Volume 01, No. 01

International Journal of Scientific Engineering and Technology Research

Jul-Dec 2012, P.P. 32-35

www.semargroups.org

IP CORE DEVELOPMENT OF AMBA AHB BUS TRACER

DEVAPATI HEMANTH VEERA KUMAR[1], A.BALAJI NEHRU[2], J.B.S KUMAR[3]

[1]M.Tech Student of CMR Institute of Technology, Hyderabad, AP-India, hemanthveera2010@gmail.com,
[2] HOD,ECE Dept, CMR Institute of Technology,Hyderabad, AP-India.

Abstract: This paper proposes an IP core development of AMBA AHB on-chip bus tracer for versatile system-on-chip (SoC) debugging and monitoring. The bus tracer is capable of capturing the bus trace with different AHB signals, all with efficient built-in compression mechanisms, to meet a diverse range of needs. In addition, it allows users to switch the trace signals dynamically so that appropriate signal levels can be applied to different segments of the trace. It mainly contains of Event Generation Module, Abstraction Module, Compression Modules, and Packing Module. The Event Generation Module controls the start/stop time, the trace mode, and the trace depth of traces. This information is sent to the following modules. Based on the trace mode, the Abstraction Module abstracts the signals in both timing dimension and signal dimension.

Keywords- *AMBA,SoC,IP,AHB,ASIC.*

I. INTRODUCTION

Over the years, System-on-Chip (SoC) designs have evolved from fairly simple uni-processor, single-memory designs to massively complex multiprocessor systems with several on-chip memories, standard peripherals and ASIC blocks. As more and more components are integrated into these designs to share the ever increasing processing load, there is a corresponding increase in the communication between these components. Inter-component communication is often in the critical path of a SoC design and is a very common source of performance bottlenecks. Silicon densities, both for ASICs and FPGAs, can now support true system- on-chip (SoCs). This level of design requires busing systems to connect various components, including one or more microprocessors, memory, peripherals, and special logic. AMBA, the Advanced Microprocessor Bus Architecture, is ARM's on-chip busing solution.

ARM processors have several unique advantages over traditional microprocessor solutions in terms of performance, small die size, and extremely low power consumption. ARM provides developers with intellectual property (IP) in the form of processor core designs, cache and SoC designs, application-specific standard products, and related software development tools. Bus protocol play an important role in the field of intellectual property reuse by standardizing the interface of hardware components, they simplify the task of transplanting a module from one system to another. The bus is the mechanism by which a processing element communicates with other processing elements, with memory and with devices. A bus is, at minimum a collection of wires, but the bus also defines a protocol by which the processing elements, memories and devices communicate.

THE ON-CHIP bus is an important system-on-chip (SoC) infrastructure that connects major hardware components. Monitoring the on-chip bus signals is crucial to the SoC debugging and performance analysis/ optimization. Unfortunately, such signals are difficult to observe since they are deeply embedded in a SoC and there are often no sufficient I/O pins to access these signals. Therefore, a straightforward approach is to embed a bus tracer in SoC to capture the bus signal trace and store the trace in an on-chip storage such as the trace memory which could then be off loaded to outside world (the trace analyzer software) for analysis. Unfortunately, the size of the bus trace grows rapidly. For example, to capture AMBA AHB 2.0 bus signals running at 200 MHz, the trace grows at 2 to 3 GB/s. Therefore, it is highly desirable to compress the trace on the fly in order to reduce the trace size. However, simply capturing/ compressing bus signals is not sufficient for SoC debugging and analysis, since the debugging/ analysis needs are versatile: some designers need all signals at cycle-level, while some others only care about the transactions. For the latter case, tracing

all signals at cycle-level wastes a lot of trace memory. Thus, there must be a way to capture traces at different abstraction levels based on the specific debugging/analysis need.

This paper presents a real-time multi-IP core development of AHB on-chip bus tracer, the bus tracer have compression mechanisms to achieve high trace compression ratio. It support cycle-to cycle tracing by capturing traces at different timing and signal abstraction levels to match specific debugging/ analysis needs. Given a trace memory of fixed size, the user can tradeoff between the granularities and trace length to make the most use of the trace memory. In addition, the bus tracer is capable of tracing signals before/after the event triggering. This feature provides a more flexible tracing to focus on the interesting points.

II.Advanced High-Performance Bus

AHB (Advanced High-performance Bus) is the latest generation AMBA (Advanced Microcontroller Bus Architecture) bus. It is intended to address the requirements of high performance synthesizable designs. Many system-on-a-chip designs in the portable electronics, telecommunications, and embedded systems markets use AHB to interface application specific design blocks with standard microcontrollers. It is a high-performance system bus that supports multiple bus masters and provides high-bandwidth operation.

An AHB bus master has the most complex bus interface in an AMBA system. Typically an AMBA system designer would use predesigned bus masters and therefore would not need to be concerned with the detail of the bus master interface.

