

HIBI-based Multiprocessor SoC on FPGA

Erno Salminen, Ari Kulmala, and Timo D. Hämäläinen

*Institute of Digital and Computer Systems
Tampere University of Technology
P.O. Box 553, FIN-33101 Tampere, Finland
erno.salminen@tut.fi*

Abstract — FPGAs offer excellent platform for System-on-Chips consisting of Intellectual Property (IP) blocks. The problem is that IP blocks and their interconnections are often FPGA vendor dependent. Our HIBI Network-on-Chip (NoC) scheme solves the problem by providing flexible interconnection network and IP block integration with Open Core Protocol (OCP) interface. Therefore, IP components can be of any type: processors, hardware accelerators, communication interfaces, or memories. As a proof of concept, a multiprocessor system with eight soft processor cores and HIBI is prototyped on FPGA. The whole system uses 36402 logic elements, 2.9 Mbits of RAM, and operates on 78 MHz frequency on Altera Stratix 1S40, which is comparable to other FPGA multiprocessors. The most important benefit is significant reduction of the design effort compared to system specific interconnection networks. HIBI also presents the first OCP compliant IP-block integration in FPGA.

I. INTRODUCTION

As size of system-on-chip (SoC) devices grow, reusing pre-designed and verified components (intellectual property, IP) is a necessity for successful products. The importance of simple integration and efficient communication between components increase rapidly with the size of system. Standardized interfaces, such as OCP (Open Core Protocol) [1], aim at plug-and-play integration of all IP components regardless of the source. Also, the type of IP (processor, hardware accelerator, memory) should be irrelevant as long they conform to common interface.

Easy and dependable integration and communication mechanism for different types of IPs eases architectural exploration, that is selecting the most appropriate components separately for each application (domain). Instead of long architecture exploration, new products can be designed on top of existing technology platform that defines the basic architecture and the IPs but that can be configured and augmented with additional hardware. Still, the interface is a key issue also in platform-based designs.

FPGA (field programmable gate array) devices, originally used only for glue logic or small prototypes, are replacing ASICs (application specific integrated circuit) in many products due to their increased capacity, smaller NRE (non-recurring engineering) costs, and programmability [2][3]. However, most FPGA IPs are currently vendor-specific limiting their reusability [4]. This paper presents a method for efficiently connecting IP

components on FPGA by using HIBI Network-on-Chip (NoC) [5]. A eight-processor system is prototyped. HIBI is, to our knowledge, one of the first communication networks on FPGA providing OCP interface.

II. RELATED WORK

FPGA devices can have parts implemented in ASIC technology for high performance, for example embedded processors and multipliers. A recent innovation in usage of IP components is the introduction of synthesizable *soft processors*, such as Nios [6], that can be (ideally) targeted to any implementation technology. The number of soft cores can be freely chosen which is not the case with embedded hard cores. The performance may be enhanced with special instructions and other hardware accelerators or by building a multiprocessor system. Technological advances enable implementation of multiprocessor System-on-Chip (MPSOC) on a single modern FPGA or, in other words, System-on-Programmable-Chip (SOPC).

There are many multiprocessor systems but only few are implemented on a single FPGA. Some are listed in Table 1. Socrates system includes two ARM7 compliant processors and memories [7]. Martina *et al.* have developed a DSP core and prototyped a system with eight DSP cores running basic signal processing algorithms [8]. Wang and Ziavras have presented a 6-processor system optimized for LU factorization for matrices. All three utilize soft processors of the same type. A single bus is used in [7] and [8] but [9] uses fully connected, multimaster Altera Avalon bus [11]. In a such bus, each slave can be connected to multiple masters through an

TABLE 1: FPGA MULTIPROCESSORS

MPSOC	FPGA	Num of CPUs	CPU type	Area	Freq. [MHz]
Socrates [7]	Virtex 1000	2	ARM7 compatible	7 234 slices / 653 349 gates	15
Martina [8]	Virtex 1000	8	DSP	á 3 024 LUT	89
Wang [9]	Apex 20KE	6	Nios+ FPU	á 5 900 LE	40
Hermes [10]	Virtex II 1000	3	R8	3058 slices, 6x18 Kbit	25

