International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 01 | Jan-2016 www.irjet.net p-ISSN: 2395-0072

A PROFICIENT RECOGNITION METHOD FOR ML-AHB BUS MATRIX

K.Priya[1], I.Jesintha[2], J.Kannadasan[3], T.Sivasakthi[4], M.Surya[5], G.Thiraviya Suyambu[6]

[1,2,3,4,5,6]Assistant Professor

[1,2,3,4,5,6]Roever College of Engineering & Technology-Perambalur, Tamilnadu, India.

Abstract— Bus matrix with multilayer projected by ARM is a highly efficient on-chip bus that allows the parallel access between multiple masters and slaves. The proposed ML-AHB bus matrix utilizes the use of slave-side arbitration. The ML-AHB bus matrixes of ARM suggest only transfer based fixed priority and round robin arbitration method. In this paper, we intend the design and completion of a flexible arbiter for the ML-AHB bus matrix to sustain three priority policies, fixed priority, round robin, and dynamic priority method. The projected arbiter, who is self-annoyed, selects the possible arbitration method based upon the priority-level notice and the necessary transfer length from masters so that the arbitration leads to maximum on the whole performance. The projected arbiter reduces area overhead and increases the throughput rate by making comparison with other schemes.

Keywords- *Multilayer AHB (ML AHB) busmatrix, on-chip bus, self Annoyed (SA) arbitration scheme, slave-side arbitration, system-on-a-chip (SoC)*

1. Introduction

1.1 AMBA

The Advanced Microcontroller Bus Architecture (AMBA) defines an on chip communications for designing highly efficient embedded microcontrollers. A typical AMBA system which is shown in figure 1.1.Three different buses is defined within the AMBA specification:

- The Advanced High-performance Bus (AHB)
- The Advanced System Bus (ASB)
- The Advanced Peripheral Bus (APB).

Advanced High-performance Bus (AHB)

The AMBA AHB is for highly efficient, high clock frequency system modules. The AHB acts as the highly efficient system bus. AHB supports an efficient connection of processors, on-chip and off-chip external memory crossing point with low-power peripheral macro cell functions. AHB is also specified to make sure ease of use in a capable design flow using synthesis and automated testing techniques.

Advanced System Bus (ASB)

The AMBA ASB is for high performance system modules. It is an alternative system bus suitable for the uses where the highly efficient features of AHB are not required. It also supports the efficient connection of the processors, on chip and off chip memory interfaces with the low power macro cell functions.

Advanced Peripheral Bus (APB)

The AMBA APB is for low power peripherals. AMBA APB is optimized for minimal power consumption and it will reduce the interface complexity to support the peripheral functions. APB can also be used in the conjunction with any version of the system bus.

1.2 Objectives of the AMBA specification

The AMBA has been derived to satisfy the four key requirements:

- To facilitate the time development of embedded microcontroller products with one or more CPUs or signal processors
- To be technology independent and ensure that highly reusable peripheral and system macro cells can be migrated across a diverse range of IC processes and it be appropriate for full custom, standard cell and gate array technologies.
- To minimize the silicon infrastructure that is required to support an efficient on chip and

off-chip communication for both operation and the manufacturing test.

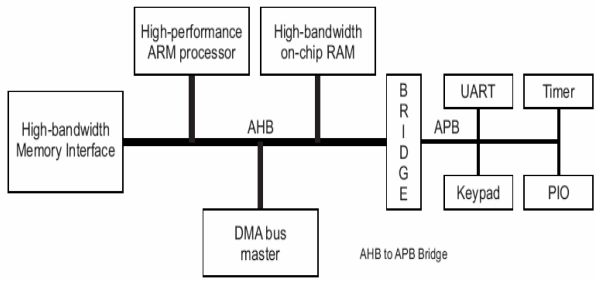

Fig 1.1 A Typical AMBA System

A typical AMBA-based microcontroller

An AMBA microcontroller is typically consists of a highly efficient performance system bus which is able to sustain the external memory bandwidth, on which the CPU, on chip memory .This microcontroller bus provides a high bandwidth interface between the elements with the majority of transfers. It also located on the high efficient bus is a bridge to the lower bandwidth.

