International Journal on Emerging Technologies 11(1): 396-402(2020)

ISSN No. (Print): 0975-8364
ISSN No. (Online): 2249-3255

# Design of AMBA AXI4-Lite for Effective Read/Write Transactions with a Customized Memory

**A. Sainath Chaithanya[1], Sameera Sulthana[2], B. Yamuna[2] and Ch Haritha[2]**
[1]Assistant Professor, Department of ECE, RGUKT-Basar (Telangana), India.
[2]B.Tech Student, Department of ECE, RGUKT-Basar (Telangana), India.

(Corresponding author: Sainath Chaithanya)
(Received 04 November 2019, Revised 27 December 2019, Accepted 02 January 2020)
(Published by Research Trend, Website: www.researchtrend.net)

**ABSTRACT: The mean for communication is to convey respective ideas and intentions which should be a duplex to be effective; the same goes for the electronics world where we find various types of interfaces among electronic devices. Communication protocol which entails the transformation of data within or out of a system, especially in System-on-Chip (SoC) boards wrapping different subsystems like microprocessors, memories, bus architectures, is always considered as one of the biggest challenges for engineers to elect a right bus protocol for the application. To a great extent, AMBA buses developed by ARM played a prominent role as interconnect for functional blocks on chipboard.**

**The three generations of AMBA -APB, ASB/AHB, AXI had their eminent part in providing effective transactions among peripherals residing on-chip, released its next-generation bus protocol AMBA4.0-AXI4 a memory-mapped protocol and its variant AXI4-Lite for simple memory peripherals. Although AXI furnishes dynamic connectivity among master devices like Processors, Controllers with their Slave devices i.e., on-chip memories or Memory systems like SRAM, ROM, FIFO registers in AMBA IO Systems, during which interconnect may be developed for Single/ Multiple Master or Slaves as per requirements.**

**The motto of this paper is to design and interfacing AXI4-Lite Master with a Customized Memory slave by adopting a set of five different channels from AXI4 for scheming a simple interconnect between Master and Slave peripherals in performing effective READ/WRITE transactions. The design is progressed in Verilog HDL where functionality checking is done by ISIM simulator and synthesized using Xilinx ISE 14.4 tool.**

**Keywords:** AMBA AXI, AXI4Lite, Bus protocol, Burst, Interconnect, SoC.

**Abbreviations:** AMBA, Advanced Microcontroller Bus Architecture; APB, Advanced Peripheral Bus; ASB, Advanced system bus; AHB, Advanced high-performance Bus; AXI, Advanced Extensible interface; SoC, System on Chip.

## I. INTRODUCTION

SoC is not only an integration of chip, software, and different components but also how they interconnect for realizing a chipset. Due to its low power consumption, these are significant in embedded systems, VLSI IC realizations, accommodating Control Units – micro/Digital signal processors, controllers, functional blocks-on-chip Memories, DMA, Timing Units-Oscillators, and PLLs. All these subsystems housed on a chip are interacted by Bus a Single set of wires used to connect multiple subsystems [17].

The Bus which acts as a communication bottleneck relies on Bus Protocols [3] that determines features like clock timings, signaling lists, allocating the resources, types of transactions, addressing and Arbitration schemes for achieving on-chip communication. Bus Protocols like AMBA, Avalon Bus, IBM Core Connect, IDT IP Bus, Open Core Protocol; Wishbone, etc. among which AMBA protocols got widely accepted by the SoC designers [14].

AMBA, a system bus released in 1996 by ARM Ltd. is an open standard on-chip communication standard bus protocol for connection and management of functional blocks residing as System-on-Chip (SoC), have its generations depicted in Fig. 1 [1].

| AMBA1 | AMBA2 | AMBA3 | AMBA4 |
|---|---|---|---|
| | | | ACE |
| | | | ACE-Lite |
| | | | AXI4 |
| | | AXI3 | AXI4-Lite |
| | | ATB | AXI4-Stream |
| | AHB | AHB-Lite | |
| ASB | | | |
| APB | APB2 | APB3 | APB4 |
| **1995** | **1999** | **2003** | **2010** |

**Time**

**Fig. 1.** Generations of AMBA.

The Advanced Peripheral Bus is a simple non-pipelined bus protocol recommended for connecting low bandwidth peripherals i.e. keyboard, UART, etc. characterized for minimal power consumption and reduced complexity.

