



**Fermi National Accelerator Laboratory**

**FERMILAB-Conf-95/329**

**Extensions to the VME Hardware and Software Standards  
for the Physics Community**

R. Pordes

For the US VME-P and Cern VSC Committees

*Fermi National Accelerator Laboratory  
P.O. Box 500, Batavia, Illinois 60510*

October 1995

Proceedings of the *Computing in High Energy Physics 1995 (CHEP 95)*,  
Rio de Janeiro, Brazil, September 18-22, 1995

## Disclaimer

*This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.*

The actual instance of any Component is extended to include additional information needed for the implementation to read or write data over the VME bus - e.g. the address modifier to use, the ethernet path to the VME intelligent controller which does the read/write etc.

A first implementation of this extended API is being coded by the CDF experiment at Fermilab in C. The Working Group is seeking a second site for implementation - preferably in C++ - to test the correctness and completeness of the interface defined. We expect a first implementation of the API by the end of this year.

## 5 Status and Plans

VIPA efforts have assisted in heightening interest in a number of areas throughout VSO and VITA. The organization is highly motivated to promulgate extensions to the VME and VME64 standards which meet the needs of the Physics Community as it moves to develop a large proportion of its backplane modules into the VME standard.

The VIPA Committee is working on a combined VME-P and VSC document on VME standards in HEP, for release within six months. The VSC document "Recommended Practices for the Use of VMEbus in High Energy Physics" is being adopted by CERN as the standard by which the LHC Experiments will design and implement VME modules for the immediate future.

## 6 Acknowledgements

We acknowledge the members of the VIPA and VSC committees who are working on defining and coming to consensus on the details of the hardware and software specifications.

## 7 References

1. VIPA url: <http://www-ese.fnal.gov> and [http://www.cern.ch/ecp-ess/vsc/vipa/vipa\\_welcome/html](http://www.cern.ch/ecp-ess/vsc/vipa/vipa_welcome/html)
2. VITA url: <http://www.vita.com> John Rynearson (VITA Technical Director), VMEbus International Trade Association, 10229 North Scottsdale Road, Scottsdale, US-AZ 85253-1437
3. VSC url: [http://www.cern.ch/ecp-ess/vsc/vsc\\_welcome.html](http://www.cern.ch/ecp-ess/vsc/vsc_welcome.html). Convenor: Chris Parkman CERN/ECP, ECP Division, CERN, CH-1211 Geneva 23
4. VME-P url: <http://www-ese.fnal.gov/vme-p/web/vme-p.htm>; Chairman: Ed Barsotti FNAL, Fermi National Accelerator Laboratory, PO Box 500, Wilson Road, Batavia, US-IL 60510
5. url: <http://www-online.kek.jp/VIPA/>; Masaharu Nomachi, National Laboratory for High Energy Physics - KEK, Oho 1-1, Tsukuba, JP-Ibaraki 305
6. VME64 Extensions Specification (VITA 1.1-199x, latest draft), VITA, 10229 N. Scottsdale Road, Suite B, Scottsdale, AZ 85253-1437, USA
7. [http://www-cdf\\_online.fnal.gov:62000/vme/vme\\_software.html](http://www-cdf_online.fnal.gov:62000/vme/vme_software.html). Contact Jim Pangbur, Fermilab, POBox 500, Batavia, Illinois 60510.
8. <http://www.vita.com/esseindex.html>

Table 1: Proposed VME API

vmeInfo()	information including: detailed error info, API revision implemented by software, controller hardware name and revision, etc.
vmeConfig()	control behavior of transfer routines: default to "best" protocol, but can explicitly set single-word transfer, etc. plus information specific for the access method.
vmeIntConnect()	specify an application routine to invoke upon receipt of VMEbus interrupt.
vmeComponent()	"housekeeping" (load, unload, etc.) and access to "extended functions" possibly provided by access method
vmeListBegin()/ ..End()/..Exec()	(optional) list-building - perform multiple actions with a single call.

The API introduces the concept of a "Component" to which is tied all details of the access method (direct memory mapped, use dma controller, VME controller accessed remotely over Ethernet etc). This paradigm allows the routine interface for the basic read and write operations to be independent of the access method. It provides for an Object Oriented paradigm where the routines which actually "do the work" can be overloaded by the most efficient implementation of the particular access. Not only is a data structure maintained for each "Component" but at run time different routines can be downloaded and/or invoked in an embedded controller depending on the Component being used.

