ALMA a highly programmable PCI to VMEbus bridge

Claude Sitbon, Herve Mougel(*), Francis Bredin, Christophe Delapchier, Guy Vanneuville (*) Gerard Boudon
IBM France, Component development Laboratory, Dept 1040-Y1T 91105 Corbeil-Essonnes , France
(*)CETIA - BP.244 - 83078 Toulon, France
e-mail : sitbon @ vnet.ibm.com Tel: (33) 1 6088 5327 Fax: (33) 1 6088 4920
e-mail : boudon @vnet.ibm.com Tel: (33) 1 6088 6382

1.0 ABSTRACT

The role of the PCI (Peripheral Computer Interconnect) Bus is becoming very important in real-time embedded solutions. It can be find in the form of mezzanine Bus such as PMC (PCI Mezzanine Card) on a VME board, or in the form of more advanced tentative to standardize the mix of the VME and PCI busses in future solutions such as the Compact PCI.

PCI is being used as local bus for most modern VME board designs. The CVME 603/604 Single Board Computer from Cetia designed with the high speed PowerPC 603 or 604 processor core implements a PCI local bus and offers numerous PCI slots thanks to the use of a PCI-PCI bridge component.

A key component which complies with this architecture trend is the Bridge between PCI and VME busses. For that purpose, IBM France and Cetia, jointly developed a component bridging 32-bit PCI bus and VMEbus, named ALMA.

High data transfer rate between both busses, has been made possible by using decoupling FIFOs, in combination with Write posting and Read-prefetch operation modes. This has permitted to eliminate bottleneck of the slower VMEbus

ALMA can play as a a master or a slave on both VME and PCI ports, it also performs VME and PCI bus controller functions. Implementation is real time oriented with a focus on features such as DMA and interrupt handling which have been embedded into the bridge. VME and PCI cycles characteristics are fully programmable from both ports with large flexibility given on addresses mapping

The design is mapped to a 3.3V ASIC in 0.7um CMOS technology which supports 5V and 3.3V bus operations and meets the low power requirement of VME board designs. Product is packaged into 25mm BGA or 40mm CQFP, with 229 signal I/O pins.

2.0 OVERVIEW

- ALMA component is a highly integrated single chip solution which interfaces a 32-bit VMEbus with VME64 [1] capability and a 32-bit PCI Bus [2] which are very different in nature:

- Slow/medium data transfer rate, asynchronous bus operations and Big Endian data bytes ordering over the VMEbus.
- High data transfer rate, synchronous bus operations and Little Endian data bytes ordering over the PCI bus.

This product allows VMEbus single board computer and I/O board vendors to develop solutions with PCI support, taking thus advantage of the growing family of PCI components on the market and PMC standardization, for integration into VMEbus systems[3]. Another class of application includes systems where main bus is on the PCI side while VME is used as secondary bus. This is one way to protect current investments taking advantage of the large variety of existing VME boards.

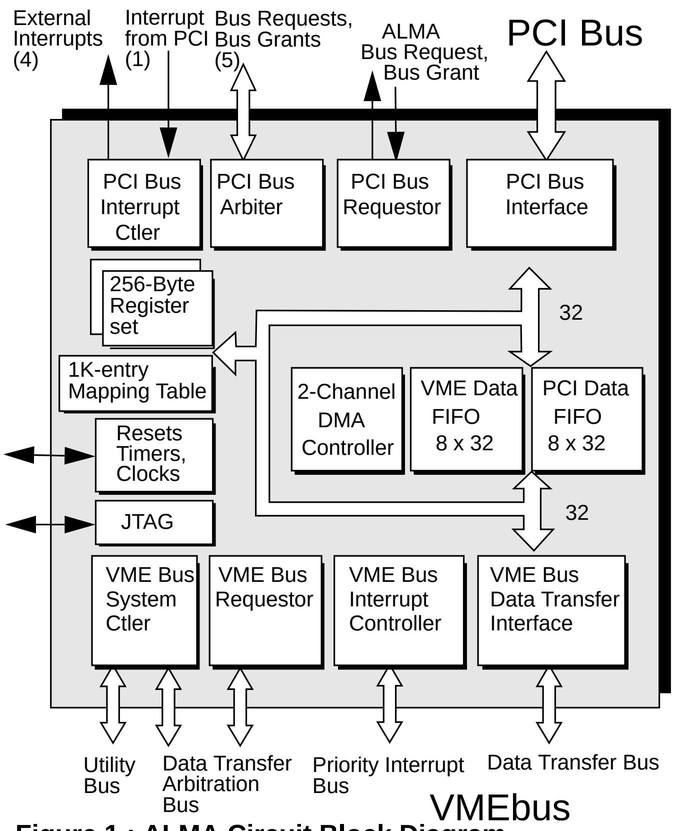

Figure 1 : ALMA Circuit Block Diagram

The main features of this high performance PCI to VME bridge shown on Figure 1 are summarized below:

