ANSI/VITA 3-1995

Approved as an American National Standard by

American National Standard
for Board Level Live Insertion for
VMEbus

Secretariat
VMEbus International Trade Association

Approved January 12, 1996
American National Standards Institute, Inc.

VMEbus International Trade Association
7825 E. Gelding Drive, Suite 104, Scottsdale, AZ 85260 USA
PH: 602-951-8866, FAX: 602-951-0720
E-mail: info@vita.com, URL: http://www.vita.com
American National Standard
for Board Level Live Insertion for VMEbus

Secretariat
VMEbus International Trade Association

Approved January 12, 1996
American National Standards Institute, Inc.

Abstract
This document recommends practices to implement board level live insertion with existing VMEbus boards.
Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether they have approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

Published by

VMEbus International Trade Association
7825 E. Gelding Drive, Suite 104, Scottsdale, AZ 85260

Copyright © 1996 by VMEbus International Trade Association
All rights reserved.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without prior written permission of the publisher.

Printed in the United States of America - R1.1
ISBN 1-885731-20-5
ABSTRACT

Advances in technology allow smaller board sizes to perform the functionality that a few years ago required a larger board. Alternatively, these advances allow increased functionality on same sized boards. Thus, VMEbus boards and system designs are greatly increasing in capability. Where once a radio room had six separate chassis containing one radio each, that radio room today has one chassis housing six radios. This size reduction has created the demand for highly reliable and easily maintainable systems. The Telecommunications and Defense applications require a more fault tolerant, live insertable VMEbus. Currently, live insertion and extraction is not supported by the IEEE 1014 VMEbus standard. The VMEbus and Futurebus+ Extended Architectures International Trade Association (VITA) has responded by forming the Recommended Practices Guide to Board Level Live Insertion for the VMEbus Working Group to examine solutions for a Board Level Live Insertable VMEbus. This document recommends practices to implement board level live insertion with existing VMEbus boards.
FOREWORD

The purpose of the Board Level Live Insertion Working Group is to develop a recommended practices document for Board Level Live Insertion on VMEbus systems. This document identifies methodologies through which a faulty board can be removed from a system and a replacement board can be inserted while the system continues to operate.

The primary motivation for supporting Board Level Live Insertion within the VMEbus environment is to enhance the current VMEbus standard while maximizing the use of existing off-the-shelf VMEbus products. As more functionality is compressed in a smaller package, the system operator cannot afford to shut the system off while a failed board is replaced. Board Level Live Insertion has been driven by customer demand in both the telecommunications, military, and aerospace industries. It is hoped that a standard live insertion methodology will be adopted by industry at large as hardware and software costs to implement the technology decrease. With the VMEbus being used in many mission critical applications, the need for live insertion becomes critical.
ACKNOWLEDGMENTS

The sacrifice of time that the members of the Live Insertion technical committee provided in developing and producing this document is greatly appreciated. Without their support, time and talents this document would still be an idea instead of a working methodology.

Members of the Working Group

Ray Alderman       Pep
Martin Blake       Bicc Vero
Dick DeBock        Matrix
Mika Sudoh         VMEBUS Member
Mike Galloway      AMP INC
Elwood Parsons     AMP INC
Jing Kwok          DY 4
Mansour Zarreii    AMP INC
Bob Blau           Mercury Computer
Jim Koser          Berg Electronics
Wayne Fischer      Force Computers
Frank Hom          Electronic Solutions
David Robak        Harting Elektronik Inc
Michael Thompson   Schroff INC

Contributing Authors:
Malcolm Airst      MITRE
Colin Davies       Radstone Technology
Wayne Fischer      Force Computers
Ben LaPointe       Motorola GEG
Robert McKee       MITRE
Kelvin Rawls       MITRE

Draft Editor:
Robert Mckee       MITRE
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.1</td>
<td>Discussion on this Document</td>
</tr>
<tr>
<td>1.2</td>
<td>Live Insertion Methodologies</td>
</tr>
<tr>
<td>1.3</td>
<td>Levels of Live Insertion for the VMEbus</td>
</tr>
<tr>
<td>1.4</td>
<td>Fault Tolerance Phases</td>
</tr>
<tr>
<td>1.4.1</td>
<td>Normal Operation</td>
</tr>
<tr>
<td>1.4.2</td>
<td>System Fault Detection</td>
</tr>
<tr>
<td>1.4.3</td>
<td>Fault Identification</td>
</tr>
<tr>
<td>1.4.4</td>
<td>Fault Isolation</td>
</tr>
<tr>
<td>1.4.5</td>
<td>Board Replacement</td>
</tr>
<tr>
<td>1.4.6</td>
<td>Bringing the VMEbus Board Back On-Line</td>
</tr>
<tr>
<td>1.4.7</td>
<td>Returning to Normal Operation</td>
</tr>
</tbody>
</table>

| 2 | Normal Operation | 2-1 |
| 2.1 | Introduction | 2-1 |
| 2.2 | Definition of Failure, Fault, and Error | 2-1 |
| 2.3 | Definition of Normal Operation | 2-1 |

| 3 | System Fault Detection | 3-1 |
| 3.1 | Introduction | 3-1 |
| 3.2 | Detection Mechanisms | 3-1 |
| 3.2.1 | Voting or Checking Mechanisms | 3-1 |
| 3.2.2 | Error Logs | 3-2 |
| 3.2.3 | Watch-dog Timer | 3-2 |
| 3.2.4 | On-Line "BIT" or Health Monitoring | 3-2 |
| 3.3 | VMEbus Backplane Specific Failure Modes | 3-3 |
| 3.3.1 | System Slot One Controller Failure | 3-4 |

| 4 | Fault Identification | 4-1 |
| 4.1 | Introduction | 4-1 |
| 4.2 | Discussion of Fault Identification | 4-1 |
| 4.3 | Types of VMEbus Faults, Failures, and Errors | 4-2 |
| 4.4 | Possible Methods for Fault Identification on the VMEbus | 4-4 |
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 Fault Isolation</td>
<td>5-1</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>5-1</td>
</tr>
<tr>
<td>5.2 Fault Confinement</td>
<td>5-1</td>
</tr>
<tr>
<td>5.3 Isolation Techniques</td>
<td>5-2</td>
</tr>
<tr>
<td>5.3.1 Logical Isolation</td>
<td>5-2</td>
</tr>
<tr>
<td>5.3.2 Electrical Isolation</td>
<td>5-2</td>
</tr>
<tr>
<td>6 Board Replacement</td>
<td>6-1</td>
</tr>
<tr>
<td>6.1 Introduction</td>
<td>6-1</td>
</tr>
<tr>
<td>6.2 Board Replacement Considerations</td>
<td>6-1</td>
</tr>
<tr>
<td>6.2.1 Board Isolation</td>
<td>6-1</td>
</tr>
<tr>
<td>6.2.2 Service Suspension Versus Interruption</td>
<td>6-3</td>
</tr>
<tr>
<td>6.2.3 Insertion / Extraction of VMEbus Boards</td>
<td>6-4</td>
</tr>
<tr>
<td>6.2.4 Slot 1 Considerations</td>
<td>6-4</td>
</tr>
<tr>
<td>6.2.5 VMEbus Conformance</td>
<td>6-4</td>
</tr>
<tr>
<td>6.3 Implementation Options</td>
<td>6-5</td>
</tr>
<tr>
<td>6.3.1 The Buffer Board Implementation</td>
<td>6-5</td>
</tr>
<tr>
<td>6.3.2 The Backplane Implementation</td>
<td>6-7</td>
</tr>
<tr>
<td>6.3.3 The Connector Implementation</td>
<td>6-9</td>
</tr>
<tr>
<td>6.4 Electrical Considerations</td>
<td>6-9</td>
</tr>
<tr>
<td>6.4.1 Isolation Circuit Characteristics</td>
<td>6-9</td>
</tr>
<tr>
<td>6.4.2 Power Ramp Timing</td>
<td>6-9</td>
</tr>
<tr>
<td>6.4.3 Power Handling Capacity</td>
<td>6-10</td>
</tr>
<tr>
<td>6.4.4 Power Consumption of Live Insertion Circuits</td>
<td>6-10</td>
</tr>
<tr>
<td>6.5 Protocol Considerations</td>
<td>6-10</td>
</tr>
<tr>
<td>6.6 Mechanical Considerations</td>
<td>6-11</td>
</tr>
<tr>
<td>6.6.1 Notification</td>
<td>6-11</td>
</tr>
<tr>
<td>6.6.2 Daisy Chain Connections</td>
<td>6-12</td>
</tr>
<tr>
<td>6.6.3 Connector Contact Bounce on the VMEbus</td>
<td>6-12</td>
</tr>
<tr>
<td>6.7 Software Considerations</td>
<td>6-13</td>
</tr>
<tr>
<td>6.7.1 Notification</td>
<td>6-13</td>
</tr>
<tr>
<td>6.7.2 Interaction Between Live Insertion Logic and Controlling Software</td>
<td>6-13</td>
</tr>
<tr>
<td>6.8 Design Example</td>
<td>6-14</td>
</tr>
<tr>
<td>6.9 Isolation timing</td>
<td>6-17</td>
</tr>
<tr>
<td>6.10 Warnings</td>
<td>6-19</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 System Back On-Line</td>
<td>7-1</td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>7-1</td>
</tr>
<tr>
<td>7.2 Insertion into the VMEbus Backplane</td>
<td>7-1</td>
</tr>
<tr>
<td>7.2.1 VMEbus Control during Insertion</td>
<td>7-1</td>
</tr>
<tr>
<td>7.2.2 VMEbus power, control, and signal cycle</td>
<td>7-2</td>
</tr>
<tr>
<td>7.3 Power up Built In Test</td>
<td>7-2</td>
</tr>
<tr>
<td>7.4 Establishing an on-line status with software</td>
<td>7-3</td>
</tr>
<tr>
<td>7.5 Resumption of software tasking</td>
<td>7-3</td>
</tr>
<tr>
<td>7.6 Normal Operation</td>
<td>7-3</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-1</td>
<td>VMEbus Signal Groups 6-2</td>
</tr>
<tr>
<td>6-2</td>
<td>The Buffer Board Implementation 6-6</td>
</tr>
<tr>
<td>6-3</td>
<td>The Backplane Implementation 6-8</td>
</tr>
<tr>
<td>6-4</td>
<td>An Example System Configuration 6-15</td>
</tr>
<tr>
<td>6-5</td>
<td>An Operational Flow Chart 6-16</td>
</tr>
<tr>
<td>6-6</td>
<td>A VMEbus Timing Diagram of the Isolation Interruption 6-18</td>
</tr>
</tbody>
</table>
RECOMMENDED STANDARDS


