A networked mezzanine processor card

Guirao A., Toledo J., Dominguez D., Bruder B., and Müller H.

CERN EP-ED, Geneva, Switzerland

***Abstract*-- The Monitoring and Control Unit (MCU) card has been developed as an IEEE P-1386.1 mezzanine card for networked monitoring and control applications. It is implemented as Peripheral Component Interconnect (PCI) host processor according to the VITA-32 draft standard for processor mezzanines. The first application of the MCU is the LHCb Readout Unit, which requires remote configuration of its Field Programmable Gate Arrays (FPGAs) and PCI devices. The MCU hardware system consists of a "PC-on-chip" with a Synchronous Dynamic Random Access Memory (SDRAM) and Local Area Network (LAN) controller on a mezzanine card, configured for booting LINUX as a diskless client from a networked server. The MCU's control functions are either implemented via the standard P-1386 PCI bus connectors or via a dedicated programmable I/O connector. The MCU's target application and its general-purpose feature will be presented.**

I. Introduction

THE Monitoring and Control Unit (MCU) has been conceived from the beginning as a mezzanine for configuring the LHCb Readout Unit [1], and provide In System programmability (IsP) capabilities, with remote operation.

The MCU is a networked x86 processor system with remote boot capability, designed for a 3.3V PCI environment. However, interface protocols available in its I/O connector are independent of the PCI bus, making the MCU a versatile system to use in motherboards hosting Industry Standard Association (ISA), Universal Serial Bus (USB), Joint Test Action Group (JTAG) IEEE 1149.1/P1149.4 protocol and I^2C bus choices.

II. System Architecture

Standards for embedded CPUs like PC104 and Industrial PC are neither specified nor adequate for use in the modular bus standards used in High-Energy Physics (HEP) experiments. The only common mezzanine standard adopted by the VME community, apart from IP, is the PMC IEEE P-1386.1 standard which is also "defacto" used in a mechanically compatible format by LHC experiments for the CERN S-Link card form factor [4].

The MCU was designed to comply with the PCI based-PMC mezzanines standard proposed by VITA-32 as a draft [5]. Using an I/O user connector apart from the PCI connector, this standard is module and company independent. It allows for components of the main board below most of its covered area. The connector area is specified to comply with a front-panel area like on a VME card. PMC connectors are both mechanically and electrically proven to work reliably and without noise problems.

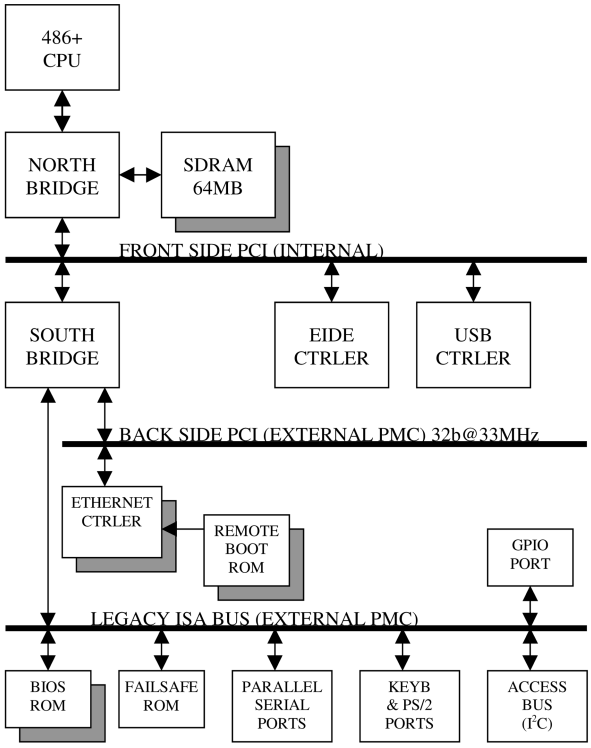

Fig. 1. General system architecture. Shadowed modules are external to the System-On-a-Chip (SoC) processor.

A. Guirao is with the European Organization for Particle Physics (CERN) as doctoral student of the Polytechnic University of Valencia (UPV). E-mail: Angel.guirao.elias@cern.ch

J. Toledo is with Polytehnical University, Valencia. He is now with the Department of Electronics, and also in CERN. E-mail: Jose.toledo@cern.ch

D. Dominguez was with CERN. He is now with Polytechnic University, Valencia, working in CERN NA60 experiment. E-mail: David.dominguez@cern.ch

B. Bruder was with CERN, and he is now with Alcatel Microelectronics, Paris.

H. Müller is with CERN, working as leader in several projects for LHCb and NA60 experiments. E-mail: Hans.Muller@cern.ch

A. System core

The "MachZ" System-On-a-Chip (SOC) [6] is a complete processor and peripheral subsystem requiring only external clocks, SDRAM, and BIOS ROM/Flash.