- VME64 capability, up to 70 MBytes/sec (Fully compliant IEEE-std-1014-1987)
- PCI bus interface 32-bit, 33MHz fully compliant with PCI Specification Revision 2.0
- Glueless PCI to VME interface
- External VME buffers
- Stand alone capability: fully programmable from VMEbus
- 256 byte-register set fully accessible from both the PCI bus and the VMEbus ports
- Two-channel programmable DMA Controller (Up to 256 KB block length)
- IEEE 1149.1 JTAG testability support
- Transmit and Receive FIFOs for data transfer decoupling between the PCI bus and VMEbus.
- 1K-entry on-chip PCI to VME Mapping Table (Address translation, bus cycles parameters)
- Full VMEbus system controller and PCI bus arbiter
- VMEbus and PCI bus interrupter and interrupt handler

3.0 DESIGN TRADE-OFFs

- One 1K-entry Mapping Table is used for the mapping of PCI I/O and PCI MEMORY cycles into a VME cycle. That table, occupying about 15% of the chip silicon area, allows the user to program any address translation, VMEbus address and data transfer modes, Write posting and Read-prefetch modes, Little/Big Endian byte ordering modes with a 8MB granularity in the 4GB PCI I/O and MEMORY address space.

- Two 8 by 32-bit data FIFOs are used as a trade-off between bus bandwidth maximization, low silicon cost and low complexity dataflow and control logic designs.

- small chip size (7 x 7mm) allowing for 25mm BGA packaging and low power consumption (below 1Watt)

4.0 APPLICATIONS

- ALMA has been designed for both VME Single Board Computer designs and for VME I/O card designs (see figure 2). Another type of application is for PCI-VME adapter cards that permit PC desktops or servers or RISC workstations running UNIX to communicate with the wide range of VMEbus cards available on markets such as telecommunications, medical imaging, robotics, among many others. In this type of applications, PCI is the main bus while VME is secondary bus.

- The PCI bus can be connected directly to ALMA, while it is necessary, due to high loading, to interconnect the VMEbus through a buffering circuitry fully controlled by dedicated ALMA control signals

Stand alone capability

For low cost or non-intelligent peripherals, the ALMA bridge may be used with no processor on board, since it can be thoroughly configured from the VMEbus via VME A16 accesses.

Regarding interrupts processing, ALMA can direct local PCI interrupts (issued from a pin or via register access) to a VME mailbox interrupt or to VME IRQ interrupts.

At least, ALMA may supply PCI local bus arbitration thanks to its internal PCI bus arbiter.

So, ALMA stand alone capability, allows for easy design of I/O or non-intelligent boards implementing a PCI local bus with only PCI slave devices attached to it.

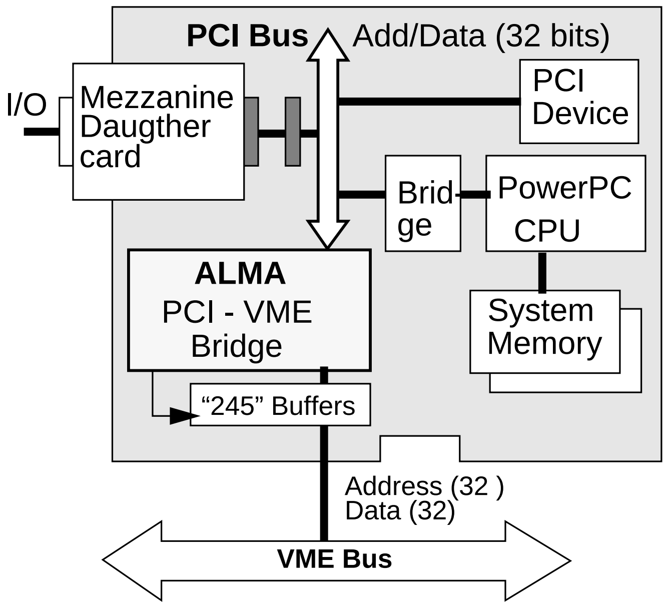

Figure 2 : ALMA typical application on a VME SBC

5.0 FUNCTIONAL DESCRIPTION

The ALMA VME/PCI bridge is a highly flexible component that supports master and slave bus operations on both the PCI bus and VMEbus ports.

5.1 Address Recognition and translation

This flexibility is a key feature to accommodate a whole range of uses, from a simple SBC VME card to a complex VME multiprocessor architecture: in the majority of VME computer designs, software compatibility (at user and kernel level) with existing standard desktop workstations or PCs is a major issue which is solved thanks to the ALMA addressing flexibility allowing the VME computer to use the same address mapping convention as the one defined by the chosen microprocessor architecture.

