International Journal of Scientific Engineering and Technology Research

ISSN 2319-8885
Vol.03,Issue.27
September-2014,
Pages:5438-5445

www.ijsetr.com

Including the Promiscuous mode in the design of MAC controller based on AXI bus and Verification with System Verilog

G. VEMAN[1], A. SATHEESH[2]

[1]PG Scholar, Vaagdevi College of Engineering, JNTUH, Warangal, Telangana, India.
[2]Assistant Professor, Vaagdevi College of Engineering, JNTUH, Warangal, Telangana, India.

Abstract: Integrated circuits have entered the era of System-on-a-Chip (SoC), which refers to integrating all components of a computer or other electronic system into a single chip. With the increasing design size, IP is an inevitable choice for SoC design. And the widespread use of all kinds of IPs has changed the nature of the design flow, making On-Chip Buses (OCB) essential to the design. This paper intends to design and verification of Ethernet MAC controller based on AXI bus, which is extended to include the concept of the Promiscuous mode, Which is used to monitor the media in networks and diagnose the connectivity issues in the networks. The main content of this project named as AMBA IO System Architecture design of configurable 10/100/1000Mbps data transfer rates. The Ethernet MAC controller of configurable10/100/1000Mbps supports the family of configurable PHY interfaces MII/GMII/RMII/RGMII to communicate with an external Gigabit/ Fast Ethernet PHY. This paper supports AXI 3.0 protocol from the ARM Advanced Microcontroller Bus Architecture (AMBA). AMBA bus protocol has become the main standard SoC bus. That means more and more existing IPs must be able to communicate with AMBA 3.0 bus. Based on AMBA 3.0 bus, this design is an Intellectual Property (IP) core of AXI (Advanced eXtensible Interface). The Verification of this AMBA IO System Architecture design is done by layered verification platform using System Verilog, it followed the methods of Direct random test, Constrained random test technologies to tape out the design successfully.

Keywords: SoC, Media Access control, AXI bus, MII.

I. INTRODUCTION

The Advanced Microcontroller Bus Architecture (AMBA) is an open-standard, on-chip interconnect specification for the connection and management of functional blocks in system (SoC) designs. It facilitates development of multi-processor designs with large numbers of controllers and peripherals. Since its inception, the scope of AMBA has despite its name gone far beyond microcontroller devices, and is now widely used on a range of ASIC and SoC parts including applications processors used in modern portable mobile devices like smart phones. AMBA is a registered trademark of ARM Ltd.AMBA was introduced by ARM in 1996. The first AMBA buses were Advanced System Bus (ASB) and Advanced Peripheral Bus (APB). In its second version, AMBA 2, ARM added AMBA High-performance Bus (AHB) that is a single clock-edge protocol. In 2003, ARM introduced the third generation, AMBA 3, including AXI to reach even higher performance interconnects and the Advanced Trace Bus (ATB) as part of the Core Sight on-chip debugs and trace solution. In 2010 the AMBA 4 specifications were introduced starting with AMBA 4 AXI4, then in 2011 extending system wide coherency with AMBA 4 ACE. In 2013 the AMBA 5 CHI (Coherent Hub Interface) specification was introduced, with a re-designed high-speed transport layer and features designed to reduce congestion. These protocols are today the de facto standard for 32-bit embedded processors because they are well documented and can be used without royalties. The AMBA specification defines an on-chip communications standard for designing high-performance embedded microcontrollers

- Advanced eXtensible Interface (AXI)
- Advanced High-performance Bus (AHB)
- Advanced System Bus (ASB)
- Advanced Peripheral Bus (APB)
- Advanced Trace Bus (ATB)

AXI, the next generation of AMBA interface defined in the AMBA 4.0 specification, is targeted at high performance, high clock frequency system designs and includes features which make it very suitable for high speed sub-micrometer interconnect separate address/control and data phases

- Support for unaligned data transfers using byte strobes
- Burst based transactions with only start address issued
- Issuing of multiple outstanding addresses
- Easy addition of register stages to provide timing Closure.

In this paper, We present a design and verification of a MAC controller based on AXI bus and Verification platform using the Layered architecture of System Verilog hardware description and verification language. The promiscuous mode is added to the design.

II. DUT DESCRIPTION

A. Project background

This paper introduced to design and verification of Ethernet MAC controller based on AXI bus, which is extended to include the concept of the Promiscuous mode, which is used to monitor the media in networks and diagnose the connectivity issues in the networks. This Architecture design of configurable 10/100/1000Mbps data transfer rates. The Ethernet MAC controller of configurable 10/100/1000Mbps supports the family of configurable PHY interfaces MII/GMII/RMII/RGMII, these media independent interfaces works at different frequencies with different data rates. The Advanced Microcontroller Bus Architecture (AMBA) AXI 3.0 bus protocol is used in this paper. This is the most essential on chip bus(OCB) architecture. The verification platform using the System Verilog hardware description and verification language by direct random test and constrained randomization verification technologies.

