USB Bus Architecture for ARM Processor based System-on-Chip (SoC) Communication

Ajay Singhal[*] Vineet Sahula[†]

ajayaks24@gmail.com sahula@ieee.org

Department of Electronics and Comm. Engineering

Malaviya National Institute of Technology, Jaipur

1. Introduction

Scaling in feature size makes possible the implementation of end product as single chip. A SoC contains large number of components including IP cores-hard and soft and basic standard cells and ASICs. Short time to market window and complex functionality requirement has driven the design reuse of IPs. These IPs from different sources reduces design time significantly. Reuse of IPs brings new challenges also, including rapid integration of communication logic between IPs.

ARM processors are being used in SoCs. Also most of the end user peripherals/functions are USB based/compatible. In this paper we propose an integration strategy for interfacing USB compatible functions with ARM processor system. For the purpose, we consider an ARM compatible memory model and other I/O interfaces.

We build an interface for ARM system components to USB using as core interface protocol. We presume that the memory and other I/O devices have built in support for AMBA protocol, as it is usual for ARM processor system component to have support for AMBA by default. Our contribution is in the generation of wrappers for each component interface so that they can be directly plugged into USB 2.0 interface. This involves basically the protocol conversion between AMBA and USB. The models of protocol and the converters are described using SystemC.

2. System Components

SoC is the collection of several components, whose combined operation implements a useful function. Components are always heterogeneous in nature and their interaction may be

---

[*] Presently working with Tata Consultancy Services. This work was performed, when author was pursuing his M. Tech. (Microelectronics) at Department of ECE, Malaviya National Institute of Technology, Jaipur-302017.

[†] Reader (Assoc. Prof.), Department of ECE, Malaviya National Institute of Technology, Jaipur-302017.

regulated by some simple or complex communication architecture. The majorities of such components are programmable and thus may be of hardware or software types.

Typical SoC architecture as in Fig. 1 is widely used for SoCs, consisting of software components viz. processors and memory together with hardware components viz. DMA, ASICs and I/O devices. These hardware and software components are connected by bus, which means that each component could be either bus master or bus slave. Processors and DMA controllers are masters while memories and I/O components would be slaves. An ASIC can be a master or a slave depending on the functions performed by it. When two or more bus masters are connected to a single bus we definitely require a bus arbiter to ensure that only one master controls the bus at a time.

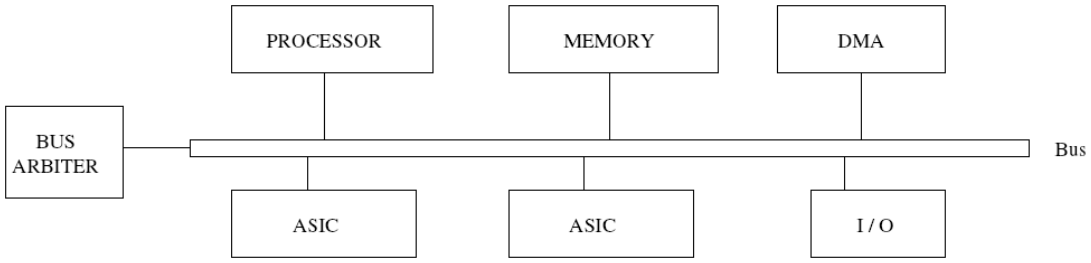

Fig.1. **Typical SoC Architecture**

The bus as an interconnect component, can be designed for specific application. Various bus architectures like AMBA, IBM Core Connect or PCI can be used. In this proposal, we have used Universal Serial Bus (USB).

3. Universal Serial Bus (USB)

USB supports data exchange between a host computer and a wide range of peripherals that can be accessed simultaneously. It supports three speeds, a high-speed mode of 480 Mbps, a full speed mode of 12Mbps and a low speed mode of 1.5 Mbps. USB is host controlled by USB host. There can only be one host per bus at a time. Data travel from the unique host to the endpoint on the selected device through USB. Data communication in USB is in the form of packet. USB support many types of transaction depending on the device requirement, such as Bulk, Control, Interrupt and Isochronous.