1.3 Other buses

Avalon is the Altera's interface bus which is used by the Nios embedded processor. The Avalo switch fabric which has a set of pre defined signal types which has a user can connect one or more intellectual property blocks. The wizard-based Altera's SOPC Builder system used to develop the tool which is automatically generates the Avalon switch using fabric logic. The generated switch fabric logic which includes the chips that elect the signals for the data-path multiplexing, address decoding, wait-state generation, interrupt priority assignment, dynamic bus sizing, multi master arbitration logic and advanced switch fabric transfers. The advanced transfers including streaming transfers, read transfers with latency and the bus control signals. Avalon masters and slaves who interact with each other that are based on a technique which is called the slave side arbitration. The Slave side arbitration will determines which master gains access to a slave, in the event that multiple masters could attempt to access the same slave at which the same time.

Core Connect is an IBM developed on chip bus communications will link that enables chip cores from multiple sources to be interconnected to create the entire new chips. The Core Connect technology eases the integration and it reuse of processor, system, and peripheral cores within the standard product platform which is designs to achieve the overall greater system performance with more efficient. The Core Connect bus architecture which includes: *processor local bus (PLB)*-The PLB bus it addresses the high performance, low latency system modules and it will provides the design flexibility which is needed in a highly integrated SOC. *on-chip peripheral bus (OPB)*-The OPB bus is an optimized bus to connect to the lower speed peripherals and the low power consumption. *Device control register bus (DCR)*-The Device Control Register (DCR) bus is designed to transfer the data between the CPU's general purpose registers (GPRs) and the DCR slave logic's device control registers (DCRs). The DCR bus will removes the configuration registers from the memory address map, which reduces the loading and it improves the bandwidth of the PLB. The fully synchronous DCR bus will provides 10-bit address bus and 32-bit data bus. The DCR bus is typically implemented as a distributed mux across the chip.

WISHBONE.

Its purpose is to advance design which is reuse by alleviating system on- a-chip integration problems. This is proficient by creating a common, logical interface between the IP cores. This will improves the portability and the reliability of the system, and it will results in the faster time to market for the end user. It is recommended by the Open Cores as the interface to all the cores that require interfacing to other cores inside a chip (FPGA, ASIC, etc.). The Wishbone specification is different from the other specifications, as it makes use of RULES, RECOMMENDATIONS, SUGGESTIONS, PERMISSIONS and OBSERVATIONS. So we can say that Wishbone is to be a simple, open, and highly configurable interface. Where others have to be separate the interfaces for the high and low speed peripherals, Wishbone defines only one interface. This shouldn't create a problem, as when you need a high and low speed bus, you can create 2 separate wishbone interfaces. Because of the highly configurable interface, users might have to create their own substandard of wishbone, specifying data order, meaning of tags, and additional features. This is probably also the cause why e.g. a generic Wishbone-to-AMBA bridge still hasn't been developed yet.

2 EXISTING METHOD

Recently, the complexity design has become higher; SoC designs are in the need of a system bus with high bandwidth to perform multiple operations in parallel. To solve these bandwidth problems, there are several types of high-performance on-chip buses had been proposed, such as the multilayer AHB (ML-AHB) busmatrix from ARM, the PLB crossbar switch from IBM, and CONMAX from Silicore. Among them, the ML-AHB busmatrix has been widely used in many SoC designs. This is because of the straightforwardness of the AMBA bus of ARM, which create a center of attention to the many IP designers, and the good architecture of the AMBA bus for applying embedded systems with low power.

The ML-AHB busmatrix is an interconnection scheme which is based on the AMBA AHB protocol, which enables the parallel access paths between the multiple masters and slaves in a system. This is accomplish by using a more complex interconnection matrix and furnish the benefit of both increased overall bus bandwidth and a more flexible system structure. In particular, the ML-AHB busmatrix uses slave-side arbitration. Slave-side arbitration is different from master-side arbitration in terms of appeal and contribution signals since, in the former, the master simply starts a burst transaction and it will waits for the slave response to proceed to the next transfer. Therefore, the unit of arbitration can be a transaction or a transfer. The transaction based arbiter multiplexes the data transfer which is based on the burst transaction, and the transfer based arbiter switches the data transfer that based on a single transfer. For a highly efficient on chip bus, several revision related to the arbitration scheme have been proposed, such as table-lookup-based crossbar arbitration, two-level time-division multiplexing (TDM) scheduling, token-ring mechanism, dynamic bus distribution algorithm, and LOTTERYBUS. Yet, these approaches employ master-side arbitration. Therefore, they can only control priority policy and also it will have some limitations when handling the transfer-based arbitration scheme since master-side arbitration uses a centralized arbiter.