The Advanced system bus designated for pipelined operations [9] and burst transfers accepting multiple bus masters for connecting with other high-performance blocks.

The Advanced high-performance bus dedicated to high-performance designs which carries numerous bus master agents of high clock frequencies outfitted for burst and split transfers for wider data (64/128 bits) bus configurations.

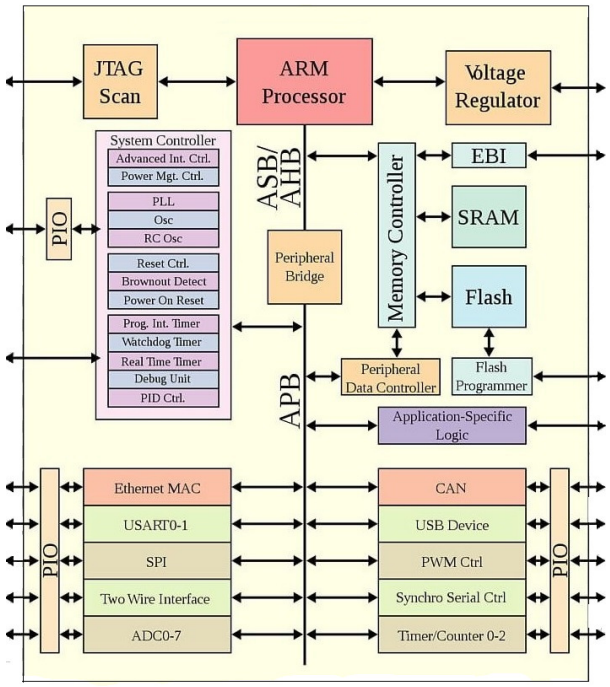


**Fig. 2.** AMBA system bus in SoC [17].

Very popular AHB or ASB slaves are peripheral Bridge, APB or any memory interfaces. However some low-bandwidth peripherals will rest on the APB side, A Bridge regularly engaged for connecting the low bandwidth designs with a high-performance system bus as shown in Fig. 2.

Fig. 3 below exhibits hypothetical SoC's holding functional blocks that are interconnected using a flavor of an Advanced Microcontroller Bus Architecture (AMBA) [2], supporting single-cycle data transfers along with multiple masters for bus arbitration.

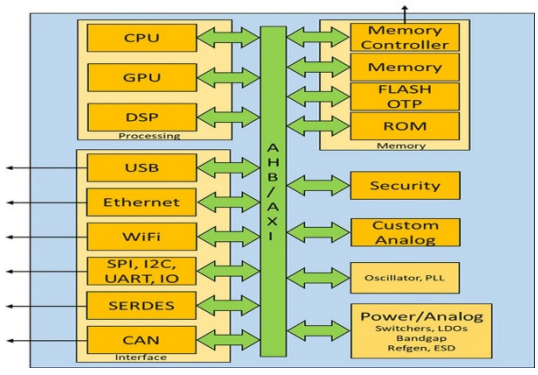


**Fig. 3.** Typical SoC [13].

The recent period of AMBA - AXI, The Advanced Extensible interface a point to point interconnect offers services for high bandwidth and low latencies, succeeding the constraints of a shared bus protocol in terms of the number of agents. Improvement from AHB in offering multiple outstanding data transfers with separate read and write paths of different bus widths, pipe-lined burst data transfers [16].

## II. LITERATURE SURVEY

The discourse will be carried out on the critical works and contributions made by various researchers, on employment of on chip bus protocols as an interface for different applications continuing with, projecting the present work i.e. Design and interface of AXI4-Lite Master core with a customized Memory.

**Survey on AMBA Buses:** In System on Chip (SoC) design various components of computer or other electronic sub-systems are mounted onto a single chip. The Notorious Advanced Microcontroller Bus Architecture (AMBA) treated as an active agent in dispensing constructive framework in SoC designs, contributes for the digital glue that binds IP process together adequately. Some of the previous works which are relevant and carry basic information for the present work is depicted.