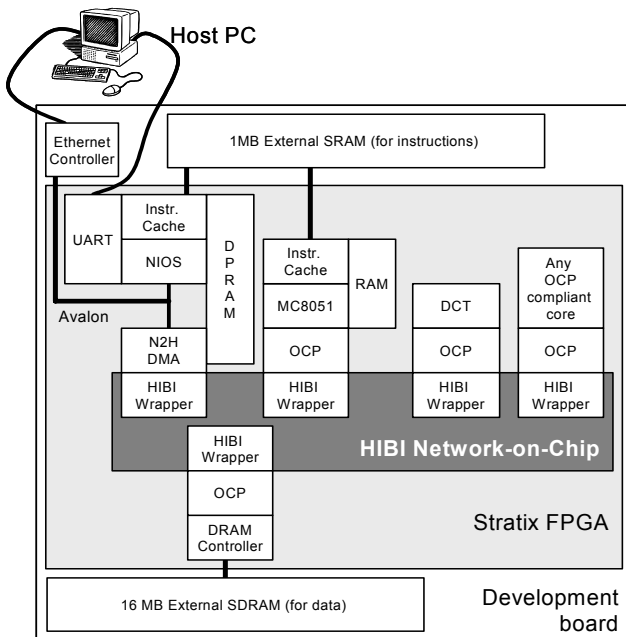


Figure 1. The structure of HIBI-based MPSOC architecture on FPGA development board.

arbiter. There may be several simultaneous transfers if masters access separate slaves. Avalon bus is utilized in Nios processors for interfacing peripherals. No information was found on using Avalon on other technologies than Altera FPGAs. Hermes NoC has been demonstrated with 2x2 mesh configuration having three R8 processors and serial interface [12].

III. INTEGRATION WITH HIBI NOC

HIBI (Heterogeneous IP Block Interconnection) [5] is a wrapper-based, scalable, and parameterizable Network-on-Chip targeted for efficient integration of coarse grain (e.g. several kilogates) components. The main properties of HIBI are summarized in Table 2. The interface to IP side can be selected to be (basic level) OCP, FIFO interface, or DMA (direct memory access) controller according to IP. HIBI is implemented as soft core so that it can be targeted to any implementation technology. The topology can be chosen and optimized according to application type and system-level requirements. Many parameters can be configured at run time if needed. The IP components can be processors, hardware accelerators, communication interfaces, or memories. HIBI does not limit the number nor the type of integrated components.

An example of SoC architecture based on HIBI NoC is shown in Fig. 1. Two examples of processor integration are shown on the left of Fig. 1. Nios includes local instruction cache and data memory, DMA controller, and some other peripherals. On the other hand, MC8051 does not utilize DMA but processor controls all transfers. Both processor sub-systems utilize OCP interface so that they are network independent. Local bus protocol of a processor, such as Avalon of Nios, is converted to OCP with wrappers.

Furthermore, several hardware accelerators, for example DCT (discrete cosine transform), can be easily added as they all conform to same protocol. In this example, the cache controllers are directly connected to off-chip instruction memory. The off-chip memory (controller) can also be connected to network with wrappers just like any other component as shown for data SDRAM memory.

IV. PROTOTYPE ARCHITECTURE

The multiprocessor SoC prototype consists of eight synthesizable 32-bit Nios processors [6] connected with a 32-bit HIBI. MPSOC is implemented using Altera development board that has a Stratix 1S40 FPGA, SRAM, SDRAM and flash memories, and several peripherals such as an ethernet controller as shown in Fig. 1. Stratix FPGA has 41250 logic elements (LE), each consisting of 4-input look-up-table (LUT) and a flip-flop, and 56 18x18 bit