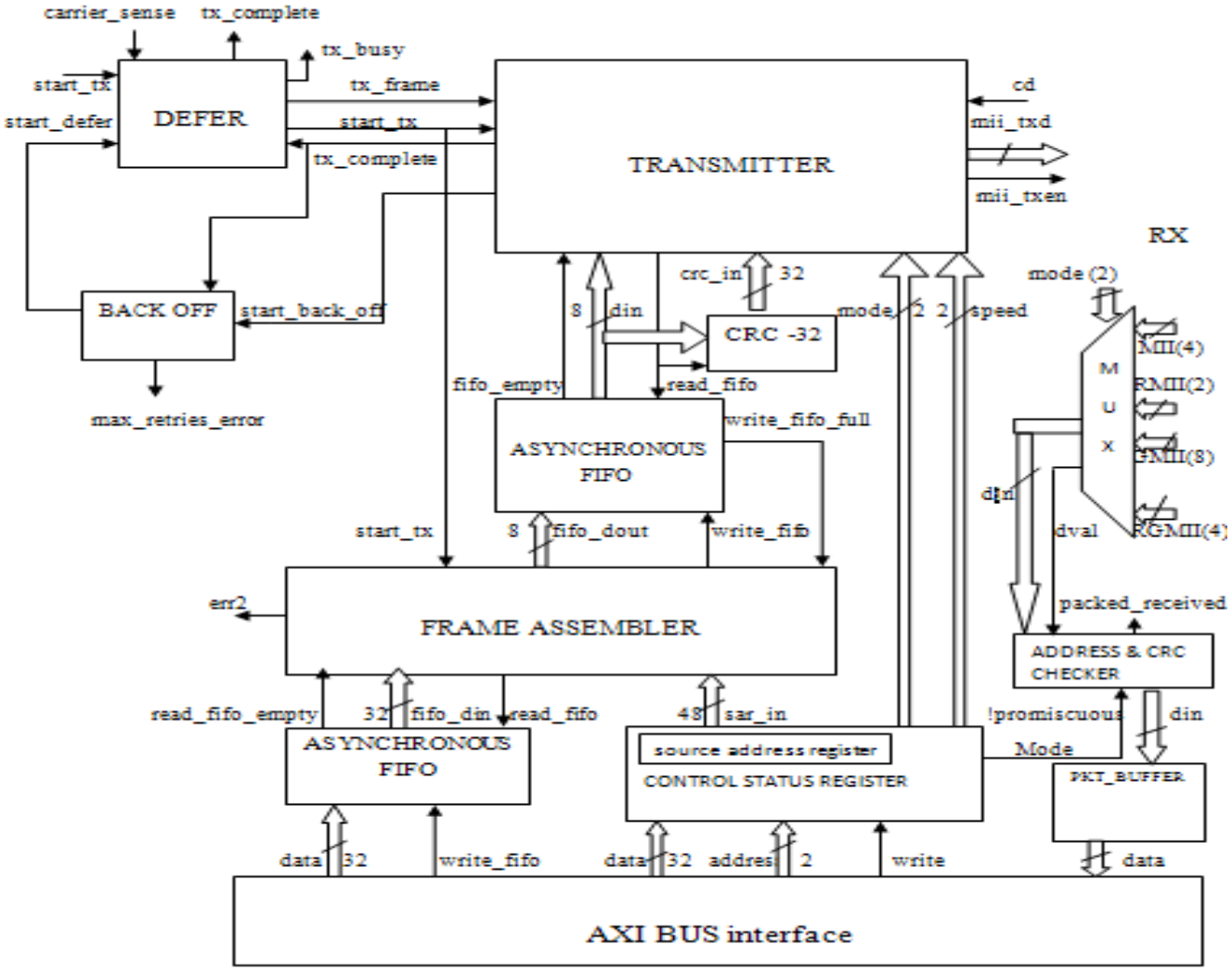

Fig1: MAC controller module with AMBA IO system Architecture.

