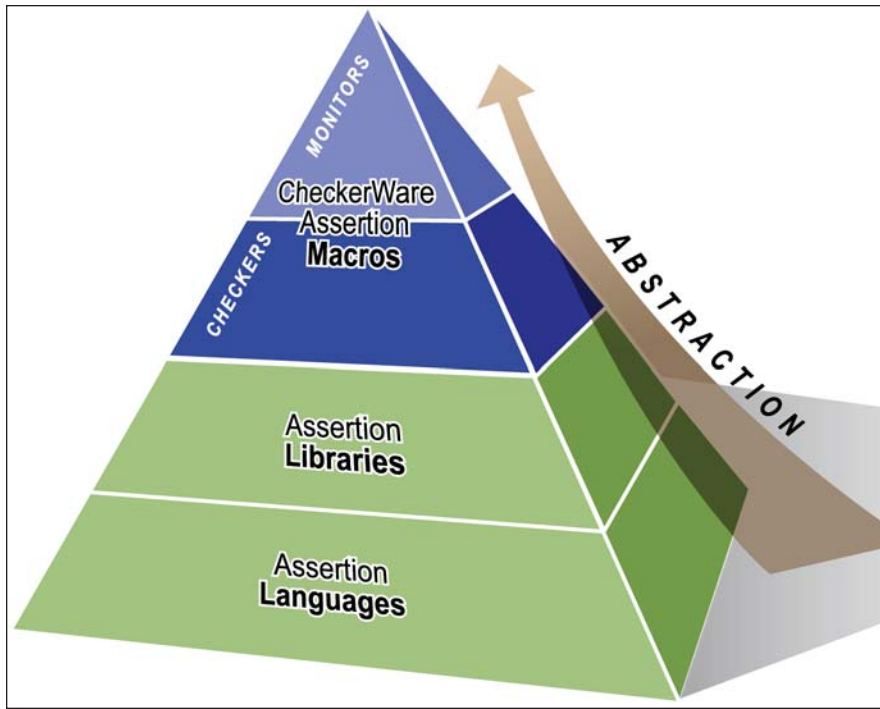


CheckerWare



With broad support of assertion languages and libraries, CheckerWare assertion macros simplify adoption of ABV and increase verification productivity through the automated generation of standard assertions.

The CheckerWare® library of advanced verification IP provides a rich set of pre-verified, standards-based assertion macros that increase productivity through automated assertion specification, maintenance, and coverage. CheckerWare assertion macros include protocol monitors and structural checkers that facilitate the adoption of the assertion-based verification (ABV) and coverage-driven verification (CDV) methodologies, enabling verification at a higher level of abstraction.

All CheckerWare components are complementary to and fully interoperable with the leading assertion standards, such as the Open Verification Library (OVL), Property Specification Language (PSL), and SystemVerilog Assertions (SVA). This provides engineers the flexibility to use the optimum combination of custom and pre-packaged assertions within a single verification environment.

CheckerWare assertion macros greatly simplify the automated specification of assertions. They are easily added to designs and can be used without

Major product features:

- Complete library of assertion macros for design checking and protocol monitoring
- Unique design inference capability for ease of adoption and maintainability of assertions
- Automatic coverage generation and collection quickly identifies verification holes
- Total interoperability with all standard assertion formats, including PSL, SVA, and OVL
- Advanced management of checkers and assertions throughout design lifecycle
- Supports simulators, emulators, and formal verification tools from Mentor Graphics and third parties

```

module zi_control;
  // 0in arbiter -req top.arb.rbus -gnt top.arb.gbus -fair -req_until_grant
  // 0in overflow -var error_count -mod top.stats

  zi_cw_pci_mon (
    .pci_ad_en_n (top.pci_ad_en_),
    .pci_cbe_en_n (top.pci_cbe_en_),
    ... );
endmodule

```

```

module A (clk, reset, ...);
  reg [3:0] pointer; // 0in gray_code

  // 0in req_ack -req req -ack ack -req_until_ack

  ...
  always @(posedge clk)
    if (en && (pointer == 4'b1011)) begin
      next_state <= `read; // 0in < sequence -val `read `wait `idle
    end
  ...
endmodule

```

CheckerWare assertion macros use standard assertion language constructs for complete interoperability and flexibility.

modification throughout the verification flow—including simulation, emulation, and formal functional verification. The assertion macros automatically generate standards-based assertions, coverage metrics, and formal verification constraints. Because they are standards based and pre-verified, they reduce many of the risks associated with design verification.

CheckerWare macros include structural checkers and protocol monitors that verify design and interface intent as well as design integrity. Assertion checkers verify common register-transfer level (RTL) structures and interfaces. Protocol monitors track the complete cycle-by-cycle protocol rules for complex, standard buses and interfaces.