In the Design of AHB Based Memory Controller having main memories i.e., SRAM and ROM are treated as AHB slaves in multi- master and multi- slave communication model where the AMBA AHB is acting as central multiplexer interconnector, in which master's drive out the address and control signals indicating the type of transfer, arbiter takes care of granting bus to master, a decoder for selecting appropriate signals from the slave. In general master generates the data and control signals but these cannot directly communicate with any generic memory, hence data processed through slave then through slave interface, a FIFO is employed for data/control buffering for information to get preserved. Further data is communicated through RAM or ROM for read/write transactions.

The work also substantiates a memory controller responsible for all timings, commands, featured with a look ahead functionality where the arbitrator can notify upcoming commands, ready for execution [5].

Scheming of AMBA ASB APB Bridge, A 32 bit AMBA Bridge provides an interface between the ASB and the APB. The wait states are inserted for burst of read/write transfers where the ASB must wait for the signals of APB ultimately latencies are created. The bridge is meant to respond for transaction requests from the currently enabled ASB master

The ASB transactions are converted into APB transactions asynchronously as the APB peripherals do not need a clock input that is the APB access is timed with a strobe signal generated by the ASB to APB bridge interface [6].

The modern SoC's capes multi-core clusters, sophisticated peripherals for which the present AHB protocol supporting low complexity shared buses, cannot keep up with the requirements of today's high speed SoC, with certain more impediments in protocol i.e., the AHB has only one outstanding transaction, no full-duplex mode, single channel-shared bus which directed to employee AXI bus. A comparison is drawn with the advanced AXI bus more realistically, by determining how the five-channels in the bus operate independently along with handshaking concept [7].

Portraying the significance of AMBA AXI4 bus which is best in terms of throughput, latency, high-performances/frequencies systems is utilized with either single or multiple channels, where the interconnect block encapsulates the arbiter, decoder, and multiplexers.

The arbiter meant for monitoring the priority to access/release the bus to one of the multiple masters which initiates transactions simultaneously, by arbitration algorithm. The decoder decodes the address and control, sent by master and passes the transaction to sit in one out of 16 slaves for simple and burst mode read/write operations [8].

**Current task:** Due to the increased customer demands design complexity of system on chip (SOC) increases day by day. Hence there is always a productivity gap and right protocol is chosen for the respective applications. To Improve interconnect performance, Improve Quality of Service, Reduces wiring congestion migration is necessary from AXI4 to AXI4-Lite which allows the processor or masters to access the registers(small and mini peripherals) we prefer AXI4-Lite. And also taking the aid of above references and having an eye, a scenario arises where the Master agent frequently want to access information that might be in small for which it uses a simple registers like cache to interact, an effective and proper bus interface have to be picked for satisfying the need. This had motivated the work "Design and interface of AXI4-Lite Master core with a customized Memory" and been developed in Verilog HDL [15].

AXI4-Lite READ operation takes place from Slave to Master and the WRITE operation takes places from Master to Slave. Current task emphasizing design of interface for AXI4-Lite Master core with a customized Memory, the work is detailed in further sessions.

### III. AMBA AXI4

The Fourth generation AMBA 4.0, released in 2010, came up with two types of AXI4 interface which are namely AXI4 memory map (AXI4 or AXI4-lite) and AXI4-stream.

• AXI4— provides a burst of up to 256 data transfer cycles with just a single address phase.

• AXI4-Lite— is a light-weight, uncomplicated, low-throughput single-bit memory map transaction.

• AXI4-Stream— supports high-speed streaming data. It eliminates the requirement for an address phase and allows unlimited data burst size [18].

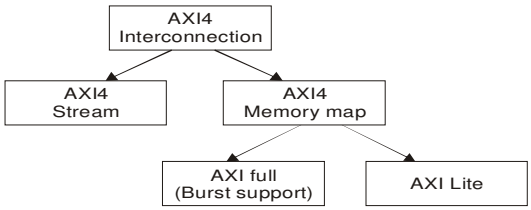


**Fig. 4.** AMBA AXI Categories.

AXI4-Lite specification includes a subset of AXI4 for communication with simple control registers whose Key functionalities are

— All transactions are of burst length 1 [11]
— All data accesses use the full width of the data bus
— AXI4-Lite supports a data bus width of 32/64-bit
— All accesses are Non-modifiable, Non-bufferable
— Exclusive accesses are not supported [10]
— AXI4-Lite supports multiple outstanding transactions [12], but a slave can restrict this by the appropriate use of the handshake signals

The current task "Design and interface of AXI4-Lite Master core with a customized Memory" in which READ operation of AXI4-Lite takes place from Slave to Master and the WRITE operation takes from Master to Slave i.e., a customized Memory file.

