ISSN:2229-6093

Chetan Sharma, Int. J. Comp. Tech. Appl., Vol 2 (2), 248-252

General Purpose AHB-DMA Controller

Chetan Sharma

(M. Tech-VLSI Department, JSS Academy of Technical Education, Noida)

Chetan2042@gmail.com

Abstract: This paper reviewed the design of a general purpose AHB-DMA controller which has two channels to perform read/write operations between the memory and customizable interface (SPI, I2C, USB or SDIO etc.). Each DMA channel enables movement of data from a source AHB address to the IP or IP to the destination AHB address, for a given programmed count. When the programmed bytes (data) are transferred to/from the system memory, an **"**End of transfer interrupt" is generated by the DMA for each channel.

Keywords: AMBA, AHB bus, Tx-FIFO, Rx-FIFO, IP Interface.

Introduction: AMBA stands for Advanced Microcontroller Bus Architecture. It is widely used as the on chip bus in SOC (System On Chip) designs. AMBA is three types AHB(advance high performance bus), ASP(advance peripheral bus), ASB(advance system bus) among which AHB is used for high speed, low latency and high frequency operation.

AHB supports 32, 64, and 128-bit data-bus implementations with a fixed 32-bit address bus. It is a synchronous bus that supports bursts and pipelining of accesses to improve throughput. AHB supports multiple masters. The arbiter has the task of determining which master gets to do an access. Every transfer has an address/control phase and a separate data phase. They are both pipelined (able to start the next transfer's arbitration and address phase while finishing the current transfer)

A typical AMBA AHB system design contains the following components:

AHB master: A bus master is able to initiate 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.

ISSN:2229-6093

AHB arbiter: The bus arbiter insures that only one bus master at a time is allowed to initiate data transfers.

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.

DMA is used to transfer of data between two peripheral or memory. The same work can be done by CPU but for doing free to CPU inplace of it, DMA controller is used. The main benefit of DMA controller is clear that it use for saving the time or for reducing the complexity of CPU. In the same time of transferring the data CPU can do another work by using DMA. Here CPU is use for initiating the transfer of data.

Features of architecture: At the time of data receiving if error is occupied then this error will be respond by the slave of this DMA controller. It can support both of channels transmitting and receiving channel separately. It supports both half duplex and full duplex peripherals.

There are two channels thats why two FIFO are used corresponding to each channel. The pre-defined size of data is transferred through user defined FIFO size. It supports byte by byte transfer of data. At the end of transfer interrupt will give the conformation of it.

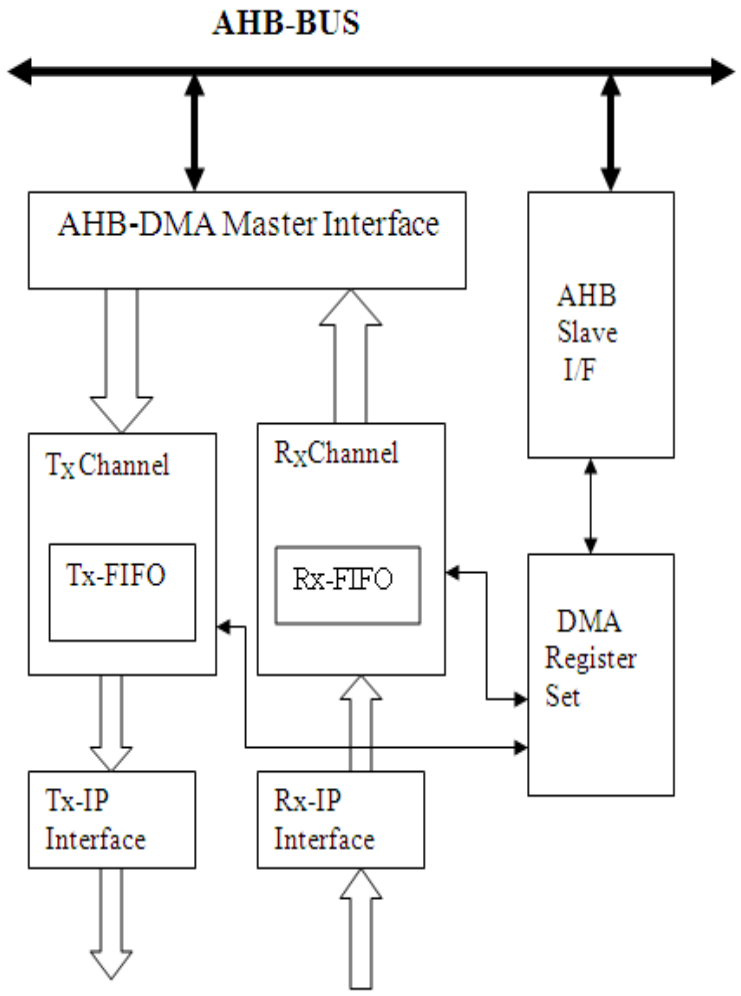

fig (1): AHB-DMA block diagram

Functional description: The AHB Slave block interfaces the AHB bus to the DMA channel registers. The AHB masters are capable of initiating single data transfers. To initiate a transfer, CPU will program all the

operational registers of the peripheral IP and all the operational registers of the AHB-DMA. After programming, it will program the control word register of DMA which will enable the DMA master controller. Data transfer with its associated channel take place using "Data Valid/Req" and "Data acknowledge" signals. When the programmed bytes (data) are transferred, an "End of transfer interrupt" (Tx or Rx) is generated by the DMA and send to the interrupt controller.