The figure 1 shows that the basic MAC controller module with AMBA IO System architecture structure. The basic function of the MAC controller is to check the status of the channel before transmitting the data on to the media to the destination address. If the channel is free then start the transmission otherwise it waits for the time to get access to the channel. In this MAC controller we use Carrier Sense Multiple Access with collision detection(CSMA/CD) protocol. Before this we have the several protocols like pure aloha, slotted aloha, and CSMA protocols are used. But the efficiency of this protocols are less compared to the CSMA/CD protocol. It follows the algorithm Binary Exponential Back off to get the access of the channel if the collision occur more times. The various modules like defer, back off, transmitter, frame assembler, asynchronous FIFO (first in first out),multiplexor, control status register(CSR), packet buffer, address and CRC checker and AMBA AXI 3.0 bus interface modules are connected each other. The function of the each block as follows. Defer block monitors the carrier sense (CS) signal provided by the physical layer and whenever the medium is busy, it defers the transmission. On receiving the strt_tx from the upper layer (LLC) this block makes the 'tx_busy' signal and starts the process of monitoring the channel for 'carrier sense' this process is called 'DEFER' .The signal 'carrier sense' is provided by the physical layer .It monitors the channel for inter frame gap

period, which is 96 bit period .This period is split up into two different time slots 60 bit period and 36-bit period. During the 60 bit time period is elapsed, the transmitter does not monitor 'carrier sense' as active then the timer is restarted .after 60 bit time period is elapsed,the transmitter does not monitor 'carrier sense' for next 36 bit period and gives the signal 'tx_frame' once the transmission is started it waits for tx_complete or start_defer to be asserted and goes to start of defer when either is asserted.

The function of back off block is, whenever transmission is aborted up on collision, his block scheduled retransmission. The scheduled is determined by a controlled randomization process called "Truncated Binary Exponentially Back off". when a transmission attempt has been terminated due to collision ,it is retried by the transmitter until it is successful or a maximum no of attempt (16) have been made the scheduling of the transmission is determined by controlled randomization process known as "Truncated Binary Exponentially Back Off " after the end of enforcing a collision (jamming) the transmitter delays before attempting to retransmit the frame .The delays is an integer multiple of slot time .The number of slot times to delay before the nth retransmission attempt is chosen as a uniformly distributed random integer r in the range

$0<=r<= 2^k$

Where

$k=min (n, 10)$

If all attempt limit attempt fail, this event is reported as an error.

It basically controls the operation of back off .whenever start_back_off is received the state machine shifts a 1 in mask register .the output of mask register is "anded" with the random number generator output .This resulted is loaded in slot counter .The state machine waits for these many slot periods and the end of counting generates start defer, thus defer process is started for next transmission attempts .After reception of tx_complete or after maximum attempt limit of 16 is reached the mask register is reset to all zeros. Frame assembler block sets value of the fields of CSMA/CD MAC frame the LLC provides these values. padding is taken care of the frame size is less than 46 bytes. The function of frame assembler is to assembler different components of the frame, viz destination address, source address and data, and supply this to the transmitter, as well as the CRC block, hence the frame assembles all the fields over which FCS is determined. The frame assembler block is controlled by the transmitter block through the strt_tx signal. The signal is low when the transmitter is idle .The strt_tx signal functions as an enable for frame assembler block. The 802.3 standards specify a minimum frame size of 64bytes .This implies a minimum data of 46 bytes. If the length field indicates a length less than 46, the frame assembler pads extra bytes At the end of valid data bytes to complete a data field size of 46 bytes. The pad bytes contain all 10101010's in this implementation.

There is also a maximum data field limit of 1500 bytes. If the length passed to the frame assembler violates this limit, it generates an error condition ERR2, indicates a frame overflow. In this situation, it sends 1500 bytes of data .at end of the data frame assembler, the frame assembler issues an EOF signal to the transmitter, which indicates the end of frame. On receiving start transmit active from defer block this block starts transmitting several bits at a time depending on the media independent interfaces. At the same time it gives signal transmit valid (mii_txen) to the physical layer. First it transmits 7- bytes of preamble then 1 byte of start frame delimiter(SFD) is transmitter and it also gives read fifo signal to asynchronous fifo block and CRC block after which it accepts 8-bits of data from fifo .Then it transmits 32- bits of CRC and gives the signal transmit complete to defer block and de asserts the signal mii_txen and' 'fifo read' .Since CRC block works on bytes , 'frame assembler' gives 8 –bits of data at the output .If the collision occurs in the channel then cd becomes 1 and it asserts start back off and desserts fifo read signals. This block also monitors the signal 'Collision Detection'(CD) provided by the physical layer if it detects Collision detect during transmitting preamble then it completed transmitting preamble and then it transmits 4 bytes of JAM sequence. If collision is detected any where ever else then preamble then 'transmitter' stops transmitting and sends JAM sequence. It also asserts the signal start back off.

A cycle redundancy check (CRC) is used by the transmit and receive algorithms to generate a CRC value for the FCS field .The frame check sequence (FCS) field contains a 4-octet C RC value .This value is computed as a function of the contents of the source address, destination address, length LLC data and pad .The encoding is defined by the following generating polynomial: X^32+X^26+X^23+X^22+X^16+ X^12+X^11+X^10+X^8+X^7+X^5+X^4+X^2+X^1+1. In the receiver section one multiplexor, packet buffer, address and crc checker. Here the promiscuous mode signal is active low. This signal is used to put the receiver in the promiscuous mode. Actually in the promiscuous mode it receives the all the frames which has different destination addresses to monitor the channel status. To rise the connectivity issues like status of the channel. So it can diagnose the connectivity issues. Control status register will give the control signals like mode, speed.etc. This protocol supports the configurable MII/RMII/GMII/RGMII PHY interfaces.

