IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 7, Issue 4, Ver. I (Jul. - Aug. 2017), PP 51-56
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Advance High Performance Bus Arbitration Techniques (AHB): A State-of-the-Art Review

Rinku[1], Pawan Kumar Dahiya[2]
[1] Scholar of M. Tech. in ECE (VLSI Design), *(Department of Electronics and Communication, Deenbandhu Chhotu Ram University of Science Technology, Sonepat, India)*
[2]A.P., ECED *(Department of Electronics and Communication, Deenbandhu Chhotu Ram University of Science & Technology, Sonepat, India)*

Abstract: *Modern computer system depend more and more on– chip communication protocol to exchange data. System-on-chip (SoC) is designed with reusable intellectual property cores to meet short time to market requirements. The communication delays in the on-chip communication architecture present a major cause of bottlenecks in many SoCs. The selection of the bus (type, width and topology) is one of the most complex tasks of SoC design.AMBA (Advance High Performance Bus) is the best suited bus for this purpose. This paper gives an informative review about the bus interfaces of Advanced Microcontroller Bus Architecture (AHB) and its Arbitration Techniques.*

Keywords: *Advance Microcontroller Bus Architecture (AMBA), System-on-Chip (SoC), Quality of Service (QoS), Time Division Multiple Access (TDMA).*

Date of Submission: 01-07-2017 Date of acceptance: 20-07-2017

I. Introduction

To share data, Computer systems relay more on highly complex on–chip communication protocol. It provides correct connections from the source components to their destinations. In addition, the communication must be fast and predictable. The extent complexity of these protocol results from handling high-performance requirements. Protocol control can be distributed, and there may be non-atomicity. The electronics industry has entered the era of multi-million-gate chips, and there is no turning back. This technology promises new levels of integration on a single chip, called the System-on-Chip (SoC) design, but also presents significant challenges to the chip designer. Processing cores on a single chip, may number well into the high ten's within the next decade, given the current rate of advancements [6]. Embedded systems design focuses on low Power dissipation and SoC. A reliable on-chip communication standard is a must in any SoC. On-chip communication architectures can have a great influence on the speed and area of SoC Designs. The important feature of a SoC is not only which block it houses, but also how they are interconnected. AMBA is a solution for the interface of every block on SoC with each other.

The objective of the AMBA specification is to:

- Open the door to right-first-time development of embedded system's microcontroller products with one or more CPUs.
- Technology independent, reuse of IP cores, peripheral and system macro-cells across diverse IC processes is allowed, encourage modular system design to improve processor independence, and the development of reusable peripheral and system IP libraries.
- Minimize area of silicon while supporting high performance and low power on-chip communication.
- to encourage modular system design to improve processor independence, providing a development road-map for advanced cached CPU cores and the development of peripheral libraries.[1]

Carrying out literature review is very significant in any research on AMBA protocol. It clearly establishes the need of the work and the background development in AMBA protocol. It generates related queries regarding improvement in the study already done and allows unsolved problems to emerge and thus clearly define all boundaries regarding the development of the research work.

This paper reviews the bus architectures of AMBA AHB. This paper discusses the detail of AMBA Protocol in the section II, section III deals with AMBA AHB and their Arbitration techniques of AHB arbiter. Finally section IV and V gives conclusion of the paper and proposed work.

II. AMBA Protocol

AMBA was introduced by ARM in 1996 and is widely used as the on-chip bus in SOC designs. AMBA is a registered trademark of ARM.

• *Ist Generation* - Advanced System Bus (ASB) and Advanced Peripheral Bus (APB).
• *IInd Generation* - Advanced High-performance Bus (AHB) is a single clock edge protocol.
• *IIIrd Generation* - AMBA 3.0 Advanced extensible Interface (AXI), to reach even higher performance interconnect and Advanced Trace Bus (ATB) as a part of core sight on chip debug and trace solution in 2003.
• *IVth Generation* – AMBA 4.0 AXI4, AXI4-Lite, and AXI4-Stream Protocol, The AMBA 4.0 protocol defines five interfaces:

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

• *Vth Generation* – AMBA 5.0 include AXI Coherency Extensions (ACE) and Coherent Hub Interface (CHI) protocol to establish hardware based coherency.

