An Integrated PCI component for IEEE 1355 Networks

F. Wajsbürt, J.L. Desbarbieux, C. Spasevski, S. Penain, A. Greiner

UPMC / LIP6, 4 pl. Jussieu 75005 Paris, France.

Abstract. This paper presents the PCI-DDC chip, a network controller for multiprocessor architectures using the IEEE 1355 HSL high speed interconnect technology. PCI-DDC is a VLSI component providing the interface between the HSL network and the PCI-local bus. PCI-DDC has been designed at Université Pierre et Marie Curie in the framework of the HPCN-EUROPRO Esprit project. RCUBE is an 8x8 dynamic crossbar with 8 IEEE-1355 bidirectional High Speed Link. The PCI-DDC component can be directly connected to the RCUBE router. PCI-DDC can be used with any CPU board providing a 33Mhz-32bit PCI-local bus. It is powered at 3.3V, although it works as well in a 3.3V signaling as in a 5V signaling environment. It implements an efficient communication protocol without intermediate copies. The communication primitive is the remote write from the local memory to the remote memory. PCI-DDC acts as a PCI master and initiates data transfer from memory to the network and conversely without processor overhead. Messages are automatically split up by PCI-DDC in a set of packets with CRC protection. This paper presents an overview of the PCI-DDC behaviour, the internal architecture, and physical features. It describes step by step the remote write primitive and gives the expected latency and throughput. PCI-DDC is currently under fabrication in the CMOS-0.5 micron Alcatel-Mietec process.

1 Introduction

The High Speed Link (HSL) technology has been developed in the framework of the OMI-HIC ESPRIT project. It is a one Gigabit/sec, bidirectional, asynchronous, point to point, highly integrated serial link, fully compliant with the IEEE 1355 standard. An Association[4] exists to support and enhance the use of this standard. The RCUBE component is a dynamic router, implementing a wormhole strategy for low latency. It contains an 8*8 non-blocking cross-bar, and 8 HSLs. This VLSI component can be used to build high performance, scalable and low-cost interconnection networks. Since the internal router latency becomes 150 ns, the global network latency is negligible versus the software latency. Classical communication protocols, such as TCP/IP introduce multiple copies of data, resulting in high latency and low throughput. This is not acceptable for high speed networks such as HSL networks.

The PCI-DDC component presented here connects the RCUBE router to the PCI local bus. It implements the "Direct Deposit State Less Receiver" (DDSLR[3]) protocol, for efficient point to point communication, without any intermediate copies, and without processor overhead. The basic primitive is the remote write: PCI-DDC directly fetches data from local memory and write them into remote memory. PCI-DDC is a master on the PCI bus. It makes DMA and does not need processor help during the transfer. Figure 1 shows a typical architecture based on the PCI-DDC/RCUBE chip set.

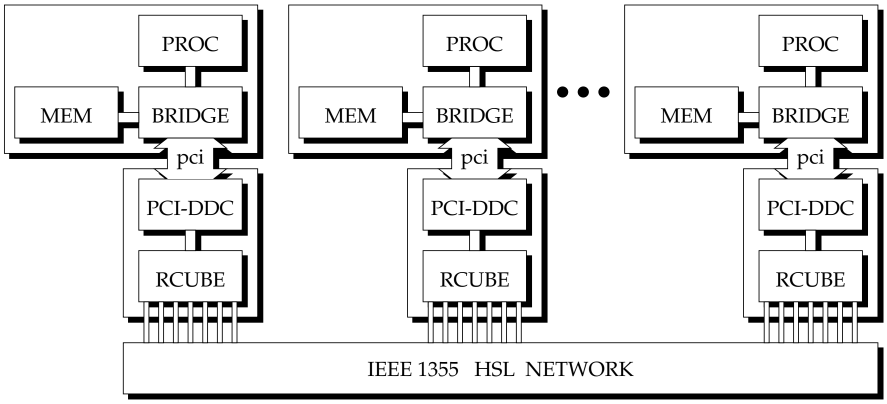

Figure 1: Parallel Computer using PCI-DDC/RCUBE chip-set

PCI-DDC offers 3 kinds of message: *standard* (any length) for data transfer, *short* (8-byte data) dedicated to signaling or monitoring, and *composed* (without data) to realize synchronization barriers. It manages data structures which describe the messages, it automatically splits the messages up in packets, it adds CRCs, it takes charge the byte realignment and it can inform whenever messages are left or arrived. It is able to operate at full PCI speed (up to 33 MHz) and HSL speed (up to 80 MHz). The main features of PCI-DDC are the following:

Low cost: A single chip is enough to connect the PCI-local bus and the RCUBE router. It integrates all the required FIFOs and Finite State Machines, and uses the host memory processor to store its working data structures. This allows to design a simple add-in card. Furthermore, a single kind of add-in card is enough to build up an HSL network.