IEEE 1014-1987  VMEbus  "versatile Backplane Bus: VMEBUS"
IEEE 1096   VSB  Multiplexed High Performance Bus Structure
IEEE P1014.1   VME 64 to FB+ Bridge
IEEE 1101.1   Convection Cooled Eurocard
IEEE 1101.2   Conduction Cooled Eurocard
IEEE 1155   VMEbus Extensions for Instrumentation; VXIbus.

MIL-C-87167 VMEC

MIL-B-87166 VMEB

For Information on VXI address correspondence to:

    VXIbus Consortium, INC.
    P.O. Box 1736
    Vancouver, WA. 98668

MITRE Corp. The VMEbus Systems Engineering and Implementation
Handbook Document # MTR 93W0000005 June 1993
SECTION 1

INTRODUCTION

1.1 DISCUSSION OF THIS DOCUMENT

This document represents a methodology or recommended practices guide for inserting (into) or extracting (out of) VMEbus cards in an operating VMEbus system. This document does not detail a point solution; this is left to the implementor. The primary directive when defining this methodology was to maintain maximum compatibility with existing off-the-shelf VMEbus boards.

Board level live insertion enables a VMEbus board to be removed and replaced without removing power from the system. The advantage of live insertion is that it allows system enhancements or fault diagnosis without the lengthy delays associated with switching off the power to a system. Thus, Live Insertion is used in applications where continuity of service is crucial. Modifications to the VMEbus system may be required to implement a live insertion solution, since insertion and removal of boards while the system is powered could cause glitches in various signal and power lines. Live insertion of a VMEbus board drawing a relatively high current could cause the power supply to temporarily lose voltage regulation. In addition, removal of a VMEbus board requires the jumpering of the BG (bus grant) and IACK (interrupt acknowledge) daisy-chain lines in order to comply with the VMEbus protocols.

Fault tolerance is the ability of a system to continue operating, possibly in a degraded mode, while a fault is detected and isolated to a single replaceable unit. Live insertion is only one element of a fault tolerant system. Even if a board can be replaced live, a poorly designed system will lose the functionality provided by the failed board unless that functionality is duplicated. For example, if a LAN interface card fails in a VMEbus based LAN server, the server will lose its LAN connectivity unless the LAN card's functionality is duplicated. A fault tolerant system would have redundant connectivity to the LAN. A fault tolerant system also requires fault tolerant software in addition to redundant hardware. The system software must switch among redundant modules, handle module failure and support initialization (rebooting) while the system continues to perform its primary function. Fault tolerance thus requires hardware which provides the means for software to detect errors and isolate the problem to a replaceable unit. Software is, therefore, a critical element of a fault tolerant system, but is NOT addressed in this document. This document only addresses live board replacement, not fault tolerance in general. There is a great deal of on-going fault tolerant research including fault isolation, detection, and identification. However, this document is limited to presenting a methodology for the enhancement of standard VMEbus system hardware to support the insertion and removal of VMEbus boards from the backplane while power is applied.
1.2 LIVE INSERTION METHODOLOGIES

Two methods of board level live insertion are feasible. The first method involves creating a "womb" around standard VMEbus boards and suggests the use of a special active backplane, or live insertion adapter with existing VMEbus boards. This method is preferred because it is backward compatible with the off-the-shelf VMEbus boards.

A second possible method for board level live insertion involves the use of a passive backplane. Existing VMEbus boards would be re-designed to implement the live insertion functionality by adding an extra connector, changing the current driver capability to include Enhance Transceiver Logic components design, and additional fault tolerant silicon circuitry. Non-live insertable VMEbus boards could operate in this type of system, without being live inserted. This method could bring additional fault tolerant features that today's standard VMEbus boards currently do not support. This method is not discussed in this document because it violates the prime objective of this recommended practices guide, which is, maintaining compatibility with existing off-the-shelf VMEbus boards.

1.3 LEVELS OF LIVE INSERTION FOR THE VMEBUS

There are three levels of VMEbus operation during a live insertion process. In the first level, there is no interruption to VMEbus backplane cycles. This operation level of Board Level Live Insertion would provide a very desirable performance capability. The drawback is the extra hardware and software required to gradually isolate a board from the system and then return a board to operation. This isolation and return to operation must be done to that specific VMEbus slot while sequencing power, bus grant, interrupt acknowledge and other signal lines in addition to not glitching or disturbing the systems operation.

First Level:

There will still be a significant software overhead.

In the second level there is minimum interruption to VMEbus backplane cycle operation. Although less desirable from a performance perspective, this level is less complex due to the reduced hardware requirements. The system controller could gain control of the bus before removal and replacement, and suspending bus traffic.

The focus of this document is in implementing the second level of live insertion for the VMEbus.

Second Level:

A hardware imposed interruption comparable with typical VMEbus read/write cycles, from a hardware point of view. There could still be a significant software overhead, but not always.
Third Level: In the third level the system could use the operator to assist in the live insertion replacement cycle. This would require a significant amount of interruption to allow the entire replacement procedure to occur. Service suspension until the hot swap is complete. This is always applicable to the hot-swap of the system processor in a single processor system. The advantage is that memory contents are preserved on boards which are not powered down.

1.4 FAULT TOLERANCE PHASES

The phases of fault tolerance are described in the following paragraphs. Understanding the phases of a fault tolerant system will enhance the user's understanding of the Live Insertion process and its role in the phases of a fault tolerant system.

1.4.1 Normal Operation

Normal operation is the desired phase of the VMEbus system. Data transfers are proceeding normally across the backplane without error. A VMEbus configured system is in normal operation when the system's preloaded tasking software and its configured hardware are performing with no interruption due to error, fault or failure.

1.4.2 System Fault Detection

The system could detect errors in normal operation by frequently running a set of diagnostics as a background task. Another method is to have the modules or cards generate a health pulse to the system monitor announcing its continued service. External monitoring devices, such as link analyzers or test sets, may also initiate the fault detection process. System level software may also play a large role in the implementation of fault detection in a configured VMEbus system.

1.4.3 Fault Identification

Once a fault is detected VMEbus system must identify an error, fault or failure by running its Built In Test (BIT) routines. The routines may be implemented in either the system software, hardware or both. Once a fault is identified, the isolation process begins. Special notice should be given to the human interfaces. The Human Computer interfaces must clearly tell the operator via indicator (i.e., LED) or screen message that the system has a failure.

1.4.4 Fault Isolation

The VMEbus system must now isolate the fault. It could do this by initiating a system-wide set of instructions and tests once the system has identified the failure. The system could then reconfigure itself to eliminate or limit the fault from its standard tasks. This operation can be carried out with either on-board firmware or by operating system
software. Alternately, the system operator may manually conduct a series of system wide
tests. Manually conducted tests reduce the amount of software required. But manual
tests increase the time required for fault isolation and decrease the ability of the system to
continue performing its assigned tasks. External test equipment may assist in isolating
the fault. If the system has redundant functionality for the faulted board, it must now
switch to the redundant board. External inputs and outputs must also be switched as
required. If the system does not have redundant capability, it must inform the user of the
failure. In either case, the system must inform the user to start the live insertion process.

