A REVIEW : AXI BUS INTERFACING BETWEEN FPGA AND HPS IN MSS HUB NETWORK

Arohi Parekh[1], Chintan Dave[2], Abhimanyu Dhiman[3]

[1]*P.G. Student, VLSI and Signal Processing (EC), VGEC-Chandkheda, Gujarat, India*
[2]*Asst. Prof., Electronics and Communication Engineering, VGEC-Chandkheda, Gujarat, India*
[3]*Scientist SC, SAC-ISRO, Ahmedabad, Gujarat, India*

ABSTRACT

In MSS (mobile satellite services) services there are different products that we are using like SGU (satellite gateway unit), VGU (voice gateway unit), TGU (traffic gateway unit) so on. These different products are based on rabbit processor technology. For advanced hub (MSS), this product needs to be optimized for a high speed on FPGA devices. In SOC FPGA, input is ethernet data (TCP/IP) and output can be serial data, HDLC data or some data conversion. In SOC FPGA, LAN/IP stack, serial data is handled by the Arm processor and protocol conversion in HDLC and compute intensive tasks can be done in parallel, pipelined fashion in FPGA to transfer high speed ethernet traffic in a SOC/FPGA. To route ethernet traffic from Arm processor to FPGA and vice versa ***AXI interface*** *is used that gives high speed. We have a AXI master on HPS side, so to connect FPGA, QSYS will automatically connect adapters of AXI slave and MM master that consumes most of the logic elements and reduces the speed of transfer. So, I will make AXI slave on FPGA side that will solve this both the problems.*

Keyword: - *Rabbit Processor, SOC, AXI interface, FPGA, Avalon MM interface*

1. INTRODUCTION

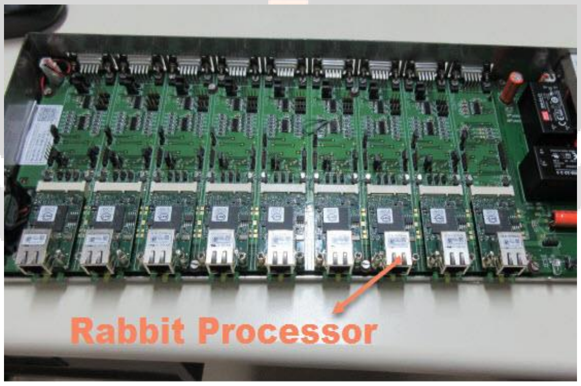
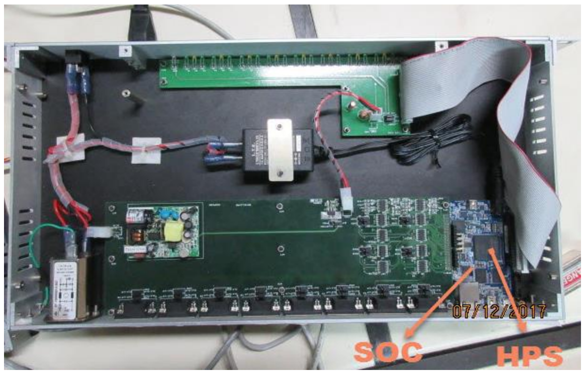


Fig. 1.1 Rabbit Processors based SGU

At before Rabbit Processor 6000 is used in SGU. There are 9 Rabbit processor in the figure. This is for 8 channel, 1st one is controlling other 8 processors. Disadvantage is the Rabbit contains 1MB of internal high-speed 16-bit RAM and network connectivity — a full 10/100 (10 to 100 mbps) Base-T Ethernet MAC and PHY built into the device. Out of Six serial ports four async/clocked serial and **Two Port E-F for HDLC data.**