The AXI4 [4] specifications and the bus stipulate an interface between a single AXI master and a single AXI slave, familiarly called an *Interconnect* block as shown in Fig. 5, where IP cores exchange information.

Both AXI4 and AXI4-Lite interfaces wrap five distinct transaction channels in its architecture:

(i) Write Address Channel
(ii) Write Data Channel
(iii) Write Response Channel
(iv) Read Address Channel
(v) Read Data Channel.

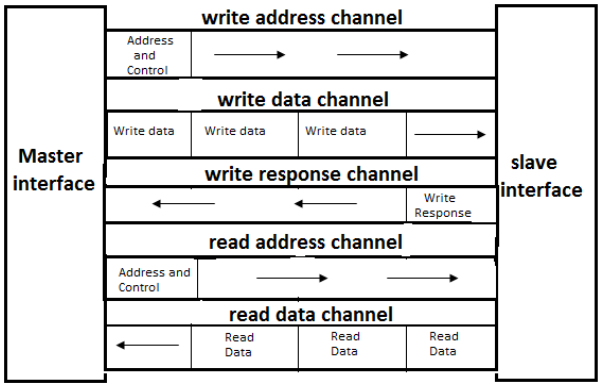


**Fig. 5.** Interconnect Architecture of AXI4-Lite.

As mentioned, the protocol includes five different channels that accommodate several sets of signals shown in Table 1 for facilitating the data transfers between the devices, the detailed description about the transaction channels with its associated signals can be found in reference [16].

**Table 1: Signals in AXI4-Lite.**

| | | | | |
|---|---|---|---|---|
| **Global** | ACLK | ARESETn | — | — |
| **Write address channel** | AWVALID | AWREADY | AWADDR | AWPROT |
| **Write data channel** | WVALID | WREADY | DATA | WSTRB |
| **Write response channel** | BVALID | BREADY | BRESP | — |
| **Read address channel** | ARVALID | ARREADY | ARADDR | ARPROT |
| **Read data channel** | RVALID | RREADY | RDATA | RRESP |

## IV. WORKING METHODOLOGY

AXI4-Lite Master creates instructions on the AXI4-Lite bus. The important tasks or functions carried out by the master are as follows:

— During write operation initially Master will send AWVALID Signal with logic high indicating that there is write transaction ready. It sends the particular address by using AWADDR signal to a customized Memory (16 - 32'bit registers) to where it wants to write the data, the address from the Master remains stable until the AWREADY is high. The number of transfers in a burst is given by the AWLEN signal. The AWSIZE highlights the size of each transaction, 32 bits, which has its own ID that will be sent through the AWID to the slave.

— Slave memory gives acknowledgment to Master by asserting AWREADY indicating that it accepts the address.

— The Master sends the data to a particular address in Memory using a data bus through the write data channel.

—When valid data is present in data bus then the WVALID signal goes high.

—WREADY signal goes high indicating that memory accepts the data.

—If the transaction is last in a burst then it is defined by WLAST signal. In AXI4-Lite it has one data transfer transaction per burst,

—In the response channel, there will be two signals BVALID and BREADY. BVALID signal will be high as a token for successful completion of data received during a write operation. The BREADY signal will be high when Master wants to accept valid signal which indicates the acknowledgment from the Master to the Customized slave Memory.

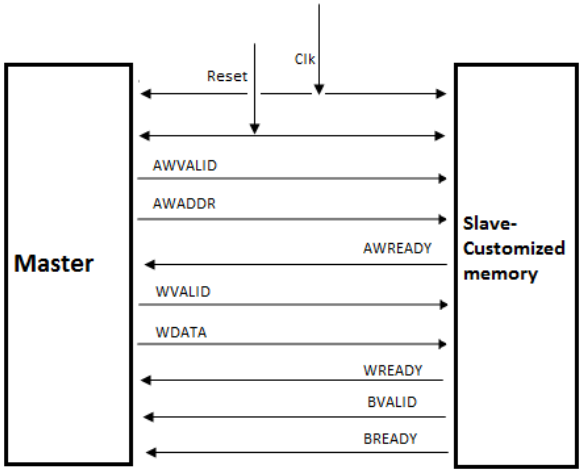


**Fig. 6.** AXI4-Lite architecture during write operation.

**AXI4-Lite Slave** is a simple Memory having 16 register locations of 32'bit data content, the reading operations happen from customized Memory to Master which in turn reading the data in memory. The important tasks or functions carried out by the slave and Master are as follows:

— During reading operation Master will send a specified address from where it wants to read the data in memory file/peripheral through ARADDR signal in read address channel, it also checks whether it is a valid address or not.