The AMBA 3.0 defined in protocol 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.[2]

A simple transaction on the AHB consists of an address phase and a subsequent data phase (without wait states: only two bus-cycles). Access to the target device is controlled through a Mux (non-tristate), thereby admitting bus-access to one bus-master at a time. AMBA 4.0 protocol includes definition of an expanded family interconnect protocols including AXI4, AXI4-Lite and AXI4-Stream. The AXI4 protocol adds support for longer bursts and Quality of Service (QoS) signaling, and is a natural extension of the AMBA 3 specification. Long burst support aids integration of devices with large block transfers, while QoS signaling provides the ability to manage latency and bandwidth in complex multi-master systems. The new AMBA 4 specification and the AXI4 protocols have also been extended to meet the needs of FPGA implementations. AXI4-Lite is a subset of the full AXI4 specification for simple control register interfaces, reducing SoC wiring congestion and simplifying implementation, while the AXI4-Stream protocol provides an efficient streaming interface for non address-based, point-to-point communication, such as video and audio data. After that AMBA 5.0 came into existence. AMBA 5.0 is to maintain cache coherency in today's multi-core chips.

III. AMBA 2.0 Protocol

AMBA specification, developed by ARM, defines an on-chip communications standard for designing high performance embedded microcontrollers. Three distinct buses are defined within the AMBA specification:

- The AMBA AHB is for high-performance, high-clock-frequency system modules.
- The AMBA ASB is for high-performance system modules. where the high performance features of AHB are not required.
- AMBA APB is optimized for minimal power consumption and reduced interface complexity to support peripheral functions. APB can be used in conjunction with either version of the system bus.[2]

An AMBA-based microcontroller is shown in Figure 1 typically consists of a high-performance system backbone bus AMBA AHB or AMBA ASB, able to sustain external memory bandwidth, on which the CPU, on-chip RAM memory and other Direct Memory Access (DMA) devices reside. This bus provides a high bandwidth interface between that are involved in the majority of transfers. Also located on the high performance bus is a bridge to the lower bandwidth APB, where most of the peripheral devices in the system are located.

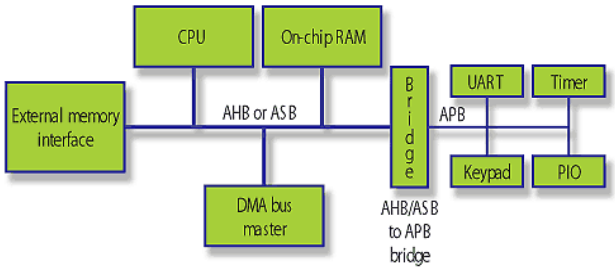

Fig. 1. AMBA 2.0 Protocol

Bridge work as in intermediate between high bandwidth and low bandwidth. [2].

IV. AMBA AHB

AMBA AHB is intended to address the requirements of high-performance synthesizable designs. It is a high-performance system bus that supports multiple bus masters and provides high bandwidth operation. AMBA AHB implements the features required for high-performance, high clock frequency systems including:
• burst transfers
• split transactions
• single-cycle bus master handover
• single-clock edge operation
• non-tristate implementation
• wider data bus configurations (64/128 bits).

