An OCP Compliant Network Adapter for GALS-based SoC Design Using the MANGO Network-on-Chip

Tobias Bjerregaard, Shankar Mahadevan, Rasmus Grøndahl Olsen* and Jens Sparsø

Informatics and Mathematical Modelling
Technical University of Denmark (DTU), 2800 Lyngby, Denmark
{tob, sm, jsp}@imm.dtu.dk, *s961394@student.dtu.dk

Abstract

The demand for IP reuse and system level scalability in System-on-Chip (SoC) designs is growing. Network-on-chip (NoC) constitutes a viable solution space to emerging SoC design challenges. In this paper we describe an OCP compliant network adapter (NA) architecture for the MANGO NoC. The NA decouples communication and computation, providing memory-mapped OCP transactions based on primitive message-passing services of the network. Also, it facilitates GALS-type systems, by adapting to the clockless network. This helps leverage a modular SoC design flow. We evaluate performance and cost of 0.13 μm CMOS standard cell instantiations of the architecture.

I. Introduction

Recent research has demonstrated the advantages of using shared, segmented interconnection networks – so called networks-on-chip (NoC) [1][2] – to implement global communication in system-on-chip (SoC) designs. By pipelining, NoC enables parallelism and counteracts the physical effects of long wires. NoC thus facilitates a scalable design approach, while accomodating the advert effects technology scaling has on wire performance. Managing the design flow of large complex chips however presents a non-trivial challenge in its own right. In order to effectively exploit the growing amount of on-chip resources a move from ad hoc SoC design to modular design is necessary. Orthogonalization of computation and communication is essential in order to enable fast design space exploration and IP reuse [3][4].

There are two basic paradigms of communication in computer systems: message-passing and memory-mapping. While much published NoC research focusses mainly on the implementation of message-passing networks, IP blocks for SoC designs, such as microprocessors and memories, are generally characterized by memory-mapped interfaces. There exist a number of socket specifications to this end, such as OCP (*Open Core Protocol*) [5] and AXI (*Advanced eXtensible Interface*) [6]. Since most NoCs are message-passing by nature, an adapter is needed.

There are a number of works published on the design of novel network architectures, but few publications have addressed issues particular to the design of an adapter module. In [7] an adapter implementing standard sockets was presented for the Æthereal NoC. The adapter however has quite a high forward latency. In [8] an OCP compliant adapter for the ×pipes NoC was touched upon. The adapter has a low area but it supports only a single outstanding read transaction. Also, both Æthereal and Xpipes are purely clocked designs, and as such do not address issues related to global synchronization in large-scale SoC designs.

In this paper we present an OCP compliant network adapter (NA) for the MANGO NoC (*Message-passing Asynchronous Network-on-chip providing Guaranteed Services over OCP interfaces*) [9]. The NA is a key component of a modular SoC design approach. Our contributions include identifying key issues of NA design and developing an efficient NA architecture. On the basis of several 0.13 μm CMOS standard cell instantiations, we evaluate the area and performance overheads of implementing NA tasks. The presented architecture enables GALS-type SoCs (*Globally Asynchronous Locally Synchronous*) [10], in that the NA implements synchronization between the clocked OCP sockets and the asynchronous or *clockless* MANGO NoC. Its design is a balanced mix of clocked and clockless parts, appropriately leveraging the advantages particular to either design style. It has a very low forward latency, is highly flexible with regards to the choice of transaction protocol and network packet format, and handles any number of outstanding transactions on the network.

The paper is organized as follows: in Section II we give an overview of MANGO. In Section III we describe the basic functionality of an NA, and in Section IV we detail the implementation of the MANGO NA. In Section V we present results, and Section VI provides a conclusion.

