Design and Use of a PPMC Processor as Shared-Memory SCI Node

D.Altmann, A.Guirao, H.Müller and J.Toledo

CERN, 1211 Geneva 23, Switzerland
EP-ED group[1]

Abstract

The MCU mezzanine was designed as a networked processor-PMC for monitoring and control in the LHCb Readout Unit (RU) with remote boot capability. As PCI monarch on the RU, it configures all PCI devices (FPGA's and readout network interface) that can then be used by user programs running under the LINUX operating system. A new MCU application is within the LHCb L1-Velo trigger where a CPU-farm is interconnected by a 2-dimensional SCI network, with event data input from RU modules at each row of the network: the SCI interface on the RU is hosted by the MCU which exports and imports shareable memory with the trigger farm in order to quasi become part as one of it's CPU. After this initialisation, the hardware DMA engines of the RU can transfer trigger data, by using physical PCI addresses that directly map to the remote CPU memory.

I. INTRODUCTION

Due to the absence of the historical backplane buses in electronics for trigger and Data Acquisition, crate-based modules like the LHCb Readout Unit (RU)[8] need embedded processors with network support for their integration in the Detector Controls System. In the case of the RU, an additional requirement is the initialisation of the onboard PCI bus segments and the PCI-based, readout network interface (NIC). In order stay technology-independent, the standard PMC-type "mezzanine approach" appeared most adequate as carrier for an integrated CPU subsystem with standard network support. The PMC mezzanine standard, apart from providing a proven electrical and mechanical framework, uses the PCI bus as interface bus between a host and it's PCI devices like FPGAs. Whilst originally CPUs were considered part of the main module, and PCI devices could be added as plug-in mezzanines, it became apparent that the inverse approach, namely putting the host system on the mezzanine card allows for considerable more flexibility and CPU upgrading. Hence processor PMCs (PPMC) were defined by VITA as mezzanine PMC cards with embedded CPU and PCI arbitration lines via an additional mezzanine connector.

With the integration level of today, it is possible to embed a complete system in a chip, using the standard IA32 architecture and thus avoiding difficulties with the OS and the hardware. At the design time of the MCU, in the year 2000, the low power "PC-on-a-chip" from "ZF Micro-embedded" was the most advanced technology choice.

The LHCb 1MHz VELO trigger test-bed at KIP in Heidelberg had chosen the DMA-based RU module as input stage to the shared-memory CPU cluster [7], also because the RU provided with it's MCU card an embedded processor with PCI bus monarch functionality which was necessary for the initialisation of SCI node interface to become a logical part of the shared memory domain. This meant that the embedded PMC processor on the RU had to run the same operating system (LINUX) as all CPUs of the cluster and that the commercial SCI drivers had to be adapted to the embedded system.

II. SYSTEM ARCHITECTURE

In order that the RU's become part of the shared memory domain of the cluster (Fig. 1), the embedded MCU on each RU needs to take part in the SCI initialisation of all CPU's.

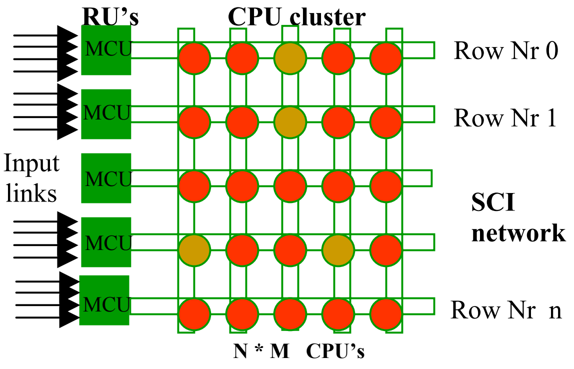

Figure 1: The 2-dimensional SCI cluster at KIP consists of n*m SCI nodes, each of which contains a farm CPU.

The tests reported here have been performed with one RU and one CPU in order to demonstrate the principle that can easily be applied to the whole cluster [10].

A. MCU – Embedded processor card

Designed around a 120 MHz "PC-on-a-chip", the MCU mezzanine card is a fully compatible PC system [1]. Conceived as a diskless Monitoring and Control Unit (MCU) of the PCI bus subsystems on the LHCb Readout Unit (RU), it boots the LINUX operating system from a remote server [2]. It's implementation as a general-purpose PMC card has allowed using it also in other applications than the one described here. A photo of the MCU is shown in Fig. 2.