Bridging between this higher level of bus and the current ASB/APB can be done efficiently to ensure that any existing designs can be easily integrated. An AMBA AHB design may contain one or more bus masters, typically a system would contain at least the processor and test interface. However, it would also be common for a *Direct Memory Access* (DMA) or *Digital Signal Processor* (DSP) to be included as bus masters. The external memory interface, APB bridge and any internal memory are the most common AHB slaves. Any other peripheral in the system could also be included as an AHB slave. However, low-bandwidth peripherals typically reside on the APB. A typical AMBA AHB system design contains the following components:
AHB master A bus master is able to commence read and write operations by providing an address and control information. Only one bus master is allowed to actively use the bus at any one time.
AHB slave A bus slave responds to a read or write operation within a given address-space range. The bus slave signals back to the active master the success, failure or waiting of the data transfer.
AHB arbiter The bus arbiter ensures that only one bus master at a time is allowed to start off with data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as *highest priority* or *fair* access can be implemented depending on the application requirements.
An AHB would include only one arbiter, although this would be trivial in single bus master systems.
AHB decoder The AHB decoder is used to decode the address of each transfer and provide a select signal for the slave that is involved in the transfer. A single centralized decoder is required in all AHB implementations.

V. Arbitration Protocol

The bus arbiter ensures that only one bus master at a time is allowed to originate data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as *highest priority* or *fair* access can be implemented depending on the application requirements. An AHB would include only one arbiter, although this would be trivial in single bus master systems. The arbiter performs this function by observing a number of different requests to use the bus and deciding which is currently the highest priority master requesting the bus. The arbiter also receives requests from slaves that wish to complete SPLIT transfers. Any slaves which are not capable of performing SPLIT transfers do not need to be aware of the arbitration process, except that they need to observe the fact that a burst of transfers may not complete if the ownership of the bus is changed.

A. Signal description

A brief description of each of the arbitration signals is given below:
HBUSREQx The bus request signal is used by a bus master to request access to the bus. Each bus master has its own **HBUSREQx** signal to the arbiter and there can be up to 16 separate bus masters in any system.

HLOCKx The lock signal is asserted by a master at the same time as the bus request signal. This indicates to the arbiter that the master is performing a number of indivisible transfers and the arbiter must not grant any other bus master access to the bus once the first transfer of the locked transfers has commenced. **HLOCKx** must be asserted at least a cycle before the address to which it refers, in order to prevent the arbiter from changing the grant signals.

HGRANTx The grant signal is generated by the arbiter and indicates that the appropriate master is currently the highest priority master requesting the bus, taking into account locked transfers and SPLIT transfers. A master gains ownership of the address bus when **HGRANTx** is HIGH and **HREADY** is HIGH at the rising edge of **HCLK**.

HMASTER[3:0] The arbiter indicates which master is currently granted the bus using the **HMASTER[3:0]** signals and this can be used to control the central address and control multiplexor. The master number is also required by SPLIT-capable slaves so that they can indicate to the arbiter which master is able to complete a SPLIT transaction.

HMASTLOCK The arbiter indicates that the current transfer is part of a locked sequence by asserting the HMASTLOCK signal, which has the same timing as the address and control signals.

HSPLIT[15:0] The 16-bit *Split Complete* bus is used by a SPLIT-capable slave to indicate which bus master can complete a SPLIT transaction. This information is needed by the arbiter so that it can grant the master access to the bus to complete the transfer.

B. Requesting Bus Access

A bus master uses the **HBUSREQx** signal to request access to the bus and may request the bus during any cycle. The arbiter will sample the request on the rising of the clock and then use an internal priority algorithm to decide which master will be the next to gain access to the bus. Normally the arbiter will only grant a different bus master when a burst is completing. However, if required, the arbiter can terminate a burst early to allow a higher priority master access to the bus. If the master requires locked accesses then it must also assert the **HLOCKx** signal to indicate to the arbiter that no other masters should be granted the bus. When a master is granted the bus and is performing a fixed length burst it is not necessary to continue to request the bus in order to complete the burst. The arbiter observes the progress of the burst and uses the **HBURST[2:0]** signals to determine how many transfers are required by the master. If the master wishes to perform a second burst after the one that is currently in progress then it should re-assert the request signal during the burst. If a master loses access to the bus in the middle of a burst then it must re-assert the

HBUSREQx request line to again get access to the bus. For undefined length bursts the master should continue to assert the request until it has started the last transfer. The arbiter cannot predict when to change the arbitration at the end of an undefined length burst. It is possible that a master can be granted the bus when it is not requesting it. This may occur when no masters are requesting the bus and the arbiter grants is given to a default master. Therefore, it is important that if a master does not require access to the bus it drives the transfer type **HTRANS** to indicate an IDLE transfer.