- **Why we are using SOC?**



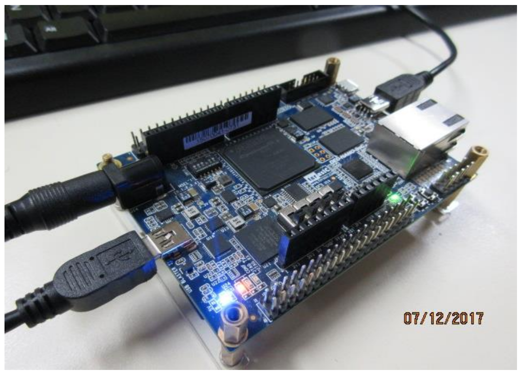


Fig. 1.2 Rabbit Processors based SGU **Fig. 1.3 SoC based SGU**

Altera Cyclone V FPGA with ARM cortex-A9 Hard Processor
The Cyclone V device is a single-die system on a chip (SoC) that consists of two distinct parts—

- A Hard processor system (HPS)
- A FPGA

In virtual every aspect, CPUs and FPGAs are radically different devices. And yet, they often compete for some of the same embedded system tasks. FPGAs are user configurable hardware logic, while CPUs are fixed arithmetic engines executing user programs.

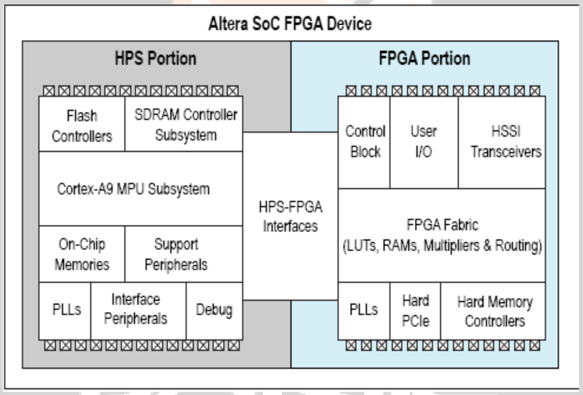


Fig. 1.4 Altera SoC FPGA Device

ARM cortex-A9 Hard Processor main advantages are, it has 925MHz Dual-core and a full 10/100/1000 Base-T Ethernet. Mainly Cortex- A9 is working on linux based kernel and Bare metal. We can also use it as a custom linux. SoC is hardware programmable device so we can use it as a both type of logics, functional logic and interfacing logic. But Rabbit Processor works on fixed OS.

Our job target is to design AXI slave on FPGA side so we can eliminate adapters and connect directly AXI slave to the AXI master so it will increase the speed of transfer and also remove the problem of logic elements.

- **FPGA-to-HPS bridge (reverse link) —** a high-performance AXI bus with a configurable data width of 32, 64, and 128 bits, allowing the FPGA fabric to master transactions to the slaves in the HPS.
- **HPS-to-FPGA bridge (forward link) —** a high-performance AXI interface with a configurable data width of 32, 64, and 128 bits, allowing the HPS to master transactions to slaves in the FPGA fabric.
- **Lightweight HPS-to-FPGA Bridge—** an AXI interface with a 32-bit fixed data width, allowing the HPS to master transactions to slaves in the FPGA fabric.

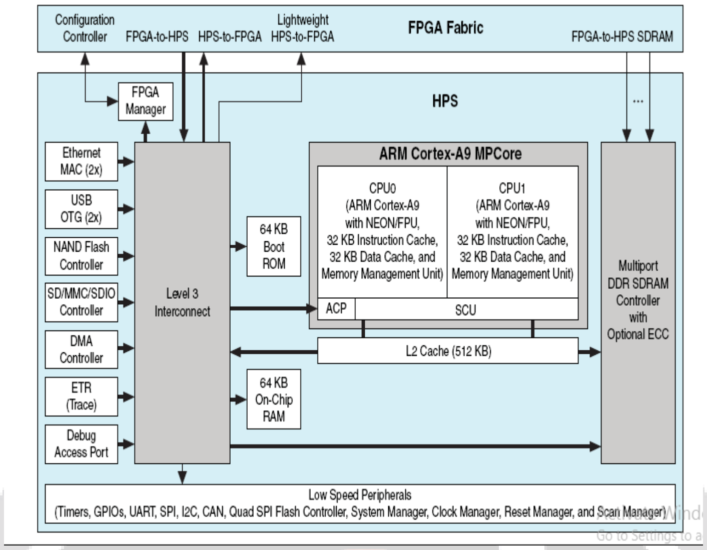
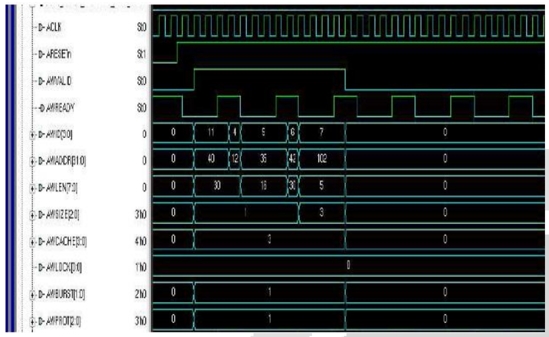

