ISSN 2348 – 9928
IJAICT Volume 5, Issue 6, June 2018 Doi:01.0401/ijaict.2015.11.13 Published on 05 (6) 2018

ETHERNET IP CORE VERIFICATION USING SYSTEM VERILOG HDL

Ms. P. Abinaya
UG Student,
Electronics and Communication Engineering,
KPR Institute of Engineering and Technology,
Coimbatore, Tamilnadu, India

Ms. S. Bala Jeevitha
UG Student,
Electronics and Communication Engineering,
KPR Institute of Engineering and Technology,
Coimbatore, Tamilnadu, India

Ms. M. Janani
UG Student,
Electronics and Communication Engineering,
KPR Institute of Engineering and Technology,
Coimbatore, Tamilnadu, India

Ms. S. Prema,
Assistant Professor,
Electronics and Communication Engineering,
KPR Institute of Engineering and Technology,
Coimbatore, Tamilnadu, India

Abstract— Functional verification of IP is an essential process in the chip/ System on Chip (SoC) design process, since this would ensure correct functionality of the IP with the SoC. This paper intends to highlight the functional verification process of an Ethernet IP core, with emphasis on the Ethernet MAC.The Ethernet IP has been adopted from OpenCores. The IP interfaces to an Ethernet Media Access Control (MAC) compatible Physical layer interface (PHY) on one side and on the other it interfaces to the host system through a Wishbone bus (WB). The IP has a MAC control module, host interface and Media Independent Interface (MII). The design and verification environment are built using Verilog HDL, an industry standard Hardware Description Language (HDL). A PHY model is implemented to model the behavior of the PHY in the system. A host model is implemented with a Wishbone interface. A basic set of four tests have been developed. They are Unicast, Multicast, Broadcast and basic Transmit and Receive tests. Additional tests are being added to these basic tests to complete the verification of the IP.Open Source VerilogHDL simulator, iverilog, is used to perform this verification process.

Keywords— MAC, MII, HDL, PHY.

I. INTRODUCTION

Ethernet is the most widely-installed local area network (LAN) technology. Specified in a standard, IEEE 802.3, Ethernet was originally developed by Xerox from an earlier specification called *Alohanet* (for the Palo Alto Research Center Aloha network) and then developed further by Xerox, DEC, and Intel. An Ethernet LAN typically uses coaxial cable or special grades of twisted pair wires. Ethernet is also used in wireless LANs. The most commonly installed Ethernet systems are called 10BASE-T and provide transmission speeds up to 10 Mbps. Devices are connected to the cable and compete for access using a Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol.

1.1 Types of Ethernet

i) 10Mbps

All Ethernet connections are limited by the slowest component, be that the hub, the Ethernet card or the Ethernet cable. While 10Mbps Ethernet long remained the standard for local-area networking and wide-area networking, it was ultimately held back by the type of cable it used.

ii) Fast Ethernet

Fast Ethernet supports all Ethernet frame types and software which makes turning traditional Ethernet to fast speed Ethernet straight forward. Usually this can be done only with replacing the Network Interface Card with a new fast Ethernet card and loading the drivers for the new card. It is designed to address the need for more bandwidth, increased performance, and additional users in Ethernet networks.

iii) Gigabit Ethernet

What took decades to do previously -- to increase the speed of a network by 10 times only took three years to do again. Gigabit routers and Gigabit options in computers are becoming more common, however, and your office can take advantage of very-high-speed transfers on a Gigabit system.

II. ETHERNET IP CORE

The Ethernet IP Core consists of five modules:

- The MAC (Media Access Control) module, formed by transmit, receive, and control module

- The MII (Media Independent Interface) Management module
- The Host Interface

2.1 Ethernet IP Core Operations

The core consists of five modules

- The host interface connects the Ethernet Core to the rest of the system via the WISHBONE (using DMA transfers). Registers are also part of the hostinterface.
- The TX Ethernet MAC performs transmit functions.
- The RX Ethernet MAC performs receive functions.
- The MAC Control Module performs full duplex flow control functions.
- The MII Management Module performs PHY control and gathers the status information from it.

All modules combined deliver full-function 10/100 Mbps Media Access Control. The Ethernet IP Core can operate in half- or full-duplex mode and is based on the CSMA/CD (Carrier Sense Multiple Access / Collision Detection) protocol.

When a station wants to transmit in half-duplex mode, it must observe the activity on the media (Carrier Sense). As soon as the media is idle (no transmission), any station can start transmitting (Multiple Access). If two or more stations are transmitting at the same time, a collision on the media is detected. All stations stop transmitting and back off for some random time. After the back-off time, the station checks the activity on the media again. If the media is idle, it starts transmitting. All other stations wait for the current transmission to end.