1.4.5 Board Replacement

The operator must prepare the system for board replacement. The chassis is opened for
access to the VMEbus boards and the operator visually identifies the faulty board. This
can be accomplished via the operator's terminal or via LED(s) on the board or chassis
front panel. The board is prepared for live insertion by disconnecting any wiring or
connectors on the front panel. The operator could initiate the live insertion process by
closing a switch contact or other means. The live insertion system provides feedback to
the operator, indicating when the system is ready for live insertion. The faulty board is
removed and the replacement board is inserted. The system must present its status via a
standard interface to allow the operator to be called. The human interface must clearly
tell the operator via indicator (i.e., LED) or screen message the system's live insertion
phase and progress. The operator must remove the board only when the system indicates
that it is ready.

1.4.6 Bringing the VMEbus Board Back On-Line

The replacement board is brought back on-line by sequencing power and signals with
minimal interruption of the VMEbus system's operation. The board then informs the
VMEbus system that the replacement is operating correctly. When this is accomplished,
the VMEbus system can issue additional software tasks for the new inserted board.

1.4.7 Returning to Normal Operation

When the system controller detects that the replacement board is initialized and operating
correctly, normal operation returns. A VMEbus configured system is in normal
operation when the system's preloaded tasking software and its configured hardware are
performing with no interruption due to error, fault or failure.
SECTION 2

NORMAL OPERATION

2.1 INTRODUCTION

Normal operation is determined by the configured VMEbus system running its operational software and completing the tasks it was assigned in an acceptable timeframe. During normal operation, no violation to the VMEbus protocol or the configured system's established protocol can occur. Normal operation over a VMEbus backplane assumes that the data and calculations performed by the configured system are 100% fault free.

2.2 DEFINITION OF FAILURE, FAULT, AND ERROR

To understand normal operation, it is helpful to understand the definition of a failure, fault and error. The terminology used by D. P. Siewiorek of the University of Texas, a recognized expert in the field of fault tolerance and reliability, to define these terms is as follows:

2.2.1 "A failure is a physical change in hardware.

2.2.2 A fault is a deviation of logic function from the design value. Faults can be hard or transient.

2.2.3 An error is a manifestation of a fault by an incorrect value".

2.3 DEFINITION OF NORMAL OPERATION

Normal operation of a configured VMEbus system occurs when the system's preloaded tasking software and its configured hardware are performing with no interruption due to error, fault or failure.
SECTION 3

SYSTEM FAULT DETECTION

3.1 INTRODUCTION

A fault has occurred. As yet, the nature of the fault is not important to the systems’ performance. However, how the fault manifests itself and how it can be detected are vitally important to the reliability of the system. The detection is application specific and, as a result, the key to define the allowable time before detection; the latency. Erroneous operation may have taken place and may not be a problem. For example, end to end communication protocols will correct a few corrupted bytes; often in data communications corrections are not even considered; data is simply discarded. Life threatening errors imply a different set of ground rules. It is important that the fault is detected before an erroneous system operation occurs. This section looks at some high level (system) and low level (VMEbus specific) detection mechanisms. VMEbus in itself provides very little help with fault detection since parity checking and other similar techniques are not supported by the current standard. As a result of this limitation to the VMEbus specification, many of the detection mechanisms discussed tend to be of a supervisory nature.

3.2 DETECTION MECHANISMS

There are many fault detection mechanisms used in fault tolerant systems today. Some mechanisms will be in use continuously, and as a result, could impose a processing overhead on the system. Fault detection mechanisms must be non-invasive and not disrupt normal system operation. Often, this means that the tests are background tasks designed to perform health monitoring.

3.2.1 Voting or Checking Mechanisms

The voting mechanism involves multiple processing units performing the same task and voting to determine the outcome. This can either be a simple majority or a unanimous vote. Simplistically, if a vote is not unanimous it is an indication of an error. Implementations of these voting mechanisms require dedicated hardware or significant software for result analysis. Results of the checking mechanisms are often compared by means of a bus which, for reasons of system integrity, is separate from the VMEbus. It should also be noted that processor boards, particularly those executing from dynamic RAM, will perform tasks at slightly different rates. As a consequence, result analysis, or synchronization points, can be difficult to arrange efficiently. The majority of VMEbus board manufacturers do not add any fault detection voting hardware to their board designs.
3.2.2 Error Logs

Communications protocols (e.g., TCP/IP) and associated management services (e.g., SNMP) support error logging and re-transmission of information. Another system management function involves monitoring log-ins and detecting system trends and resource demands. For example, increased re-transmission rates may be an indication of a failing connection. This checking mechanism offers the system designer the ability to detect a failure and to reconfigure the system to accommodate the application's requirement. Many VMEbus applications are embedded systems that communicate to a LAN or a WAN, but few have a requirement for this level of system monitoring. The software and hardware requirements to implement this mechanism in a VMEbus are not beyond the capabilities of the VMEbus. Many system integrators of mission critical and fault tolerant applications have implemented this protocol over the VMEbus backplane.

3.2.3 Watch-dog Timer

A watch-dog timer could be a count down timer that is re-initialized by system software within a given period of time. If the counter reaches zero, it will signal the system monitoring software that a fault has occurred and initiate a fault identification process. Historically the watch-dog has been used to generate a RESET signal as a way of restarting a processor. This is particularly important for dealing with processor failures and software related errors (bugs and program corruptions).

3.2.4 On-line "BIT" or Health Monitoring

Fault detection could be accomplished with Built-In-Test (BIT) hardware and software routers. This BIT software usually consists of routines running on each board of the system and a BIT Master. This section discusses system level BIT in VMEbus based systems. BIT is required to support maintenance and repair. Without BIT, mean-time-to-repair (MTTR) times can be excessive and system availability can be significantly degraded. This is due to the complexity of systems made possible by the VMEbus.

With appropriate design rules, many errors can be detected on an individual VMEbus board. Typically, this involves the use of parity protected DRAM (or better) and careful use of health monitoring firmware. During system initialization, a set of diagnostics often will be run to identify faults. This fault analysis is quantified in terms of the coverage achieved, which will typically be in excess of 95% of the hardware. To achieve such high levels of coverage, it is normally necessary to exercise hardware by reconfiguring the hardware and moving data through the system. Some BIT testing could be considered destructive in nature because subsequent reconfiguration of system hardware may impose a system overhead and sometimes a loss of performance. The limitations of your system software and hardware BIT should map back to your system fault tolerance, maintainability requirements.

System level BIT can be made up of a collection of board self test sequences. A system level BIT Master controls the sequencing of self test operations and the collection and display of system wide BIT information. The BIT Master is application dependent, but...
certain aspects of BIT will be common among most system implementations. The BIT Master can be any embedded microprocessor in the system or an attached intelligent terminal or control processor. The function of the BIT Master is to control the scheduling of the VMEbus board self test sequences. The nature of this scheduling function will be application dependent. For example, in a SATCOM (Satellite Communication) system the BIT Master may schedule self test during time periods when the channel is off-line. The BIT Master must collect results of self test sequences and present them to the user through a suitable display device. These results can also be used to facilitate fault confinement and isolation as discussed later in this document. It is assumed that an integral function of the BIT master's logic involves a minimal fault diagnosis capability. This diagnostic capability should at least be able to identify the VMEbus board that is reporting a failure. An ability to diagnose ambiguous situations (i.e. board or backplane, errant board, etc.) is also desirable in the BIT Master's algorithms.

The BIT master should also monitor the VMEbus SYSFAIL* signal to detect a board that may generate this signal. The VMEbus specification (IEEE 1014-1987) allows this signal to be generated at any time by a board that detects an error or catastrophic failure. The diagnosis of the type of failure may be more difficult.

Processor and VMEbus throughput must be allocated for the BIT function. Depending on the complexity of the board, BIT overhead can be substantial and must be accounted for in the system design. However, the amount of data traffic required to transverse the VMEbus in support of BIT should be small, typically requiring only a few single cycle transfers per VMEbus board to initiate self test and obtain status.

The nature of the BIT Master's self test scheduling can be critical to fault confinement. If self test sequences are not done with a short enough period, erroneous results can be allowed to propagate through the system. On the other hand, if self tests are done too frequently, significant processor and bus bandwidth can be wasted. This is a systems level trade-off that must be carefully considered. This is a particular approach and is not necessarily the general case. In practice, "health monitoring" may be a continuous high level process designed to detect faults down to the replaceable unit level.