Configuration Controller
FPGA-to-HPS
HPS-to-FPGA
Lightweight HPS-to-FPGA
FPGA Fabric
FPGA-to-HPS SDRAM
FPGA Manager
HPS
Ethernet MAC (2x)
USB OTG (2x)
NAND Flash Controller
SD/MMC/SDIO Controller
DMA Controller
ETR (Trace)
Debug Access Port
Level 3 Interconnect
64 KB Boot ROM
64 KB On-Chip RAM
ARM Cortex-A9 MPCore
CPU0 (ARM Cortex-A9 with NEON/FPU, 32 KB Instruction Cache, 32 KB Data Cache, and Memory Management Unit)
CPU1 (ARM Cortex-A9 with NEON/FPU, 32 KB Instruction Cache, 32 KB Data Cache, and Memory Management Unit)
ACP
SCU
L2 Cache (512 KB)
Multiport DDR SDRAM Controller with Optional ECC
Low Speed Peripherals (Timers, GPIOs, UART, SPI, I2C, CAN, Quad SPI Flash Controller, System Manager, Clock Manager, Reset Manager, and Scan Manager)

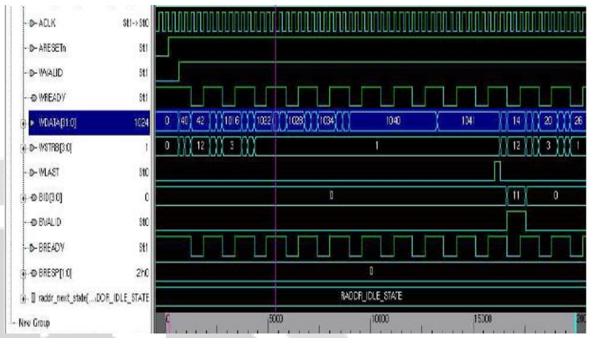
Fig. 1.5 HPS with Dual-Core ARM Cortex-A9 MPCore Processor