3 PROPOSED METHOD

In compare, it is achievable to deal with the transfer-based arbitration scheme as well as the

transaction-based arbitration scheme in slave-side arbitration. In this paper, we propose a flexible arbiter based on the self motivated (SM) arbitration scheme for the ML-AHB busmatrix which is shown in figure 3.3. Our SA arbitration scheme has the following advantages: 1) It can alter the processed data unit; 2) it modifies the priority policies during runtime; and 3) it is straightforward to tune the arbitration scheme according to the uniqueness of the target relevance. Hence, our arbiter is able to not only deal with the transfer based fixed priority, round robin, and dynamic priority arbitration schemes but it also manage the transaction based fixed priority, round robin, and dynamic priority arbitration schemes. Additionally, our arbiter provides the desired transfer length based fixed priority, round robin, and dynamic priority arbitration schemes. In addition, the proposed SM arbiter selects one of the nine possible arbitration schemes based on the priority level notifications and the desired transfer length from the masters to ensure that the arbitration leads to the maximum efficient performance.

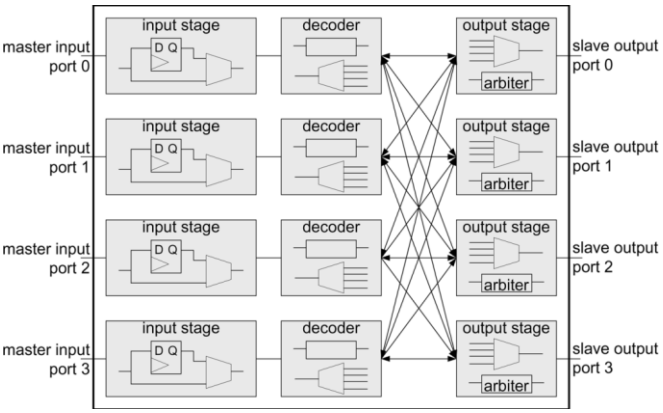

Fig 3.1 Overall structure of the ML-AHB bus matrix of ARM

The ML-AHB busmatrix of ARM which is shown in figure 3.1 is consists of the input stage, decoder, and output stage, including an arbiter. The input stage is in charge for holding the address and control in sequence when transfer to a slave is not able to originate immediately. The decoder decide which slave that a transfer is destined for which is shown in figure 3.2. The output stage is used to select which of the various master input ports is routed to the slave. Each output stage has an arbiter. The arbiter will determines which input stage has to perform a transfer to the slave and decides which the highest priority is currently. The ML-AHB bus matrix employs slave-side arbitration, in which the arbiters are located in front of the each slave port, the master will simply starts a transaction and it waits for the slave response to proceed to the next transfer.

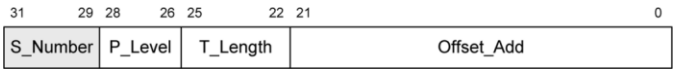

Fig.3.2 Decoding information of the 32-b address bus.

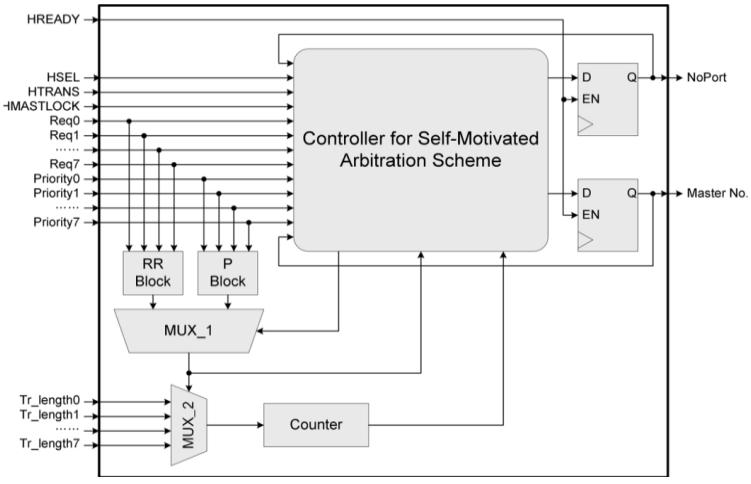

Fig 3.3 Internal structure of our arbiter.