### 3.3 VMEBUS BACKPLANE SPECIFIC FAILURE MODES

A large fault tolerant problem occurs when a VMEbus board will not give up ownership or mastership of the VMEbus. The errant VMEbus board may drive its signal or data lines at the same time as another VMEbus board legally on the bus. This failure mode is one that the VMEbus has no method of detecting because the arbiter of the VMEbus is not required to signal system monitoring software when a board has been on the bus for a long period of time. The VMEbus hardware handshake protocol and passing of control between boards on the system is not monitored by any one process. This makes it extremely difficult for the system monitoring software to detect a failure like a board stuck on the VMEbus or one driving signals illegally.
3.3.1 System Slot One Controller Failure

The VMEbus specifies that the System Controller Modules physically reside in slot one of the System. A failure of one of the System Controller Modules (Arbiter, BTO, SYSCLK Driver or IACK Daisy-chain Driver) may cause a system-wide failure. However, VMEbus also defines these modules in such a way that they may reside as submodules of a board that occupies slot one, and not require any of the other services of that board in order to function. This makes the System Controller Modules extremely reliable and less likely to fail or operate less efficiently as a result of any other failure on the same board. For future systems that may require even greater reliability, VITA/VSO has formed a working group to evaluate the ability to provide for a more fault tolerant VMEbus system.
SECTION 4

FAULT IDENTIFICATION

4.1 INTRODUCTION

The ability of a VMEbus system to detect and identify a problem is critical to the overall performance and reliability of the system configuration. The ability to identify a system problem may be as important as the physical act of fixing or repairing the system problem. This section identifies and discusses the issues with the current VMEbus standard (IEEE 1014) in identifying system faults and suggest possible methods for fault identification.

4.2 DISCUSSION OF FAULT IDENTIFICATION

Based on the definitions for fault, failure and error, our definition for fault identification for the VMEbus is:

The ability of the configured VMEbus system to recognize or detect an error and determine if that error was generated by a fault or a failure.

A fault-tolerant system must detect errors across the backplane and the local buses of the system's boards. The VMEbus standard does not lend itself to identifying system errors easily. The original designers of the VMEbus realized that backplane transfers were extremely accurate and chose not to slow the VMEbus transfer rate with parity or error correction. This means that the current VMEbus standard, IEEE 1014-1987, does not have a specified means of identifying backplane transfer errors.

The system monitoring software of the VMEbus system does not have a standard that specifies a means to detect system errors. The system monitoring software, in this reference, is responsible for the health and well being of a configured VMEbus system. It may have additional requirements to monitor the status of the system or to run system tests to determine reliability. The system monitoring software could be a hardware processor board in the VMEbus system. Since there are no standards which describe procedures of identifying a fault with the VME system, one must be developed by the user or system design engineer. This generally means that the system monitoring software relies on health information that it obtains from individual VMEbus boards. When a board in the VMEbus system provides an error message, the system monitoring software now has the arduous task of identifying the error as a failure or a fault, and then locating the fault within the configured system. The transfer method of this error information from the boards to the monitoring software card is a task for the system designer to resolve. The bandwidth limitations of the VMEbus backplane and its vulnerability to failure add to the problem of reporting the fault to the system monitoring software. To help the reader understand the magnitude of the problem, the next few
paragraphs will deal with the types of faults, failures, and errors the VMEbus system may encounter.

NOTE: This system may reside on any board or all boards in a system. Which board monitors the health of the system may be defined by the systems software, and may be a static assignment or determined by a dynamic scheme.

4.3 TYPES OF VMEBUS FAULTS, FAILURES, AND ERRORS

There are two types of failures that could cause an error in a VMEbus system. The first is a physical failure, where the hardware is destroyed or physically changed. A physical or hardware failure may be permanent or transient. The second type is a software related failure, “bug within the application code or the operating system. There are three primary places that these hardware and software failures can occur within the VMEbus system.

The first is the application firmware and operational software.

The second area of failure is the VMEbus board and its input / output interfaces that make up the VMEbus system configuration.

The third area is the VMEbus backplane, which includes the power supply and cable hook up. The VMEbus backplane, has four separate internal bus structures where a failure could occur: the Utility bus, Arbitration, Interrupt bus and Data Transfer Bus (DTB).

Software errors could fill an entire chapter in this document. The key to identifying software errors is the ability to determine that the errors were actually caused by the software. There is an abundance of off-the-shelf operating systems and software packages available that have built in tests and health monitoring to detect and identify software errors. Some software packages assume all errors are software problems and perform extensive testing to determine the cause and location of the error. However, the majority of available software assumes that errors are generated by the hardware and initiates tests to determine the location of these failures. These software initialized tests are only as good as the reporting method of the VMEbus system. To be able to identify software failures, the VMEbus system must have a method for testing and identifying hardware failures. Until this is accomplished, the system monitoring software will have no way of knowing if a fault was generated by a bug in the software or by a hardware failure.

The VMEbus board adds another dimension to fault identification. The board may be connected to an outside source receiving data, and if there is no means built into the VMEbus card for identification of errors for the incoming data, then there will be no way of determining if the data received is correct. The VMEbus board could also have a local bus with a processor, memory, and I/O which could cause a local bus or local bus error. The VMEbus interface logic is another area that could cause a system error. This
VMEbus bus interface could drive a signal when it is not supposed to be driven, or it could pick up noise and interpret it as a valid signal. A failure on the VMEbus interface logic board could cause a data line or address line to go bad and corrupt the entire VMEbus backplane. Again, the system monitoring software for the configured system has no means of identifying or detecting that error without a feedback circuit. However, if the VMEbus board was designed with these considerations in mind, then it could report an error, fault or failure to the system monitoring software via a SYSFAIL* signal. This would require additional application specific software, which could, depending on the implementation, cause additional processing overhead. The ideal solution is to have a feedback circuit built into the configured VMEbus system. This could be achieved by using the VMEbus backplane serial bus pins on the P1 connector. The system designer could select a serial bus architecture from those available for the P1 row B pin 21 and pin 22 connector, or use a standard RS232, RS422 serial bus feature of the VMEbus card.

The Utility bus provides the VMEbus power and critical error signaling control lines. This bus has no sensor signal lines to identify a failure with any of the voltages supplied by the power supply. However, a power supply that monitors its own voltage input and its DC voltage output could send an error message to the system monitor through the SYSFAIL* or ACFAIL* lines on the Utility bus of the VMEbus backplane. The system could then run some routines to identify a failure in the VMEbus system.

The Arbitration bus is another of the four independent buses that make up the VMEbus architecture. The Arbitration bus requires circuitry that is normally located on the VMEbus system controller, or slot one board. This presents a dependency problem for the VMEbus architecture. If the slot one or system controller has a hardware fault, the Arbitration circuit on that board could cause a larger system failure. There is nothing that a VMEbus system monitor could do to detect an Arbitration error, since there is no direct feedback from the Arbitration bus to the system monitoring software. The system's application system software may not even detect that there is a problem. Another difficulty in identifying arbitration bus faults is the four possible arbitration levels of the bus. There are no signal lines routed back to the system monitoring software identifying the arbitration level of the VMEbus backplane or what two boards are communicating on the bus. The Arbitration bus level is handled through a hardware daisy chain configuration built into the VMEbus backplane architecture.

The Interrupt bus is very much like the Arbitration bus because it also incorporates a hardware daisy chain configuration for its means of identifying and transferring control to an interrupter. The Interrupt bus depends on the system controller to contain the necessary IACK daisy chain to begin the daisy chain propagation. Again the system monitoring software has no direct feedback connection from the Interrupt bus. Therefore, it has no way of detecting what board has interrupted the Interrupt Handler or what level interruption has occurred.
NOTE:

Even though the IRQ lines cannot be actively shared among Interrupt Handlers, any board can monitor the IRQ lines and determine when an IACK cycle occurs and what level is being acknowledged. The only thing this IRQ monitor can't do is determine which Interrer on that level acknowledged the interrupt unless it is also monitors the Data Transfer Bus.

The Data Transfer Bus, which passes data between two boards over the VMEbus backplane has a unique hand shake to ensure that the transfer completed. However, there is no protocol or procedure to inform the system monitoring software that the data transferred was correct. Due to the absence of a parity line or error correction circuitry, the system monitoring software has no idea what type of transfer took place and whether the data was correctly transferred. Many mission critical VMEbus application system integrators implement TCP/IP like protocol over the VMEbus backplane to assure data integrity.

4.4 POSSIBLE METHODS FOR FAULT IDENTIFICATION ON THE VMEBUS

The VMEbus specifications calls out two serial bus pins on the P1 connection of the VMEbus backplane (B21 and B22). This serial bus was never explicitly defined and very few, if any, VMEbus manufacturing companies use it in their implementation of a VMEbus board design. However, if we examine the need for fault identification as well as isolation, then the serial bus could be a feedback loop that will improve the VMEbus ability to identify and isolate failures. A serial bus could provide communication between VMEbus boards and the system monitoring software.

One application where a serial bus could be used immediately is in system health monitoring. A health monitoring byte or word would be sent by every board within a given time period to the system monitoring software. This health status will identify the availability of individual VMEbus boards. If the system monitoring software does not receive a health monitoring status, then it can assume the VMEbus board is dead or a failure has occurred. The serial bus now gives the system designer the ability to take immediate action to correct the problem, even if the backplane is locked up. The serial bus could be used to isolate VMEbus boards, to initiate additional test commands, or to provide additional status information.