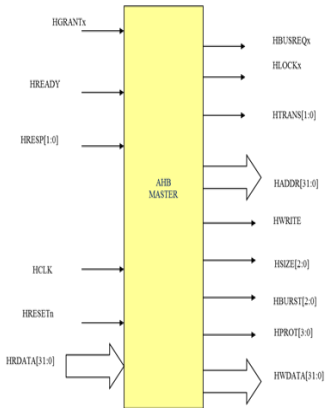

Fig 1: AMBA AHB Master

III.AMBA AHB TRACER

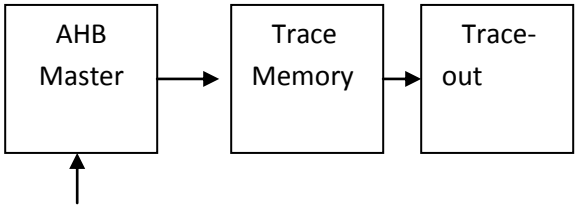

Fig 2: Existed AMBA AHB Tracer

It mainly contains of: Event Generation Module, Abstraction Module, Compression Modules, and Packing Module. The Event Generation Module controls the start/stop time, the trace mode, and the trace depth of traces.

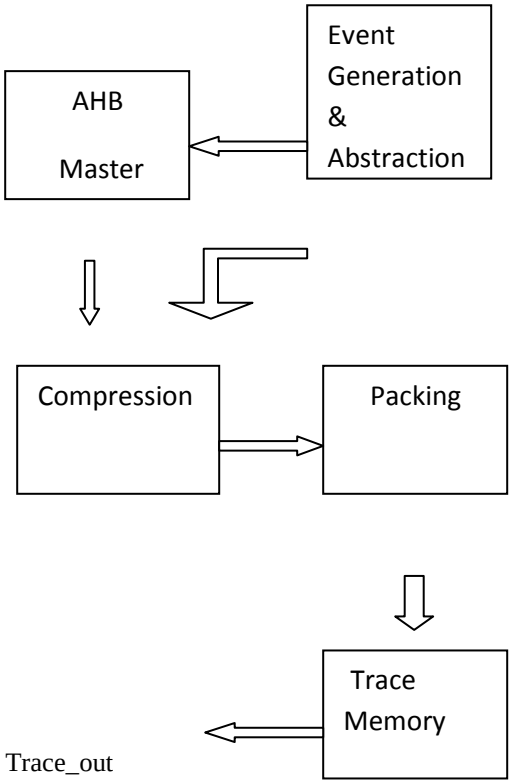

Fig 3: Proposed AMBA AHB Tracer

This information is sent to the following modules. Based on the trace mode, the Abstraction Module abstracts the signals in both timing dimension and signal dimension. The abstracted data are further compressed by the Compression Module to reduce the data size. Finally, the compressed results are packed with proper

headers and written to the trace memory by the Packing Module.

Event Generation Module: The Event Generation Module decides the starting and stopping of a trace and its trace mode.The module has configurable event registers which specify the triggering events on the bus and a corresponding matching circuit to compare the bus activity with the events specified in the event register

Compression Module: The purpose of the Compression Module is to reduce the trace size. It accepts the signals from the abstraction module. To achieve real time compression, the Compression. Module is pipelined to increase the performance. Dictionary-Based to reduce the size, we take the advantage of the temporal locality. Temporal locality exists since the basic blocks repeat frequently (loop structure), which implies the addresses of the Location. We can use the dictionary- based compression. The idea is to map the data to a table keeping frequently appeared data, and record the table index instead of the data to reduce size of tracing memory. The dictionary keeps the frequently appeared branch/ target addresses. To keep the hardware cost reasonable, the proposed dictionary is implemented with a ROM based Memory.

Packing: The compressed results are packed with proper headers and written to the trace memory by the Packing Module. The Packing Module is the last phase. It receives the compressed data from the compression module, processes them, and writes them to the trace memory. It is responsible for three jobs: packet management, circular buffer management, and mode change control. For packet management, since the compressed data length and type are variable, every compressed data needs a header for interpretation. Therefore, this step generates a proper header and attaches it to each compressed datum. In this paper, we call a compressed data with a header as a packet. Since the header generation takes time, to avoid long cycle time, the header generation is implemented in one pipeline stage.

Trace Memory: The tracing packet come out from the packet module stored into the Trace memory. This trace memory we can design using FIFOs.The trace for circular buffer management, it manages the accesses to the trace memory. Since the size of a packet is variable but the data width of the trace memory is fixed, this module collects the trace data in a first-input, first-output (FIFO) buffer and outputs them to the trace memory until the data size in the FIFO buffer is equal/larger than the data width. If the tracing stops and the data size in the FIFO buffer is smaller than the data width, one additional cycle is required to output the remaining data to the trace memory.