B. AXI bus

AXI is part of ARM AMBA, a family of micro controller buses first introduced in 1996. The first version of AXI was first included in AMBA 3.0, released in 2003 provides the procedure for obtaining the ARM specification. Consult those specifications for the complete details on AXI operation. The AXI specifications describe an interface between a single AXI master and a single AXI slave, representing IP cores that exchange information with each other. Memory mapped AXI masters and slaves can be

connected together using a structure called an Interconnect block. The Xilinx AXI Interconnect IP contains AXI-compliant master and slave interfaces, and can be used to route transactions between one or more AXI masters and slaves. The AXI Interconnect IP.

Table1: Differences among the different PHY interfaces

Mode	Work Frequency	Data width	Rate
MII	25MHz/2.5MHz	4	10/100Mbps
GMII	125MHZ	8	1000Mbps
RMII	50MHZ	2	10/100Mbps
RGMII	125/25/2.5MHz	4	10/100/1000Mbps

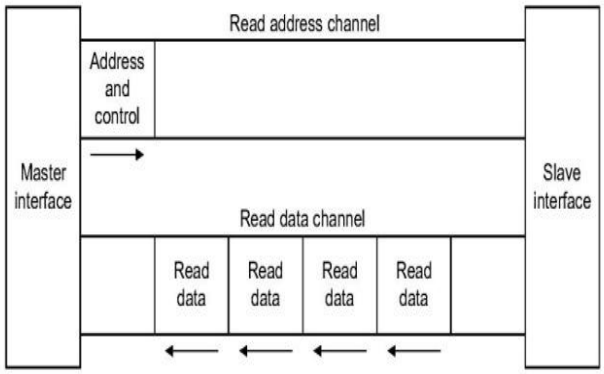

Fig2: Channel architecture of read transaction.

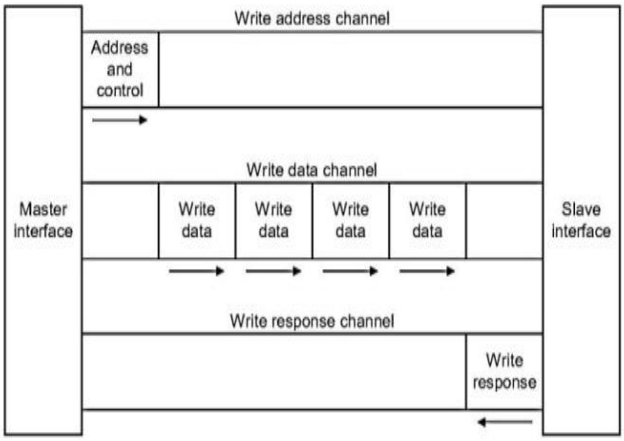

Fig3: Channel architecture of write transaction.

Influential On-Chip-Bus structure mainly comprises AMBA bus from ARM corp.[2], Core Connect bus from IBM corp.[3], Wishbone bus from Silicore corp.[4], Avalon bus from Altera corp.[5], Micro Networks from Silicon corp.[6] , Palm bus from Palm corp.[7] OCP bus from OCP-IP[8], etc. On-chip-bus IPs in China include L*bus from Chinese Academy of Science, C*BUS from SUZHOU GUOXIN corp., etc. Every On-chip-bus is developed for special application, each has his strong point, and it is difficult to compare with their advantages and disadvantages. AMBA bus own many third party supporters, is used in SoC system based on ARM core by 90% cooperation of ARM corp. ,it is becoming one of On-Chip-Bus industry standards gradually. AXI AMBA Advanced extensible Interface Bus protocol is important content of AMBA 3.0, which is designed for high- performance, high-frequency system designs, and includes a number of features that make it suitable for a high-speed submicron interconnect. So, AXI bus is selected in our SOC design.