With a serial bus, the VMEbus now has an alternate path (feedback loop) to communicate errors, get status of VMEbus boards, and to communicate commands. The serial bus could also be used to pass data to and from other boards in a VMEbus system. The software to perform these tests would have to be added to the VMEbus system. The way the software is implemented using the serial bus may even reduce the throughput requirements of the VMEbus in the configured system application.
NOTE:
The use of a serial bus may not be backwards compatible with boards built with no serial bus connection. Serial bus notification is not mandatory with live insertion using isolation and could violate the backward compatibility goal of this document.
SECTION 5

FAULT ISOLATION

5.1 INTRODUCTION

It is important to isolate failed components from the rest of the system. Fault isolation is the reconfiguration of system logic to remove or limit the possibility of failures from propagating faults that result in errors in the system application. Fault confinement is defined as the ability to limit the propagation of faults without explicitly initiating reconfiguration actions. Fault confinement is typically treated at the systems level as part of a system level fault tolerance approach. Fault isolation techniques can be used as part of the fault confinement strategy and/or as a precursor to fault replacement.

Fault isolation in a multi-drop bus system (such as the VMEbus environment) can be achieved at three levels: logical, electrical, and physical. Logical isolation can be achieved using software manipulated flags to make a board appear not to be in the system. Electrical isolation can be implemented with a hardware generated or software derived signal used to control interface circuits that have the ability to electrically isolate a board from the backplane. Physical isolation would require a suitable fault detection approach coupled with manual or electro-mechanical repositioning of a board or module. The following sections will explore the implementation of logical and electrical isolation. Physical isolation will not be discussed in this document because of the complexity and limited practical utility. In addition, we will address some of the key issues associated with fault isolation in a VMEbus environment.

5.2 FAULT CONFINEMENT

Once a fault has been detected, some means must be used to confine the erroneous data to a small subset of the applications. Fault confinement can be divided into two basic approaches: masking confinement and temporal confinement. Masking confinement involves redundant resources and comparison logic (in hardware or software). For example, two devices can be configured to generate the same result. These results can be compared for equality. If the two are not equal, the circuit that enables an output from either of the two devices can be turned off. Thus, the comparison logic can be used to control confinement of erroneous results. Comparison logic either can be designed into the VMEbus board (hardware approach) or implemented using a significant commitment of logical resources (processor time and software). Temporal confinement can be implemented with BIT sequences coupled with application specific software check points and retry logic at the system level. For example, a processor can be instructed to perform a self test operation after each output data block (e.g., 1024 16-bit words). If the self test passes, the data block is saved as a "check point" that a down stream process can request if required to "roll back" the application. The data block is assumed valid since the subsequent self test indicated no failures in the processing element. If the self test fails,
the system can be informed that no new data block is available and a roll back of the application is required. The system control function can then reconfigure the application (i.e., assign this process to a known good processor) and roll back the application to a known good check point. Check points are saved data sets from previous time periods that can be used to roll back an application and restart (warm start) it without incurring the overhead (time) to completely restart (cold start, or hard reset) the application. The temporal approach can only be used when sufficient temporal margin (i.e., application timeline reserve) is available. In some applications, some amount of data simply can be ignored when BIT determines that erroneous data may have been generated.

5.3 ISOLATION TECHNIQUES

Reconfiguration of system logic can be used as a means of fault isolation. The goal of reconfiguration is to remove or limit the possibility of failures from propagating errors to the system application. Reconfiguration is typically achieved with software. However, hardware assisted reconfiguration is possible.

5.3.1 Logical Isolation

As discussed earlier, a VMEbus board can be logically isolated from the system. For example, a board that responds to memory accesses could have its memory space set to addresses that are not used in the application. Thus, unless an erroneous address is generated, the address space that the board thinks it now occupies will never be accessed and the board is effectively isolated from the system. This approach is only possible when the failed board still has functional Data Transfer Bus or serial bus logic and the board depends on other bus masters to initiate backplane activity. This technique only works if the failed board's address decoding logic is software programmable.

Many VMEbus boards do not provide the ability to software program their location in the VMEbus address spaces. Instead they hard code their slave locations in programmable logic, or provide limited selection via jumpers. The software on these boards should periodically check an on-board memory location to determine if they are still a part of the system. The system software should also notify all other boards in the system not to include the failed board. This effectively isolates the failed board from the rest of the system.

5.3.2 Electrical Isolation

Electrical isolation can be accomplished using special circuitry in the board's VMEbus interface logic or by installing a buffer circuit between the VMEbus board and the backplane. One example of electrical isolation is DY4 Systems Inc.'s use of the IRQ* pin on the VMEbus P1 connector to implement their "Bi-mode" mode of operation. In this approach, the IRQ* signal is routed to the VMEbus transceivers and disables activation of any VMEbus signals. An example of the buffer board approach is the Radstone and MUPAC’s live insertion module which can isolate (using a manual switch) a board from the VMEbus backplane. In both the buffer board approach and DY4 approach, the same
BIT derived fault indications used to initiate logical isolation can be used to instruct the board to be electrically isolated from the system. In one case you instruct the on-board logic to "turn off" and in the other you instruct the buffer board to "turn off".

In both the logical and electrical isolation cases, the VMEbus is still affected by the electrical load (inductance and capacitance) presented by the VMEbus board's (or the buffer board's) interface circuits. If a buffer board is used to isolate the VMEbus board, some signal delay will be experienced from the VMEbus board's bus interface logic through the buffer board's interface logic. This logic can be implemented with fast FETs and should be minimal. However, some degradation in VMEbus throughput would be experienced.
SECTION 6

VMEBUS BOARD REPLACEMENT

6.1 INTRODUCTION

The VMEbus live insertion method described in this document allows standard, unmodified VMEbus boards to be used in a live insertion environment. Modifications required to achieve live insertion should be performed at the system level. This includes modifying system hardware such as VMEbus connectors or backplanes.

This section describes general considerations when live inserting and extracting VMEbus boards; several implementation options for VMEbus board replacement detailed electrical, protocol, mechanical and software consideration; and a design example.

6.2 BOARD REPLACEMENT CONSIDERATIONS

6.2.1 Board Isolation

In order to avoid damage to boards or the configured system, it is recommended that the VMEbus boards are powered down or isolated from the VMEbus before they are removed from the system. In addition, the resulting empty slot should remain isolated and unpowered until the replacement board has been inserted. The VMEbus specification defines the electrical characteristics of devices which may be used to interface to the bus when voltage rails are within tolerance. It does not, however, define the required behavior when power is not applied to a board. Thus, the behavior of an unpowered VMEbus interface will vary from board to board because of the variety of interface hardware.

Some new boards may employ FCTT devices which behave in a controlled manner and may have a negligible effect on the bus when they are unpowered. When some backplane devices are unpowered but still attached to an active bus, they may disturb that bus. For example, an unpowered CMOS gate or buffer will present a forward biased diode to an active system bus signal, locking it at approximately 0.7V. Because this specification requires unmodified VMEbus boards to be accommodated, unpowered boards on the backplane are not recommended; some form of isolation is required.

After a board is powered down, it must be removed. This is a process which could introduce contact bounce "noise" for several tens of milliseconds and could interfere with VMEbus traffic. This again implies that buffering is required to isolate each VMEbus interface line so that an unpowered VMEbus board is unable to disturb the continuing bus operation. This technique also provides a limited means for fault isolation in the case of a failed board or VMEbus interface.
Figure 6-1. VMEbus Signal Groups
From figure 6-1 it can be seen that three groups of VMEbus signals exist: bi-directional signals, single direction signals, and power. Isolation logic must be designed to cope with all signal groups in a uniform manner. This implies the use of a series Field Effect Transistor (FET) or similar technology rather than logic gates. If isolation takes place while a board is participating in a VMEbus cycle either as a master or as a slave, an illegal VMEbus operation could be generated. Ideally, isolation should take place during a period of bus inactivity. This is guaranteed by the isolation logic gaining control of the VMEbus and not releasing it until board isolation has occurred.

Slot isolation should take place in response to an incoming signal from BIT master or external control which either has been generated locally or passed over a suitable communication link. The exact nature of the mechanism concerned is application dependent; several options are discussed in 6.3. However, reliability is essential in the slot control logic and to this end, interlocks and error checking must form an integral part of the system design. The fact that many board features could be duplicated in the live insertion logic means that the detection of faults in the isolation circuits may be a complex task which imposes a significant software overhead and/or operator involvement.

6.2.2 Service Suspension Versus Interruption

There are two levels of live insertion that have been identified by the live insertion committee. Level one does not allow the backplane bus transactions to be interrupted. Level two allows the backplane bus to be interrupted. It is critical that when the backplane bus is interrupted it is done in the quickest manner possible. This document only covers level two. Level one will be explored in the forthcoming System-level Live Insertion document. For level two we define two possible methods: Service suspension and service interruption.