3.1 SM ARBITRATION SCHEME FOR THE ML-AHB BUSMATRIX

An theory is made that the masters can alter their priority level and it can issue the desired transfer length to the arbiters in order to

implement the SA arbitration scheme. This assumption should be valid because of the system developer that generally recognizes the features of the target applications. For example, some of the masters in the embedded systems are required to complete their job for given timing constraints, resulting in the satisfaction of system level timing. The computation time of each master is predictable, but it is not easy to predict the data transfer time since the on chip bus that is usually shared by the several masters. Previous works solved this issue by decreases the latencies of numerous latency critical masters, but a side effect of these methods is that they can boost the latencies of the other masters; hence, they may violate the given timing constraints. Unlike existing works, our scheme can keep the latency close to its given constraint by adjusting the priority level and transfer length of the masters by its own. Fig. 3.4 shows an example. In this example, the service latencies (latency-limit times) of M1, M2, and M3 are 4, 8, and 2 cycles (T14, T8, and T10), respectively.

The requests for three masters are all initiated at T0, and M3 is the most latency-sensitive master. Fig. 3.4 (a) shows an arbitration scheme that does not use latency constraints for arbitration.

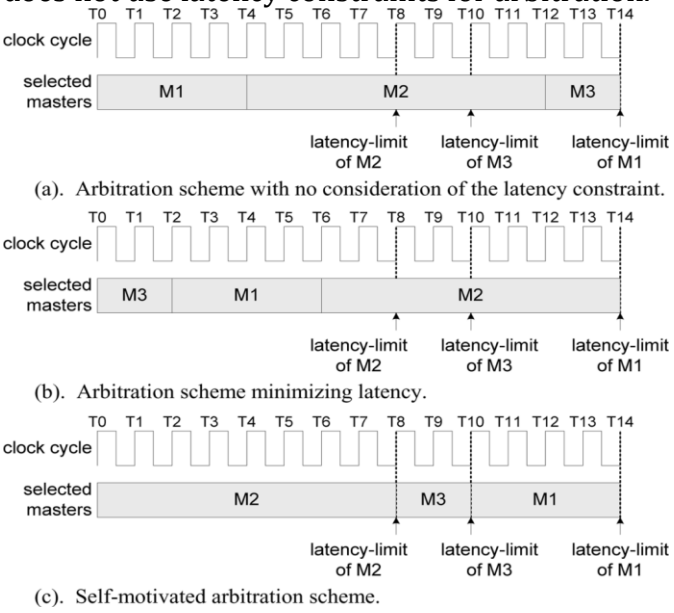

Fig 3.4 Arbitration scheme examples in an embedded system.
(a) Arbitration scheme with no consideration of the latency constraint. (b) Arbitration scheme minimizing latency.
(c) SM arbitration scheme.

Therefore, M2 and M3 will violate the latency constraint as the masters are selected in the ascending order. Only M1will meets the constraint. Fig. 3.4(b) shows the scheduling of a typical latency minimizing arbiter. It minimizes the latency of the most latency sensitive module, namely, M3, which causing M2 to violate its constraint. Although neither of these two arbitration schemes can meet the latency constraints for all the three masters, in the SM arbitration shown in Fig. 3.4(c), all masters use the buses with no violations by configuring the priority levels (transfer lengths) of M1 is lowest, M2 is highest , and M3 is an intermediate priorities (4, 8, and 2), respectively. We use part of a 32-bit address bus of the masters to update the arbiters of the priority level and also the desired transfer length of the masters. Decodinh information for our address bus is shown in Fig. 3.2.