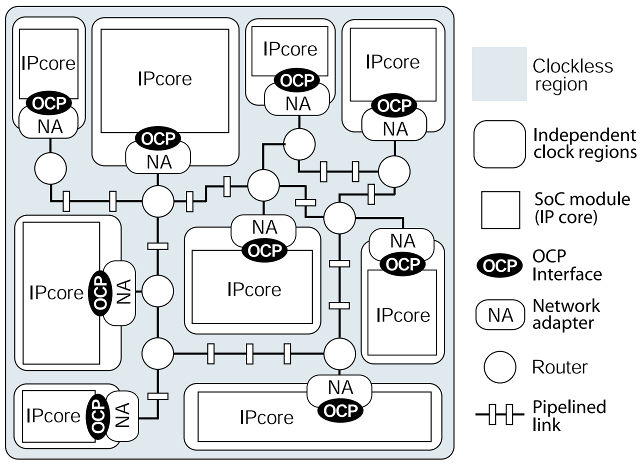

Fig. 1. A MANGO-based SoC.

II. MANGO Overview

Figure 1 illustrates a MANGO-based SoC. IP cores reside in locally clocked regions, connected by a heterogeneous network through NAs. The NAs use the primitive message-passing functionality implemented by the network, to transparently provide the cores with read/write transactions. Key features of MANGO are (i) clockless implementation, (ii) guaranteed communication services and (iii) standard socket access points. These are briefly presented below. For more details, please refer to [9][11].

A clockless implementation of the network routers and links enables a GALS-type system, in which the integration of cores with different timing characteristics is inherently supported. An advantage of clockless circuits is that they use zero dynamic power when idle. Also, their forward latency can be made much lower than that of comparable clocked circuits, helping to minimize the performance overhead of using a network for e.g. memory accesses.

MANGO provides connection-less best-effort (BE) routing, as well as connection-oriented guaranteed services (GS), in terms of hard latency and bandwidth bounds. The predictability of GS makes system analysis much easier, leading to advantages at all levels of SoC design.

This paper addresses the third key feature: standard socket access points. While the choice is arbitrary, we implement these as OCP sockets. OCP is a family of synchronous point-to-point interfaces for memory-mapped read/write transactions. Support for other similar interfaces such as AXI is a trivial extension.

III. NA Functionality

Though much simplified, the layered communication approach enabled by NoC is similar to the OSI model used in macro networks. The transport of messages in the network is decoupled from higher level transactions. The basic datagram of network-level flow control is called a *flit* (flow control unit). In the NA, OCP transactions are packetized and transmitted across the network as a stream of flits. A layered approach enables a mix of different sockets in the system, enhancing system-level composability. The overhead – of packetizing and depacketizing – is tolerated, as the approach enhances design-productivity: it becomes possible to design and verify SoCs hierarchically, and to plug together off-the-shelf IP cores from different vendors.

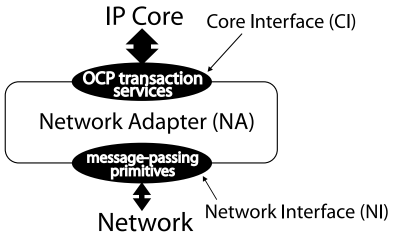

Fig. 2. The Network Adapter provides high-level transactions to an IP core, based on the message-passing primitives of the network.

As illustrated in Figure 2, the NA implements a Core Interface (CI) at the core side and a Network Interface (NI) at the network side. The functions implemented in the NA are *transaction handshaking* according to the CI protocol (in this case OCP), *encapsulation* of the transactions for the underlying packet-switched network, and *synchronization* between the IP core and the network.

IV. NA Implementation

Read/write transactions require two types of NAs: an *initiator* connecting to a master core such as a microprocessor, and a *target* connecting to a slave core such as a memory. Figure 3 shows the structural design of the NAs, illustrating how the functions mentioned in Section III are implemented in separate modules. It is also seen how each NA consists of a *request* and a *response* path, and – orthogonally to this – of a *clocked* and a *clockless* part. The presented version of the MANGO NA supports all the required OCP v2.0 basic signals and a consistent subset of burst, thread and interrupt extensions, allowing support for single reads and writes, single-request burst reads and writes, threads, connections and interrupts. The initiator is configured through its CI, the target through its NI.