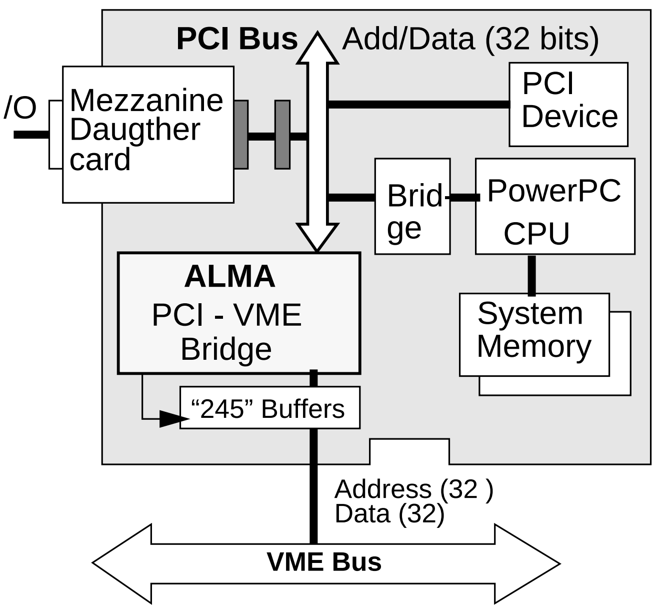

Figure 3 : ALMA typical application on a VME card

VME to PCI access

As a VME Slave, ALMA decodes if its VME slave image address and address modifiers are the target of the VME access, through use of 8 decoding channels whose address recognition range depth is programmable (min depth is 1MB, max depth is 4GB). These features accommodate well with the big parcelling out of the VME address map sometimes found in VME systems.

ALMA responds to the VME access under the following conditions:

- VME address 12 Msb match the address field of at least one channel.
- VME address modifiers match the address modifiers field of the channel that has been hit.
- The channel that has been hit is enabled.

When more than one is hit, an arbitration is performed between the channels.

PCI address is obtained through a translation of the VME address with a granularity of 64KB. Thus, the offset field of the selected channel is added to the 16 Msb of the VME address. While the PCI bus command is obtained from the PCI bus command field of the selected channel

These capabilities allows for high mapping flexibility of VME addresses into the 4GBytes PCI address MEMORY and I/O spaces.

PCI to VME access

ALMA as a PCI slave implements 6 PCI Base Address Registers for bus address and command decoding. At Reset these registers are initialized such as to define three ranges of 256 MB mapped in the MEMORY space and two ranges of 256 MB in the I/O space, while one range of 256 B mapped in the I/O space is dedicated for decoding of those PCI I/O cycle accessing the register set.

Furthermore, an additional filtering based upon PCI bus command (MEMORY and I/O) is performed, with a granularity of 8MB, through a 1K-entry programmable Mapping Table in which the low-order 512 entries are used to validate (accepted/rejected) MEMORY accesses while the high-order ones validate I/O accesses. PCI address 9 most significant bits are thus used to index one or the other of these 512-entry blocks, according as the PCI cycle is MEMORY or I/O.

ALMA selects itself as the target of the PCI access when both Base Address Registers and Mapping Table are hit.

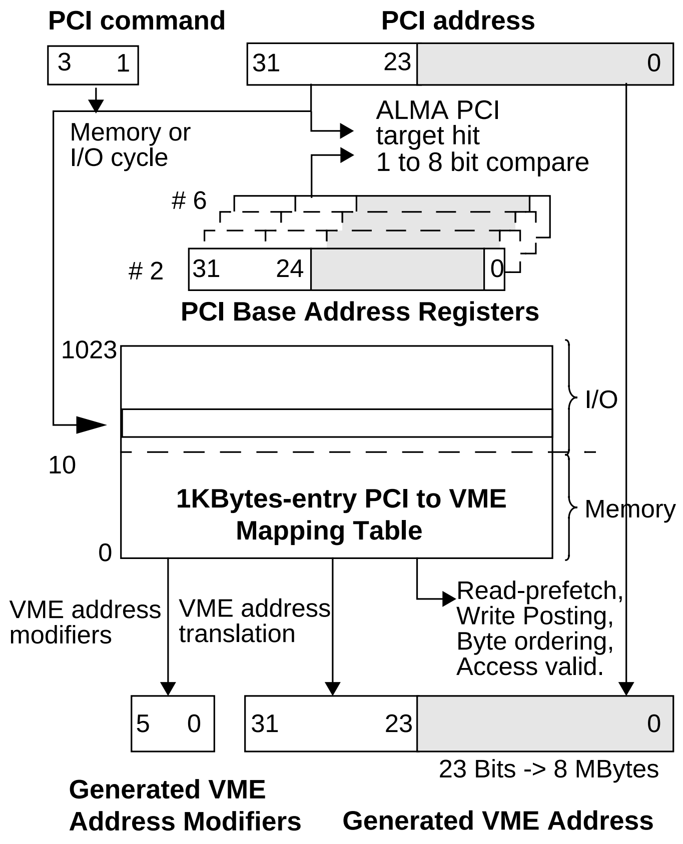
PCI command
PCI address
3 1
31 23 0
Memory or I/O cycle
ALMA PCI target hit 1 to 8 bit compare
# 6
# 2
31 24 0
PCI Base Address Registers
1023
10
0
I/O
Memory
1KBytes-entry PCI to VME Mapping Table
VME address modifiers
VME address translation
Read-prefetch, Write Posting, Byte ordering, Access valid.
5 0
31 23 0
23 Bits -> 8 MBytes
Generated VME Address Modifiers
Generated VME Address