Simplifying Assertion Specification

To verify higher-level RTL structures—such as arbiters, linked-lists, FIFOs, and standard buses—a collection of related assertions is required. Typically this involves a design or verification engineer writing all of the assertions from scratch and updating them whenever the design changes.

CheckerWare overcomes this labor intensive effort by automatically encapsulating all of a design's assertions based on the checker configuration required to fully verify the RTL structure. Each assertion macro automatically generates from 2 to 4000 assertions and coverage points per macro, depending on the checker configuration. Then, Mentor's unique design inference technology auto-

matically extracts the necessary design information to update the assertions. Combined with integrated coverage measurements, this makes CheckerWare the easiest to use ABV solution available.

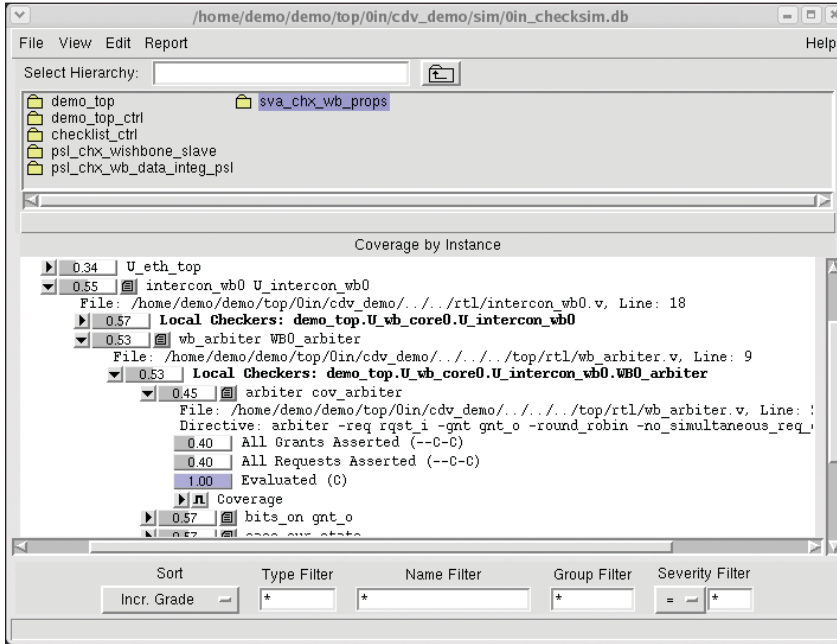
Rich Selection of Checker and Monitor Types

The CheckerWare library contains a rich selection of assertion checkers that map to RTL structures found in every design. The checker library has over 100 checker types. Each checker type has multiple configuration options and can be cascaded, providing a rich set of functions. Based on customer feedback, new checkers and options are added with each release.

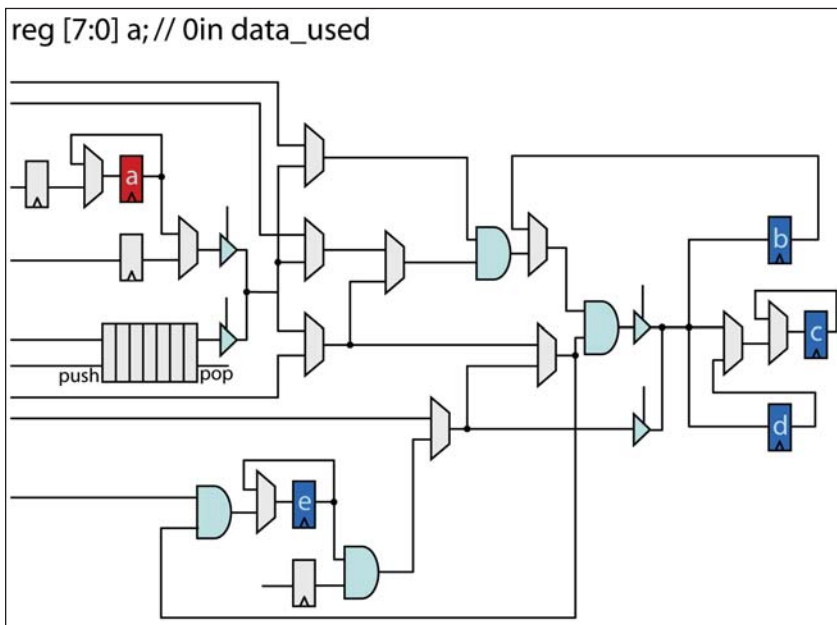
CheckerWare protocol monitors capture the protocol behaviors of industry standard interfaces. In addition to performing function checks, each monitor collects and reports detailed statistical and transaction information about the interface. The collection of 0-In monitors is continually expanded, based on customer demand. A few of the many monitors currently available are USB 2.0, AMBA AXI, OCP, SATA, and PCI Express. The complete listing of CheckerWare monitors can be found on www.mentor.com.