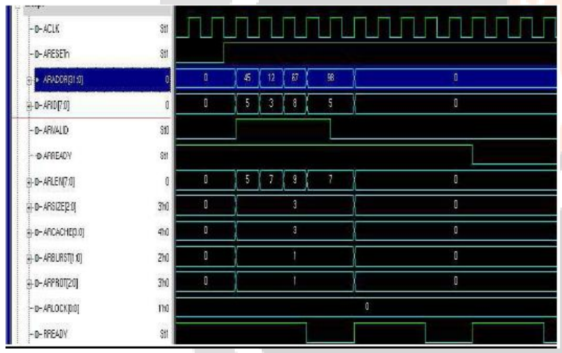
2. LITERATURE REVIEW

Anitha. H.T et. al. [1] With the need of application, chip with a single processor can't meet the need of more and more complex computational task. We are able to integrate multiple processors on a chip. Multi-Processor System on Chip (MPSoC) which gives a solution to this requires efficient on-chip communication architectures to support high data bandwidth and increase parallelism. However, traditional buses only allow one master to access one slave at one time, which badly restricts the performance of the whole system. In this paper we focus on the design and implementation of a multi slave interface for AXI bus, which translates data in burst, maximal length of which is up to 16 transactions. Besides, it only needs to translate the head address of the burst in this transaction. Owing to that feature, multiple masters accessing multiple slaves at one time becomes possible in sharing address bus architecture. Here we design and implement AXI interface for two slaves i.e., internal memory and register bank with one master. The address/data is transferred from master to slave using write address/data channel and read the data from slave using read data channel. Here the read and write transactions occur simultaneously because there are five different channels for read and write and no multiplexing is required. The entire design is described using Verilog HDL.

Shaila S Math et. al. [2] Advanced microcontroller bus architecture (AMBA) protocol family provides metric-driven verification of protocol compliance, enabling comprehensive testing of interface intellectual property (IP) blocks and system-on-chip (SoC) designs. The AMBA advanced extensible interface 4 (AXI4) update to AMBA AXI3 includes the following: support for burst lengths up to 256 beats, updated write response requirements, removal of locked transactions and AXI4 also includes information on the interoperability of components. AMBA AXI4 protocol system supports 16 masters and 16 slaves interfacing. This paper presents a project aimed to do data transactions on SoC bus using AMBA AXI4 protocol modelled in Verilog hardware description language (HDL) and simulation results for read and write operation of data and address are shown in Verilog compiler simulator (VCS) tool. The operating frequency is set to 100MHz. Two test cases are run to perform multiple read and multiple write operations. This paper conclude that AMBA AXI4 is a plug and play IP protocol released by ARM, defines both bus specification and a technology independent methodology for designing, implementing and testing customized high-integration embedded interfaces. The data to be read or written to the slave is given by the master and is read or written to a particular address location of slave through decoder. In this work data transactions were carried out using AMBA AXI4 protocol modeled in Verilog hardware description language (HDL) and simulation results for read and write operation of data and address are shown in Verilog compiler simulator (VCS) tool. To perform single read operation module consumed 160ns and for single write operation 565ns.


(a)


(b)


(c)

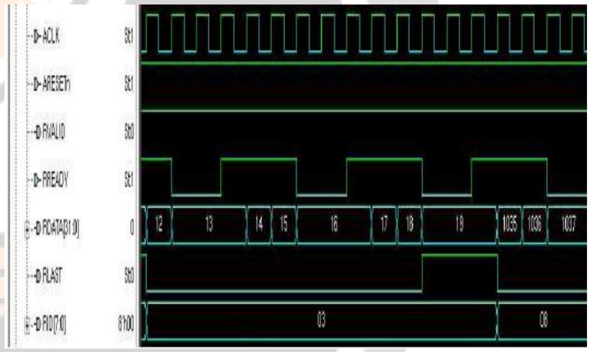

(d)

Fig. 2.1 (a) Simulation result of write address operation (b) Simulation result of single write data operation (c) Simulation result of read address operation (d) Simulation result of single read data operation

Ramesh Bhaktavatchalu et. al. [3] This paper describes the design and implementation of programmable AXI bus Interface modules in Verilog Hardware Description Language (HDL) and implementation in Xilinx Spartan 3E FPGA. Successful System-On-Chip Communication between two masters and four slaves has been accomplished in this paper.All the interface modules are reconfigurable with the data size, burst type, number of transfers in a burst. Multiple masters can communicate with different slave memory locations concurrently. An arbiter controls the burst grant to different bus masters based on Round Robin algorithm. Separate decoder modules are implemented for write address channel, write data channel, write response channel, read address channel, read data channel. The design can support a maximum of 16 masters. All the RTL simulations are performed using Modelsim RTL Simulator. Each independent module is synthesized in XC3S250EPQ208-5 FPGA and the maximum speed is found to be 298.958 MHz. All the design modules can be integrated to create a soft IP for the AXI BUS system. ARM Cortex processors are used in smart phones which demand high data transfer speeds. Thus the need for high speed, high throughput protocols is increasing. In this paper, AXI bus interface modules were implemented in Modelsim using verilog HDL and simultaneous read and write data transfers among a Soc of two masters and four slaves were simulated in Modelsim SE and all the modules were synthesized on a Spartan 3E FPGA. The maximum throughput achievable with the AXI protocol occurs for 16 data transfers. The calculated throughput is 5.51 bits/clock cycle for write and 7.619 bits/clock cycle for read transfer each of 6 transfers.

