

# An innovative verification approach for Serial ATA 1.0a host controller

George Attokaran, Anil Sewani, Sandip Solanki, Sal Vemuri,



George Attokaran is director, design services at QualCore's Sunnyvale, USA operation. He holds an MSEE from San Jose State University and has more than 11 years of experience in the design, development and testing of ASIC designs



Sandip Solanki is a Member Technical at QualCore Hyderabad. He holds a BTech from DDIT at Gujarat University, Nadiad and has more than five years of experience in the design, development and testing of ASIC designs



Anil Sewani is a member technical at QualCore's Hyderabad, India operation. He has a BTech from MBM Engg College, Jodhpur, India and has more than five years of experience in the design, development and testing of ASIC designs



Sai Vemuri vice president of engineering at QualCore Hyderabad and holds an MSEE from, IIT, Chennai, India. He has more than 20 years of experience in the ASIC Industry

## The company

QualCore Logic is a leading provider of digital, mixed-signal and analog IP for system-on-chip (SoC) designs. It can take an engineering or register transfer level (RTL) specification to GDSII through final silicon. QualCore operates design and support centers in Sunnyvale, California, and Hyderabad, India. It has more than 150 design engineers.

## The project

Parallel ATA has begun to show signs of age with its high pin count, serious cabling headaches and a 5V signaling requirement. This has spurred the introduction of Serial ATA (SATA) 1.0a. This specification is designed to overcome the limitations of Parallel ATA while enabling the storage interface to scale with the growing media-rate demands of PC platforms.

SATA ensures compatibility with existing operating systems and drivers, and also adds performance headroom. It reduces voltage and pin count requirements and can be implemented with thin and easy-to-route cables. Moreover, SATA 1.0a provides a transfer rate of 150Mbyte/s. That speed will double with the release of its next generation interface.

In developing a host controller for SATA 1.0a, QualCore found that such a core's design, implementation, verification and hardware validation required a considerable investment to ensure total compliance to the standard.

The company also needed to adopt an innovative approach to verification to ensure the absolute quality of deliverables, while optimizing effort and resources needed to complete the project. This process is outlined below on a segment-by-segment basis.

Specific deliverables for the host controller were:

- Fully synthesizable Verilog RTL source code.
- Documentation — datasheet, user guide, verification description document.
- Self-checking verification suite.
- Synthesis scripts.

This article outlines the verification flow and joint verification and design procedures used by QualCore Logic to develop a host controller core for the emerging Serial ATA standard within a six-month time constraint.

- Scripts for static timing analysis and design for testability (optional).

Now successfully completed, the core has been hardware validated with the help of a Xilinx HW-V2P-ML321 Design Kit, and is currently being demonstrated to customers.

## Verification methodology

As with any intellectual property (IP) development project, the QualCore team had to go through the various steps of specification, design, verification, synthesis and timing analysis. However, in view of the available resources and time constraints for the project, the team took a number of novel steps with regards to verification.

The main difference from traditional techniques came in the form of a bottom-up, protocol-layered, incremental integration approach to verification where all reusable vectors were written in proprietary C language syntax. So significant was the impact of this verification process that it has ultimately come to be seen as a major contributor to the quality of the finished core.

Figure 1 shows the verification environment of the SATA Core using Bus Functional Models (BFMs), Bus Monitors, stimulus generated using C syntax commands and self-checking capability.

### Verification team's role in architecture and design reviews

QualCore's verification engineers were involved from the beginning of the project. As a first step, the design and verification staff discussed the specification threadbare to ensure that the entire team had an agreed and documented interpretation of the features and the numerous protocol details. This helped coordinate work on many tricky design issues across the development phase.

Beyond that, verification engineers then participated in all specification, architecture and design reviews. The intense level of interactions helped in quickly bringing out the corner and critical test cases. Further, the verification team was able to get a clinical insight into the intricacies in the implementation of the complete core and, this simplified the task of building a comprehensive verification strategy with an extensive set of test cases.

# oach for a

## QualCore Logic

### Protocol layered incremental integration approach for verification

The SATA core was block partitioned into three basic protocol layers — application, transport and link — with a bottom up verification strategy.