In full-duplex mode, the Carrier Sense and the Collision Detect signals are ignored. The MAC Control module takes care of sending and receiving the PAUSE control frame to achieve Flow control (see the TXFLOW and RXFLOW bit description in the CTRLMODER register for more information).

The MII Management module provides a media independent interface (MII) to the external PHY. This way, the configuration and status registers of the PHY can be read from/written to.

2.1.1 Resetting Ethernet Core

The RST_I signal is used for resetting all sub-modules except the MIIM module. Setting the MIIMRST bit in the MIIMODER register to 1 resets the MIIM(Media Independent Interface) module. To reset the PHY, assert its RESET signal either through the boars system control register or by writing an appropriate bit in the PHY register.

2.1.2 Host Interface Operation

The host interface connects the Ethernet IP Core to the rest of the system via the WISHBONE bus. The WISHBONE serves to access the configuration registers and the memory. Currently, only DMA transfers are supported for transferring the data from/to the memory.

2.1.3 Configuration Registers

The function of the configuration registers is transparent.

2.1.4 Buffer Descriptors (BD)

The transmission and the reception processes are based on the descriptors. The Transmit Descriptors (TxD) are used for transmission while the Receive Descriptors (RxD) are used for reception.

The buffer descriptors are 64 bits long. The first 32 bits are reserved for length and status while the last 32 bits contain the pointer to the associated buffer (where data is stored). The Ethernet MAC core has an internal RAM that can store up to 128 BDs (for both Rx and Tx).

The transmit and receive status of the packet is written to the associated buffer descriptor once its transmission/reception is finished.

2.1.5 Frame Transmission

To transmit the first frame, the RISC must do several things, namely:

- Store the frame to the memory.
- Associate the Tx BD in the Ethernet MAC core with the packet written to the memory (length, pad, crc, etc.).

2.1.6 Frame Reception

- Set the receive buffer descriptor to be associated with the received packet and mark it as empty.
- Enable the Ethernet receive function by setting the RECEN bit to 1.

2.2 Ethernet MAC

Ethernet MAC will be divided into two types. These are

- TX Ethernet MAC
- RX Ethernet MAC

2.2.1 TX Ethernet MAC

The TX Ethernet MAC is divided into several modules that provide the following functionality:

- Generation of the signals connected to the Ethernet PHY during the transmission process
- Generation of the status signals the host uses to track the transmission process
- Random time generation used in the back-off process after a collision has been detected
- CRC generation and checking
- Pad generation
- Data nibble generation

2.2.2 RX Ethernet MAC

The RX Ethernet MAC module is divided into several sub-modules providing the following functionality:

- Preamble removal
- Data assembly (from input nibble to output byte)
- CRC checking for all incoming packets
- Generation of the signal that can be used for address recognition (in the hash table)
- Generation of the status signals the host uses to track the reception process

2.3 MAC Control Module

The MAC Control Module performs a real-time flow control function for the full-duplex operation. The control opcode PAUSE is used for stopping the station transmitting the packets. The receive buffer (FIFO) starts filling up when the upper layer cannot continue accepting the incoming packets. Before an overflow happens, the upper layer sends a PAUSE control frame to the transmitting station. This control frame inhibits the transmission of the data frames for a specified period of time.

When the MAC Control module receives a PAUSE control frame, it loads the pause timer with the received value into the pause timer value field. The Tx MAC is stopped (paused) from transmitting the data frames for the "pause timer value" slot times. The pause timer decrements by one each time a slot time passes by. When the pause time number equals zero, the MAC transmitter resumes the transmit operation. The MAC Control Module has the following functionality:

- Control frame detection
- Control frame generation
- TX/RX MAC Interface
- PAUSE Time
- Slot Timer

2.4 Features

- Performs MAC layer function as per IEEE 802.3 and Ethernet standard
- Supports full duplex and half duplex modes
- Flow control and generation of control frames in full duplex mode (IEEE 802.3x)
- Collision detection and auto re-transmitting on collision in half duplex mode (CSMA/CD protocol)
- Complete status for TX/RX packets
- 32 bit CRC generation and checking
- Delayed CRC generation
- Preamble generation and removal
- Automatically pad short frames on transmit
- Transmit, receive FIFO of 32 x 16(width x depth)
- Interrupt generation

III. FUNCTIONAL VERIFICATION

A primary purpose for functional verification is to detect failures so that bugs can be identified and corrected before it gets shipped to costumer. If RTL designer makes a mistake in designing or coding, this results as a bug in the Chip. If this bug is executed, in certain situations the system will produce wrong results, causing a failure. Not all mistakes will necessarily result in failures. The bug in the dead code will never result in failure. A single mistake may result in a wide range of failure symptoms. Not all bugs are caused by coding errors. There are possibilities that error may in the specification itself. Sometimes miscommunications between teams may lead to wrong design.