[1] http://ep-div-ed.web.cern.ch/ep-div-ed/

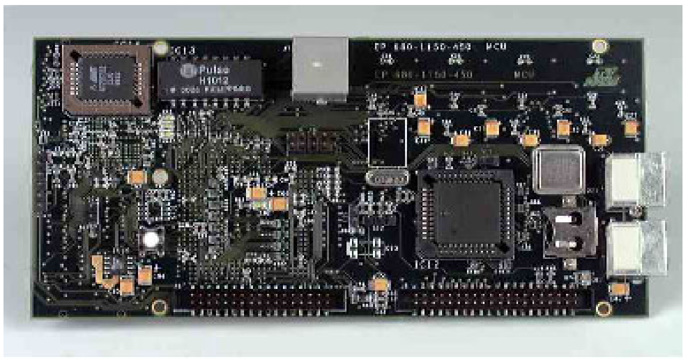
Figure 2: The MCU mezzanine card (PMC format)

The MCU's core is based on a Cyrix 486 core architecture which integrates a peripheral subsystem which is divided in two large blocks (Fig 3): embedded interfaces with PCI bus and I/O extensions. The embedded interfaces are serial and parallel ports, watchdog timers, EIDE, USB and floppy controllers, access bus (I2C compatible) interface, keyboard and PS/2 mouse systems. Extra additions to this are the 10/100 Mbit Ethernet and user programmable I/O. The latter are available via the VITA-32 user connector (P14) and used as a general JTAG port on the RU board.

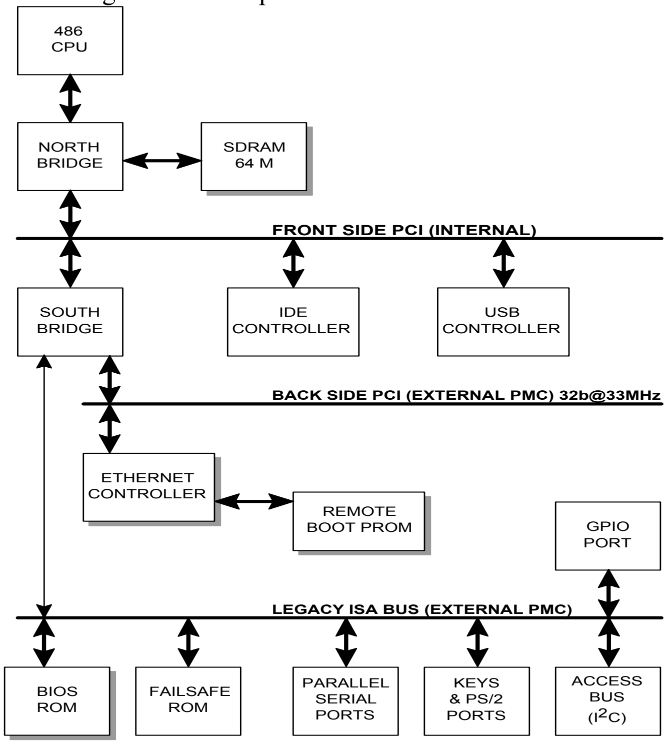

Figure 3: Architecture of MCU subsystems. Un-shaded blocks are integral part of the "PC-on-a-chip".

The bus architecture of the Readout Unit is depicted in Fig. 4. The MCU being a PCI monarch on the Readout Unit, it scans and initialises the PCI bus during the boot operation, finding and configuring:

1. All FPGAs and their resources
2. The SCI network interface card
3. All buffers and registers mapped via the FPGA's

After the hardware configuration, the operating system is loaded from a server that also configures the SCI network card according to the procedures defined for PCI, i.e. via access to the PCI configuration space of the SCI card.

B. *DMA architecture on RU*

The output FPGAs in each RU are connected to a 64 bit wide input buffer, each of which is filled from 2 Slink [6] input stages (Fig. 4). These input buffers are dual port RAMs, (used like a circular buffer) which are filled on one side from Slink and emptied on the other side by FPGA-based DMA engines. The RU has two parallel DMA engines, which are multiplexed via the common PCI bus. This dual data path architecture was chosen in order to increase the I/O performance via interleaved DMAs, where two PCI burst coming from each data path combine their payloads in a single SCI packet which gets sent to the network cluster.