The AXI bus protocol is an enhanced bus protocol of the existing Advanced High-performance Bus (AHB). AXI is targeted at high performance, high-frequency system designs. AXI protocol has five independent unidirectional channels that carry the address/control and data. Each channel uses a two-way valid and ready handshake mechanism. The five independents channels are the Address Read (shortening, AR) channel, Address Write (AW) channel, Read Data (RD) channel, Write Data (WD) channel, and write response channel (B). AW and AR channel convey the address and control of write and read transactions. The control signals of such channels describe the nature of the read and write transactions. A transaction can be a burst of a different length, or it can be atomic. A burst is composed of a number of transfers whose length is defined. The data is transferred between master and slave port, and vice versa using WD and RD channels respectively. Write response channel (B) allows a salve to signal completion of the write transaction or an error. One of the features of AXI bus is a burst transaction with only the start address issued. The split transaction AXI protocol enables out-of-order transaction completion; it provides a "transaction ID" field assigned to each transaction. Transactions from the same master port, but with different IDs have no ordering restriction while transactions with the same ID must be completed in order. The AXI protocol enables address information to be issued ahead of the actual data transfer, supports for multiple outstanding transactions ' supports for out-of-order completion of transactions. Out-of-order transactions completion improves system performance ' it allows a complex slave like memory return data out-of order. Out of order execution and interleaving are the two main features of AXI bus that provide high throughput.

C. Verification Platform

In the design verification role, System Verilog is widely used in the chip-design industry. The three largest EDA vendors (Cadence, Mentor and Synopsys) have incorporated System Verilog into their mixed-language HDL-simulators. Although no simulator can yet claim support for the entire System Verilog LRM, making testbench interoperability a challenge, efforts to promote cross-vendor compatibility are underway. In 2008, Cadence and Mentor released the Open Verification Methodology, an open-source class-library and

usage-framework to facilitate the development of re-usable test benches and canned verification-IP. Synopsys, which had been the first to publish a System Verilog class-library (VMM), subsequently responded by opening its proprietary VMM to the general public. Many third-party providers have announced or already released System Verilog verification IP. The Figure 4 shows the verification flat for for MAC controller module design with AMBA IO system architecture. Here DUT is nothing but our design MAC controller based on AXI bus. The Interface module is used to interface the all modules in the verification environment. Generator generates test cases and then driver drives to the design. Checker will check the outputs.

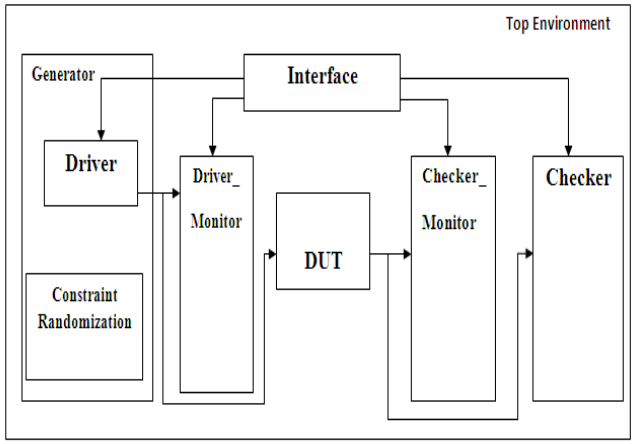

Fig4: Verification flat form.

Environment contains the instances of the entire verification component and Component connectivity is also done. Steps required for execution of each component is done in this. When building a verification environment, the verification engineer often starts by modeling the device input stimulus. In Verilog, the verification engineer is limited in how to model this stimulus because of the lack of high-level data structures. Typically, the verification engineer will create a array/memory to store the stimuli. SystemVerilog provides high-level data structures and the notion of dynamic data types for modeling stimulus. Using SystemVerilog randomization, stimulus is generated automatically. Stimulus is also processed in other verification components. System Verilog high-level data structures helps in storing and processing of stimulus in an efficient way.

The generator component generates stimulus which are sent to DUT by driver. Stimulus generation is modeled to generate the stimulus based on the specification. For simple memory stimulus generator generates read, write operations, address and data to be stored in the address if its write operation. Scenarios like generate alternate read/write operations are specified in scenario generator. System Verilog provided construct to control the random generation distribution and order. Constraints defined in stimulus are combinatioural in nature where as constraints defined in stimulus generators are sequential in nature. Stimulus generation can be directed or directed random or automatic and user should have proper controllability from test case. It should also consider the generation of stimulus which depends on the state of the DUT for example, Generating read cycle as soon as interrupt is seen. Error injection is a mechanism in which the DUT is verified by sending error input stimulus. Generally it is also taken care in this module.

Generally generator should be able to generate every possible scenario and the user should be able to control the generation from directed and directed random test cases. The drivers translate the operations produced by the generator into the actual inputs for the design under verification. Generators create inputs at a high level of abstraction namely, as transactions like read write operation. The drivers convert this input into actual design inputs, as defined in the specification of the designs interface. If the generator generates read operation, then read task is called, in that, the DUT input pin "read_write" is asserted. Monitor reports the protocol violation and identifies all the transactions. Monitors are two types, Passive and active. Passive monitors do not drive any signals. Active monitors can drive the DUT signals. Sometimes this is also refered as receiver. Monitor converts the state of the design and its outputs to a transaction abstraction level so it can be stored in a 'score-boards' database to be checked later on. Monitor converts the pin level activities in to high level. The monitor only monitors the interface protocol. It doesn't check the whether the data is same as expected data or not, as interface has nothing to do with the date. Checker converts the low level data to high level data and validated the data. This operation of converting low level data to high level data is called Unpacking which is reverse of packing operation. For example, if data is collected from all the commands of the burst operation and then the data is converted in to raw data , and all the sub fields information are extracted from the data and compared against the expected values. The comparison state is sent to scoreboard.