Performance: The remote write primitive allows the system designer to implement very efficient message passing protocols from user memory to user memory without any copy by the system. It is so possible to use the HSL speed to the full.

Interoperable: The PCI-DDC/RCUBE chip-set can be used with any processor providing a PCI-local bus, i.e. with Pentium, Alpha and PowerPC.

The PCI-DDC component has been designed at UPMC in the framework of the HPCN-EUROPRO (High Performance Computing and Networking) Esprit project[5]. Its goal is to build an high speed communication network for an heterogeneous parallel computer dedicated for both signal and data processing. It is based on PowerPC and Shark DSP VME boards. PCI-DDC will be also used at UPMC for the MPC[6] project, that is an high performance yet low cost server based on standard Pentium mother board interconnected by an HSL network.

2 Functional Description

This section describes the PCI-DDC functional behaviour. We present the "direct Deposit State Less Receiver" protocol, as well as the data structure managed by PCI-DDC, and we detail the remote write primitive.

2.1 The "Direct Deposit" protocol

All memory addresses handled by PCI-DDC are physical addresses. PCI-DDC does not provide any mechanism to translate virtual to physical addresses, this task is left to the software.

From the application software point of view, the transfer unit is the message. Since a message is in the virtual memory, it can be distributed on several zones in physical memory. PCI-DDC only manages physical contiguous memory zones in both local and remote nodes. For PCI-DDC, such a zone is called a page, and then a message consists of a list of one to several pages.

PCI-DDC manages a list of pages to emit (LPE), which is a structure placed in main memory that contains page descriptors. The software driver writes page descriptors into the LPE (List of Pages to Emit). All page descriptors corresponding to the same message must have consecutive entries in the LPE. PCI-DDC has pointers to manage the LPE. A page descriptor contains a Message Identifier (MI: 24 bits), a destination node number called Routing Part (RP: 16 bits), a local physical address (PLSA: 32 bits) which is the start address of the page in the local memory, a Page Length (PL: 16 bits) in bytes and a remote physical address (PRSA: 32 bits) which is the start address for the page in the remote memory. There are also few more bits to define protocol details.

Pages are the smallest entities from the software point of view but not from the hardware point of view. Pages are split up in packets by PCI-DDC. The packet is the physical exchange unit. The packet length is variable and dynamically adapted by PCI-DDC, depending on the PCI burst length and the HSL network load. Each packet contains all the informations necessary to reconstruct the message in the remote memory (i.e. PRSA and MI).

Each message is identified by a number called Message Identifier (MI). All pages associated to the same message have the same MI. The MI is used by PCI-DDC to detect message completion. There are two possible methods to detect a message completion. In adaptive networks, packets can be received in a different order than they have been sent. PCI-DDC uses a List of Received Messages (LRM) implemented in main memory. The LRM allows to keep the current packet number for each received message. PCI-DDC uses the MI as an index in the LRM. The last packet of a message contains the number of packet sent which is the expected packet number from the receiver point of view. Therefore, a message is completed when the received packet number is equal to the expected packet number. In non-adaptive networks, it is possible to detect message completion without using the LRM. This can save memory (the LRM is a large table in physical memory) and decrease the traffic on the PCI bus. In this case, message completion is detected by the Last Packet bit contained in the last packet of a message.

Beside the LPE and LRM, the last data structure used by PCI-DDC is the LMI (List of Message Identifier). That structure is used by PCI-DDC to indicate the message received without error. In case of short messages, the same structure receive the 8-byte data.

2.2 The "remote write" primitive

From the hardware point of view, a message transmission procedure consists of three stages: preparation, transmission and reception (see Fig 2).

Preparation: (1) Page Descriptors are pushed by software into the LPE. Then (2), the software informs the PCI-DDC component of the LPE modification by writing in a PCI-DDC configuration register.

Transmission: (3) The sending PCI-DDC asks for the PCI-bus and reads the LPE descriptor using DMA accesses. Then (4), it performs data transmission, always using DMA accesses. There is no overhead for the processor during transmission. The physical remote address is transmitted with every packet. When the last page is sent, PCI-DDC can notify the sending processor with an interrupt signal (IT1).