Once the board level isolation process is initiated, the objective is to accomplish the isolation, removal and reinsertion with the minimum impact to the host system. One of two timing scenarios could be acceptable; system operation either will be suspended or interrupted.

With service suspension, a human operator suspends system operation and triggers the board isolation process. After a board update or replacement, the operator restarts the system. Service suspension is applicable to the hot swap of a processor or key system component without which normal system operation cannot proceed. The live insertion ensures that memory contents or dynamic configuration data are not lost. Service suspension is easier to implement than service interruption.

The objective of service interruption is to interrupt system operation for the minimum possible time. Thus, normal VMEbus operation is only interrupted for the slot isolation process and the slot de-isolation, or return, process. Normal VMEbus operation resumes during the period between these events. Service interruption is applicable to the vast majority of live insertion applications. When implementing live insertion in the service
6.2.3 Insertion / Extraction of VMEbus boards

The physical act of removing and inserting a VMEbus board into or out of a backplane could potentially cause a series of problems. The first is the capacitance, inductance and load of the VMEbus board being removed or added to the system configuration. This could cause power supplies to crowbar their supply voltages and change the shape of the signals being transmitted over the backplane. An easily overlooked problem is Electro-Static Discharge (ESD). When being inserted into the backplane the VMEbus board could generate a charge that is transmitted to the backplane. This charge could result in spikes on the power and signal lines. Proper ESD precautions can prevent this problem.

6.2.4 Slot 1 Considerations

Given that the system controller functions normally reside in slot 1, live insertion of this slot would require special attention to arbitration, daisy chain switching, etc. Historically, VMEbus has not had a standard CSR definition. Nor has the VMEbus had the ability to automatically re-allocate the system controller function. A simple way to gain this ability is moving the arbiter off the slot 1 board and onto the backplane or onto a central services module. Arbitration may now be organized as a star arrangement rather than a daisy chain. This could improve performance and functionality. But the star arbitration arrangement need not affect the working of existing boards which would still receive their bus grant signals in the same way. This solution could require a special slot one or system controller board or Central Services Module (CSM). The CSM could be attached or mounted to the backplane where it could be easily replaced and require no additional VMEbus slot. A concern with this solution is the active components required for the isolation circuit on the modified backplane. Those components could reduce the Mean Time Between Failures (MTBF) of the backplane, decreasing the reliability and maintainability of the configured system.

6.2.5 VMEbus Conformance

Since isolation circuits must be included for each signal trace, the signal trace length to each slot will increase. In extreme cases, some solutions may contravene Rule 6.24 (IEEE 1014-1987), the 2” rule. Extreme care must be taken in the system design if this rule is violated. It may be necessary to restrict the number of slots or take other appropriate measures in order to compensate for irregularities in signal trace lengths. Propagation delays introduced by isolation circuits or increased signal trace lengths must remain within the timing budgets of the VMEbus specification. To avoiding these propagation delays the user could reduce the maximum slot count. This is not a recommended solution, but one that should be considered for some VMEbus applications.
6.3 IMPLEMENTATION OPTIONS

This specification defines protocols and techniques without restricting the physical implementation. As a result, several implementations are possible and are described in this section. A specific example with additional detail is described in section 6.8. Given today’s technology, three distinct isolation implementations are possible:

(i) Mount live insertion logic on a short "stub" board (buffer or paddle board) positioned between a standard backplane and a standard VMEbus board.

(ii) Place the live insertion logic on an active backplane.

(iii) Mount the live insertion logic on the connector.

6.3.1 The Buffer Board Implementation

The buffer board is illustrated in figure 6-2. The overall depth of 220mm is chosen because, not only is this an achievable format, it is the next increment in depth from 160mm according to IEC 297.3 and IEEE 1101.1. Each live insertion slot in the system will require a buffer board. This solution has a number of advantages which include:

(i) Both the backplane and VMEbus boards are standard items (unmodified) and comply with IEEE 1014-1987 and ANSI/VITA-1 April 1995 (VME64)

(ii) If one of the buffer boards fails it may not cause a complete system failure.

(iii) The Mean Time To Repair for a failed slot (VMEbus board or buffer board) will not change dramatically. However, the Mean Time to Failure (MTBF) calculations may decrease due to the fact we are adding components to the overall systems design.

One disadvantage of this solution is that the extension of the VMEbus board signals may be in violation of the Rule 6.24 of the IEEE 1014-1987 standard and ANSI/VITA-1 (VME64). Another disadvantage is that there is no standard mechanical definition or standard for a daughter card solution identified in IEEE 1101.1 or IEEE 1101.10 that meets this application requirement. Connector wipe compliance should be taken up at the system level design to be sure this type of solution will work in your application.
This method can be extremely cost effective when hot swap or live insertion is only required on specific slots. However, this implementation does have the disadvantage that the increased card depth (220mm) dictates the use of modified racking and possibly a modified enclosure. This solution also extends the pin length of the VMEbus board P2 connections, which could be detrimental to the board's P2 connector I/O. The buffer board implementation must contravene this rule, however, good system engineering and design could be applied to allow for this violation. A possible solution is to populate
every other slot in a VMEbus backplane, or build a backplane with a 1.6 inch slot spacing. The 1.6 inch slot spacing guarantees that the loaded backplane impedance and slot to slot signal skews will not exceed those assumed by the VMEbus specification and limits the maximum number of slots to 11. Spacers can fill in the front panel spaces left when using standard boards at this larger pitch. Alternatively, live insertion may not be supported across the entire backplane. This solution should not simply be overlooked because of the Rule 6.24 violation; certain users may find this approach advantageous for certain applications.

6.3.2 The Backplane Implementation

Figure 6-3 shows a simplified component placement for a five slot backplane with up to two buffers per signal. This creates the isolation circuit needed for the VMEbus board in a Live Insertion backplane.

This solution requires a modified backplane but has a number of advantages which include:

(i) Standard VMEbus boards may be used without modification.

(ii) The system enclosure may remain unchanged.

(iii) The solution need not contravene the 2 inch rule.

The main disadvantage is that the active components on the backplane could reduce the Mean Time Between Failure (MTBF) figure. The Mean Time To Repair (MTTR) of the total system could increase as well unless the location of the backplane is designed for easy access.

In order to minimize the MTTR impact, control logic could be mounted on a Central Services Module (CSM) which is plugged into either the front or the rear of the backplane as a removable unit. The CSM minimizes the number of active components on the backplane and may support the user interface and/or slot 1 functionality in addition to live insertion control. This solution then reduces the MTTR because of the modular design and access to the CSM. It improves the possibility of the slot one controller being live insertable. Slot 1 functionality is provided so that hot swap of the slot 1 board may be achieved using current arbiters which do not support active reconfiguration.

Another possible solution is to create a new VMEbus backplane that has a CSM and live insertion slots that are user configurable. Instead of placing a paddle board into the backplane and then the standard VMEbus board into the paddle board, we would now place the isolation circuit on the back of the backplane physically on a separate new module. This new module would connect to both P1 and P2 from behind the backplane. This solution would provide the user more flexibility in his system design and he would be less concerned about the two inch rule. This solution does not place active components on the backplane.
Figure 6-3. The Backplane Implementation
6.3.3 The Connector Implementation

The active connector implementation is similar to the backplane implementation. Instead of moving the Live Insertion circuit to the backplane, however, the connector is used to house the new circuitry. Some manufacturers have developed a new connector that can house surface mount components in the shell of the connector, making it ideal for the new Live Insertion circuitry. The current connector implementation for the new circuitry is in the board's DIN connector, not the backplane mating DIN connector. This solution would require a redesign of all boards that were to be considered for the application using Live Insertion that did not have this solution already embedded in its DIN connector.

Another solution that should be considered is the Enhanced Transceiver Logic (ETL). This new circuitry has been specifically designed with Live Insertion in mind. Replacing the current VMEbus board's backplane drivers and receivers with ETL should also provide a Live Insertion capability. The viability of this solution will depend upon the level of redesign needed.

6.4 ELECTRICAL CONSIDERATIONS

6.4.1 Isolation Circuit Characteristics

The recommended Isolation circuit should meet the following characteristics:

- Propagation delay (signal lines) \(1\) ns max
- Total slot capacitance \(18\) pF
- Daisy chain link time \(20\) ns max

6.4.2 Power ramp timing

System integrity demands that power to boards be applied in a controlled fashion. It is recommended that the +5v, +12v and -12v rails are ramped up in a linear fashion with a rise time on the order of 4 milliseconds. The rise time is chosen to minimize two effects:

(i) If the rise time is too short, below 2 ms for example, then voltage rail transients can cause problems.

(ii) If the rise time is too long, above 10 ms for example, then the logic families becoming "operational" at different rates or voltages is an important factor and can cause initialization failures on some boards.