The NAs provide a number of input and output network ports, each corresponding to a GS connection or BE service. Output ports are pointed to using the OCP signal *MConnID*, and several threads may use each port without restrictions. Transmitting on a BE port, packets need a header flit for routing information. On GS connections no routing information is needed, as these implement a virtual circuit to another NA in the network [9].

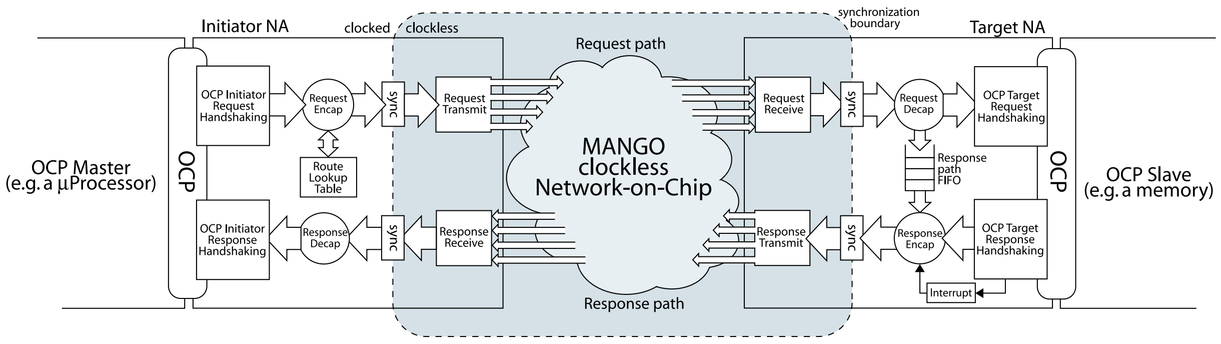

Fig. 3. The initiator NA connects to a master core, the target NA connects to a slave core.

An OCP transaction originates at a master core. In the initiator NA, the *OCP Initiator Request Handshaking* module implements part of the OCP slave socket with which the master transacts. The *Request Encap* module maps the OCP signals to packet signals. During BE transmissions it also uses the 8 MSBs of the OCP address field (*Maddr*) to look up a routing path, appending this to the payload (the routing path lookup table can be programmed through the OCP interface). This all takes 1 OCP clock cycle. In order to retain the flexibility of synthesized design, clocked circuits are used here. In any case, there is little performance to be gained by using clockless circuits in the handshaking and encapsulation modules, since the OCP interface is clocked itself. Serialization of packet flits is done by the *Request Transmit* module, in the clockless domain. This makes the flit serialization independent of the OCP clock speed. Also, the synchronization overhead is minimized by synchronizing an entire packet to the clockless domain in one go, instead of one flit at a time.

In the target NA packets are reassembled in the clockless *Request Receive* and decapsulated in the clocked *Request Decap* modules. The *OCP Target Request* and *Reponse Handshaking* modules then conduct an OCP transaction accordingly. Packet reassembly and buffering is done independently per port, such that a packet is forwarded to the clocked part of the NA only when the entire packet has arrived. During bursts however the packet is forwarded as soon as the control information and the first data word has arrived. If the transaction requires a response (e.g. a read), the response is encapsulated by the target NA, serialized in the clockless *Response Transmit* module and transmitted across the network. While a request is being processed by the slave core, the *Response path FIFO* stores the return path. Response packets are reassembled in the *Response Receive* module, decapsulated in the *Reponse Decap* module, and the transaction is completed by the *OCP Initiator Response Handshaking* module.