TABLE 2: HIBI v.2 NETWORK-ON-CHIP CHARACTERISTICS

Property	HIBI v.2 implementation
Topology	Hierarchical bus with wrappers
Interface	OCP, FIFO, DMA ctrl, all asynchronous if needed
Description language	Synthesizable VHDL and SystemC
Switching type	Circuit-switching (within segments) and wormhole packet-switching (between segments)
Clocking	Multiple clock domains
Arbitration	Distributed, pipelined
Arbit. algorithm	Priority, round-robin, TDMA, combination
Compile-time configurable parameters	FIFO sizes, data width, addresses, initial configuration, number of config pages and their type (RAM/ROM), included properties
Run-time configurable	All arbitration parameters and algorithm, cycle counters, power mode
Quality-of-Service	TDMA, Send limit+priority/round-robin, multiple priorities for data, fast reconfiguration with pages, synchronization
Bus signals	Clock, Reset, Data/Address, Address valid, Command, Lock, Target full
Signal type	Unidirectional, all shared - no point-to-point
Bus resolution	OR network
Addressing	Multiple addresses per agent, multiplexed with data, allows multicast
Commands	8 commands: read request (split), write/multicast data (w/ prior), write/read config
Verification	stand-alone simulation, HW/SW co-simulation, hardware emulation, FPGA prototype
Test applications	10 processor H.263 encoder, WLAN baseband engine, H.263+WLAN, synthetic test cases, MPEG-4

embedded multipliers. In addition, it has over 3 Mbits of embedded dual-port SRAM. The development board is connected to host PC via 100 Mbit/s ethernet. All processors have local memories (32 Kbyte for instructions, 2 Kbyte for boot monitor program, and 8 Kbyte for data), timer and serial connection (UART) peripherals. Only one processor is connected to ethernet controller. Also, the external memories residing on development board could be used. Single HIBI segment is utilized, although hierarchical structures can be easily generated for high-performance systems.

A. Nios processor

Nios is a soft processor targeted for Altera FPGA devices. It can be configured by selecting the data width (16 or 32 bits), amount of memory, and which peripherals are included. Nios has a 5-stage pipeline and Harvard memory architecture. It supports also custom instructions for enhancing performance. Over 20x speedups have been reported [9]. C compiler allows easy portability of programs from other environments.

B. Nios-to-HIBI DMA controller

Nios processors use Avalon bus for connecting memories and peripherals. HIBI does not support Avalon protocol directly, and therefore, the developed DMA controller also implements the conversion between FIFO interface and Avalon. DMA controller is interrupt-driven so that it may operate on different clock than Nios. In transmit operation, the DMA controller reads data from dual-port RAM (DPRAM) and writes it into FIFO of HIBI wrapper. When receiving data from HIBI, controller writes data into

DPRAM until the received address changes or the user-defined maximum transfer size is reached. After that it sends an interrupt to Nios and gives a pointer to received data. After Nios has processed the received data, it notifies the DMA controller that the buffer is free for other transfers. There may be several pending transfers so that Nios can process data from one buffer while controller writes received data to other(s). The controller is implemented with parameterizable RTL VHDL.

V. RESULTS AND CONCLUSIONS

FPGA synthesis results for some NoC proposals from literature are shown in Table 3. In addition, an arbiter for multimaster Avalon was synthesized and is also shown in the table. Comparing results between different FPGA devices is not straightforward, and therefore, our synthesis results in Table 4 are shown for several technologies including 0.18 μ m CMOS ASIC technology. Syntheses were carried out using Precision RTL by Mentor Graphics and Design Compiler by Synopsys. There are two implementation styles for memories and buffers: FF-based uses flip-flops and EAB-based uses embedded memories. Multilayer AMBA is a similar concept as multimaster Avalon. In AMBA, full connectivity leads to quadratical increase in area when number of connected modules increases [3]. The NoCs [12][12][13] are all mesh-based, packet-switched networks. The buffers of HIBI were implemented with registers to leave maximum amount of memory for processors. Buffers take nearly half of the HIBI area. When IP is hardware accelerator needing only little memory, HIBI buffers can be