Integrated Coverage Metrics

Measuring coverage is critical to reaching verification closure in modern ASIC and SoC designs. To support a complete coverage-driven verification methodology,



The 0-In View assertion management interface displays the recorded structural coverage for an arbiter. In this example, simulation has not exercised all channels of the arbiter, so additional simulation or formal testing is required.



Registers b, c, d, and e consume a, depending upon circuit activity. To perform the check, the load conditions and clocks for all of these registers must be inferred.

CheckerWare library elements contain detailed, implementation-specific, structural coverage metrics for functional RTL units as well as transaction coverage metrics for standard interfaces. Using the 0-In View assertion management interface, engineers can view and manage protocol functional checks, protocol corner cases, and protocol statistics. These important capabilities ensure standards compliance, verify corner-case behaviors, and generate statistical information.

Design Inference Technology

As a design evolves, maintaining assertions typically requires significant verification overhead. CheckerWare overcomes this problem with its unique design inference technology. This allows CheckerWare to extract important design information—such as names, widths, clocks, resets, and activation conditions—directly from the RTL code. This not only makes specification significantly easier, but also allows CheckerWare components to automatically adapt to RTL design changes and perform tests that would otherwise be difficult or impossible to specify.

For example, as shown in the following diagram, the *data_used* check ensures that the datapath register *a* is always consumed prior to being over written. The specification of the test is trivial from the

user perspective, but the design inference performed can quickly become extremely complex.

Simulation

User-specified assertions and assertions promoted from automatic assertion checking are simulated with the design and test environment. They passively monitor activities inside the design and on the interfaces to ensure the design is working correctly. Whenever an assertion is violated, it is reported immediately.

Formal Verification

By instrumenting a design's interfaces with CheckerWare components, advanced formal verification techniques can be applied quickly and with minimal effort. All CheckerWare protocol monitors and checkers include constraints for formal verification.

Compatibility

CheckerWare libraries are compatible with the following simulation and emulation products:

- Mentor Graphics Questa™ and ModelSim®
- Synopsys VCS
- Cadence Verilog-XL, NCSim,

CheckerWare Assertion Macros

MONITORS	AGP	QDR SRAM	SigmaRAM
	PCI-X	DDR-II SDRAM	AMBA
	DDR SDRAM	SPI4-2	Gigabit Ethernet
	UTOPIA Level 1	USB 1.1	OCP
	POS-PHY Level 3	POS-PHY Level 2	SCSI
	UTOPIA Level 4	Infiniband	USB 2.0

CHECKERS	interface assert_follower, assert_timer, bus_driver, change_timer	user custom, mutex, same_bit, timeout,
	control channel_data_integrity, multi_port_memory_access	basic driven, known, known_driven, three_state
	interface and control assert_together, multi_clock_fifo, scoreboard	coverage coverage, xproduct_bit_coverage, xproduct_value_coverage
	datapath arithmetic_overflow, data_used, encoder, serial_to_parallel	cdc cdc_dsel, cdc_fifo, cdc_sample

This small sample only hints at the full selection of monitors and checkers available in the CheckerWare library of assertion macros.

Incisive, and Palladium

— Verisity Axis

Hardware Platforms

- Solaris (32 and 64 bit)
- Linux (RedHat 7.1+)
- HP-UX 11

Visit our web site at www.mentor.com/fv for more information.

© 2006 Mentor Graphics Corporation.

0-In, CheckerWare, Mentor Graphics, and ModelSim are registered trademarks and Questa is a trademark of Mentor Graphics Corporation.

All other products or services mentioned in this data sheet are covered by the trademarks, service marks or product names as designated by the companies that market those products.

Corporate Headquarters
Mentor Graphics Corporation
8005 S.W. Boeckman Road
Wilsonville, Oregon 97070 USA
Phone: 503-685-7000
North American Support Center
Phone: 800-547-4303
Fax: 800-684-1795

Silicon Valley
Mentor Graphics Corporation
1001 Ridder Park Drive
San Jose, California 95131 USA
Phone: 408-436-1500
Fax: 408-436-1501

Europe
Mentor Graphics
Deutschland GmbH
Arnulfstrasse 201
80634 Munich
Germany
Phone: +49.89.57096.0
Fax: +49.89.57096.400

Pacific Rim
Mentor Graphics Taiwan
Room 1603, 16F,
International Trade Building
No. 333, Section 1, Keelung Road
Taipei, Taiwan, ROC
Phone: 886-2-27576020
Fax: 886-2-27576027

Japan
Mentor Graphics Japan Co., Ltd.
Gotenyama Hills
7-35, Kita-Shinagawa 4-chome
Shinagawa-Ku, Tokyo 140
Japan
Phone: 81-3-5488-3030
Fax: 81-3-5488-3031



Printed on Recycled Paper

6-06 MGC

1025050-w