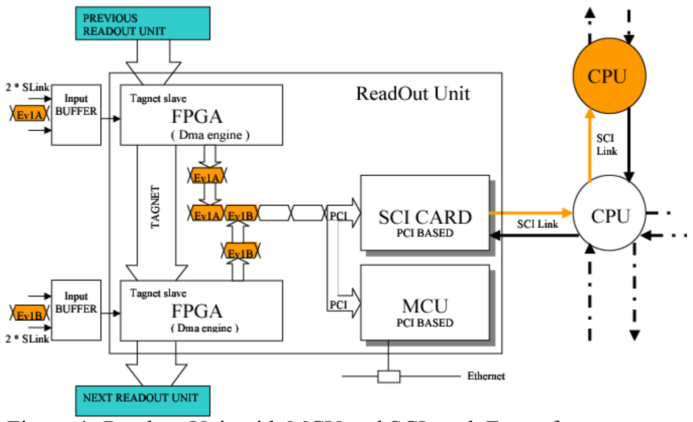

Figure 4: Readout Unit with MCU and SCI card. Event-fragments are received in 2 input buffers of which each has 2 Slink inputs. The two buffers are combined in the SCI card via the PCI write combining protocol: both DMA engines send their fragments in close succession to the same SCI card buffer.

In order to synchronize the DMAs in all RU's, a special twisted-pair protocol named "TAGnet" was developed [9]. Via this protocol, the identifier of the next destination CPU in the cluster is received in the FPGA-resident TAGnet slave and converted into a physical PCI address for the DMA engine. The physical PCI address has been obtained through the SCI initialisation procedure of the MCU, mapping local PCI memory space into a remote CPU memory. The transport of data from the local PCI bus to the remotely mapped memory is entirely performed by the SCI network hardware and requires only several microseconds.

C. *SCI Initialisation*

SCI is a scalable network standard [4] with a raw bandwidth of currently ca. 6 Gbit/s. SCI allows in particular implementing a shared memory paradigm between remotely connected CPUs. Interfaced via the PCI bus, SCI network bridges need to be initialised via a dedicated driver that will map remote memory segments into the local computer address space.

For the embedded applications, PMC versions of the SCI brides exist (Fig 5) from a leading manufacturer [3] of SCI technology.

Figure 5: Photo of Dolphin SCI card (PMC version). SCI packets are received and transmitted via two 16 bit wide differential LVDS connectors. The PCI mezzanine connectors are configured for 64 bit and 66 MHz.

III. DATA TRANSFER FROM RU TO CPU

Once initialised by the MCU, the RU becomes part of the shared memory system of the CPU farm. A hardware DMA in an FPGA can directly send data to remote CPU memory by using a local physical PCI address (figure 6). Using a modified version of the IRM driver for SCI on the MCU, it exports and imports shareable memory with the other CPU nodes in the farm. The SCI adapter on the Readout Unit shares its 64 bit@66 MHz PCI bus segment with an FPGA-resident DMA engine. The latter requires a physical PCI address to copy trigger data to remote, exported memory in a farm CPU node. The corresponding physical address is available after the shared memory initialization has been completed..

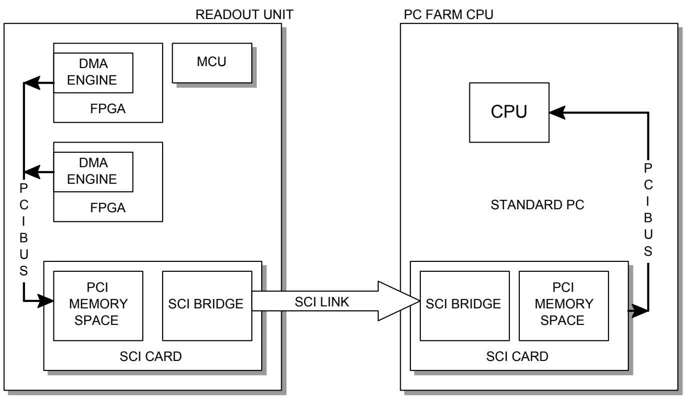

Figure 6: Data transfer between Readout Unit and CPU farm. Two alternating DMA engines in the Readout Unit transmit event-data to a mapped memory space within the SCI bridge. The SCI link transmits the combined DMAs to the PCI bus of the remote CPU.