C. Granting Bus Access

The arbiter indicates which bus master has currently the highest priority requesting the bus by asserting the appropriate **HGRANTx** signal. When the current transfer completes, as indicated by **HREADY** HIGH, then the master will become granted and the arbiter will change the **HMASTER[3:0]** signals to indicate the bus master number[2].

VI. Arbitration Techniques

There are many different arbitration schemes created for different applications. These arbitration schemes i.e., scheduling strategies can be applied not just on the bus arbiter, but also more generally at hardware and software scheduling. These are: static priority, round-robin, Weighted round robin and time division multiple access (TDMA), Lottery Bus Architecture.

A. Static Fixed Priority:

In this arbitration scheme, the masters are given fixed priorities. If several masters are requesting to access the same slave at the same time, access is given to the highest priority master. This is a simple arbitration scheme and it is one of the most commonly used small area cost, flexibility and faster arbitration time. It provides high performance for the high priority masters. In a crowded bus, there is a possibility that some lower priority masters have to wait a long time for the access or the access might never be granted. This protocol is used in shared bus communication architectures[9].

B. Round Robin Algorithm and Weighted Round Robin:

Round Robin Algorithm, in this scheme token is passing the priorities change at every transfer. The master with the highest priority (i.e. with the token) changes in circular (round-robin) fashion. If the master with the highest priority does not need to access the slave the token is given to the master with the next highest priority. [6] depicts a basic round-robin arbiter with four masters (M1-M4) and one slave (S) see Figure 2.

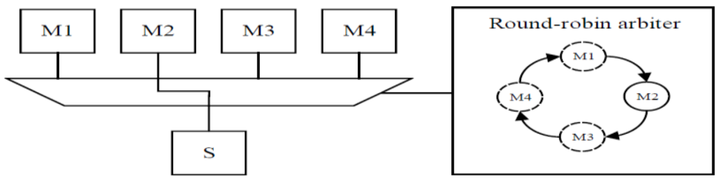

Fig 2: Round-robin arbiter

Round-robin arbitration scheme guarantees that every master is given opportunity for equal bandwidth. It gives fair access for every master to the slave. The maximum wait time is predictable and proportional to the number of masters. The scheme also ensures that the unused time slots are usable by the lower priority masters. The basic round-robin arbitration scheme can be slower than static priority scheme for some masters. This comes from the nature of fairness where all masters are given equal bandwidth. There exists variations of the round-robin such as the Weighted-Round-Robin (WRR) that are meant to tackle this issue. In WRR, every master is given "credits" or "shares" and after each transfer, one credit is deducted. When master runs out of credits the next master is selected in round-robin fashion. Critical masters can be given more credits, which means they get more bandwidth. After all masters have run out of credits, they are reloaded. The drawback of the WRR arbitration is increased complexity compared to the basic round-robin. [7]

A. ***Time Division Multiple Access (TDMA) :***

In TDMA arbitration scheme masters are given fixed time frames for transfers. Masters that need more bandwidth can be given longer time frames. The scheme ensures high bandwidth for critical masters while ensuring that lower priority masters are eventually served. TDMA arbitration is pictured in Figure 3 [8].