The USB 2.0 specification by itself does not support any form of multi master arrangement. The later USB On-The-Go (USB OTG) specification, which is built upon USB 2.0, introduces a “Host Negotiation Protocol” (HNP). HNP allows two devices to compete for

playing the role of the host. USB host is responsible for undertaking all the transactions. Data can be sent by various transactions-methods using token-based mechanism.

One of the original objectives of USB was to reduce the amount of interconnections on SoC to reduce the overall area and power consumption in battery operated SoC applications. This entails sacrificing speed of communication. USB uses a tiered star topology for connecting devices which is ideal for SoC applications

Every USB device must connect to a hub. A hub is a key element in the plug and play architecture of USB. USB 2.0 hubs consist of three portions: the hub controller, hub repeater and transaction translator. The hub repeater is a protocol switch between upstream and downstream ports. The hub controller provides communication to and from the host. Transaction translator provides the mechanism that supports full/low speed devices behind the hub, while transmitting all device data between host and the hub at high speed.

4. System Integration

As discussed earlier, there is an urgent need to optimize design time/TTM while the product complexity increases the pressure to reuse complex building blocks known as Intellectual Property or IP increases. In this paper, we try to implement the interface between communicating IPs that uses different signaling conventions. To separate the communication from behavior of the IP, the block has been abstracted to transaction level.

Fig 2 shows the communication between an abstract producer and receiver using an interface.

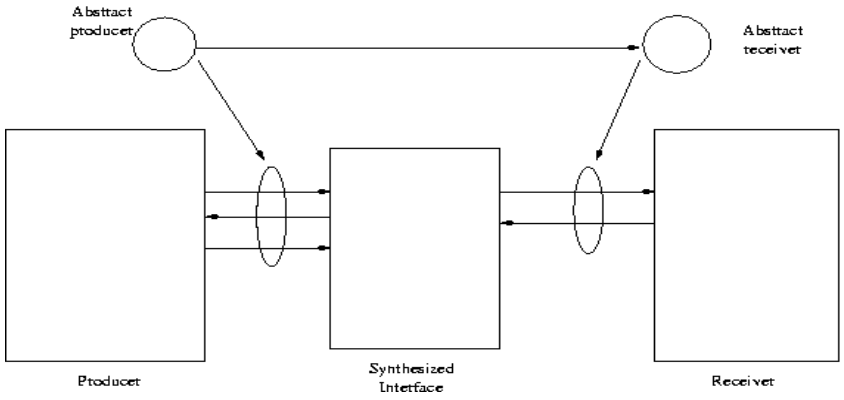

Fig. 2. **Interface between incompatible protocols**

Also we have tried to implement the communication between ARM processor and memory through universal serial bus instead of parallel bus architecture e.g. AMBA that is supported by ARM. Our implementation methodology is shown in Fig. 3.