Figure 4 : PCI to VME address translation

The selected Mapping Table entry configures ALMA PCI slave behavior on writes (Write posting enable/disable) and provides all the informations useful for cycle generation onto the VMEbus, such as address translation, address modifiers, Little Endian/Big Endian conversion mode and Read-prefetch enable/disable

The Mapping Table can be disabled through programming. In that mode, called «transparent mode», an ALMA register content plus some hard-coded informations replace the Mapping Table programming. The register provides two informations: the first validates the

PCI access (accepted/rejected, no distinction is made between MEMORY or I/O space accesses and the 8MB granularity filtering does not apply anymore), while the second provides address modifiers value of the VME cycle to generate. All others informations are hard-coded as follows: no address translation (PCI address is forwarded as-is onto the VMEbus), Little Endian/Big Endian conversion mode set to the Address Coherency mode, Read- prefetch and Write posting modes set to disabled.

5.2 VMEbus interface

VME Bus Master interface

The ALMA VME master module provides D32, D16, D8 and UAT under A32, A24 and A16 addressing modes plus D64 BLT and D32BLT under A32 and A24 addressing modes. Any Address Modifier (AM) is supported.

Parameters defining the VME cycle to generate (address translation, address modifiers, Read-prefetch or Write posting mode, Big-Little Endian conversion type) are taken out of values programmed into the Mapping table.

The VME block mode, D64 or D32, can be automatically started from a PCI burst. The use of internal FIFOs allows maximum speed in decoupled mode (Read-prefetch or Write posting). In the same way, even a suite of PCI single write accesses to ALMA can be translated into one VME D32 BLT or D64 BLT cycle (mode called BB2BLT).

VME Bus Slave interface

The ALMA VME slave interface supports the same addressing mode and data size as the VME master interface. Eight decoding channels are available for accessing the PCI bus from the VME bus. Each channel is independently configured in the A32, A24 or A16 address space. These channels allow the user to program the PCI bus cycle parameters with a minimum granularity of 1 Mbyte.

An additional channel available under A16 space is provided for ALMA register set access.

VME bus Requester

ALMA drives its bus requests on the 4 levels BR0 to BR3 (optional FAIR policy), release of the bus is managed according to the ROR (Release On Request), RWD (Release When Done), ROC (Release On Clear) or RNE (Release NEver) policy.

VME System controller

VME bus system controller function is enabled via pin control. The «Auto System Controller» mechanism is supported as defined into the VME64 norm[3].

ALMA VME system controller features are the following:

- VME arbiter with a programmable arbitration scheme: PRI (fixed priority) or RRS (Round Robin), plus a selectable fixed arbitration time-out of 8ms to check wether bus is effectively taken by the granted agent (elapsed time between BGOUT* and BBSY* signals assertion).
- DTB Timer. A programmable timer (4 us to 256 us) to check the data exchange duration on bus (elapsed time between DS* and DTACK* or BERR* signals assertion).
- 16 MHz SYSCLOCK generation.
- SYSRESET* generation (See Special Features).
- SYSFAIL* generation, via pin control or register access.

5.3 PCI bus interface

ALMA initiates and responds to a subset of PCI bus commands. The PCI bus command is defined by the P_CBEb[3:0] signals during the address phase of a transaction.

PCI Bus Master interface

ALMA PCI bus master interface supports I/O, MEMORY, INTERRUPT ACKNOWLEDGE and SPECIAL Cycles plus Type 0 and 1 CONFIGURATION Cycles. Any PCI bus commands can be programmed in the VME slave channels, but only protocol of those cycles are supported. Note that ALMA might initiate a bus cycle for which the bus command is part of those Reserved bus commands defined into PCI Specification rev 2.0.

Parameters defining the I/O or MEMORY bus cycle to generate (address translation, PCI bus command, Read-prefetch mode, Little/Big Endian conversion mode) are taken out of values programmed into the selected VME channel (see VME Slave interface).