III. COVERAGE

Coverage as applied to hardware verification languages refers to the collection of statistics based on sampling events within the simulation. Coverage is used to determine when the device under test (DUT) has been exposed to a sufficient variety of stimuli that there is a high confidence that the DUT is functioning correctly. Note that this differs from code coverage which instruments the design code to ensure that all lines of code in the design have been executed. Functional coverage ensures that all desired corner cases in the design space have been explored. A System Verilog coverage group creates a database of "bins" that store a histogram of values of an associated variable. Cross-coverage can also be defined, which creates a histogram representing the Cartesian cross-product of multiple variables

```
class eth_frame;
  // Definitions as above
  covergroup cov;
```

```
    coverpoint dest {
       bins bcast[1] = {48'hFFFFFFFFFFFF};
       bins ucast[1] = default;
    }
    coverpoint type {
       bins length[16] = { [0:1535] };
       bins typed[16] = { [1536:32767] };
       bins other[1] = default;
    }
    psize: coverpoint payload.size {
       bins size[] = { 46, [47:63], 64, [65:511], [512:1023],
[1024:1499], 1500 };
    }
     sz_x_t: cross type, psize;
  endgroup
endclas
```

- Constrained random generation
- Integer quantities, defined either in a class definition or as stand-alone variables in some lexical scope, can be assigned random values based on a set of constraints. This feature is useful for creating randomized scenarios for verification.
- Within class definitions, the rand and randc modifiers signal variables that are to undergo randomization. randc specifies permutation-based randomization, where a variable will take on all possible values once before any value is repeated. Variables without modifiers are not randomized.

```
class eth_frame;
   rand bit [47:0] dest;
   rand bit [47:0] src;
   rand bit [15:0] type;
   rand byte       payload[];
   bit [31:0]      fcs;
   rand bit [31:0] fcs_corrupt;

   constraint basic {
      payload.size inside {[46:1500]};
   }

   constraint good_fr {
      fcs_corrupt == 0;
   }
Endclass
```

In this code, the fcs field is not randomized; in practice it will be computed with a CRC generator, and the fcs_corrupt field used to corrupt it to inject FCS errors. The two constraints shown are applicable to conforming Ethernet frames. Constraints may be selectively enabled; this feature would be required in the example above to generate corrupt frames. Constraints may be arbitrarily complex, involving interrelationships among variables, implications, and iteration. The SystemVerilog constraint solver is required to find a solution if one exists, but makes no guarantees as to the time it will require to do so.In the directed random test method unlike constraint randomization it tries to test the all test cases

IV. RESULTS OF THE DESIGN

The following figure 4 shows the input to the MAC design. The data could be drive into fifo first. Then according to length field data is sent.

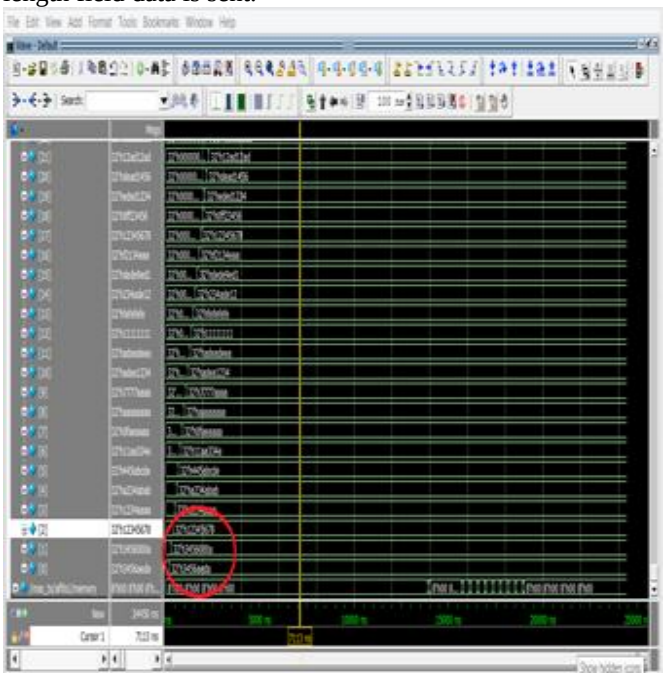

Fig4: Input data.