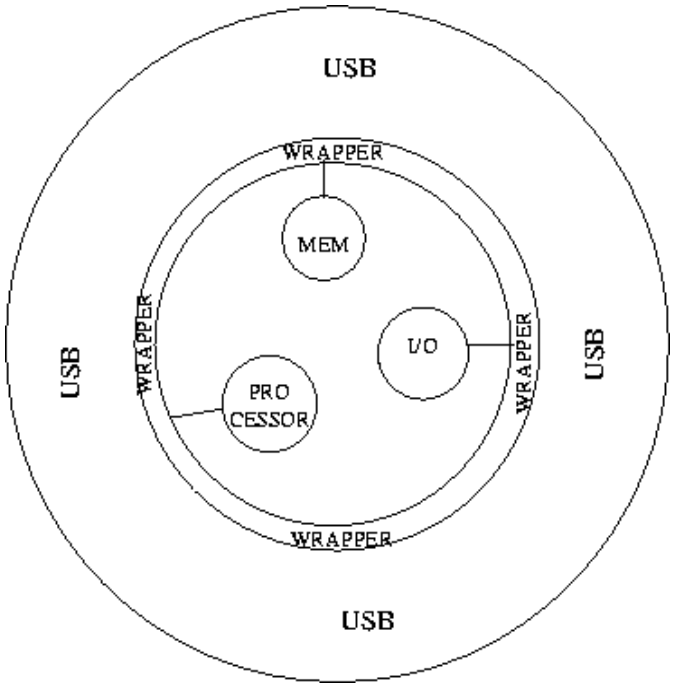

Fig. 3. **System Integration through USB**

In our implementation we consider a transaction level model of ARM and memory. We have implemented USB behavioral model in systemc for communicating between two interfaces. For the memory write operation the address of register in memory, data to be written and control signals are combined in data packet and sent to the USB to memory interface through OUT transfer of USB. For memory read operation, two USB transfers happen. An OUT transfer followed by an IN transfer to read data from memory.

For the slow speed devices such as I/O devices, we have used a hub, which is part of the USB system hierarchy. It is similar to the concept of speed division, which takes place between AHB and APB in AMBA.

5. Results

We have built the system model described above in Section 4 using SystemC constructs. Some of the SystemC simulation results are shown in Fig. 4 and Fig. 5.