INTERRUPT ACKNOWLEDGE or SPECIAL Cycle are initiated from the VME via a read or write access to a specific ALMA register.

CONFIGURATION Cycles for primary and secondary busses are initiated from the VME via an access to two specific ALMA registers used as configuration address and data registers, according to mechanism #2 of the PCI specification rev 2.0.

PCI Bus Slave interface

ALMA PCI bus slave interface decodes I/O, MEMORY and Type 0 CONFIGURATION Cycles. Note that MEMORY READ MULTIPLE, MEMORY READ LINE and MEMORY WRITE & INVALIDATE Cycles are treated as regular MEMORY READ and MEMORY WRITE Cycles.

ALMA slave image base addresses and address spaces (I/O or MEMORY) are encoded into six PCI Base Address Registers defined into its register set. Five registers are available for decoding of I/O and MEMORY accesses to the VME while one register is provided for decoding I/O accesses to a subset of its register set (192 bytes - see Register set chapter). ALMA is offering that capability as a substitute to CONFIGURATION Cycles.

In addition, a Mapping Table of 1K entries is used to enable/disable PCI MEMORY accesses (up to 4GB) as well as PCI I/O accesses (up to 4GB), with a granularity of 8MB. For any given 8MB PCI addresses block, the selected Mapping Table entry provides the following informations: PCI access accepted/rejected, Write posting mode enabled/disabled, plus all parameters required to generate a master access onto the VMEbus, such as: the address bits for PCI to VME address translation (8MB translation granularity), the VME Address Modifiers, the Little/Big Endian conversion mode for data bytes ordering, and the Read-prefetch mode enabled/disabled.

To accommodate software mapping flexibility requirements, ALMA allows for some programming of its slave image address space and range depth. So, the so-called hard-coded slave address range depth and address space (as defined into any PCI Base Address Register layout) can be selectively overridden by the software. For any of the 5 PCI Base Address Registers, address range programming range is 16MB - 2GB and address space can be set to I/O or MEMORY

PCI System Controller

ALMA implements a 6-way PCI bus arbiter (enabled/disabled via pin control) which arbitrates between 5 external devices plus ALMA itself, according to a fixed or round robin priority scheme (programmable). When the arbiter is enabled and the fixed priority mode is set, ALMA internal request has the highest priority.

When no agent is currently using or requesting the bus, the ALMA arbiter parks the bus to himself. ALMA is then the default bus owner and behaves as a parked agent as defined into PCI specification.

When the arbiter is disabled, ALMA provides request and grant pins for an external arbiter.

PCI reset or board reset (local reset) is generated by ALMA on a pin (See Special Features).

5.4 DMA

Two DMA channels are available, with a programmable priority between channels (blocks may be interlaced or not). Each channel is initialized with a source address, a destination address, a transfer count (64K words of 64 bits maximum), a block count (up to 256 words of 64 bits), plus all the necessary control data (VME Address Modifiers, PCI bus command, transfer direction, etc.). When DMA ends, the information related to its completion (normal ending or error description) is recorded into an interrupt status register and an error status register. DMA completion can be signalled via an interrupt to the PCI bus (programmable).

5.5 Interrupt management

The ALMA interrupt controller can handle the following interrupt sources:

- 7 VME interrupts (IRQ7*-IRQ1*).
- 8 addressed interrupts (occur when a specific 8-bit register is addressed in write mode from the PCI bus or the VMEbus).
- ACFAIL* and SYSFAIL* occurrence on VMEbus.
- Internal Exceptions (end of DMA, abnormal transfer terminations on PCI bus or VMEbus, VMEbus arbitration timeout...).

All these interrupts can be masked or can drive either the PCI bus interrupt pin (INTA#) or any of 3 extra interrupt output pins provided by ALMA.

As interrupter, ALMA generate any level VME Interrupts (IRQ7*-IRQ1*) when a specific 8-bit register is addressed in write mode (from the VMEbus or the PCI bus). The release mechanism of this interrupter is ROAK (Release on Acknowledge)

An external interrupt input pin is provided especially for PCI interrupts. When asserted ALMA will generate either a VME interrupt (IRQ7*-IRQ1*) or a programmable VME bus cycle (read/write, address, address modifiers and data).

5.6 Special Features

Reset controller

ALMA can be reset by different ways, as listed below:

- Power-on-reset (pin).
- VME reset (pin SYSRESET*).
- Local (board) reset (pin).
- Addressed Reset (via a specific register access).
- Reset due to ALMA Reset Watchdog function.