— If there is a valid address it Signals ARVALID will high. The address from the Master remains stable until the RREADY signal is high.

— Memory gives acknowledgment to Master by asserting ARREADY, indicating that it accepts the address, followed by placing the data on a data bus.

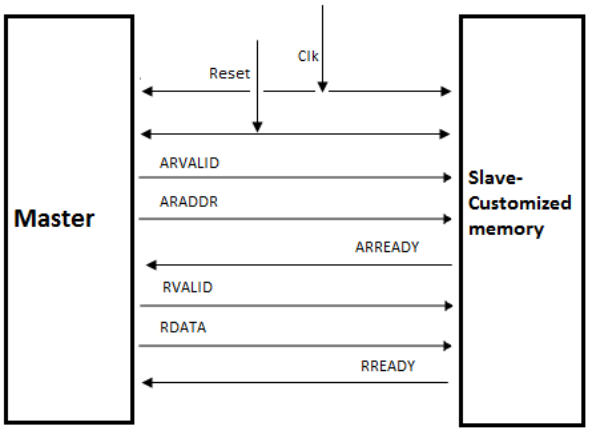


**Fig. 7.** AXI4-Lite architecture during read operation.

When valid data present in data bus then RVALID Signal goes high indicating that master accepts the data and reads the data which is present in the data bus

## V. RESULT ANALYSIS

The entire design is simulated by ISim Simulator. The conversion of RTL to Netlist synthesis is carried by Xilinx ISE 14.4 and is represented in the following Figs. 8, 9, 10 and 11.