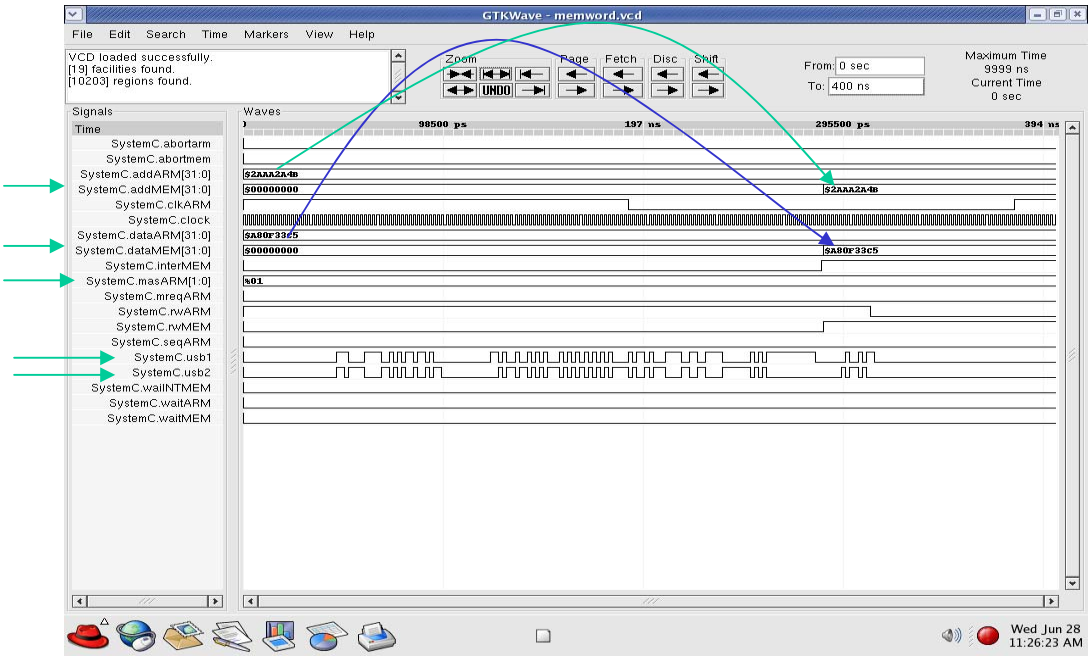

Fig. 4 Memory Write Cycle

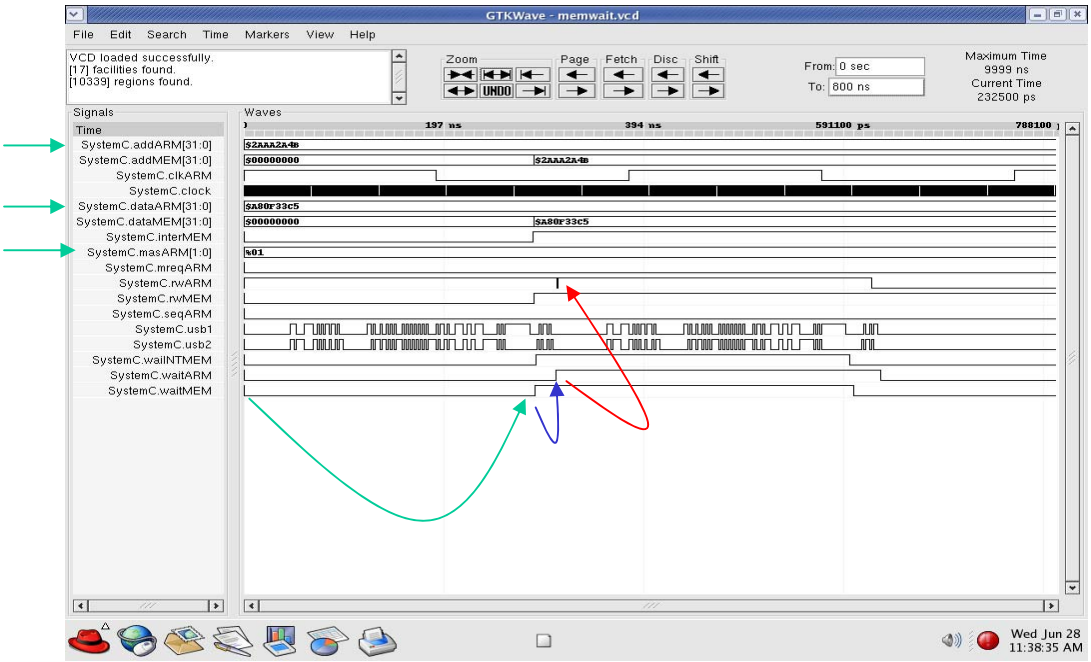

Fig. 5 Memory Write cycle with one Wait state

6. Conclusions

In proposed implementation, bus width reduces to USB width, which reduces area on chip/board. If USB interface is integrated with processor on SoC, pin count will be reduced leading to a smaller chip size. This reduction would be difficult in the absence of proposed

USB interface even if feature size is reduced, as pin size as well as pad size cannot be reduced in the same proportion as of feature size reduction.

7. Future work

In the time to come, we plan to enhance the features of our system including those features, which we have not implemented. The power estimation of the proposed system can be done once we are able to synthesize. For the purpose, we need SystemC to hardware description language (HDL) translator. Once RTL description at HDL level is available, any synthesis tool, commercially available can be used for design-synthesis. Power and performance evaluation of the proposed architecture can then be performed for comparison with existing approaches and architectures.

8. References

[1]. Kanishka Lahiri, A. Raghunathan and Sujit Dey. “Design space exploration for optimizing On-Chip communication networks.” IEEE transaction on CAD of ICS, Vol 23 No.6, June 2004.

[2]. Jan Axelson. “USB Complete 2.0” Penaram International, 2001.

[3]. A. Rajawat, M. Balakrishnan and Anshul Kumar. “Interface Synthesis: Issues and approaches” Department of CSE, IIT Delhi

[4]. Luca Benini and G. D. Michelli. “Networks on Chip: A new SoC paradigm” IEEE computers 2002.

[5]. G. Cyr, G. Bois and M. Aboulhamid. “Generation of Processor Interface for SoC using standard communication protocol” IEEE Proc. Comput. Digit Tech., 151 No. 5, Sep 2004.

[6]. Eung Shin, K. K. Ryu and Vincent J. Moonov. “A Comparison of Five Different Multiprocessor SoC Bus Architecture.” Proc. of EUROMICRO systems on digital system design, Sep 2001.

[7]. Salim Ouadjaout and Dominique Houzet. “Easy SoC Design with VCI SystemC adapters”, Proc. Of EUROMICRO systems on digital system design, 2004.

[8]. IBM CoreConnect Bus architecture http://ibm.com

[9]. AMBA specifications (Rev 2.0) http://www.arm.com

[10]. ARM 7TDMI TRM. http://www.arm.com

[11]. USB 2.0 specifications. http://usb.net.org

[12]. Functional Specification for SystemC 2.0.1. http://www.systemc.org