The OCP signal *MThreadID* is stored in request packets and returned in the corresponding response. The NAs thus handle any number of outstanding transactions, as they do not need to keep track of these themselves. If ordering is guaranteed by the network, e.g. on GS connections, multiple transactions can even be initiated on each thread.

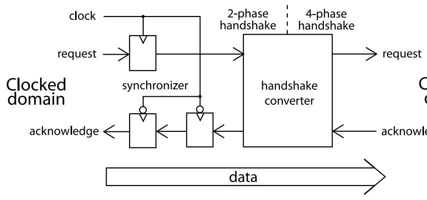

Fig. 4. Synchronizing between the clocked and the clockless domains.

Interrupts are implemented as *virtual wires*: an interrupt triggers the target NA to transmit an interrupt packet. Its destination is defined by configuring the target NA with a routing path or connection ID. Upon receiving this packet, the initiator NA asserts the local interrupt signal pin.

In order to minimize the synchronization overhead, we implement a 2-phase handshake channel, employing a two-flop synchronizer [12]. As illustrated in Figure 4 this is converted, in the clockless domain, to a 4-phase protocol compliant with the handshaking of MANGO. A complete clock cycle is available for resolving metastability in the flip-flop. Using flip-flop settling time and susceptibility window values of $\tau = 60\,ps$ and $T_W = 120\,ps$ (conservative values for a 0.13 μm standard cell technology), the mean time between failure (MTBF) at a 400 MHz OCP clock is calculated as follows [13][12]:

$$MTBF = \frac{e^{T/\tau}}{T_W f_C f_D} = 259 \times 10^9\ s(> 8000yrs)$$

T, f_C, f_D are the settling window (one clock cycle), and the clock and data frequencies, f_D set to one fourth f_C.

The presented architecture is very flexible and modular. Changing the *Handshaking* modules a different CI protocol can be implemented, and customizing the packet format is merely an exercise of mapping CI signals to packet signals in the *Encap/Decap* modules. Adapting to a different network is done by modifying the *Transmit/Receive* modules.

TABLE I. Results for 0.13 μm CMOS standard cell implementations (worst-case process corner) .

NA Type	OCP Speed	Area		Port Transmit Speed		OCP Read		OCP Write		OCP Interrupt	
		initiator	target	initiator	target	BE	GS	BE	GS	BE	GS
BE only	400 MHz	0.058 mm^2	0.020 mm^2	725 MHz	1.08 GHz	6 clks	-	4 clks	-	3 clks	-
BE + 3GS	400 MHz	0.064 mm^2	0.039 mm^2	483 MHz	633 MHz	8 clks	6 clks	5 clks	4 clks	4 clks	3 clks
BE + 3GS	200 MHz	0.060 mm^2	0.037 mm^2	483 MHz	633 MHz	6 clks	6 clks	3 clks	3 clks	3 clks	3 clks

V. Results

NAs with 32-bit OCP data and address fields and 32-bit network ports have been implemented using 0.13 μm standard cells. Netlist testing was done using the OCP tool CoreCreator[14], which provides OCP compliancy detectors. In the test setup an initiator NA was connected head-to-head with a target NA. Table I shows three classes of results for a number of NA instantiations: area, port transmit speed and transaction latency overhead.

The area values are purely cell area (pre-layout). Due to the GS input buffers, the area of the target NA supporting GS ports is twice that supporting only a BE port. The area of the initiator NA however is not very different in the two configurations, its main part being the BE routing path lookup table. A pure GS initiator would be smaller.

The port transmit speed indicates the flit speed on the clockless output ports. Since these are independent of the OCP clock, flits can be transmitted fast, even from a slow core. The NAs with only a BE port are simpler than those with multiple GS ports, hence their port speed is higher.

Finally, the table shows the latency overhead of conducting transactions through the NAs, relative to transactions of a master and slave core connected directly. The latency is displayed in OCP cycles. The overhead in the clockless circuits is independent of the OCP clock, hence the overhead in terms of *cycles* is less for slower cores.