Support for the concepts of "configuration" and "autoconfiguration" lead to support for "Plug and Play" systems where prior knowledge of the hardware configuration is not needed. Changes to the hardware configuration are propagated up from the devices themselves and automatically taken account of and used by the software layers. Towards these goals, the Component Types - or standard access methods - proposed so far are listed in Table 2 below:

Table 2: VME Access Methods

std	standard VME addressing.
geo32	A32-based geographical addressing.
autoslot	autoslot-ID from VME64 specification.
autoconf	autoconfiguring system: reads database of module IDs, discovers what is in a crate.
cblt	for new chained block transfer (CBLT) protocol.
redundancy	to provide redundancy functions, to provide alternate access parths for critical systems.

### 3 Hardware Activities

Since “Computing in High Energy Physics” is a conference that concentrates on computer architecture and software we will say little about the activities of the committees in the hardware arena. Suffice it to say that the VIPA (VME-P and VSC) committees are very active in the following areas: Master Terminated (known data length) and Slave Terminated (unknown data length) “2 edged Block Transfers” (2eblt); data broadcast protocols; live insertion of boards (longer precharge ground & voltage pins); more ground pins for a quieter backplane (new 160-pin 5-row connectors); geographical addressing pin assignments & implementation; P0/J0 connector (between P1/J1 & P2/J2 connectors); and board & subrack keying. The VME64 standard was approved by ANSI as of May 5th, 1995 [6].

### 4 Software Activities

The VIPA Software Working Group [7] is exploring various areas in which standardization or consistency will be beneficial. Additionally the group reviews the hardware proposals for software impact and requirements. The group is interested in the CSR and ROM definitions; the higher level protocols needed to transfer data between devices; standard layers of software to access VME devices and control VME interfaces; and in software support for the upcoming “Plug and Play” hardware system implementations. Members of the committee are becoming active in the Embedded Systems Software Environment (ESSE) [8] - a newly formed initiative from VITA which is attempting to address the issues of standards for Embedded Systems Software over a very wide range.

The VIPA Software Working Group is starting at the simplest level to define a software API to provide a very simple infrastructure for VME I/O. While this interface can be used for the simplest memory-mapped access to VME modules, it is being defined and implemented to allow support for the new advanced functions (e.g. slave terminated block transfers). Additional to reads and writes the API includes provision for connecting routines to be executed on receipt of a VME interrupt.

The basic API shown in Table 1 below provides a set of bindings to perform VME I/O independent of any particular “access method” being used to perform the operation.

Table 1: Proposed VME API

vmeOpen()	establish access to VME module, name argument specifies access method to use.
vmeClose()	terminate access to something opened.
vmeRead()/ vmeWrite()	transfer routines; different for each access method.
vmeMemMap()	establish mapping between VME space and process address space.
vmeSync()	ensure I/O completion, needed for vmeMemMap() and asynchronous operation.

# **Extensions to the VME Hardware and Software Standards for the Physics Community**

Ruth Pordes for the  
The US VME-P and Cern VSC committees

The United States VME for Physics (VME-P) Committee and the European ESONE/CERN VME Steering Committee (VSC) are working together with the commercial sector on extending the VME standard. This paper reports on the current status and directions of this standards effort .

## **1 Goals**

With the ubiquitous availability of commercial VME modules and interface chips many experiments in all branches of Experimental Physics are turning to VME as the module packaging and backplane protocol of choice for application specific modules. The current VME backplane, electrical, and mechanical standards are clearly lacking when it comes to implementing the analogue and digital front end modules that Physics experiments rely on to provide the needed high speed and intelligent solutions to their data collection requirements.

To address these needs the European ESONE and American VME-P committees are collaborating with the ANSII and ISO standards committees, and the VME VITA/VSO manufacturers associations, to define standard extensions to the VME protocols for such HEP needs as "sparse data scans", standard uses and implementations of previously user defined pins and connectors, standardization of the 9U form factor VME cards etc.

In addition, based on the overall positive feedback to the definition of CAMAC and FASTBUS software interface standards, definitions of standard software access mechanisms for use with VME modules are being defined.

## **2 Organization**

The VMEbus International Physics Association (VIPA) [1] is an umbrella organisation grouping interests in Europe, North America, and Japan together to work with VITA and the VITA Standards Organisation (VSO) concerned with the use of VMEbus in physics applications. The VMEbus International Trade Association (VITA) [2] promotes open standards, and is the "guardian" of VMEbus.

VIPA aims to encourage a consensus on the VMEbus standard within the physics community - including the forthcoming extensions to the standard such as "VME64 Extensions" and requirements specific to HEP applications - such as support for sparse data collection from multiple devices.

In Europe, VIPA is represented by the ESONE/CERN VMEbus Steering Committee - VSC [3]. In the United States and Canada, VIPA is represented by the NIM VME-P committee [4]. Japan is represented by a VME-P member organisation at KEK [5].