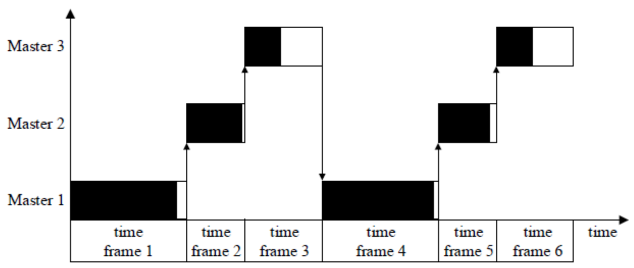

Fig 3: TDMA Arbitration

The darkened areas of the time frames in Figure 3 represent the actual used transfer times by the masters. The decision on the lengths of the time frames is crucial for ensuring high bandwidth allocation. As an example, in Figure 3 the Master 3 uses only under 50 % of the available time frame. This causes a lot of wasted time and bandwidth. The advantage of TDMA arbitration is that it guarantees fixed bandwidths for the masters. When properly designed, it gives high performance for critical masters but with unpredictable or changing traffic, it cannot match the bus allocation of a round-robin arbiter.

B. Lottery Bus Architecture:

In this protocol a centralized lottery manager accumulates request for ownership of shared communication resources from one or more masters, each of which has assigned static or dynamic lottery tickets. Master owning the maximum number of tickets will be granted the access of bus.

VII. Features Comparison

Table 1, shows features comparison between all techniques of arbitration of AMBA AHB. This comparison is based on literature survey of these techniques.

Table 1. Comparison of Techniques

Parameter	Static Fixed Priority	Round Robin	Weighted Round Robin	TDMA
Simplicity	High	Moderate	Low	Moderate
Cost	Low	Moderate	High	Moderate
Architecture	Shared Bus Architecture	All masters have equal bandwidth	Require more bandwidth than Round Robin	Require high and fixed bandwidth to all masters
Performance	High	Moderate	Low	High

VIII. Future Work

These comparison techniques will be verified using Hardware Description Language and synthesis tool and some more techniques will also be verified.

References

Journal Papers:

[1] ARM, "AMBA Specification Overview", available at *http://www.arm.com/*.

[2] ARM, "AMBA Specification (Rev 2.0)", available at *http://www.arm.com*.

[3] Anurag Shrivastava, G.S. Tomar and Ashutosh Kumar Singh," Performance Comparison of AMBA Bus-Based System-On-Chip Communication Protocol," *International Conference on Communication Systems and Network Technologies* , IEEE DOI 10.1109/CSNT.2011.98,p.449.

[4] Sandhya Sunpal and Prof. Mohammed Arif," A Review on High-Performance Microcontroller Bus Architecture," *International Journal of Digital Application & Contemporary Research* Website: www.ijdacr.com (Volume 4, Issue 2, September 2015).
[5] Benini and D. Bertozzi, "*Network-on-chip architectures and design methods*," IEEE Proc.-Comput. Digit. Tech., Vol. 152, No. 2, March 2005.
[6] Shin E., Mooney J. & Riley G. "*Round-robinarbiter design and genera-tion*" In: IEEE 15th International Symposium on System Synthesis, (October 2002 , Kyoto, Japan, p. 243-248)
[7] Sonntag S. & Helmut R. "An Efficient Weighted-Round-Robin Algorithm for Multiprocessor Architectures" In: IEEE 41st Annual Simulation Symposium, (April 2008, Ottawa, Canada, p. 193-199).
[8] Cilku B., Crespo A., Puschner P., Coronel J. & Peiro S. " A TDMA-Based Arbitration Scheme for Mixed-Criticality Multicore Platforms" In: IEEE First International Conference on Event-Based Control, Communication, and Signal Processing, (June 2015 17-19, Krakow, Poland, p. 1-6).
[9] D.Shanthi, Dr.R.Amutha , "Design Approach to Implementation Of Arbitration Algorithm
[10] In Shared Bus Architectures", Computer Engineering and Intelligent Systems: (ISSN 2222-1719 Volume 2, 2011).

IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) is UGC approved Journal with Sl. No. 5081, Journal no. 49363.

Rinku. "Advance High Performance Bus Arbitration Techniques (AHB): A State-of-the-Art Review." IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) 7.4 (2017): 51-56.