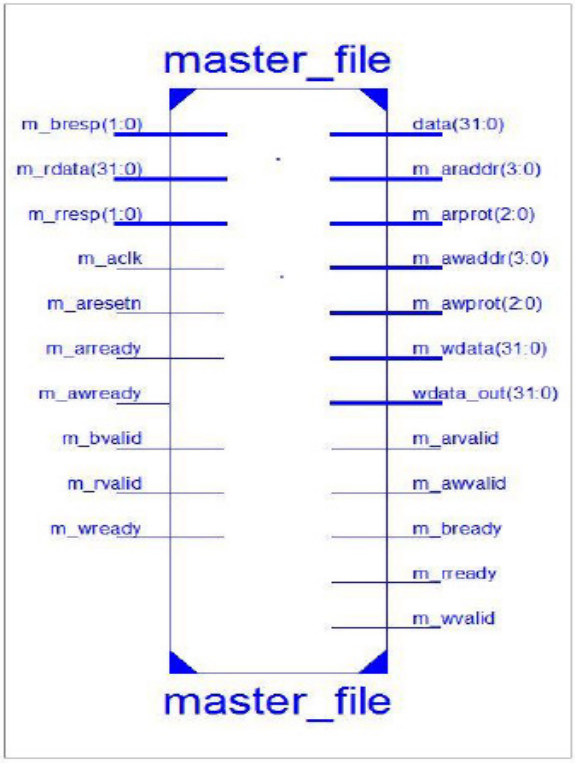


**Fig. 8.** Synthesis Diagram of AXI4-Lite Master.

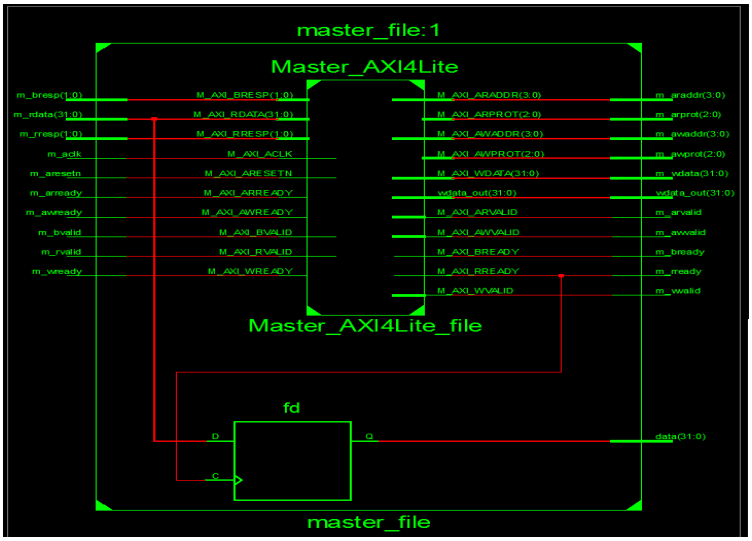


**Fig. 9.** Internal schematic of master.

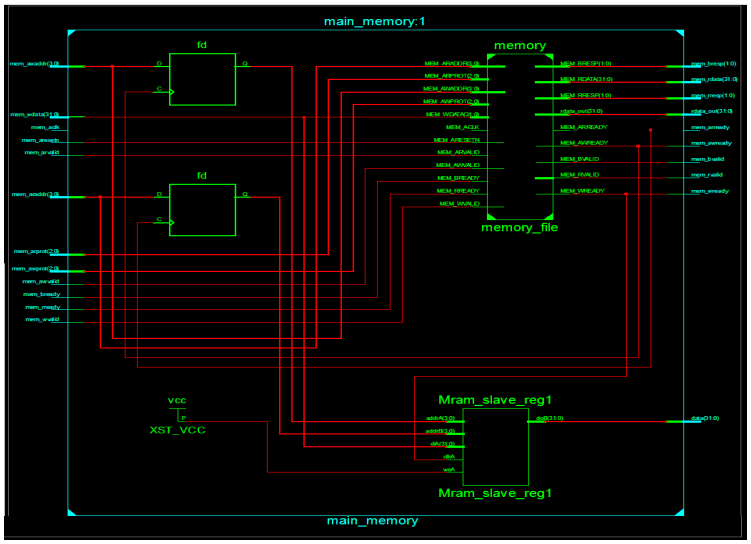


**Fig. 10.** Internal schematic of customized memory.

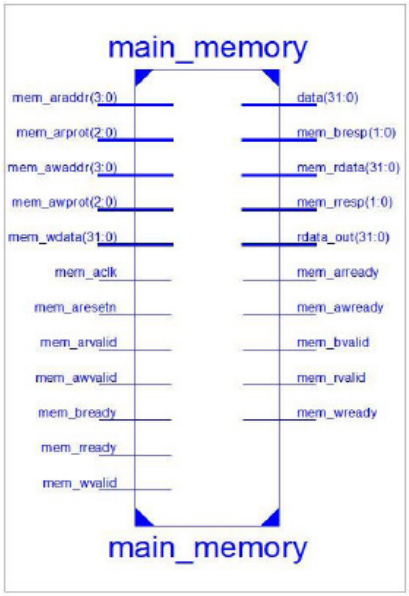


**Fig. 11.** AXI4-Lite Slave Synthesis diagram.

## VI. DISCUSSION ON SIMULATION RESULTS

**During READ Operation:**

— During WRITE Operation Master sends a particular address (in this case it is decimal 13) to a slave data file. If the address is present in the write address channel then AWVALID Signal will be high.

— Slave data file/Memory acknowledges to Master by asserting AWREADY indicating that it accepts the address. The address from the Master remains stable until the AWREADY is high.

— Then Master will send the data to a memory file using a data bus. When valid data is present in data bus then the WVALID signal goes high.