IV. SIMULATION RESULT's

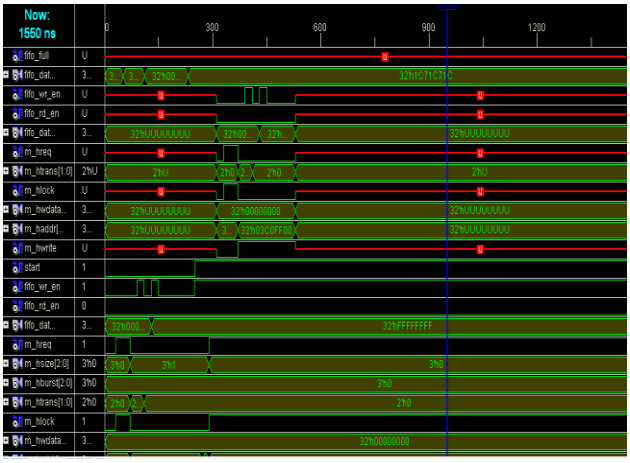

Fig 4: Simulation Results of AHB Tracer without Compression.

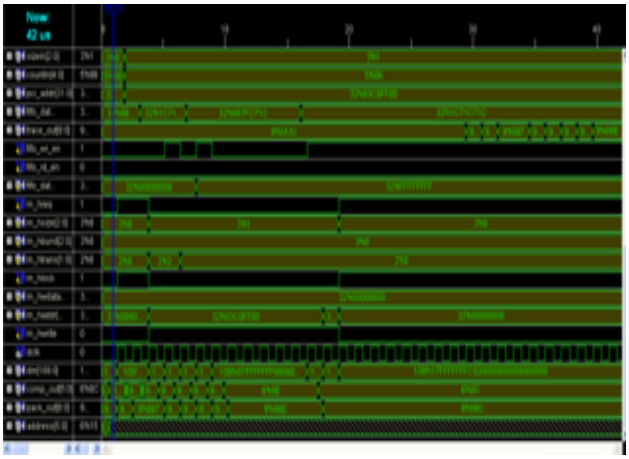

Fig.5: simulation results of AMBA AHB Tracer with compression

Project File:	pt11vl076.ise	Current State:	Synthesized
Module Name:	top_tracer	• Errors:	No Errors
Target Device:	xc3s250e-4tq144	• Warnings:	13 Warnings
Product Version:	ISE 9.1i	• Updated:	Tue Sep 4 02:34:52 2012

PT11VL076 Partition Summary
No partition information was found.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	283	2448	11%
Number of Slice Flip Flops	230	4896	4%
Number of 4 input LUTs	502	4896	10%
Number of bonded IOBs	122	108	112%
Number of BRAMs	1	12	8%
Number of GCLKs	2	24	8%

Fig 6: Design summary report for AHB Tracer

The Fig 4. Shows the AMBA AHB Tracer without compression. Fig 5. Shows the The Fig 4. Shows the AMBA AHB Tracer without compression. Fig 5. Shows the AMBA AHB Tracer with compression Technique. The Design report of Tracer of Spartan 3e FPGA showed in fig.6

V. CONCLUSION

Tracing of AMBA AHB Bus successfully developed using VHDL. Compression Technique is successfully implemented to reduce the storage of Trace Memory size. And Total design functionally verified using ISE simulator and Synthesized by Xilinx 9.1i.

VI. REFERENCES

[1]A Multi-resolution AHB Bus Tracer for Real-time Compression of Forward/Backward Traces in a Circular Buffer Yi-Ting Lin, Wen-Chi Shiue, and Ing-Jer Huang Department of Computer Science and Engineering National Sun Yat-Sen University Kaohsiung 804, Taiwan

[2]An On-Chip AHB Bus Tracer With Real-Time Compression and Dynamic Multiresolution Supports for SoC Fu-Ching Yang, Member, IEEE, Yi-Ting Lin, Chung-Fu Kao, and Ing-Jer Huang, Member,IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 19, NO. 4, APRIL 2011

[3] ARM. Example AMBA SYstem User Guide ARM DUI0092C, Aug. 1999.

[4] ARM. Embedded Trace Macrocell Architecture Specification, Feb. 2006.

[5] ARM. AMBA AHB Trace Macrocell (HTM) Technical Reference Manual ARM DDI 0328D, 2007.

[6] First Silicon Solutions. Preliminary Technical Data for AMBA Navigator AMBA On-Chip Bus Analyzer for AHB Bus Systems.

[7] J. Gaisler, E. Catovic, M. Isomaki, K. Glembo, and S. Habinc. GRLIB IP Core User's Manual. Gaisler Research.

[8] R.-T. Gu, T.-C. Yeh, W.-S. Hunag, T.-Y. Huang, C.-H. Tsai, C.-N. Lee, M.-C. Chiang, S.-F. Hsiao, Y.-N. Chang, and I.-J. Huang. A low cost tile-based 3D graphics full pipeline with real-time performance monitoring support for OpenGL ES in consumer electronics. In Proceedings of IEEE International Symposium of Consumer Electronics, 2007.

[9] W.-J. Huang, N. Saxena, and E. J. McCluskey. A reliable LZ data compressor on reconfigurable coprocessors. In IEEE Symposium on Field-Programmable Custom Computing Machines, 2000.

[10] Infineon Technologies. TC1775 TriCore User's Manual System Units section 20, on-chip debug support, Feb. 2001.

[11] E. E. Johnson, J. Ha, and M. B. Zaidi. Lossless trace compression. IEEE Trans. Comput., 50:158–173, Feb. 2001 ol. 147, pp. 539–547 (1994).