BE transactions necessitate transmitting the routing path in a header flit, thus the total latency is increased. For a slow core however, an entire packet can be transmitted in less than one OCP clock cycle, even when needing a routing header, hence the overhead of BE transactions is not greater than for GS transactions.

In comparison with that of the Æthereal [7], the MANGO NA has a shorter total forward latency – only a single cycle for packetization. The area is also smaller, but this is not directly comparable; the Æthereal and the MANGO networks both implement support for BE and GS routing, but since this functionality is realized differently, the tasks required in the NAs are different. Compared with ×pipes [8], the MANGO NA implements more complex functionality in terms of GS connections, interrupts and support for multiple outstanding transactions. In the clockless MANGO NoC, there is no synchronization overhead in the forward path of entering the network, hence the total synchronization overhead is at least one clock cycle faster than both these purely synchronously clocked counterparts.

VI. Conclusion

We have presented an OCP compliant NA architecture for the MANGO NoC. The NA enables modular, GALS-type SoC design by providing synchronous, memory-mapped interfaces, based on the clockless message-passing services of the network. The flexible architecture, which mixes clocked and clockless circuits, can easily be configured for different sockets and/or networks. Several 0.13 μm standard cell designs were implemented. Simulation results indicate that the performance and area overheads, of a layered approach to on-chip communication, are reasonable.

References

[1] L. Benini and G. D. Micheli, “Networks on chips: A new SoC paradigm,” *IEEE Computer*, vol. 35, no. 1, pp. 70 – 78, 2002.

[2] A. Jantsch and H. Tenhunen, *Networks on Chip*. Kluwer Academic Publishers, 2003.

[3] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli, “Addressing the system-on-chip interconnect woes through communication-based design,” in *Proceedings of the 38th Design Automation Conference (DAC’01)*, June 2001, pp. 667 – 672.

[4] K. Keutzer, A. Newton, J. Rabaey, and A. Sangiovanni-Vincentelli, “System-level design: Orthogonalization of concerns and platform-based design,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 1523 –1543, 2000.

[5] “Open Core Protocol Specifi cation, Release 2.0,” www.ocpip.org, OCP-IP Association, 2003.

[6] ARM, “AMBA AXI Protocol Specifi cation, version 1.0,” www.arm.com, ARM, March 2004.

[7] A. Radulescu, J. Dielissen, K. Goossens, E. Rijpkema, and P. Wielage, “An effi cient on-chip network interface offering guaranteed services, shared-memory abstraction, and fexible network confi guration,” in *Proceedings of the 2004 Design, Automation and Test in Europe Conference (DATE’04)*. IEEE, 2004.

[8] S. Stergiou, F. Angiolini, S. Carta, L. Raffo, D. Bertozzi, and G. D. Micheli, “Xpipes lite: A synthesis oriented design library for networks on chips,” in *Proceedings of the 8th Design, Automation and Test in Europe Conference*. IEEE, 2005.

[9] T. Bjerregaard and J. Sparsø, “A router architecture for connection-oriented service guarantees in the MANGO clockless network-on-chip,” in *Proceedings of Design, Automation and Testing in Europe Conference 2005 (DATE05)*, 2005.

[10] D. Chapiro, “Globally-asynchronous locally-synchronous systems,” Ph.D. dissertation, Stanford University, 1984.

[11] T. Bjerregaard and J. Sparsø, “A scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip,” in *Proceedings of the 11th IEEE International Symposium on Advanced Research in Asynchronous Circuits and Systems*, 2005.

[12] R. Ginosar, “Fourteen ways to fool your synchronizer,” in *Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems (ASYNCŠ03)*. IEEE, 2003.

[13] C. Dike and E. Burton, “Miller and noise effects in a synchronizing fip-fbp,” *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 849–855, 1999.

[14] www.ocpip.org/socket/corecreator, OCP International Partnership.