— When Memory is ready to accept the information it gives WREADY high then Memory will receive that data i.e., 32'd26 present in the data bus. At that particular address, the data present in Memory can be written newly or can be overwritten. Figs. 12, 13 are empty memory file, written memory file respectively and Fig. 14 depicts the simulated timing diagram during the write operation

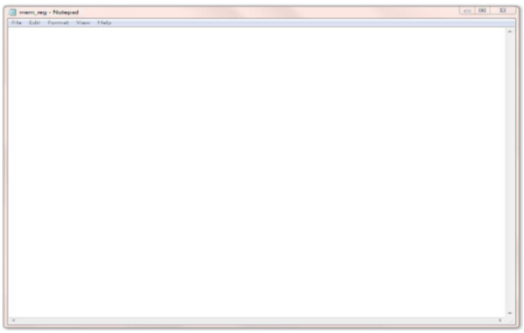

**Fig. 12.** Memory File before WRITE Operation.

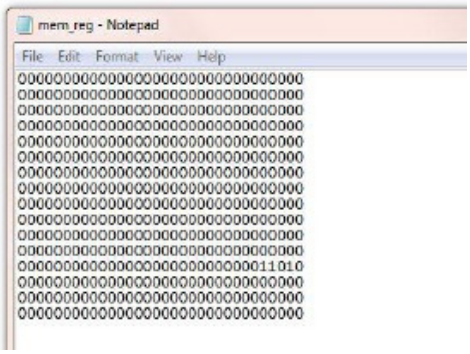


**Fig. 13.** Memory File after WRITE Operation.

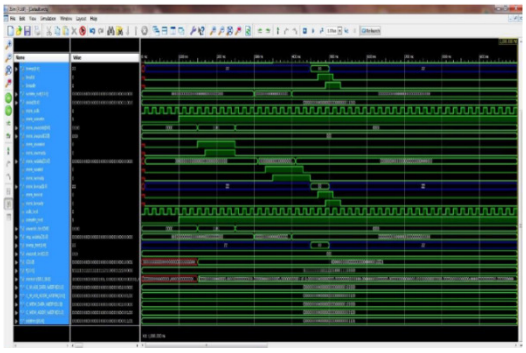

**Fig. 14.** Timing diagram during WRITE Operation.

**During READ Operation:**

— In READ Operation Master will send a particular address location from where it would like to read the data i.e., 'd13. Whenever an address is present in the read address channel then ARVALID Signal will be high.

— The address from the Master remains stable until the RREADY is high. Memory gives acknowledgment to Master by asserting ARREADY indicating that it accepts the address.

— The Memory will receive that Address and puts the data i.e., 32'd26 on the data bus. When valid data is present in the data bus then RVALID signal goes high.

— When the RREADY signal is high, it allows the Master to accept the data (whose timing diagrams are as shown in Fig. 15) which reads from memory file shown in Fig. 16.

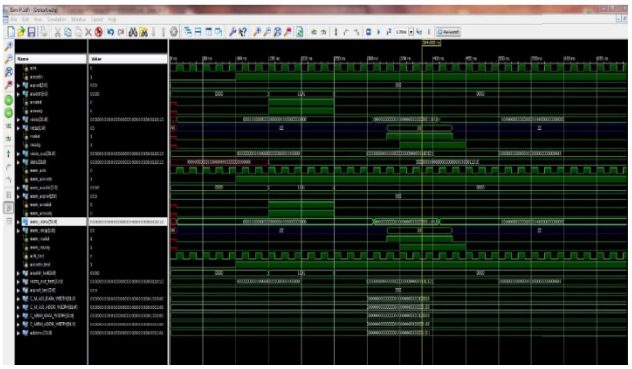

**Fig. 15.** Timing diagram during READ Operation.

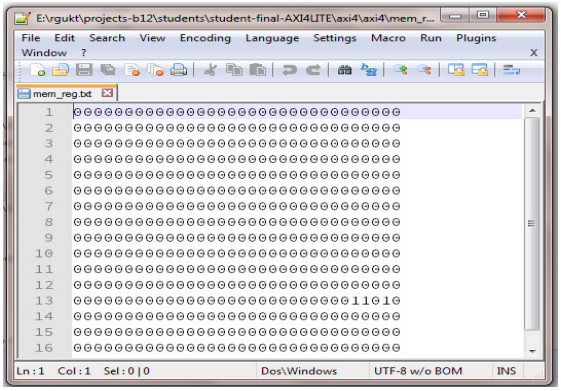


**Fig. 16.** Information in Memory during read Operation.

## VII. CONCLUSION

This paper started out with a discussion about SoC and Bus protocols demonstrating AMBA significance, along with its versions which meant for adequate interface for a given application in glimpse. The present digital era has enormous heterogeneous peripherals on system board with numerous processor cores; among which the AMBA-AXI of ARM's an open-standard, on-chip communication standard supports a rich set of bus signals, administrating several peripherals for a framework on a chip.