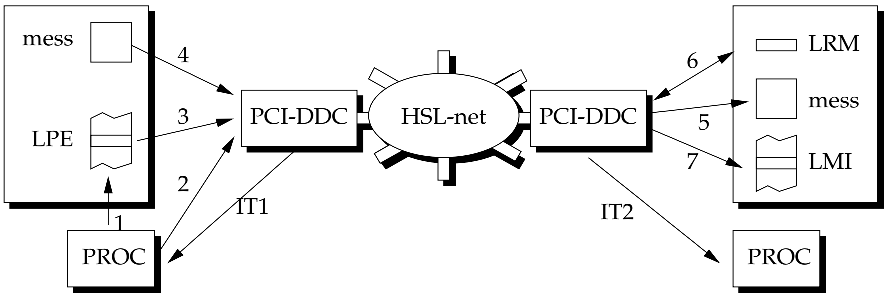

Figure 2: Standard transaction steps

Reception: (5) At the receiving side, as soon as PCI-DDC begins to receive packets, it asks for the PCI-bus and writes data at the specified address into the main memory using DMA accesses. (6) In case of adaptive networks, PCI-DDC uses the LRM in order to count received packets. There is no overhead for the processor during reception. (7) When the last packet is written, PCI-DDC can notify the processor, either with an interrupt signal (IT2), and/or by writing the LMI.

2.3 Message types

PCI-DDC offers three kind of messages : standard, short and composed messages.

standard message: There is one transmitter and one receiver. A standard message may be any length and consists of one to several pages described in the LPE. The receiving PCI-DDC component may use the LRM to count the received packets and the LMI to write the report. Messages are not necessarily word-aligned.

Short message: A short message contains 8 bytes of data. Data are fetched directly in the LPE and written directly in the LMI. The protocol is simpler since several PCI bus transactions are saved (i.e. 4, 5 and 6).

Composed message: There are several transmitters and one receiver. The receiver detects the message completion when each transmitter have sent one message. This kind of message does not transfer any data. It is used to realize synchronization barriers. All piece of a composed message must have the same Message Identifier (MI). The receiving PCI-DDC's LRM is used for giving the expected transmitter number.

2.4 Internal architecture

The PCI-DDC internal (Fig 3) structure consists of finite state machines and FIFOs. The actual PCI bus throughput is 132 Mbytes/sec. The HSL throughput is 80 Mbytes/sec for each direction. In order to solve the throughput mismatch, PCI-DDC has two FIFOs (one for each direction).

When PCI-DDC emits data from the PCI-bus to RCUBE, it can absorb the speed difference until TX FIFO is full. Whenever the PCI-bus data throughput is larger than the RCUBE throughput, the TX FIFO size defines the packet length, since the packet is cut as soon as the FIFO is full. When PCI-DDC receives data from the HSL network, the RX FIFO allows PCI-DDC to quickly flush the network. Each FIFO contains 400 bytes.

The configuration bloc manages all configuration registers, such as LPE, LMI and LRM pointers. It also contains registers to initialize and monitor the RCUBE router. The reception bloc (RX) extracts packets from the RX FIFO, checks CRC, performs DMA write for data,

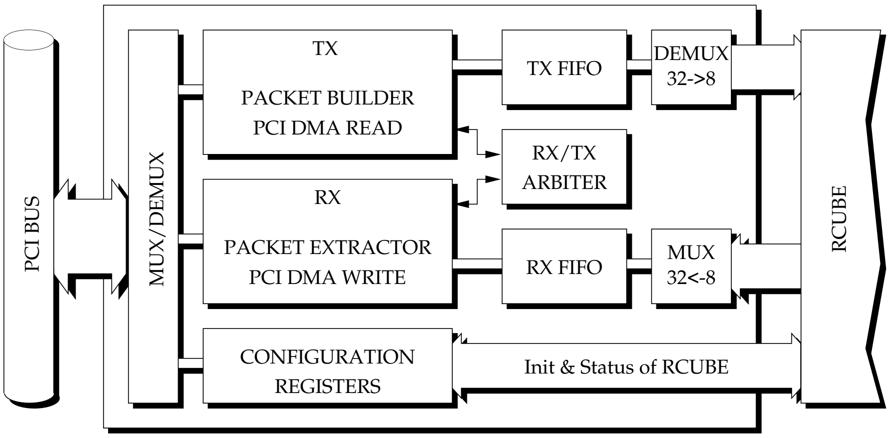

Figure 3: PCI-DDC Internal Structure

manages the LRM and LMI structures. This bloc contains a small cache to keep the last packet number and message identifier in order to save LRM access (if the current received packet belongs to the same message as the previous one, it is not necessary to perform LRM access since informations are already there). The transmission bloc (TX) reads LPE descriptor as soon as software move LPE pointers, performs DMA read for data and builds packets. It adds CRC and takes charge of data alignment if local address and remote address are not aligned on the same byte number. At last, the arbiter grants RX and TX blocs permission to request the PCI bus. This arbiter may be tune to favour either RX or TX.

3 Results

3.1 Performances

The PCI-DDC performances closely depends on the PCI bridge features and network load. Table 1 presents few evaluations based on reliable cycle precise simulation in a test bench including a complete PCI environment (memory, bridge, ...) with PCI-DDC as unique peripheral device. For this simulation, the output and input RCUBE ports loop back in order that all outgoing packet immediately come back. This simulates a node with equal transmission and reception load.