It consists of the following major blocks:

- Industry standard 32 bit processor core with integrated floating point co-processor and 8KB L1 cache,
- A North Bridge (system controller) with "Frontside" PCI Master / Slave Arbitration and SDRAM interfaces,
- A custom South Bridge with "Front-side" PCI interface to the North Bridge and "Backside" PCI Master/Slave system interface, Enhanced Integrate Device Electronics (EIDE) and USB controllers, floppy controller, serial ports, access bus, PC/AT sub-system, parallel port and general purpose I/O.

The idea is to use a fully compatible PC system, which runs Linux with only minor changes, and thus allows the software programmers to work without problems due to a platform change.

The MCU is carrying a 486+ core processor, running up to 120MHz. Peripherals, like the PCI bus, run at 33MHz.

B. Peripherals

Peripherals in the MCU are divided in two large blocks. ZFLinux's embedded interfaces and MCU's own extensions.

Embedded interfaces globe serial and parallel ports, watchdog timers, Enhaced IDE (EIDE), USB and floppy controllers, Access bus (I^2C compatible) interface, keyboard and PS/2 mouse systems. A general purpose I/O system completes this wide fan of choices, allowing the designer to use the x86 SOC with minimal external parts.

For easy integration in embedded systems, the processor offers also a programmable chip select logic for high integration systems. This concept avoids external logic and the complex address methods used in an x86 platform.

The MCU's own peripherals are the embedded 10/100Mb Ethernet unit, with MAC and PHY units for wired Ethernet, and JTAG interface for Readout Unit test. The parallel port has been disabled, although EIDE and floppy interfaces are available for using the MCU as independent computer, outside the Readout unit application.

A video output was not required for the diskless philosophy, and only a serial terminal output for debugging purposes is offered.

1) Ethernet and remote booting

An embedded Ethernet 100Mb/s chip, integrating MAC and PHY layers is included for a full diskless computer, with remote booting capability across the network.

Remote booting is a comfortable way to get a central, common and fast control over software running in a large number of computers.

Diskless computers rely on this procedure to load their operating system images into memory, in several steps. The drawback using remote booting is the network load. However, benefits are important, because expensive, fragile and cumbersome equipment like hard drives is not required.

The boot negotiation is accomplished in three steps. First, the diskless computer broadcasts an IP address request. Either the BOOTP or DHCP protocol is running in one or more central servers, which assign an IP address for a particular MAC address.

Once the MCU has its IP address, it requests for an operating system image. The Central server transmits it in a packet based basic protocol (TFTP).

The operating system is loaded into memory this way, and executed locally in each MCU. When running an operating system capable of mounting a network file system, like Linux, the MCU will mount the file system across the network.

This way, many MCU will share the same file system, with exception of some critical resources, which are unique for each of them.

Once each MCU is running Linux, eCos or any real-time system (RTOS), slow control and IsP funcionalities are available.

Following this philosophy, Readout Units installed in the LHCb experiment will be centrally controlled and configured in an inexpensive and fast way.

C. Software

The MCU is meant to use a Linux system, which eases the diskless operation thanks to the use of a powerful networking file system. The team policy is to keep all software development within the frame of a public source. Linux operating system is widely known and very well supported.

With a special license-free compiled kernel, the MCU boots Linux using a console redirection to the serial port for local debugging.

Etherboot free software [8] is used to link with the existing DHCP and TFTP protocols in Linux. Intel's PXE software [7] is also available to the public. Nevertheless, it requires a

licensed programming environment, not always preferred by all users.

III. Applications

A. The Read Out Unit

The LHCb Readout Unit (RU) is in general a module for receiving and buffering event data on 4 Slink inputs at up to 1 MHz trigger rate [1].

The overall throughput product of "eventsize" rate is approximately 160 Mbytes/s. The internal subevent building process generates newly formatted subevents on the RU's output port, which may be either PCI or Slink.

The RU has been conceived from the beginning as a 9U module with data input channels in the front panel and output channels on the rear panel. There is no use of the backplane, except for power supply purposes. The crate system chosen is Fastbus (IEEE960).

In absence of backplane-resided controller, an embedded controller is required in each RU. This controller has therefore to be inexpensive and networked.

After using non-standard commercial controllers at the beginning, a widely used standard was chosen [3] for the final Readout Unit design: The PCI based-PMC mezzanines with user I/O extension as is proposed by the VITA-32 draft standard. This standard is module and company independent, allows for components on the main board below most of its covered area and the connector area is specified to comply with a frontpanel area like on a VME card.

The MCU mezzanine was thus designed by the RU team as a networked, PMC processor module for FPGA configuration via PCI and for user-specific protocols via the VITA I/O connector.

Task of the MCU include monitoring the Sub Event Building (SEB) buffer, error handling and reporting, remote control of the operating parameters of the RU, PCI bus initialization, re-load FPGA configuration bitmaps, access status registers inside the FPGAs, implement high-level data transport protocols or emulate RU functions before implementing them in programmable logic hardware.

We opted for a double stack height (i.e. double-width RU modules) in Fastbus crates. This allows for mounting components & connectors up to 14 mm on the visible side of a PMC.