The work focuses on the designing of AXI4-Lite which is meant for low memory peripheral requirements Concluding with the synthesis of AXI4-LITE Master Core and a Slave Memory peripheral along with the simulation waveforms exhibiting how a master

performing effective Memory-mapped data transactions of one burst length integrated for 32-bits at 13th address location in the customized slave peripheral/data file. All the synthesis reports are carried out in Xilinx ISE 14.4 tool, followed with simulation using ISim simulator.

## VIII. FUTURE SCOPE

The fact that axi4/lite supports for limited bursting and data transfers subjected with address phase, but for certain applications it always necessitates a protocol like AXI4 stream for video streaming in which a very number of frames have to be accessed and processed i.e., unlimited bursting of data is required at which the address is not so crucial, is considered to be as future scope for present work.

**Conflict of Interest.** No.

## REFERENCES


[1]. Shrivastava, A., Pandit, A. K., Kakkar, V., & Tomar, G. S. (2012). Trends and Trade-offs in designing and performance evaluation of different on-chip AMBA Bus. *International Journal of Engineering Sciences & Management*, *2*(1), 66-72.

[2]. AMBA™ Specification (Rev 2.0) © Copyright ARM Limited 1999. All rights reserved.

[3]. Patil, R. P., & Sangamkar, P. V. (2015). A review of system-on-chip bus protocols. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *4*(1), 271-281.

[4]. ARM, AMBA AXI reference guide (UG761 (v13.1) (2011). [Online]. Available at http://www.arm.com

[5]. Venni, E., & Rao, K. G. (2018). Developing of AMBHA AHB Based Memory Controller using VHDL. *International Conference on Recent Innovations in Science Engineering and Management*: (*ICRISEM*), *12,* 897-904.

[6]. Manu, B. N., & Prabhavathi, P. (2013). Design and implementation of AMBA ASB APB Bridge. In *2013 International Conference on Fuzzy Theory and Its Applications (iFUZZY)* (pp. 234-238). IEEE.

[7]. Gandhani, P., & Patel, C. (2011). Moving from AMBA AHB to AXI Bus in SoC Designs: A Comparative Study. *International Journal of Computer Science & Emerging Technologies (IJCSET)*, *2*(4), 476-479.

[8]. Kandiya, M. M. N., Harniya, M. M. K., & Govani, K. K. (2014). Implementation of Read/Write operation for AMBA AXI4 Bus using VHDL. *IJFTMR*. *I*, IV, 1-3.

[9]. Processor Pipeline by Nima Honarmand-stony brook university, Spring 2016: CSE 502 – Computer Architecture

[10]. AMBA-AXI-exclusive-access available at blogs.synopsis.com/vip-central/2016/08/24/amba-axi-exclusive-access-de-mystified/ (online)

[11]. AMBA, A. (2011). AXI and ACE Protocol Specification. *ARM IHI D*, *22*, 147. (Pp 118).

[12]. Krishna, N. V. (2017). Design and Development of AMBA-AXI4 Bus Based System on Chip VIP Protocol Using UVM. *International Journal of Innovative Research in Science, Engineering and Technology, 6*(4). (Pp 8).

[13].https://www.intrinsix.com/blog/how-chiplets-provide-a-path-forward-for-custom-semiconductor-design

[14]. Understanding-amba-bus-architechture-protocols available at http://verificationexcellence.in/amba-bus-architecture/ (online)

[15]. Samir Palnitkar. (2003). *Verilog HDL:* A Guide to Digital Design and Synthesis, Second Edition Publisher: Prentice Hall PTR

[16]. AMBA® AXI Protocol Version: 2.0 Specification Copyright © 2003-2010 ARM. All rights reserved. (Page No.18-20,25-30, 26,45 & chapter-13)

[17]. https://en.wikipedia.org/wiki/System_on_a_chip

[18]. AMBA® 4 AXI4-Stream Protocol Version: 1.0 Specification, 2010 (Pp 14).



**How to cite this article:** Sainath Chaithanya, A., Sulthana, Sameera, Yamuna, B. and Haritha, Ch (2020). Design of AMBA AXI4-Lite for Effective Read/Write Transactions with a Customized Memory. *International Journal on Emerging Technologies*, *11*(1): 396–402.