The figure of 4 ms represents a recommended, mid ground figure and takes into account the system dependent variations found with the upper and lower times. The application
of power to the system may be treated as a special case of board insertion. In this case, all boards in the system have been inserted at the same time.

### 6.4.3 Power handling capacity

Each slot must employ its own power control circuitry with the recommended minimum values:

<table>
<thead>
<tr>
<th>DIN Connector rating:</th>
<th>Recommended Slot Power:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 AMP @ 20 degrees Celsius</td>
<td>+ 5v at 8 Amp</td>
</tr>
<tr>
<td>1 AMP @ 70 degrees Celsius</td>
<td>+ 12v at 1.5 Amp</td>
</tr>
<tr>
<td>0.5 AMP @ 100 degrees Celsius</td>
<td>– 12v at 1.5 Amp</td>
</tr>
</tbody>
</table>

The power control circuits should only introduce a maximum series "resistance" of 0.025 ohms on the + 5v trace and 0.4 ohms on the +12v and -12v traces.

### 6.4.4 Power consumption of Live Insertion circuits

The Live Insertion circuitry will require power to operate. This power will turn into heat and can generate design problems for system integrators. The lowest power consuming Live Insertion circuitry will, of course, require less power and produce less heat. The ideal Live Insertion circuitry should draw a minimum of power and should use component technology that will provide "clean" signals.

**NOTE:**

The issue of switching the load onto the bus is not addressed. If the voltage level of the bus line differs from the level of the signal on the isolation side of the FET (or similar technology), when the FET (or similar technology) is turned on a transient current is induced. Under some conditions, this transient can produce voltage spikes greater than a volt. If the protocol insures no other activity on the bus while the FET (or similar technology) is switched, this is probably OK, but this problem should still be considered further.

FET switches also present an inline impedance which in some operating environments, can make signals more susceptible to cross talk.

### 6.5 PROTOCOL CONSIDERATIONS

The VMEbus is an asynchronous bus and therefore, no one can predict exactly what a failed board will do to the system protocol or operation. We must consider what effect the insertion and extraction of a VMEbus board will have on the protocols during reduced system functionality.
We assume that the Live Insertion circuitry requests the VMEbus and receives an acknowledgment prior to board insertion or extraction. Because of this acknowledgment we do not have to concern ourselves with the data transfer protocols. However, if the insertion or extraction of a board creates an interrupt or a bus error, then the system monitoring software must be intelligent enough to ignore the spurious transmission. These spurious signals could be prevented by proper Electro-Static Discharge (ESD) procedures and by cycling the control signal power from the address and data signal power. Power cycling or separating will allow the board to stabilize before any control signals have power applied, reducing the possibility of spurious signal generation.

Another solution that could be implemented to reduce protocol violations is to hot swap the slot 1 system controller. This will require a star type arbitration and interrupt scheme dictating a modified backplane design. Individual boards will keep the standard VMEbus specification defined arbitration and interrupt connections and the backplane will route the signals from each board to a Central Services Module (CSM). This solution would require the majority of user definable signal lines on the P2 row A and C DIN connector of the card or a Central Services Module (CSM). The CSM card could be the slot one controller and would require a specially built VMEbus board to implement. There is work going on now that is addressing additional fault management and dynamic reconfiguration for the VMEbus. Please contact the VITA/VSO for further information on the Fault Management Dynamic reconfiguration.

### 6.6 MECHANICAL CONSIDERATIONS

#### 6.6.1 Notification

Mechanisms are required in the live insertion process to recognize when a board is about to be removed. Hardware in the system or on the backplane could detect that a board is being removed and initiate the necessary board isolation and power down before the board leaves the backplane. However, current mechanical tolerances make this solution extremely difficult to realize. In addition, only a very short period of time is available for VMEbus acquisition and power down. This technique could be used if, for example, the switch was located in a special racking extrusion and operated by the board locking screws. Another option is to use strain gauges in order to detect the first signs of a board removal.

A simpler option is operator driven hardware (i.e., a push switch). This would be relatively easy to integrate and use, with time-outs and safety interlocks easily incorporated. Figure 2 depicts this method which requires the operator to initiate the insertion or extraction procedure. The chosen method will be dictated by the application and by the sophistication of the environment. The operator must be instructed on which board is bad and how to properly remove and insert the boards.

Another alternative is to add a simple RS232 connection to a CSM that may initiate and control the hot-swap process. This may be the most cost effective solution but may not meet all of the users application requirements.
6.6.2 Daisy Chain Connections

In addition to detecting that a board is ready for removal, mechanisms are required to link the daisy chain signals of a removed board so the bus can continue its operations without interruption. There are currently backplanes and connectors that automatically link VMEbus daisy chain signals mechanically when a board is removed from the backplane. A potential problem is that switching with these mechanical backplane connectors takes time on the order of milliseconds and could possibly generate electrical noise on powered signals. Signal isolation is not recommended to be used with mechanical switching. However, connectors and backplanes with active components may have a role in the three potential isolation implementations:

(i) The buffer board implementation

An important feature of this solution could be the ability to live insert the buffer board itself if it has failed. Since the buffer board is essentially a set of isolation transistors, live insertion may be possible provided that the VMEbus is quiescent and daisy chain signals are automatically linked through by the backplane connectors. Note that the removal of the buffer board suspends service.

(ii) The backplane solution

If VMEbus operation can be suspended until the card in question has been removed, then automatic connectors could be used to link the daisy chain signals.

(iii) The connector implementation

The connector implementation would use the same logic circuitry the backplane or paddle board solution uses to isolate the VMEbus board from the backplane. Therefore, the connector implementation could also use connectors to link the daisy chain signals if VMEbus operation is suspended when the card is removed or inserted.

NOTE: There are other alternatives to the daisy chain problem. The VMEbus backplane could be modified to house a CSM which would determine system health and identify and isolate failures. This CSM slot would also have the "slot one" features of arbitration and interrupt handler. Instead of using the current VMEbus daisy chain solution, star arbitration and interrupt handling could be employed.

6.6.3 Connector Contact Bounce on the VMEbus

Contact bounce occurs when the DIN connectors of the backplane and board first meet and continues until they become fully seated. This bounce can cause problems on both the power rails and the data transfer bus. Testing shows that a window of 30 ms to 50 ms of contact bounce can exist. Additionally turning power on, then off again on a single slot could generate signals over the backplane. Another unseen problem is the possible
damage by contact bounce which could occur to the interface and buffer circuits when power is applied and then removed several times during the insertion or extraction of the board. The isolation circuit solution solves these problems by having the power cycled off to the slot when the board is inserted or extracted from the DIN connectors. Once the connections are made, the power is reapplied. A significant time delay prior to continuing operation of the system is added if the bus is held during this time period. It is recommended that a minimum amount of time, 250 ns, be used to gain control of the bus and isolate the VMEbus board. Once the board is isolated, the bus should be released.

6.7 SOFTWARE CONSIDERATIONS

6.7.1 Notification

Software can be used to notify the system that a live insertion is about to take place. An operating system task, a console port or a modem connection can be used to modify a status register when the software detects a faulty board. This is a high level function that has been successfully used in the live insertion of similar bus standards (i.e., Multibus II, IEEE 1296-1987).

One difficulty is that some logic must reside on the VMEbus backplane which does have an impact on MTBF. In addition, this logic is difficult to define because, at present, no standard Control and Status Register (CSR) definition exists for VMEbus. This is, however, a viable long term solution when a VMEbus CSR definition has been defined and is in widespread use. Another difficulty with this technique is the above average degree of operator skill required. This skill requirement may make it unsuitable for some applications.

6.7.2 Interaction Between Live Insertion Logic and Controlling Software

The live insertion process will create an empty slot where a board previously existed. This vacated slot will be referred to as a NULL SLOT. If a DMA device attempts to access a NULL SLOT it will receive a BUS ERROR. Most DMA devices will stop transferring data when they receive an abnormal termination, such as a BUS ERROR, and leave unmodified the address counter where the error occurred. Software can read the address to determine that the previous functions associated with the board at that address are no longer available.

The simplest solution is for the live insertion logic to do nothing. When the controlling software attempts to access the board that was in the NULL SLOT, the NULL SLOT does not respond so the VMEbus system controller will time out and drive BUS ERROR* true.

A second approach is for the live insertion logic to generate an interrupt to the controlling software when the board is removed. This requires installation of an appropriate interrupt handler together with the standard BUS ERROR handler. The advantage of this solution is that action can be taken in advance of an access to a NULL SLOT, either by operator
feedback or simply by the avoidance of lengthy fault isolation procedures. The major disadvantages of this method are requirements for an extra software driver and possibly extra hardware which can have a significant impact upon reliability.