The figure 5 shows that the output of the MAC

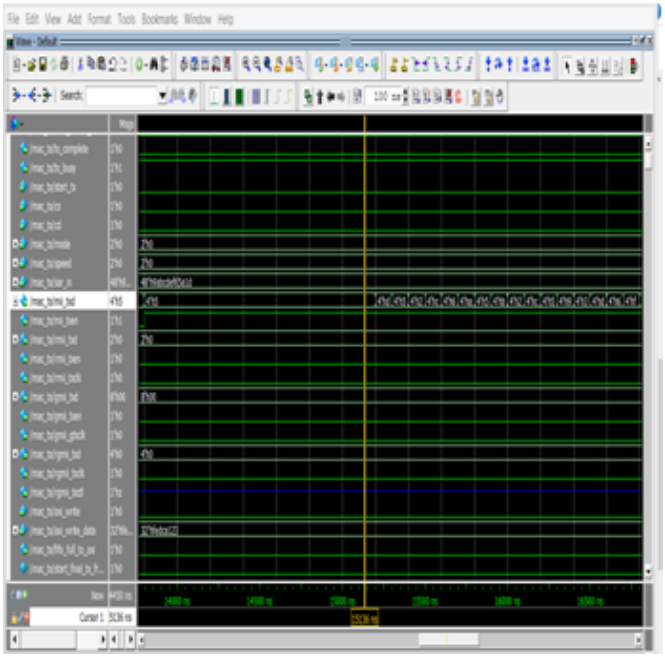

Fig5: output data

The figure 6 shows the padding and crc values. Also this waveforms shows that the configurable MII/RMII/GMII/RGMII PHY interfaces as shown in fig 5 and fig 6. So this design support RMII along with remaining PHY interfaces.

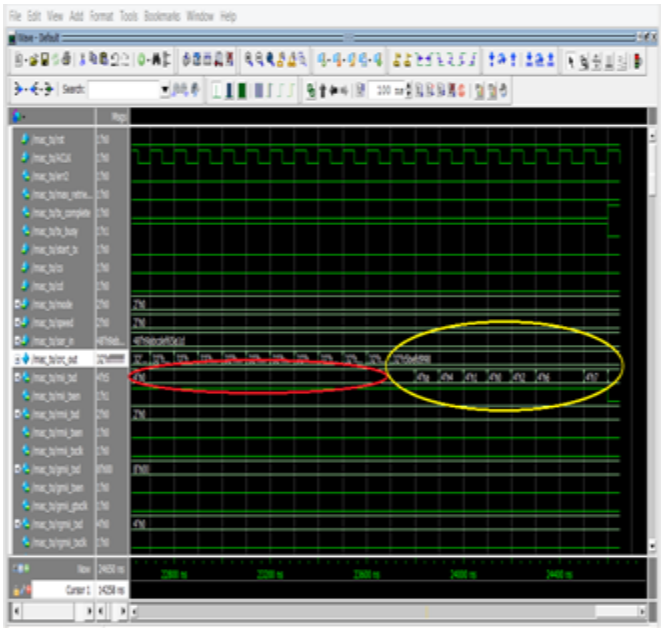

Fig6: Padding and crc

The Figure 7 shows the receiver section shown the promiscuous mode to monitor the media.

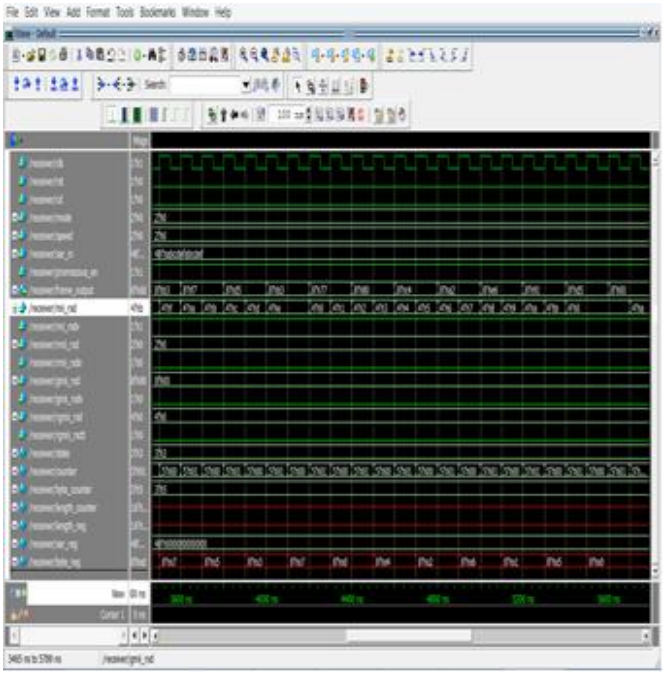

Fig 7: Promiscuous mode in the receiver section.

