

AMBA AHB *e*VC

Automated, Coverage-Driven Verification IP

AHB *e*VC

- ✓ AMBA Bus Specification Revision 2.0
- ✓ Supports AHB Lite
- ✓ Supports AHB Multi-Layer
- ✓ Functions as:
 - Master(s)
 - Slave(s)
 - Arbiter
 - Decoder
- ✓ Supports any valid AHB topology, up to 16 slaves and 16 masters
- ✓ Supports multiple bus widths
- ✓ Supports big and little endian
- ✓ Arbitration with three levels of priority
- ✓ Supports all valid burst types
- ✓ Total control over bus traffic generation using sequence interface including injection of errors
- ✓ Built-in bus traffic monitors
- ✓ Built-in coverage driven verification support (CDV)
- ✓ Scoreboard checking of input and output
- ✓ HDL independent

e Verification Component Overview (*e*VC)

e Verification Components are reusable, configurable, pre-verified, plug-and-play verification environments. They offer the easiest to use, most complete module, chip and system level verification solution available. *e*VCs integrate automatic stimulus generation, assertion checking, and functional coverage analysis all within a single, extensible component. *e*VCs drastically reduce the time needed to compose a verification environment.

The philosophy underlying *e*VCs differs significantly from alternative products. Rather than use thousands of directed tests, the *e*VC employs automatic generation and a coverage driven methodology. Using automated scenario generation the *e*VC can typically achieve 90-95%+ coverage of the protocol. With the addition of a few tests the remaining corner cases are then exercised. This approach uncovers more bugs faster and frees engineering time to focus on testing the Device Under Test (DUT) proprietary functionality.

Quality and Productivity Gains

With *e*VCs verification environments are created in days instead of weeks or months. You can begin running tests much earlier and achieve a much higher quality product. Furthermore, *e* Verification Components can

be reused without expending any extra effort. This enables you to retain your investment when moving from module to system level verification as well as when verifying derivative products.

AHB *e*VC Overview

The AHB *e*VC verifies a device under test (DUT) in the AMBA AHB bus environment. Verisity and ARM have worked closely together to ensure that the AHB *e*VC accurately verifies AMBA's AHB protocol. Verisity's AHB *e*VC also works effectively with ARM's AMBA Compliance Testbench (ACT). The AHB *e*VC is stable and mature. It has been employed in over 400 verification projects for over 140 different design teams.

Configurable Verification Environment

The Verisity AHB *e*VC is highly configurable. This enables you to apply it to many different verification environments. Two examples are described and shown below:

Single AHB DUT

The AHB *e*VC functions as a master device, slave device, arbiter, or decoder. In addition, combinations of these functions are also possible for the same single instance. A master DUT can be verified in an *e*VC environment that provides the arbiter and decoder along with any number of masters and slaves exercising the DUT.

AMBA AHB eVC

Verity – Meet your SpeX

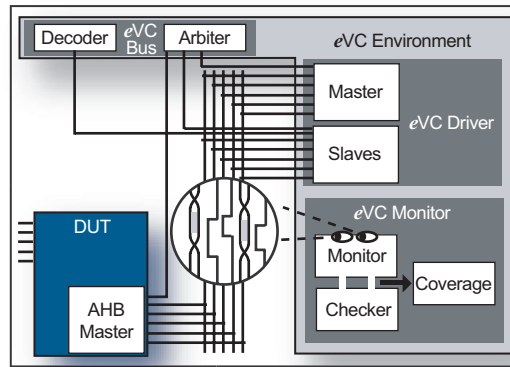
Contact Information

Verity Design, Inc.
331 East Evelyn Avenue
Mountain View, CA 94041
PH: (650) 934-6800
FX: (650) 934-6801
www.verity.com

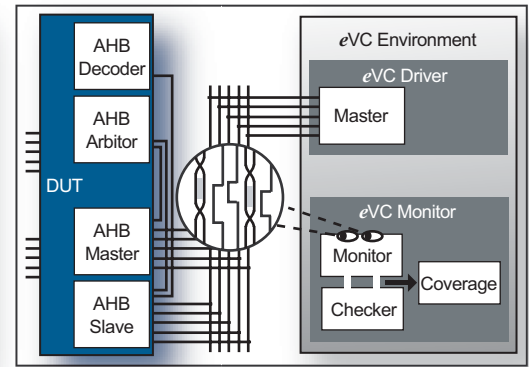


For More Information

To find out more about the AHB eVC contact your Verity account manager or see us on the web at www.verity.com.



Single AHB DUT



System-Level AHB DUT Environment

System-Level AHB DUT Environment

The DUT consists of a combination of AHB elements. The eVC is plugged into this environment to complete the system-level verification setup. For Example, the eVC can drive additional traffic on the bus and monitor the protocol.

Deliverables

- Fully verified AHB eVC code written in the e language
- Documentation, including a user guide and release notes
- Standalone introductory demonstration
- Sample, extensible sequence covering basic functionality

AHB eVC Functional Description

Master	Active emulation of a master device that generates and drives transactions into the bus
Slave	Active emulation of a slave device accepts transaction from any master
Arbiter	Active emulation of bus arbitration
Decoder	Active emulation of decoding
Monitor	Passive monitoring, checking and collection of functional coverage of bus traffic

Protocol checking and coverage collecting require the monitor element. Otherwise, each of these elements are entirely independent. You can use any or all of them selectively as required.