Some of these reset informations can be propagated (depending upon programming) to one or several of the following locations:

- to ALMA internal logic.
- to the VME (pin SYSRESET*).
- to Local (board) (pin)

Little/Big Endian byte ordering

Little Endian and Big Endian byte ordering conventions are supported by ALMA. Little Endian/Big Endian conversions are automatically performed on both directions of the data stream, according to a conversion mode programmed into the VME slave channels (for VME accesses) and the PCI Mapping Table (for PCI accesses). ALMA implements the 4 conversion modes depicted below:

- mode “Address Coherency“: data bytes are swapped, address 2 low-order bits remain as-is.
- mode “Data Coherency”: data bytes ordering is kept as-is, address 2 low-order bits are modified.
- mode “Bytes Translation with No Swapping”: data bytes are translated, address 2 low-order bits remain as-is.
- mode “No conversion“: data bytes ordering and address 2 low-order bits remain as-is.

Some dynamic bus sizing is also performed by ALMA when it performs VME 64bit to/from PCI 32 bit data transfers.

Decoupled mode

Two FIFOs (receive and transmit; 8 by 32 bits), give ALMA the capability to fully decouple data streams between VME and PCI busses. This way, PCI bursts can be translated into a VME block mode and VME block mode can be translated into PCI bursts when Write posting and Read-prefetch modes are enabled (refer to PCI slave interface and VME slave interface chapters). They give the user the capability to get the maximum performance from the PCI bus and VME bus.

Figure 4 shows PCI bus to VMEbus decoupled data transfers (8 data transfers) for a PCI write access with Write posting mode enabled or VME read access with Read-prefetch mode enabled

On a VME D32/D64 BLT read/write, ALMA will burst 8-data blocks onto the PCI bus (ALMA releases the bus between blocks).

On a PCI burst read/write, ALMA will continuously burst into D32/D64 BLT mode onto the VMEbus, in case where FIFO is full, ALMA will retry the PCI access.

In addition, a specific ALMA transfer mode called BB2BLT can convert a suite of PCI single data writes into one VME block cycle, allowing thus high performance data transfer for those non-bursting PCI master devices.

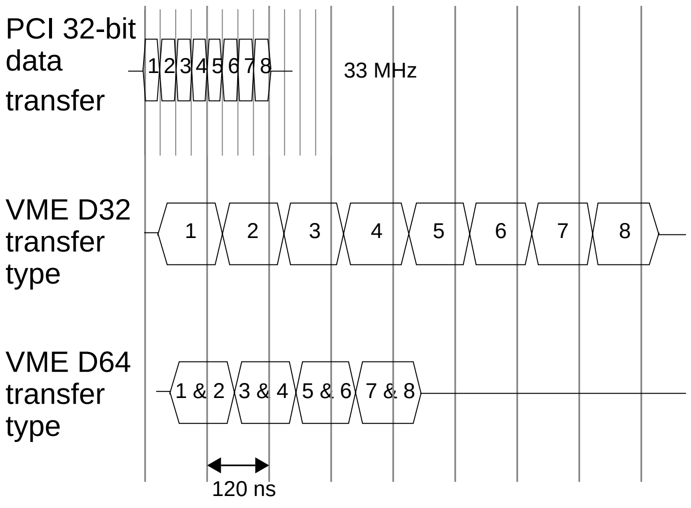

Figure 5 : In the case of a VME read access in block mode or a PCI write access in burst mode: PCI data are transmitted immediately to the VME while FIFO get filled, so as to minimize PCI bus occupancy.

5.7 Hardware semaphores

VME Multiprocessor architectures often need common resources between different VME cards to be shared. The ability for a card to have hardware mechanisms for handling semaphores is one way to solve the issue.

The standard VME specification revision C read-modify-write cycle is able to support semaphores. Unfortunately, there are a few drawbacks in using this RMW VME protocol:

- 1. RMW operation in the VME specification revision C is a slow mechanism and supporting it could even slow down non RMW VME cycles. This is due to the signalling method used on the VMEbus: in fact indivisible cycles can only be recognized at the end of the read phase.
- 2. RMW cycles are often not used nor generated by modern microprocessor architectures, and hardware emulation of such cycles is not easy to design.

For these reasons, the ALMA chip includes 4 shared 8-bit Semaphore registers so that the design of VME multiprocessor architectures can be greatly simplified.

5.8 ALMA register set

A 256-byte space named «ALMA Configuration Space» is allocated to the ALMA register set.

The first 64-byte block define the «ALMA Configuration Space Header», and the remaining 192-byte block is named «ALMA Configuration Space Specific». A VME master may access both spaces via VME A16 read/write cycles. A PCI master may access both spaces via PCI CONFIGURATION Cycles, and, access only the «ALMA Configuration Space Specific» via PCI I/O Cycles (this feature allows some PCI agents, with no means to generate PCI CONFIGURATION Cycles to access ALMA configuration registers too).