In Fig. 3.2, *S_Number* indicates the slave number of the target, *P_Level* indicates the priority level of a master, *T_Length* denotes the desired transfer length of a master, and *Offset Add- it* specifies the internal address of the target slave. Each of *S_Number* and *P_Level* consists of 3 b because the maximum number of master/slave sets is 8 [3]. Also, *T_Length* is composed of 4 b because the maximum number of burst lengths is 16. Although we used 7 b for *P_Level* and *T_Length* in the 32-b address bus to notify the arbiters of the priority level and the preferred transfer length of a master, we consider it adequate to express the internal address of a slave because the range of *Offset_Add* is from 0 to . Through the aforementioned assumption, the

priority level and transfer length can then be changed by the SM demand of each master.

Fig. 4 shows the internal structure of our arbiter based upon the SM arbitration scheme. In Fig. 4, the *NoPort* signal means that none of the masters must be selected and that the address and control the signals to the shared slave that must be driven to an inactive state, while Master Number which indicates the currently selected master number which is generated by the controller for the SA arbitration scheme. In general, our arbiter consists of an RR block, a P block, two multiplexers, a counter, a controller, and two flip-flops. MUX 1 and MUX 2 are used to select the arbitration scheme and the preferred transfer length of a master, respectively. Counter calculates the transfer length, with two flip-flops are inserted to avoid the attempts by the critical path to arbitrate. An RR block (P block) performs the round robin or priority based arbitration scheme.

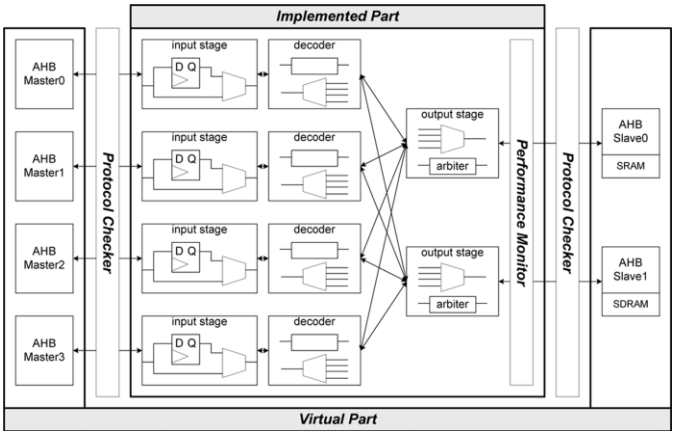

Fig. 3.5 Simulation environment for performance analysis.

Simulation Environments: Figure 3.5 shows our simulation
environment. In our simulation environment, the clock frequencies for all the components are 100 MHz (10 ns). The implemented ML-AHB busmatrix has a 32-b address bus, a 32-b write data bus , a 32-b read data bus, a 15-b control bus, and a 3-b response bus. Temporarily, the simulation environment consists of both an implemented and a virtual part. The former corresponds to the ML-AHB busmatrixes with different arbitration schemes and it will consist of four masters and two slaves. Particularly, we only considered two target slaves, which is when difference frequently happens. The masters then access these, in order to focus the performance analysis which based on the arbitration schemes of each busmatrix. The virtual part is composed of AHB masters and AHB slaves. The AHB master generates the transactions. The AHB slave responds to the transfers of the masters. Both the AHB masters and slaves are fully well-matched with the AMBA AHB protocol. For a more sensible model of a SoC design, we modeled the AHB masters after the features of the processor and DMA with VHDL at the behavioral level. For the AHB slaves, we worn the real SRAM, SDRAM, and SDRAM controller RTL models used in many applications. We also constructed the protocol checker and performance monitor modules with the VHDL and foreign language interface (FLI C module) to ensure the reliability of our performance simulations.