TABLE 3: RELATED NOC AND BUS IMPLEMENTATIONS ON FPGA

Network	FPGA	Configuration	Area	Freq. [MHz]
Multimaster Altera Avalon	Stratix	32b arbiter, N masters	$20 + 98(N-1)$ LE	220 $-12(N-1)$
Multilayer AMBA [3]	Virtex II 1000	full-connected 32b network, N modules	$230 * N^2$ slices $11385 * N^2$ gates	NA
Socrates [7]	Virtex 1000	32b, bus interface	352 slices 5873 gates	50
Hermes [12]	Virtex II 1000	8b router, 5x8-word buffers, with / without wrapper	278 slices 383 slices	25
Bartic <i>et al.</i> [12]	Virtex II Pro	16b router 5x272-word buffers	552 slices, 5 BRAMs 330000 gates	50
Rasoc [13]	Flex 10K	32b router, 5x4-word FF-based / EAB-based buffers	$1830 / 726$ LC, $745 / 90$ FF, $0 / 680$ bits	64

TABLE 4: OUR IMPLEMENTATION RESULTS

Component	FPGA	Configuration	Area	Freq. [MHz]
HIBI v.2	0.18 μ m CMOS ASIC	As below, 28b / 16b / 32b	5 200 gates / 7 600 gates / 12 400 gates	328 / 287 / 216
	Virtex II Pro	32b wrapper, 2x5+2x3-word	912 slices	39
	Flex 10K	FF-based buffers, 1-page config	2 116 LC, 1 155 FF, 0 bits	28
	Stratix	ROM	1 993 LE	124
Nios-to-HIBI DMA	Virtex II Pro	32b,	200 slices	68
	Flex 10K	4 pending transfers, 8kbit EAB	637 LC, 268 FF,	26
	Stratix	buffer	538 LE	123
Nios	Stratix	32b Nios, 8 KB instr, 32 KB data	2 316 LE, 620 Kbits	117
8x (Nios +DMA +HIBI)	Stratix	32b Nios, 32b DMA, 32b HIBI	36 402 LE, 2 911 Kbits	78

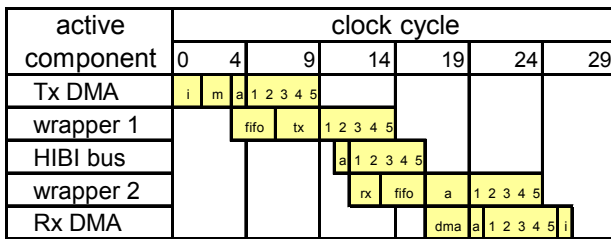


Figure 2. Timing diagram of a 5-word transfer

implemented with embedded dual-port memories to save logic elements.

The MPSOC takes 88% of logic and 87% of memory available on Stratix 1S40 FPGA. The biggest Stratix II FPGA offers logic capacity for extending the presented MPSOC architecture to over 20 processors. Both HIBI and Nios-to-HIBI provide the same maximum frequency as Nios although they are not optimized for any certain technology. The frequency of MPSOC is smaller than any of the components but that is mainly due to fact that more optimization effort was used when components were synthesized separately.

Timing of 5-word transfer over HIBI is shown in Fig. 2. At first, Nios sends an interrupt (*i*) to DMA which takes two clock cycles due to synchronization. Memory (*m*) causes one cycle read latency. After that DMA writes one datum on every cycle to HIBI wrapper. Transmit FIFO causes 3 cycle latency, and four cycles are taken due to bus contention and transfer FSM. Transfer on the bus is also six cycles, that is one cycle for address (*a*) and five for data (*1-5*). On receiver side, it takes few cycles for DMA to reserve Avalon bus and to start reading the received data. Finally, an interrupt request (*i*) is sent to receiving Nios. In this example, it takes 27 cycles for transferring 5 data, resulting in latency of 5.4 cycle/datum. For longer transfers, the latency per datum naturally decreases, for example, sending 50 data, results in latency $(50+22)/50 = 1.4$ cycle/datum. Furthermore, having two priorities causes two cycle latency in the transmitter and one cycle in the receiver. If only one priority is enough, latency decreases by three clock cycles. As a comparison, Hermes NoC requires at least 10 cycles per router and 2 cycles per each datum transfer [12]. Theoretical maximum bandwidth of HIBI prototype is $82 \text{ MHz} * 4 \text{ B} = 328 \text{ MB/s}$.