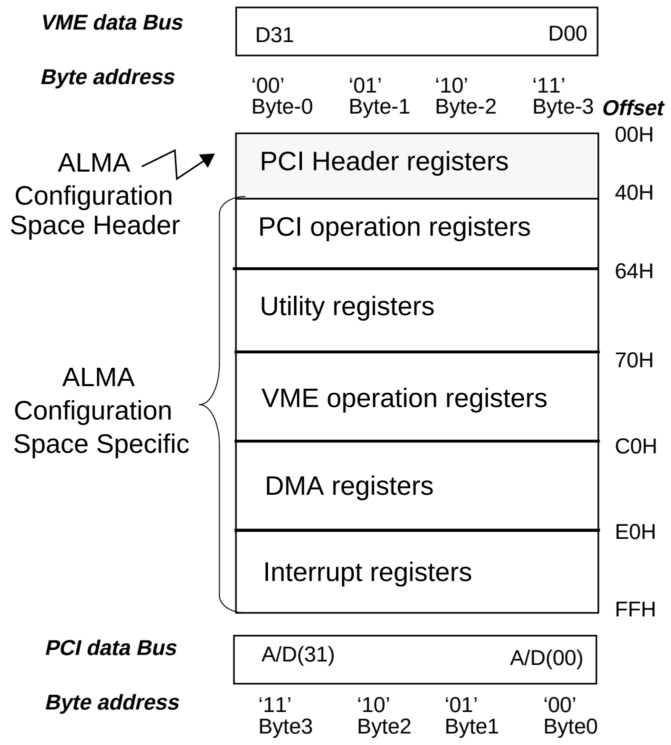

Figure 6 : ALMA registers mapping

6.0 DESIGN METHODOLOGY

The logic design has been done in synthesizable Register Transfer Level (RTL) language using Verilog tool from Cadence. The synthesis of the circuits has been performed with Synopsys Logic Synthesis tool.

Design challenges were complexity of the VME and PCI bus when all options are chosen, plus performance and timing requirements on PCI logic.

The physical design has been thoroughly done by automatic placement and routing software tools, on a 7mm x 7mm 89K cells standard cell image.

ALMA Verilog model has been simulated with cycle-based behavioral bus models from Synopsys (bus master, bus slave, bus arbiter and bus monitor models for VMEbus and PCI bus).Total simulation test-cases exercised represent about 35,000 lines of code.

ALMA CQFP prototype have been tested using Cetia VMTR2 board (Power Engine VME card) on the VMEbus and the “Hewlett Packard 2910A PCI bus exerciser” and the Cetia CVME604 on the PCI bus. A special PCI mezzanine card carrying ALMA was built to connect to boards (see Figure 7).

Following is an overview of the various types of test performed:

- ALMA register set access from both ports.
- All types of VME to PCI data transfer initiated by the VMTR2 and targeting the Hewlett Packard PCI bus exerciser and the DRAM memory of the CVME604
- All types of PCI to VME data transfer initiated by the Hewlett Packard PCI bus exerciser and targeting the DRAM memory of the VMTR2 card.
- DMA transfers between both DRAMs.
- VME system controller functions.
- PCI bus arbiter function.
- Reset operations.
- some basic handling of interrupts

Some of these tests have also been exercised with Lynx/OS running on the CVME 604 board.

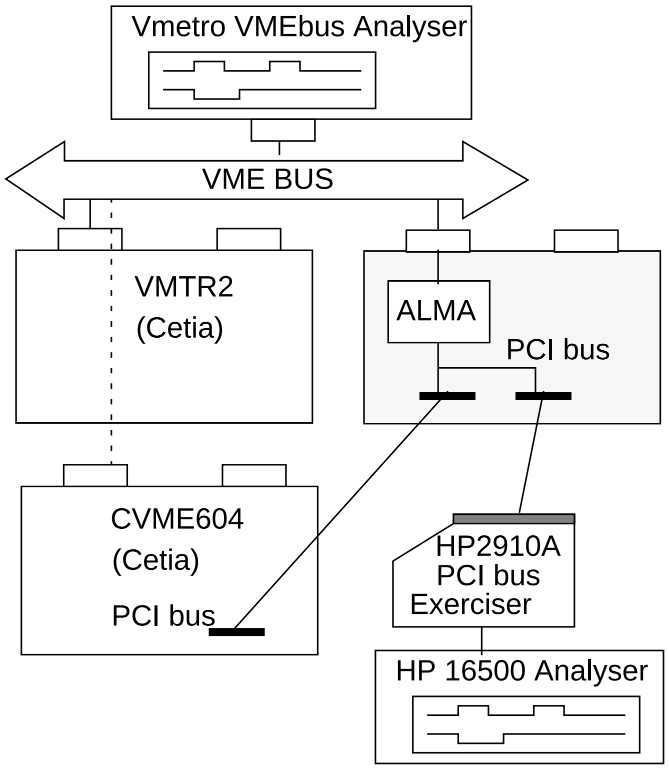

Figure 7 : Test bench for debugging and validation

Hardware results

As shown on test results of figure 6, the tested prototypes were running at 33MHz. In spite of the extensive simulation, 3 bugs escaped especially when performing DMA in case of several Retries on the PCI bus. This was without impact on prototype card operation since the occurrence of the case is extremely rare and can be avoided with appropriate software fixes.