4.CONCLUSION

In this paper, we proposed a flexible arbiter which is based on the Self Annoyed arbitration scheme for the ML-AHB bus matrix. Our Self annoyed arbiter will supports three priority policies: fixed priority, round robin, and dynamic priority, these three approaches used for data multiplexing transfer, transaction, and desired transfer length; in other words, there are nine achievable arbitration schemes. In addition to this the proposed SA arbiter selects one of the nine possible arbitration schemes that is based

on the notifications of the priority level and the desired transfer length from the masters will allow the arbitration to lead to the maximum efficient performance. Experimental results show that, the area of the proposed SA arbitration scheme is better than those of other arbitration schemes; our SA arbiter will improves the throughput compared with the other schemes. Therefore we expect that it would be enhanced to apply our SA arbitration scheme to an application for the specific system because it is easy to tune the arbitration scheme according to the features of their target system. For future work, we feel that the configurations of the SA arbitration scheme with the maximum throughput will be found automatically during the runtime.We are looking at the applicability of the proposed arbitration scheme to AMBA AXI (ver. 3.0).1

Acknowledgment

Before setting to give the review, we would like to thanks the invisible superior power for giving us the strength and perseverance to do this project. We bow our head and thanks to our almighty God for showering upon the necessary wisdom and grace for accomplishing this project. We express our gratitude to our parents first for giving health as well as sound mind and financial support for taking up this project.. Finally, we also extend our gratitude to all who helped us directly and indirectly for the successful completion of our review research work.

References

[1] Implementation of a Self-Motivated Arbitration Scheme for the Multilayer AHB Busmatrix Soo Yun Hwang, Dong Soo Kang, Hyeong Jun Park, and Kyoung Son Jhang, Member, IEEE VOL. 18, NO. 5, MAY 2010

[2] M. Drinic, D. Kirovski, S. Megerian, and M. Potkonjak, "Latencyguided on-chip bus-network design," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 25, no. 12, pp. 2663–2673, Dec. 2006.

[3] S. Y. Hwang, K. S. Jhang, H. J. Park, Y. H. Bae, and H. J. Cho, "An ameliorated design method of ML-AHB busmatrix," ETRI J., vol. 28,no. 3, pp. 397–400, Jun. 2006.

[4] ARM, "AHB Example AMBA System," 2001 [Online]. Available:http://www.arm.com/products/solutions/AMBA_Spec.html.IBM, New York, "32-bit Processor Local Bus Architecture Specification,"2001.

[5] S. S. Kallakuri and A. Doboli, "Customization of arbitration policies and buffer space distribution using continuous-time Markov decision processes," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 15,no. 2, pp. 240–245, Feb. 2007.

[6] K. Lahiri, A. Raghunathan, and G. Lakshminarayana, "The LOTTERYBUS on-chip communication architecture," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 14, no. 6, pp. 596–608, Jun. 2006.

[7] J. H. Han, M. Y. Lee, B. Younghwan, and C. Hanjin, "Application specific processor design for H.264 decoder with a configurable embedded processor," ETRI J., vol. 27, no. 5, pp. 491–496, Oct. 2005.

[8] M. Jun, K. Bang, H.-J. Lee, N. Chang, and E.-Y. Chung, "Slack-based bus arbitration scheme for soft real-time constrained embedded systems," in Proc. Int. Conf. ASP-DAC, Jan. 2007, pp. 159–164.

[9]Jesintha.I,Thiraviya Suyambu.G, Priya.K,Azhagesvaran.T "Efficient Broadcast Authentication with Highest life Span in Wireless Sensor Networks" International research journal of Engineering and Technology (IRJET),Volume 2 Issue 9, December 2015.

[10] Ambiga, S., and M. Ramasamy. "Rural Women Entrepreneurship-A Managerial Perspective." Middle-East Journal of Scientific Research 23.3 (2015): 479-484.

[11] Ambiga, S., and M. Ramasamy. "Women Entrepreneurship Development in India." Jurnal Teknologi 64.3 (2013).

[12] Ambiga, S., and M. Ramasamy. "Rural Women Entrepreneurship–A Step towards Self Contained Economy."

[13] T.Azhagesvaran ,I. Jesintha, K.Priya, G.Thiraviya Suyambu, "Privacy Protection Based On Trust Level Of Nodes In Manet"

BIOGRAPHIES

Mrs.K.Priya[1], M.E.,Assistant Professor Department of ECE, Roever College of Engineering & Technology, Perambalur, Tamil nadu, India. Her research Interests are VLSI & Antenna Design. I have 6 years of teaching experience.

Mrs.I.Jesintha[2] , M.E., Assistant Professor Department of ECE, Roever College of Engineering & Technology, Perambalur, Tamil nadu , India. Her research Interests are Network security & Antenna Design. She has 5 years of teaching experience.