The multiprocessor system was tested with the aid of Transaction Generator that executes synthetic test cases [14]. The original Transaction Generator was aimed for simulation-based design space exploration on workstation. The program code was modified so that it can be executed on MPSOC. Applications are abstracted with a Kahn process graph that has a set of communicating processes. Each process is mapped onto one Nios and each Nios can have several processes. This way different combinations of process mappings can be compared. Fair comparison of SoC architectures is difficult as results are given for different applications. Synthetic test cases provide a promising way for creating an open benchmark set as they do not possess any proprietary information found in real applications. In

addition, there is an ongoing research on implementing MPEG-4 video encoder on our multiprocessor FPGA.

VI. CONCLUSION

FPGA IP block integration based on HIBI NoC was presented and prototyped using eight soft processor cores and memories. HIBI reduces design effort due to its standardized and wrapper based interfaces as well as flexible and configurable architecture. HIBI provides low latency in transfers with moderate hardware costs. Despite FPGAs, HIBI scheme can be used to ASIC SoCs as well. Future work includes implementation of heterogeneous topologies using bridges between HIBI segments and design automation support for mapping applications on multiprocessor SoC.

ACKNOWLEDGEMENT

This work has been supported by TES and Ulla Tuominen foundation. Authors would like to acknowledge Olli Lehtoranta, MSc, for his contribution in testing the MPSOC.

REFERENCES

- [1] OCP-IP Alliance, Open Core Protocol specification, Release 2.0, 2003.
- [2] P.G. Paulin, DATE panel: Chips of the future: soft, crunchy or hard?, DATE, Feb. 2004, Vol. 2, pp. 844 - 849.
- [3] H. Kalte, D. Langen, E. Vonnahme, A. Brinkmann, U. Ruckert, Dynamically reconfigurable system-on-programmable-chip, PDP, Jan. 2002, pp. 235 - 242.
- [4] I. Mackintosh, Advancing FPGA design efficiency: a proven standard solution, FPGA and Programmable Logic journal, [online], 17th Aug. 2004.
- [5] E. Salminen *et al.*, HIBI v.2 Communication network for system-on-chip, LNCS 3133 Computer Systems: Architectures, Modeling, and Simulation, A.D. Pimentel, S. Vassiliadis, (eds.), Springer-Verlag, Berlin, July 2004, pp. 412 - 422.
- [6] Altera, Nios 3.0 CPU, Data sheet, Version 2.1, March 2003.
- [7] M. Collin, M. Nikitovic, R. Haukilahti, SoCrates - a scalable multiprocessor system on chip, MSc thesis, Mälardalen University, May 2000
- [8] M. Martina, A. Molino, F. Vacca, FPGA system-on-chip soft IP design: a reconfigurable DSP, MWSCAS, Aug. 2002, Vol. 3, pp. III-196 - III-199.
- [9] Xiaofang Wang, S.G. Ziarvas, Parallel direct solution of linear equations on FPGA-based machines, IPDPS, Apr. 2003, pp. 113 - 120.
- [10] F. Moraes, N. Calazans, A. Mello, L. Möller, L. Ost, HERMES: an infrastructure for low area overhead packet-switching networks on chip, Integration, the VLSI Journal, Vol. 38, Iss. 1, Oct. 2004, pp. 69-93.
- [11] Altera, Avalon Bus Specification Reference Manual, Version 2.3, San Jose, CA, July 2003.
- [12] T.A. Bartic *et al.*, Highly scalable network on chip for reconfigurable systems, Int'l Symposium on System-on-Chip, Nov. 2003, pp. 79 - 82.
- [13] C.A. Zeferino, M.E. Kreutz, A.A. Ssin, RASoC: a router soft-core for networks-on-chip, DATE, Feb. 2004, Vol. 3, pp. 198 - 203.
- [14] T. Kangas, J. Riihimäki, E. Salminen, K. Kuusilinna, T.D. Hämläinen, Using a Communication Generator in SoC Architecture Exploration Int'l Symposium on System-on-Chip, Nov. 2003, pp. 105 - 108.