3.1 Test Bench

A test bench or testing workbench is a virtual environment used to verify the correctness or soundness of a design or model, for example, a software product.

The term has its roots in the testing of electronic devices, where an engineer would sit at a lab bench with tools for measurement and manipulation, such as oscilloscopes, millimeters, soldering irons, wire cutters, and so on, and manually verify the correctness of the device under test (DUT).

3.2 Device under Test

Device under test (DUT), also known as equipment under test (EUT) and unit under test (UUT), is a term commonly used to refer to a manufactured product undergoing testing.

IV. ARCHITECTURE

The Ethernet IP Core consists of 5 modules

- Host Interface and the BD structure
- TX Ethernet MAC (transmit function)
- RX Ethernet MAC (receive function)
- MAC Control Module
- MII Management Module

4.1 Host Interface

The host interface is connected to the RISC and the memory through the two Wishbone interfaces. The RISC writes the data for the configuration registers directly while the data frames are written to the memory. For writing data to configuration registers, Wishbone slave interface is used. Data in the memory is accesses through the Wishbone master interface.

4.2 TX Ethernet MAC

The TX Ethernet MAC generates 10BASE-T/100BASE-TX transmit MII nibble data streams in response to the byte streams supplied by the transmit logic (host). It performs the required deferral and back-off algorithms, takes care of the IPG, computes the checksum (FCS) and monitors the physical media (by monitoring Carrier Sense and collision signals).

4.3 RX Ethernet MAC

The RX Ethernet MAC interprets 10BASE-T/100BASE-TX MII receive data nibble streams and supplies correctly formed packet-byte streams to the host. It searches for the SFD (start frame delimiter) at the beginning of the packet, verifies the FCS, and detects any dribble nibbles or receive code violations.

4.4 MAC Control Module

The function of this module is to implement the full-duplex flow control.

The MAC Control Module consists of three sub-modules that provide the following functionality:

- Control frame detection
- Control frame generation
- TX/RX Ethernet MAC Interface
- PAUSE Timer
- Slot Timer

4.5 MII Management Module

The function of the MII Management Module is to control the PHY and to gatherinformation from it (status). During the read/write operation, the most significant bit is shifted in/out first from/to the MDIO data signal.

