verilog and mixed language	• Up and running with your design in minutes
• Syntax checking	• Design metrics
• Works on incomplete designs	• Quickly navigate through your design
• File / Library / Hierarchical views	• Extensive search functionality
• Distributed directories	• Signal tracing through the whole hierarchy
• Fits in any design flow	• Includes miscellaneous files (Word, PDF, etc.)

Language Support	Operating Systems	License Configuration
• VHDL 87, VHDL 93, VHDL 2002, VHDL 2008 • Verilog 95, Verilog 2001, Verilog 2005	• Windows (64 bit) 7 / 8.1 / 10 • Linux 64 bit (x86 PC, any recent distribution)	• Node locked • Floating • FlexLM protected

HDL Works B.V.
Keplerlaan 16, 6716 BS, Ede
The Netherlands
Phone: +31 (0)318 642 022
E-mail: info@hdlworks.com
http://www.hdlworks.com