Each protocol layer block was verified independently, followed by an incremental integration verification for which the blocks were stacked together. The transport layer block was integrated with a link layer block and the combined stack of two blocks was verified. This was followed by the integration of the application layer block and then the stack of all three blocks was verified.

This verification scheme ensured an effective and complete functional verification.

### Verification of finite state machine implementation

There are two finite state machine (FSM) implementations in the SATA core. One FSM at the link layer block has 30 states and 294 possible state transitions. An equally complex FSM was implemented at the transport layer block and complete FSM verification was attempted in each case.

Stimulus was applied such that each of the states was reached and all possible transitions achieved. Additionally, a dedicated testcase was developed to reach each state. From this basic testcase, all possible transition testcases were derived.

### Use of full functional device model

A fully functional behavioral model of the SATA device was developed compliant with the 1.0a specification. This was used to function as the SATA device, and attached to the SATA host adaptor core for design-under-test (DUT).

The model was developed as a set of Verilog tasks that were fully controlled by device stimulus using built-in control registers. Within it, the two channels for transmit and receive logic were independently implemented, and a switch was implemented to decide the source of input data available from a counter logic and from a data file.

### Use of bus functional models

Two interface models – the Open Core Protocol (OCP) master bus functional model (BFM) and the OCP slave bus functional model – were developed to interact with the DUT from the host side. These models were made fully controllable by the host stimulus.

### Extensive use of bus monitors

Bus monitors were placed at all interfaces to the DUT to track and report the protocol and other data transactions. These monitors sent triggers during simulations and provided online comparison capabilities. Features of these bus monitors were controlled through programmable register bits.

SOURCE: QualCore Logic

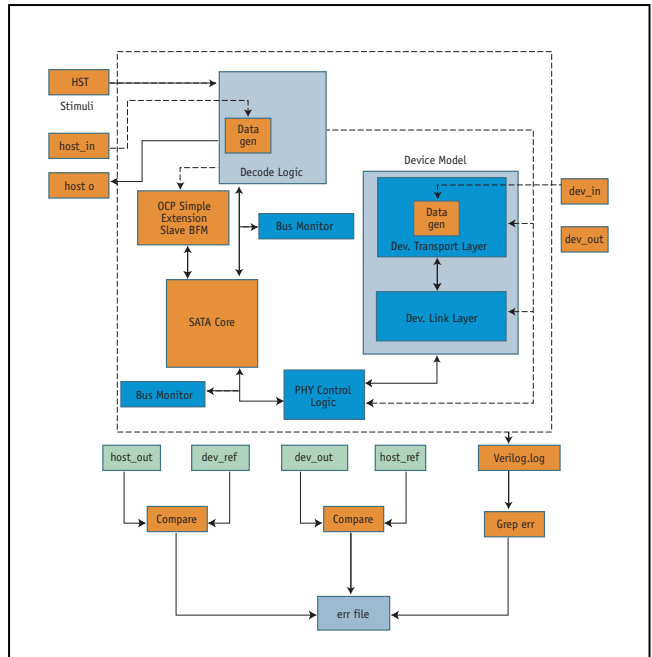


FIGURE 1 Verification environment for SATA 1.0a core

### Offline response data comparison

To save simulation time, most of the data comparison and response analysis was done offline. For this purpose, C utility programs were developed. Simulation output data was stored in files that were compared against golden reference files which were generated by exercising the Serial ATA Model.

### Vectors written at higher abstraction level

Vectors were written in C language syntax to allow use of C functions to form different testing sequences. These C format vectors were converted offline to hexadecimal datafiles using a proprietary vector format conversion utility. The datafiles were then read into the Verilog simulation environment.

Vectors were written in the form of simple register read and write commands with each line of the hexadecimal file represented as one command. These commands were decoded using a command interpreter and mapped to various control register bits.

### Reuse of block level test vectors at system level

Nearly 60% of the vectors developed during block-level verification were reused at system level verification with little modification. In fact, it was easy to shift the control register layer from the block-level verification environment to system level and do verification using the same stimulus.