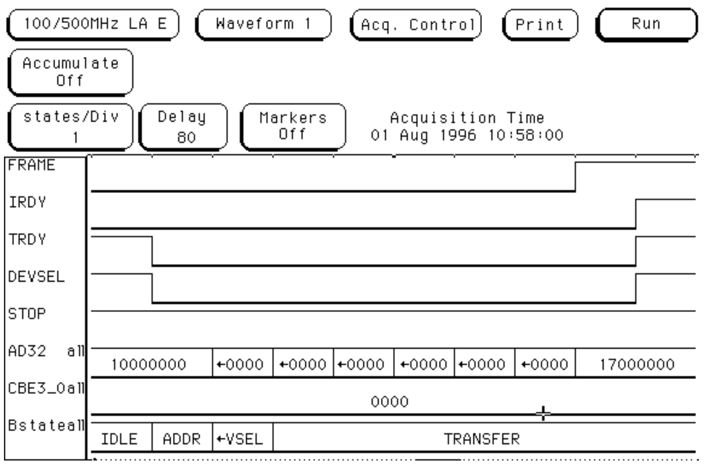

Figure 8 : Typical ALMA waveforms for VME BLOCK A32/D32 -> PCI Memory burst 33 MHz

Features

ALMA PCI to VME Bridge Specifications	
Technology	0.7 um CMOS, three levels of metal
Temperature Range	0 to 55 degree C room temp
Performance (est.)	up to 70 MByte/sec on the VME up to 132 MByte/sec on the PCI @ 33MHz
Signal I/Os	229
Power Supply	3.3V +/- 5% (supports for 5V I/Os)
Power Dissipation	0.72 W @ 33MHz (worst case)
Packaging	- 304-pin , 40mm, CQFP - 350-contact, 25mm, Ceramic BGA

Conclusion

A PCI to VME bridge providing high degree of functionality on both PCI and VME busses has been designed and tested successfully in a 0.7um CMOS ASIC technology

This integrated circuit permits design of VME cards with the minimum of additional buffer circuits, and with a maximum of flexibility to be used in most of VME applications that want to take advantage of high performance and low cost inherent to PCI extension.

Acknowledgments.

The authors would like to thanks people from Cetia and from IBM who have contributed to the definition and the design of this integrated circuit.

References:

[1] The VME64 specification - Jan, 94
[2]"PCI Local Bus specification Rev. 2.0" PCI Special Interest Group June 95
[3]"Advantages of PCI Local Bus on VMEbus CPU Boards" Chris Engels - Real-Time Magazine 95/1
[4]"Interfacing VMEbus and PCI bus" R. Negre - RTS Jan 95

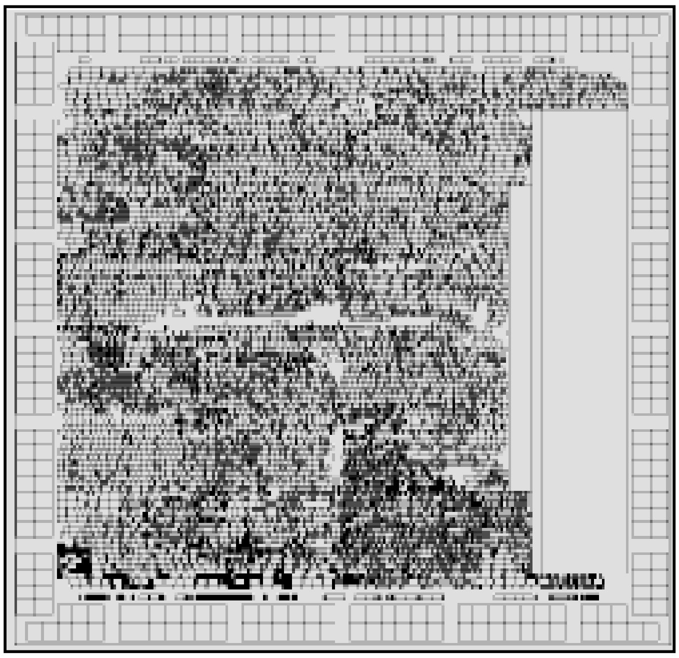

Figure 9 : ALMA 7.1 x 7.35mm Chip layout