Fu-ming Xiao et. al. [4] While the computational core is becoming faster and faster, the communication efficiency between the processors has become a bottleneck which limits the performance of multiprocessor system-on-chip (MPSoC). This paper focuses on design and implementation of AXI bus protocol-based MPSoC architecture. Firstly, the RTL models of 4 NIOS II processors using AXI communication architecture are developed. Then the MPSoC was implemented in Altera Stratix II EP2S180 FPGA. Lastly, the performance was evaluated using matrix operation benchmark and compared with previous in-house designed architecture. Experiments showed that the proposed prototype could run at 100 MHz requiring 8963 Adaptive Look-up Table (ALUTs) and the maxim speedup ratio can be up to 3.81, and performs better than the traditional bus (AHB bus) and 2-D mesh NoC architecture. This paper concludes that, we design an AXI bus which offers high-lever service at high frequency. To evaluate the performance of AXI based MPSoC, we perform a prototype which includes four cores. Two real applications are mapped on the prototype and the experimental results show that the AXI-based MPSoC has the best performance of speedup ratio 3.81 under MMs application when we integrate four cores in the prototype. The hardware overhead is measured by FPGA utilization. The entire prototype requires 7% resource of the Altera Stratix II, including 8962 ALUTs, 7234 logic registers and 2491904 block memory bits. As shown in table I, we can see the resource usage of each component. The LPS uses most ALUTs and memory bits resources of the prototype because of that it includes a NIOS II processor. And each NIOS II processor has integrated hardware multiply as computational elements, so the LPS uses 8 DSP block 9-bits elements.

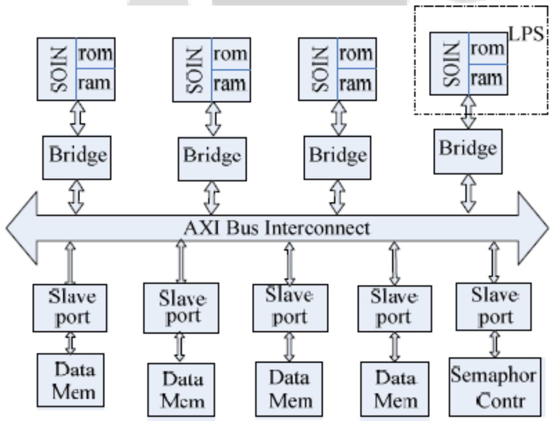


Fig. 2.2 AXI Based MPSoC Prototype Architecture

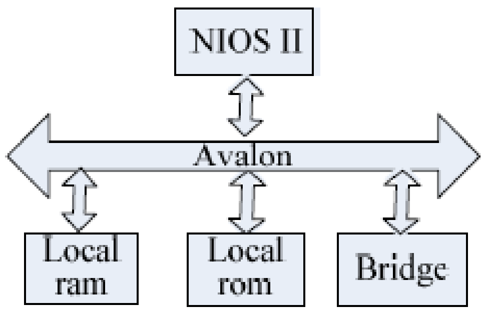
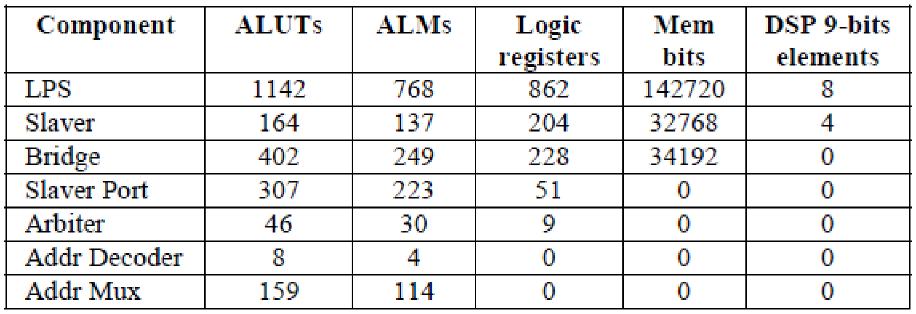