### Complete self-checking test environment

The test environment was made completely self checking using Perl and Shell scripts. There was a Pass/Fail status indication after the simulation of each testcase, based on the response analysis using bus monitors and offline data comparison utilities.

>>> continued on next page

### Regression

At each stage of verification, test vectors were graded, and those with maximum functional coverage were identified making it possible to quickly simulate these selected test vectors in a batch run after every change in the design.

### Coverage

An attempt was made to cover the maximum possible design functionality, achieved by a thorough study of design specifications by verification engineers and a careful review of testcases by block designers. The functional coverage was complemented by a tool that aided code coverage analysis.

A near 100% code coverage was achieved at the block level, while at the system level, the code coverage matrix was 95%. This coverage was achieved under design blocks, toggle, expression and state transition categories.

### Redundancy

Redundancy was purposely added to increase the effectiveness of verification. Many of the block-level testcases were mapped to system-level testcases and conditions covered at block-level verification were re-exercised at the system level.

These common testcases were developed by two different groups of test engineers with one group working at the block level and the other group working at the system level. This redundancy added to the confidence before delivery of the verified design.

### Conclusion

Ensuring quality under demanding schedules is a challenge as well as an unenviable responsibility for any management team. In this case, the task of delivering a quality core in a period of six months and with a limited team strength was achieved through meticulous planning of each of the interdependent tasks of design and verification.

The major factors which led to the success of the project were:

- Extensive brain storming of the SATA 1.0a standard prior to start of the project.
- A protocol layered incremental integration approach for verification.
- Test vectors developed at a higher abstraction level.
- The reuse of block level vectors at system level.
- Independent verification teams for block and system level.
- Judicious use of Bus Monitors and BFMs.
- Partitioning of tests for off-line and on-line verification to save on system time.
- And finally, maximizing the coverage.

QualCore Logic  
1289 Anvilwood Ave  
Sunnyvale  
CA 94089 USA  
Tel: +1 408 541-0730  
Fax: +1 408 541-0740

7-145 Nagendra Nagar  
Habsiguda  
Hyderabad  
500-007  
India  
Tel: +91-40-27174437  
Fax: +91-40-27171939

[www.qualcorelogic.com](http://www.qualcorelogic.com)

### Project Tools

#### Design and Verification

- The verification environment was built in the Verilog Hardware Description Language (HDL), complemented by structures, utilities and scripts written in C, Perl and Shell.
- NC-Verilog was used as the simulator.
- For debugging, Signalscan was used.
- For linting and code coverage analysis respectively, Verisity's SureLint and SureCov were used.

#### Synthesis

- The Serial ATA core design was targeted to the Artisan-TSMC 0.18µm library.
- Synopsys' Design Compiler was used for synthesizing the register transfer level (RTL).
- Extensive RTL code linting was performed before synthesis.
- Latches and internal tristates were avoided during synthesis.

#### Static Timing Analysis (STA)

- Synopsys' PrimeTime was used for static timing analysis.
- STA was performed for each individual block.
- STA was performed again for the integrated design.
- Multicycle paths and false paths were identified and fixed during synthesis.

#### Design For Testability (DFT)

- The memory built-in-self-test circuit for RAM cells was generated and inserted into the core design using Mentor Graphics' MBISTArchitect.
- Using Multiplexed flop scan style, full scan was inserted using Mentor's DFTAdvisor.
- ATPG vectors with 96% fault coverage were generated using Mentor's FastScan.

#### Formal Verification

- Equivalence checking was performed before and after DFT insertion in the design. Synopsys' Formality was used for this purpose.

#### Hardware Validation

- Hardware Validation was done using Xilinx HW-V2P-ML321 Design Kit, which has Xilinx Virtex II Pro XC2VP7 FPGA.
- A Seagate Serial ATA 1.0 Drive (Barracuda 7200.7, 80 Gbytes, Model No: ST380013AS) and a Western Digital Serial ATA 1.0 Drive (WD800, 80 Gbytes, Model no: WD800JD-00HKA0) were also used.
- A successful demonstration was done at a customer site, detailing features of the SATA core.