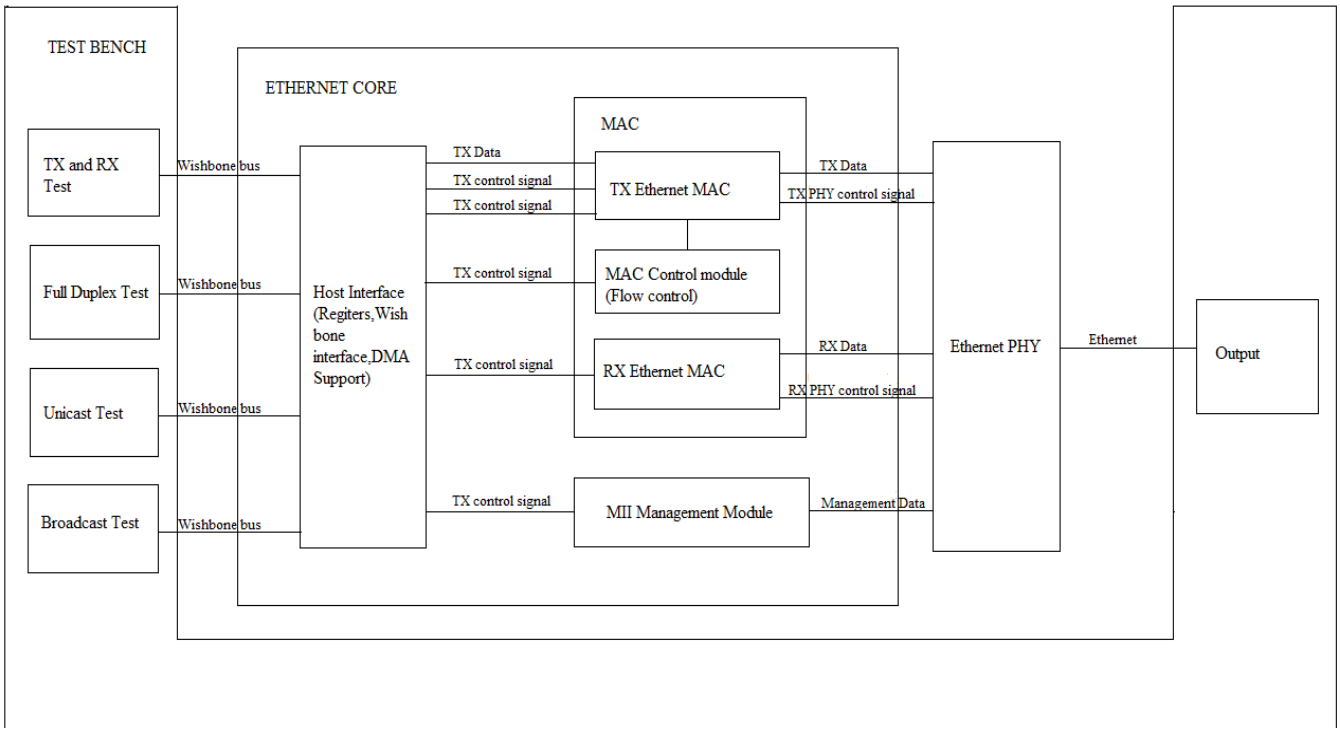

Fig 1 : Architecture of Ethernet IP core

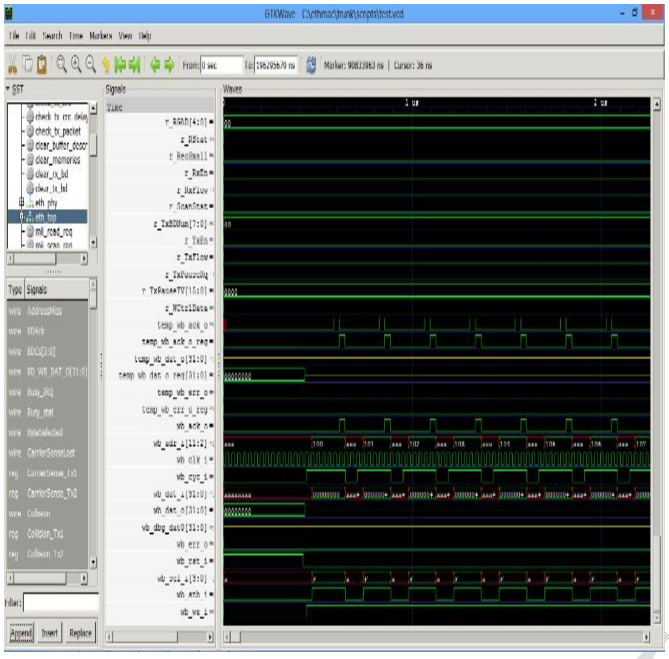

Fig 2 : Result

V. CONCLUSION

Ethernet is an easy, relatively inexpensive way to provide high performance networking to all different types of computer equipment. Ethernet provides transmission speed up to 100 megabits per second. Functional verification of Ethernet contains the description of verification platform using Verilog for the design under test. Verilog coding is used to integrate with the test bench. These test cases concentrate on verification of concurrent, coherent and multi-threaded capabilities and are designed to run efficiently in any hardware platform. It is then integrated with Device Under Test (DUT) and hence the verification will be done. In future enhancement in testing and new method called graph can be implemented.

References:

[1] W. Hermas, Approach to Coverage-Driven Functional Verification of Complex Multimillion Gate ASICs, M.A.Sc. thesis, School of Information Technology and Engineering, University of Ottawa, Ottawa, ON, Canada, 2006.

[2] J. Bergeron, Writing Testbenches: Functional Verification of HDL Models. New York: Springer, 2003.

[3] L. Bening and H. Foster, Principles of Verifiable RTL Design: A Functional Coding Style Supporting Verification Processes. Boston, MA: Kluwer, 2000 (2nd Edition).

[4] Verisity Design, Inc., "Coverage-driven functional verification", White paper, 2005 (available online: http://www.verisity.com /resources/ whitepaper/ coverage_ driven.html).

[5] J. Shen and J. A. Abraham, "Verification of processor microarchitectures", in Proc. VLSI Test Symp.,1999, pp. 189-194.

[6] A. Aharon, D. Goodman, M. Levinger, Y. Lichtenstein, Y. Malka, C. Metzger, M. Molcho, and G. Shurek, "Test program generation for functional verification of PowerPC processors in IBM", in Proc. Design Automation Conf., 1995, pp. 279-285.