Fig. 2.3 Local Processing System Architecture



Component	ALUTs	ALMs	Logic registers	Mem bits	DSP 9-bits elements
LPS	1142	768	862	142720	8
Slaver	164	137	204	32768	4
Bridge	402	249	228	34192	0
Slaver Port	307	223	51	0	0
Arbiter	46	30	9	0	0
Addr Decoder	8	4	0	0	0
Addr Mux	159	114	0	0	0

Table 2.4 Component utilizations in FPGA (Altera Stratix ii)

3. PROBLEM STATEMENT

Altera's Qsys system integration tool is able to integrate IP components using the AMBA AXI interface and ALTERA AVALON interface. So, while connecting master and slave, Qsys tool automatically added adapters that will consume more than 50% logic elements and also reduces the speed of data transfer.

4. OBJECTIVES

1. My job target is to design AXI slave on FPGA side in software.
2. If I can eliminate that adapters and connect directly to the AXI master and that will increase the speed.
3. That will remove the problem of logic elements.

5. CONCLUSIONS

1. From this research papers I conclude that The Avalon interface family defines interfaces appropriate for streaming high-speed data, reading and writing registers and memory, and controlling off-chip devices. These standard interfaces are designed into the components available in Qsys.

2. AXI interface has separate address/control and data phases, support for unaligned data transfers using byte strobes, burst-based transactions with only start address issued, separate read and write data channels to enable low-cost *Direct Memory Access* (DMA) and ability to issue multiple outstanding addresses.

3. If I can eliminate that adapters and connect directly to the AXI master and that will increase the speed of transfer. That will also remove the problem of logic elements.

6. REFERENCES

[1] Anitha H.T., Nataraj K.R. "Implementation of multi-slave interface for AXI bus" Communication and Computing (ARTCom2012), Fourth International Conference on Advances in Recent Technologies, 19-20 Oct. 2012.
[2] Fu-ming Xiao, Dong-sheng Li, Gao-ming Du, Yu-kun Song , Duo-li Zhang, Ming-lun Gao "Design of AXI Bus Based MPSoC on FPGA" 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication, 20-22 Aug. 2009.
[3] Ramesh Bhaktavatchalu, B. Sasi Rekha, G. Ananta Divya, V. Usha Sai Jyothi "Design of AXI bus interface modules on FPGA"_2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), 25-27 May. 2016.
[4] Shaila S. Math, R. B. Manjula, S. S. Manvi, Paul Kaunds, "Data Transactions on System-On-Chip Bus Using AXI4 Protocol" 2011 international conference on recent advancements in electrical, electronics and control

engineering, 15-17 Dec. 2011.
[5] http://web.eecs.umich.edu/~prabal/teaching/eecs373f12/readings/ARM_AMBA3_APB.pdf
[6] http://mazsola.iit.unimiskolc.hu/~drdani/docs_arm/IHI0033A_AMBA3_AHB_Lite.pdf
[7] https://www.terasic.com.tw/attachment/archive/941/DE0-Nano-SoC_User_manual.pdf\
[8] http://mazsola.iit.unimiskolc.hu/~drdani/docs_arm/AMBAaxi.pdf
[9] http://www.gstitt.ece.ufl.edu/courses/fall15/eel4720_5721/labs/refs/AXI4_specification.pdf
[10] http://ftp1.digi.com/support/documentation/90001108_H.pdf

Book:
[1] Verilog HDL – Samir palnitkar

Website:
[1] www.terasic.com
[2] https:/rocketboards.org/
[3] www.altera.com