The copy process from the local PCI bus to a remote PCI bus of a CPU is similar to a hardware copy to local memory, requiring only a few microseconds. For speed enhancement, the RU uses two alternating DMA engines that write two consecutive PCI bursts to the SCI bridge. The PCI write combining protocol generates a single packet from the two inputs, hence also performs data multiplexing on the fly.

D. Remotely mapped PCI address

A few lines of code are needed to find the PCI physical memory address that maps to a remote CPU on the farm. It uses the IRM SCI driver layer and calls the SCIQuery function.

```
qlseg.subcommand = SCI_Q_REMOTE_SEGMENT_IOADDR;
qlseg.segment = remoteSegment;
SCIQuery (SCI_Q_REMOTE_SEGMENT, &qlseg, NO_FLAGS,
        &error);
printf("The physical address for the remote segment is : %x\n",
        qlseg.data.ioaddr);
```

which will result in a screen dump like:

```
The physical address for the remote segment is: fd040000
```

When a CPU identifier is received via TAGnet, there is a lookup table entry that outputs its corresponding physical PCI address. The loading of the lookup table is part of the SCI initialisation procedure of the MCU.

E. Results

A successful integration of Readout Units into the shared memory domain of a CPU farm has been demonstrated by using the MCU card as host processor for the initialisation of the SCI bridge on the RU. The initialisation of SCI via an embedded 486 -type CPU required modifications [5] to the company IRM driver that was developed for server-type CPUs.

The measurement of Fig. 7 shows that the DMA engines of the RU write two successive 64 byte PCI bursts to the SCI bridge. The lower trace shows that ca. 1 us later, a 128-byte SCI packet frame appears on the SCI ring. Fig. 8 shows that the same SCI packet has correctly addressed the destination CPU and unfolds here into a 128 byte PCI burst (SCI card as PCI master) which is directed to CPU memory that was mapped from the two DMAs on the Readout Unit.

This measurement shows implicitly that:

4. The SCI initialisation between the embedded MCU card on the RU and farm PC has been correctly established
5. The PCI write combining of the two source DMA engines operates correctly and can be used for payload multiplexing
6. The 128-byte payload packet requires 200 ns of an SCI link, hence could be transmitted at up to 5 MHz in the extreme case.

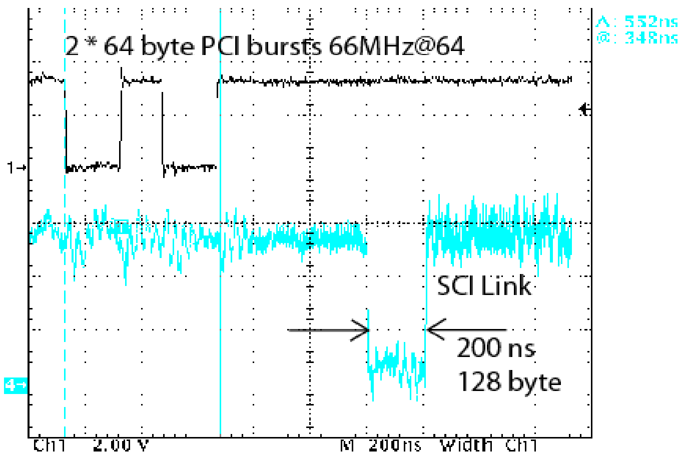

Figure 7: PCI and SCI frames at the Readout Unit (source side)

At the destination side, a 128-byte SCI packet is reconverted into a PCI burst that is directed to the destination CPU memory (figure 8).

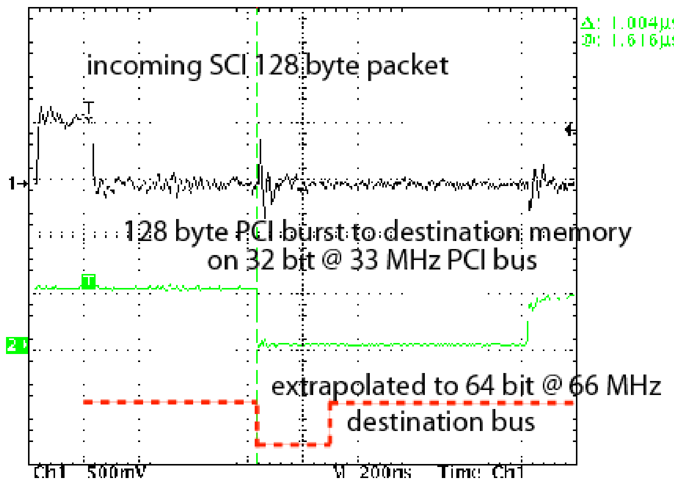

Figure 8: SCI and PCI frames at the destination CPU (farm)