V. CONCLUSION

This project presents a design and verification of configurable 10/100/1000Mbps Ethernet MAC controller based on AXI bus including the promiscuous mode in the receiver section to receive all the frames in order to monitor the channel and diagnose the connectivity issues of the media. This design can support the RMII along with MII/GMII/RGMII PHY interfaces. The verification is done by System Verilog hardware description and verification language.

VI. REFERENCES

[1]D.Wingard. Micro Network-based integration for SOCs [A]. In: Design Automation Conference, 2001. Proceedings [C] .2001:673-677.

[2]B.Cordan, An efficient bus architecture for system on-chip design[A].In: Custom Integrated Circuits, Proceedings of the IEEE 1999[C].1999:623-626.

[3] Open Core protocol specification[S/OL].: http://lad.dsc.ufcg.edu.br/ip/OCP-IP-Open Core Protocol Specification-1.0.pdf.

[4] Yang tao, Study on Key Technology of 10M/100M/1000M Adaptive.

[5] MAC controller[Master thesis], Chinese Academy of Institute of Computing Technology, 2008,12. (in Chinese).

[6] Zhou yanchao, Research on Design of On-chip Bus and IP core Integration Technology of the SOC platform [master thesis] Northwestern Polytechnical University, 2004,12.(in Chinese).

[7] ARM, Inc. AMBA® AXI Protocol Version:2.0 Specification[S].2003.

[8] Haalen R, Malhotra R, de-Heer A. Optimized routing for providing Ethernet LAN services[J].IEEE Communications Magazine,2005,43(11):158-164.

[9] Koopman P. Chakravarty T. Cyclic redundancy code (CRC) polynomial selection for embedded networks [J].Dependable Systems and Networks,2004,7:145-154.

[10] Janick Bcrgeron Eduard Ceny Alan Hunter et a1 Verification Methodology Manual for System Verilog[M] New York Springer Science 2005.

[11] Introduction to Design Verification with VMM: A Quickstart Guide, Synopsys, Inc.2008.

[12] Introduction to Design verification with VMM: A Quickstart Guide, Symopsys, Inc.2008.

[13] System Verilog Assertions Checker Library Quick Reference Synopsys 2008.

[14] Jerraya, W. Wolf. Multiprocessor Systems-on-Chips [M], 2007.

[15] D.Flynn,AMBA:enabling reusable on-chip designs[J]. IEEE Micro Magazine, July- Aug.1997.Volume:17.

[16] R.Hofman and B.Drerup.Next generation Core Connect processor local bus archit- echerur [A], Annual IEEE International ASIC/SOC Conference[C],2002: 221-225.

[17] Silicore.http://www.silicore.net.

[18]http://www.altera.com/products/ip/iup/ethernet/mmtip10/100/1000mbps_fulldup_ethermac.html. (Accessed: 20 April 2014).

[19]http://www.dcs.gla.ac.uk/~ross/Ethernet/1000MbpsAbove.html (Accessed: 30 April 2014).

[20]www.dcs.gla.ac.uk/~ross/Ethernet/protocol.html (Accessed: 3 May 2014).

[21] http://en.wikipedia.org/wiki/IEEE_802.3. (Accessed: 10 May 2014).

[22] http://en.wikipedia.org/wiki/Media_access_control (Accessed: 2 June 2014).

[23]http://www.synopsys.com/SystemVerilog/verification.html. (Accessed: 22 May 2014).

[24]http://en.wikipedia.org/wiki/Advanced_Microcontroller_Bus_Architecture.(Accessed: 22 May 2014).

[25]http://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/v13_4/ug761_axi_reference_guide.pdf (Accessed: 22 June 2014).
[26]http://en.wikipedia.org/wiki/Media_Independent_Interface. (Accessed: 22 May 2014).
[27]http://www.intel.in/content/www/in/en/ethernetcontrollers/ ethernet-controllers.html (Accessed: 5 July 2014).
[28]http://en.wikipedia.org/wiki/SystemVerilog#IEEE_Standard_Reference (Accessed: 12 July 2014).
[29]https://moodle.polymtl.ca/pluginfile.php/80905/course/section/16621/Complement_Randomisation.pdf . (Accessed: 2 August 2014).
[30]http://www.synopsys.com/Tools/Verification/Documents/assertion_based_wp.pdf (Accessed:12 Aug2014).

Author's Profile:

Mr. G.VEMAN received B.Tech degree in ECE from JNTUH in 2012 , pursuing M.Tech (2012-2014) in the stream of VLSI at Vaagdevi College of Engineering, (Affiliated to JNTUH) Hyderabad, India.

Mr. A.SATHEESH received M.Tech degree in ES from JNTUH , B.Tech degree in ECE from JNTUH. Presently working as Sr. Assistance professor in Vaagdevi College of engineering, Warangal, India.