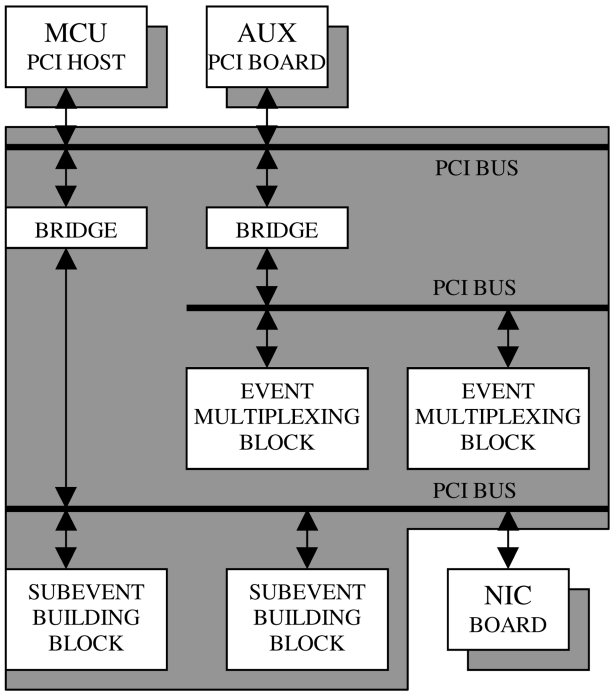

Fig. 2. The MCU in the Readout Unit Application. The main shadowed area is RU, meanwhile small shadowed components are mezzanine boards.

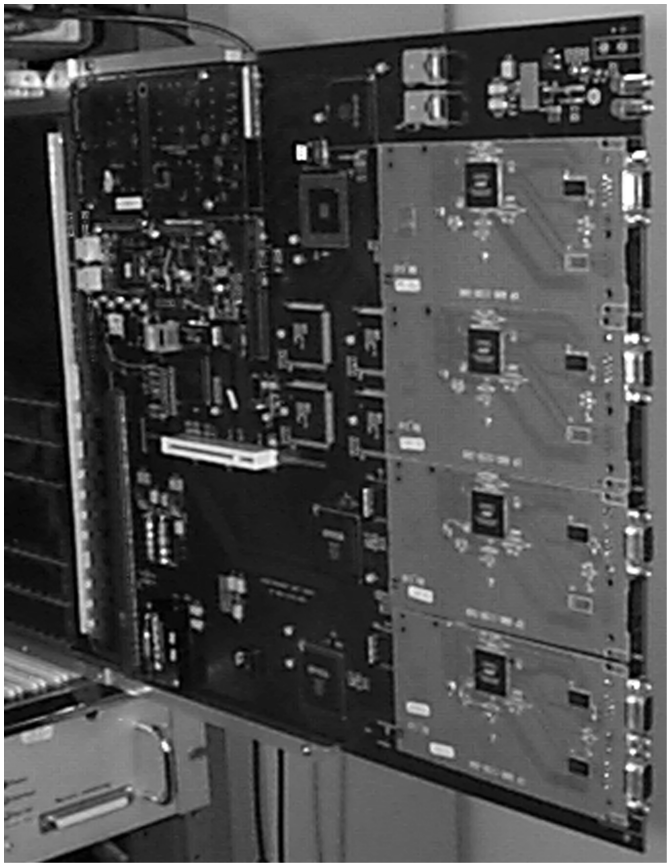

Fig. 3. The Readout Unit with all S-Link mezzanines, the NIC and the MCU cards. The MCU is on the half height, on the left.

B. *Other scenarios*

The MCU is a complete PC compatible system, quite simple and more affordable than any portable or industrial PC on the market. With some compromise on processing power, it offers low power, and small size.

The industrial PCs market, with follows the PIC standard, offers a PCI-ISA system based in a passive backplane, and a PC system plugged as PCI host. Other standards like PC104 only provide the PCI bus.

The MCU's open architecture and PHC compliance allows using it in other applications as PCI host, using a passive adapter for PMC to the appropriate standard.

Instead of a passive adapter, the Flexible Input-Output card (FLIC) [2] may be used as "intelligent" adapter, giving the designer the power of a highly customizable PCI logic added to the MCU versatility.

IV. REFERENCES

[1] J.Toledo et alt., The Readout Unit For High Rate Applications In The LHCb Experiment, 12th IEEE-NPSS Real Time 2001

[2] H.Müller et alt., A Flexible PCI Card For Data Acquisition, 12th IEEE-NPSS Real Time 2001

[3] Readout Unit history, documents, talks and minutes see http://hmuller.home.cern.ch/hmuller/ RUminutes.htm

[4] Draft Standard of physical and environmental layers for PCI mezzanine cards IEEE P1386.1 Draft 2.0 http://www.cern.ch/~hmuller/docs/PCI-SCI/pmc_draft.pdf

[5] Processor VME Standard VITA 32 199x Draft 4.01 Sept 2000, http://www.vita.com

[6] ZFLniux data source, ZF Linux DevicesInc , Palo Alto, CA, USA http://www.zflinux.com, 2000

[7] Preboot Execution Environment PXE software, Intel Corporation, http://developer.intel.com/ial/wfm

[8] Etherboot Project, http://etherboot.sourceforge.net/