The 200 ns SCI packet arrives at the PCI bus after a latency of ca. 800 ns, which includes the PCI arbitration as PCI master.

The destination CPU used has only a 32 bit @33 MHz PCI bus, hence can be extrapolated by a factor 4 to a farm-PC with 64bit @66 MHz PCI bus. The burst duration on 32bit@33MHz PCI bus is ca. 1 us, hence would be a factor 4 less on a 64bit@66MHz version, i.e. 250 ns.

The total PCI-to-PCI latency from Readout Unit to destination CPU can be combined from Fig.7 and Fig 8 and is 1,6 us + network cable delay.

IV. Conclusion

An embedded Monitoring and Control Unit PMC card has been designed and used as PCI bus monarch on the LHCb Readout Unit. The diskless mezzanine card boots LINUX from a server and initialises all FPGA–resident PCI bus resources. In the 1MHz VELO trigger application, the MCU has further been used to integrate the Readout Unit into the shared memory paradigm of the CPU cluster. After successful initialisation of the SCI network, the DMA hardware engines on the RU can use physical PCI addresses to copy trigger data to remotely mapped CPU memory. The transfer frequencies measured for aggregated data payloads of 128 byte are far beyond the required 1 MHz.

V. References

[1] A networked mezzanine processor card, Guirao A., Toledo J., Dominguez D., Bruder B., Muller H., Proceedings of 2001 IEEE Real Time International Conference, Valencia, p. 81-84

[2] MCU Monitoring Control Unit, a networked PMC processor for the Readout Unit, D.Altmann,A.Guirao, H.Muller, LHCb note DAQ-2002-006 http://hmuller.home.cern.ch/hmuller/RU/MCU/MCUnote2002.pdf

[3] Dolphin Interconnect Solutions AS, http:// www.dolphinics.com

[4] IEEE Std. 1596-1992 IEEE Standard for Scalable coherent Interface, IEEE 1992, developed between IEEE groups, industry and the CERN RD24 project. http://hmuller.home.cern.ch/hmuller/rd24.htm

[5] Installation of SCI-PCI adapters via an embedded CPU on Readout Units, D.Altmann, H.Muller, LHCB Note TRIG 2002-xxxx (to appear) http://hmuller.home.cern.ch/hmuller/RU/SCIonRU/SCInote.pdf

[6] LVDS link cards (SLINK) for multiplexers, VELO trigger and other applications in LHCb, F.Bal, H.Muller, LHCb DAQ 2002-005 http://hmuller.home.cern.ch/hmuller/RU/Links/Slinknote.pdf

[7] Architecture and Prototype of a Real-Time Processor Farm Running at 1 MHz, Inaugural doctoral thesis, Alexander Walsch, Sept 2002, KIP Heidelberg (to appear)

[8] Study and Design of the Readout Unit Module for the LHCb Experiment, Doctoral Thesis, Jose Francisco Toledo, University of Valencia, Gandia, 2001 http://hmuller.home.cern.ch/hmuller/RU/curro_thesis.pdf

[9] TAGnet a twisted-pair protocol for event-coherent DMA transfers in trigger farms, Hans Müller, Francois Bal, Sebastien Gonzalve, Angel Guirao, Filipe Vinci dos Santos, presented at the 8th Workshop on Electronics for LHC Experiments, Sept. 2002 Colmar, France http://hmuller.home.cern.ch/hmuller/RU/TAGnet/Colmartalk.pdf

[10] HS Project, horizontal slice of the Level-1 VELO Trigger, Hans Muller, Filipe Vinci dos Santos, Angel Guirao, Sebastien Gonzalv, LHCb TRIG note 2002-xxxx (to appear) http://hmuller.home.cern.ch/hmuller/RU/L1Velo/HorizontalSlice/HSproject.pdf