Table 1: Performance results

	Message length		
	4 bytes	128 bytes	4 Kbytes
Config write lpe (pointer)	0→1	0→1	0→1
DMA reed page descriptor	6→11	6→11	6→11
DMA read data	16→18	16→49	16→2026
Packet sending	26→37	26→83	26→2146
DMA write data	45→46	63→94	207→2199
LMI write (IT assert)	52→57	104→107	2209→2212
Total time transfer (μs)	1.7	3.2	66.3
Useful throughput (Mb/s)	4.7	80	124

All numbers in the top of the table are cycle numbers, which indicates when begins and ends the transaction. Column 1 gives an one-word message transfer latency from the first configuration write (step 2 on fig 2) to the interrupt assertion once the LMI transaction is completed (step 7 on fig 2). The RCUBE latency is 150 ns. For 128 processors the largest distance between two processors is 7 hops. It means that the total hardware latency is less than 3 μs. Column 2 and 3 gives results for longer messages. When the message length increases, the throughput increases close to the maximum PCI bus throughput, as shown in column 3.

3.2 Physical features

- CQFP 120 pins (among which 28 supplies),
- Estimated power consumption 2 Watts,
- 3.3 Volt supply, but usable in 5 Volt or 3.3 Volt environment
- 50,000 equivalent gates
- Process CMOS 0.5μm Alcatel-Mietec

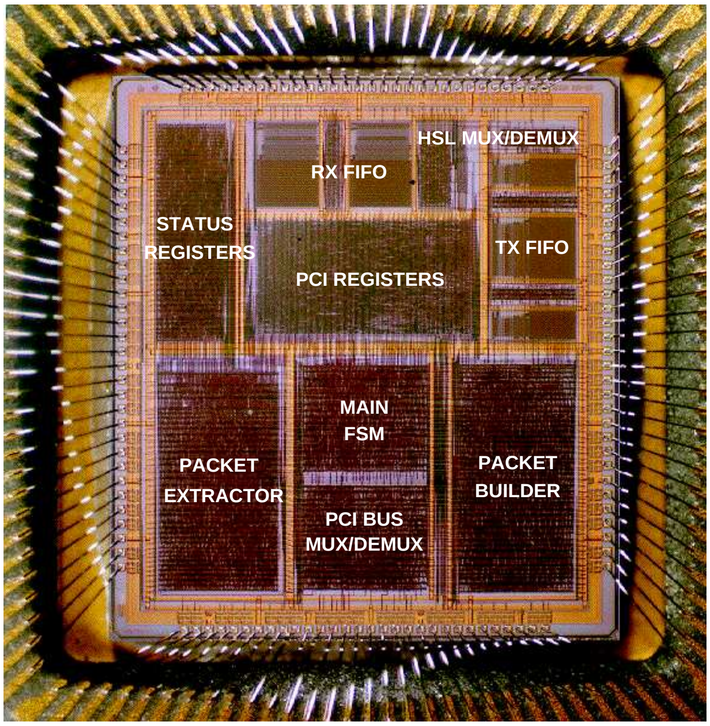

Figure 4: PCI-DDC die plot.

4 Conclusion

With the PCI-DDC/RCUBE chip-set, it is now possible to build high performance, low cost and scalable interconnection networks, based on the HSL IEEE-1355 technology. PCI-DDC is full compatible with RCUBE (same pin allocation and no GLU required). PCI-DDC does not degrade the PCI throughput, even for a small message size (80 Mb/s for 128 bytes message in and out simultaneously), and requires less than 3 μs latency to transfer a one-word message. PCI-DDC, fabricated by Alcatel-Mietec in CMOS-0.5 micron process, is expected by the end of 1997. The PCI-DDC/RCUBE chip-set will be commercialized by BULL S.A.

References

[1] PCI Special Interest Group, P.O. Box 14070, Portland, OR 97214, *PCI Local Bus Specification, Rev. 2.1*. June 1995.

[2] B. Zerrouk, V. Reibaldi, F. Potter, A. Greiner and A. Derieux, *RCube: A Gigabit Serial Links Low Latency Adaptive Router*. pp. 13-17, HOT Interconnect IV symposium, Stanford, California, August 15-17, 1996.

[3] F. Potter, *Conception et Réalisation d'un réseau d'interconnexion à faible latence et haut débit pour machines multiprocesseurs*. PhD dissertation, April 1996, Available at Université Pierre et Marie Curie, LIP6 Lab., Architecture Dept., 4 pl. Jussieu, 75005 Paris, France.

[4] 1355 Association, http://www.1355-association.org/

[5] ESPRIT HPCN Project Num 21040 - EUROPRO, http://www.cetia.com/news/EUROPRO/

[6] MPC project, PCI-DDC and RCUBE specifications, http://asim.lip6.fr/mpc/