FIFO (Tx & Rx) inside the DMA, can have either a fixed or parameterizable depth. In case of Transmit, When-ever FIFO is not FULL, Data fetched from the system memory initiating 32/16/8 bit access will be written into the TX-FIFO. Similarly, in case of receive, when-ever FIFO is not EMPTY, Data read from the RX-FIFO is stored to system memory. ARC processor programs a set of control registers through its AHB slave interface.

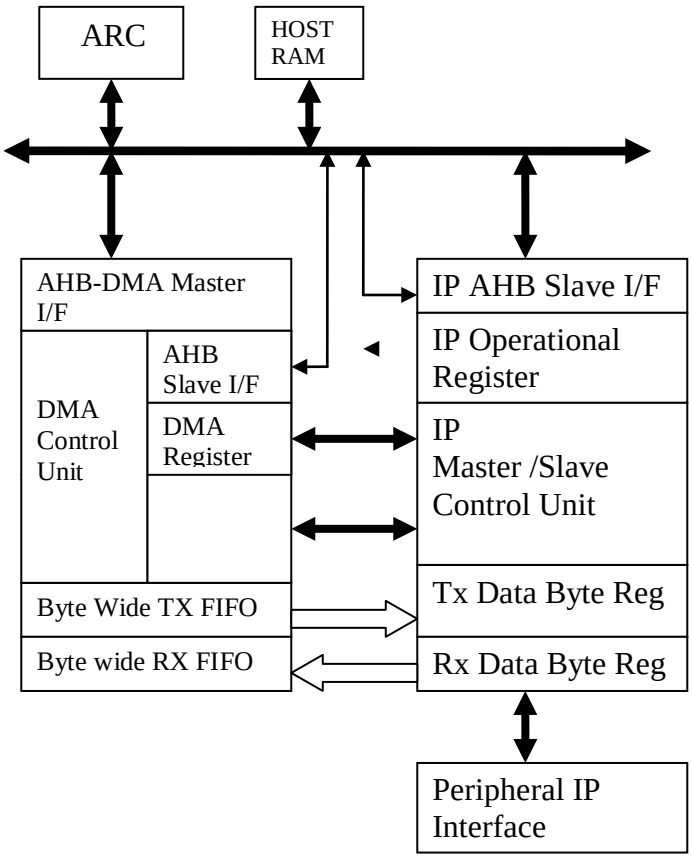

fig (2): AHB-DMA Controller & It's I/f Block Diagram

The data block size should not 32 bit aligned because by using this aligned bit, combination of 8/16/32 bit data can not receive or transmit.

Whenever the FIFO is empty, 8/16/32 bit data which comes from AHB bus are store to Tx-FIFO, prior to sending this data to peripheral device.

Similarly to Tx-FIFO, receiving data from peripheral to AHB bus firstly stored in Rx-FIFO as shown

in fig (2).

DATA Transmission (AHB to peripheral):

This operation is done in two phases: (1) AHB to Tx-FIFO (2) Tx-FIFO to peripheral.

(1) AHB to Tx-FIFO: Processor programs the IP and DMA registers. Depending upon the value of block size register, numbers of bits 8/16 or 32 are allows to AHB bus. According to the value of bus arbiter, data is read from memory and save in Tx-FIFO.

(2) Tx-FIFO to peripheral: Here IP Interface is used for transfer of data. As soon as data comes to the Tx-FIFO, it informs the availability of valid data and allow transfer it to peripheral. At the clock of last byte it informs the availability of last byte and transfer it to peripheral.

At last "END OF TRANSFER" signal is given after transmitting the whole data.

DATA Reception (peripheral to AHB):

There are two steps of transferring data from peripheral to AHB:

(1) IP Interface to Rx-FIFO: During the reception of data, processor program DMA register to enable DMA master in Rx mode. If Rx-FIFO is not full then a signal is send else first-byte-signal is shown which meaning the storing of first byte in Rx-FIFO.

(2) Rx-FIFO to AHB: Depending upon block count 8/16/32 bit data is send to AHB side from Rx-FIFO. At last byte clock, signal of last byte is shown and it sends to AHB.

At the end of transfer of last byte "END OF TRANSFER" signal conforms the sending of last byte.

Conclusion: This paper represents the high speed AHB-DMA controller which reduces the use of CPU at the time of data transmission between two peripherals or memory devices. So that at this time, CPU kept free for doing any other operation. This is done by using two FIFO and master and slave I/f.

References:

[1] Flynn, D., "AMBA: Enabling Reuseable On-Chip Designs", IEEE Micro, 17(4), July/August 1997, pp 20-27.

[2] J. Liang, Swaminathan, S, "ASOC: a Scalable, Singlechip Communications Architecture", Tessier, R. Parallel Architectures and Compilation

Techniques, 2000, Proceedings. International Conference,2000, pp 37-46.

[3] P. Laramie, “Instruction Level Power Analysis and Low Power Design Methodology of a Microprocessor”, MS Thesis, University of California at Berkeley, 1998

[4] AMBA Specification (rev2.0) and Multi Layer AHB Specification, Arm: http://www.arm.com, 2001.

[5] AMBA Bus DMA Controller Specification, Draft 1.03,Gareth Morris 2000.

[6] J. Becker, L. Kabulepa, F.M. Renner, M. Glesner,“Simulation and Rapid Prototyping of Flexible Systemson-a-Chip for Future Mobile Communication Applications”, RSP 2000, pp. 160-165.

[7] Reconfiguring Excalibur Devices Under Processor Control: Application note 298, Feb 2003, ver 1.0.http://www.altera.com

[8] http://www.xilinx.com

[9]http://www.altera.com/products/ip/processors/nios/nioindex.Html

[10] C8237 Programmable DMA Controller Core,http://www.cast-inc.com.