6.8 DESIGN EXAMPLE

A system using isolation logic mounted on a buffer board, activated by a push switch from the front panel, is illustrated in Figure 6-4 and an operational flow chart is shown in Figure 6-5.
Figure 6-4. An Example System Configuration
Figure 6-5. An Operational Flow Chart
The flow chart begins with the idle loop for the active state (board in slot). If a board needs to be removed from a slot, the activation push switch for the slot in question is pressed. The live insertion logic gains control of the VMEbus to guarantee that the slot disable phase does not cause the slot to generate short or illegal VMEbus cycles. Isolation takes place and the power down sequence is initiated. The push switch contains an LED indicator and during the power down sequence flashes to give operator feedback. After approximately 10 seconds the sequence is complete and the LED is permanently illuminated to indicate an unpowered or empty slot. The board can now be removed and the LED remains illuminated (idle loop empty slot).

If a new board is inserted into a slot or a powered down board has not been removed within a predetermined period (removal timer), in this case 2 minutes, the slot power up sequence begins. This is automatic and is signaled to the operator by the flashing LED. When the slot has been powered up and RESET, the LED indicator in the switch is turned off to indicate a functional slot.

6.9 ISOLATION TIMING

Figure 6-6 shows a possible isolation circuit interruption to the VMEbus transactions. In this implementation the design is not interrupting a data transfer cycle. The design is taking advantage of an arbitration cycle to guarantee that data is not being corrupted or lost. The important fact is that the service interruption on the VMEbus is in the 300 to 350 nanosecond time range.
Figure 6-6. A VMEbus Timing Diagram of the Isolation Interruption
6.10 WARNINGS

- This specification does not require signal conditioning on the P2 user defined pins. If these are used for I/O purposes and can source power then they could damage an unpowered VMEbus board, a condition which exists prior to board removal and immediately after board insertion.

- This specification does not define a mechanism for the live insertion or removal of P2 busses, such as VSB, (IEEE 1096-1988), RACEway (ANSI/VITA-3), and ANSI/X3 SCSI-1. Extensions to this specification may be required for operation of a P2 bus.

- Electro Static Discharge (ESD) will have an effect on the power rails and possibly the signal lines of the VMEbus Backplane. Extreme caution and correct ESD procedures should be followed when inserting and ejecting a VMEbus board in a VMEbus chassis.

- Shock hazard: Care should be taken not to contact the printed circuit traces or component leads as the board is inserted or removed, and that the specification should guarantee that when power is turned off to the slot, that no shock hazard shall exist.
SECTION 7

SYSTEM BACK ON-LINE

7.1 INTRODUCTION

This section describes the four phases that a VMEbus board goes through when it is brought back on line in a VMEbus backplane configuration. These phases are: insertion into the VMEbus backplane, powering on and running built in test, establishing on-line status with the system monitoring software board, and resumption of software tasking. This section includes user concerns when inserting a VMEbus board into a backplane and possible actions to alleviate those concerns.

7.2 INSERTION INTO THE VMEBUS BACKPLANE

The VMEbus board must be inserted into the VMEbus backplane before it can come back on-line and assume normal operation. The VMEbus board must power up its own on-board features, without violating any of the VMEbus system protocols. It must not interrupt or cause an error to the current transaction over the VMEbus backplane. It must overcome the contact bounce that could occur while the DIN connector is being inserted into the VMEbus backplane. Finally, it can not misinterpret the backplane transactions that are occurring over the VMEbus while the board is being inserted into the VMEbus backplane. We have identified two ways that could allow a VMEbus board to be inserted into a VMEbus backplane without violating any of the above concerns. The first is by controlling the VMEbus backplane during insertion; thereby preventing any transactions from being affected. This solution will stop transactions from occurring while the VMEbus board is being inserted into the backplane. The second way is to control the power, control, and signal lines of the VMEbus board, so it will not cause any violation to the current transmission or system protocol.

7.2.1 VMEbus Control During Insertion

Controlling the VMEbus during an insertion and ejection of a board is the easiest way of ensuring that no transaction will be adversely effected. The buffer board and active backplane isolation circuit could provide this feature through standard VMEbus protocols. New circuitry could request the VMEbus using standard control signals on the backplane (Bus Request and Bus Grant) and when given control of the bus indicate to the user that the system is available to insert or eject the board.

For solutions that implement the connector, backplane or buffer board isolation circuit, additional software may be required. For example, a simple keyboard terminal could be used to tell the system monarch or system monitoring software that the VMEbus should be at a quiescent state for insertion. When the CSM board has gained control of the VMEbus it would then let the user know the bus is available for insertion through the
same keyboard terminal interface. However, the software needed to bring the system to that state and its effects on the application's performance should be carefully considered.

### 7.2.2 VMEbus Power, Control, and Signal Cycle

Control of power lines, control signals and the address / data lines of the board being inserted into the backplane ensures the process does not inadvertently interrupt the current VMEbus transactions. This method could be added to the isolation circuit solutions. The isolation circuitry (on backplane, buffer board or connector) has continuous power and, therefore, can cycle the power, control and signal lines to the inserted VMEbus board. This cycling of power, control signals, address and data lines ensuring the VMEbus board does not misinterpret any of the VMEbus transactions during power up. This solution requires little hardware and practically no software intervention. The hardware circuitry required would be a simple VMEbus requester circuit that would request the VMEbus. A third solution is to control the backplane power for each slot on the VMEbus. This method would shut the power down to the slot during insertion and ejection. When the power is off in the selected slot, the system monitoring software informs the user to remove or insert the VMEbus board. There are components on the market today that may attempt to draw power through the control and address, data signal lines even when power is turned off. If this solution is being considered, the current draw of the powered signal lines has to be taken into account.

**NOTE:**

Any solution that would require a redesign of the VMEbus board could also include changes like: a DIN connector that has extended power and ground pins to reduce glitches and pins that would pre-charge the VMEbus board's circuitry; an additional connector that would provide extra pins which could indicate to the system controller or system monitoring software that the board is about to be inserted or ejected into the VMEbus system; or additional components that have been specially designed for a fault tolerant environment and are capable of live insertion. Because this requires changing the VMEbus board design, the compliance to IEEE 1014-1987 will be in jeopardy. While this method has merit for very high reliability systems it goes against the prime objective of this committee and therefore is not allowed.

### 7.3 POWER-UP BUILT-IN TEST

When a VMEbus board is powered up in any VMEbus backplane, the board may perform some type of diagnostic of its on-board features. This type of testing does not mean an in-depth look into the functionality of the board but implies that the on-board logic must come up and can access on-board features. This is necessary for any VMEbus board with on-board intelligence and requires configuration data to be installed before it can establish any type of communications. Therefore, it is recommended the intelligent board being inserted into the VMEbus backplane have power-on Built In Test. Boards used for high reliability defense or telecommunication applications would have extensive board
BIT capabilities. The simple bit capabilities available on the majority of off-the-shelf boards could be expanded to aid the system monitoring software in identifying and isolating faults. Once the VMEbus board has been inserted into the backplane and the power has stabilized, the board should perform some type of on-board diagnostics and report them to the CSM. This can be done through dual port RAM, serial bus, or with VMEbus interrupts.

7.4 ESTABLISHING AN ON-LINE STATUS WITH THE SYSTEM SOFTWARE.

The inserted VMEbus board establishes that it is functional through its BIT. The board then requests or must be polled by the system monitoring software to determine its on-line availability for system operation. This linking to the system monitoring software is an important phase in reestablishing a normal operational condition in the configured VMEbus system. It is possible to link the VMEbus board with the system monarch through a human interface by using a switch, a terminal, or a VMEbus generated software interrupt. The system monitoring software could poll the inserted VMEbus board address space expecting to receive a BERR* and/or a VMEbus time-out when the board is not in the slot or when it is doing its on board built in test. When the system monitoring software receives the DTACK* signal from the polling exercise it knows the inserted VMEbus board is an available resource in the configured system.

The first task the system monitoring software should give to the newly inserted VMEbus board is a request to download its BIT results for verification of its on-board features.

7.5 RESUMPTION OF SOFTWARE TASKING

Once the newly inserted VMEbus board has established a link with the CSM or system monitoring software, the phase of downloading and verifying the board's configuration begins. The system monitoring software should have stored the necessary information of the configured system's normal operation and the tasking sequence. The system software, after verifying that the new board is available, may begin the process of downloading the software tasks to resume normal operation. These operational procedures will be as diverse as the VMEbus applications.

7.6 NORMAL OPERATION

Once the system has downloaded the tasking to the new board there should be one last check to see, if indeed, the board is functioning as it was configured for the VMEbus application. Then without any error, fault or failure, the VMEbus system resumes normal operation.
GLOSSARY

All terms defined in the context of the VMEbus.

A16  A type of module that provides or decodes an address on address lines A01 through A15

A24  A type of module that provides or decodes an address on address lines A01 through A23

A32  A type of module that provides or decodes an address on address lines A01 through A31

ac   Alternating current. Refers to a facility power distribution system

ACFAIL*  ac power failure

Address-only cycle  A DTB cycle that consists of an address broadcast, but no data transfer. Slaves do not acknowledge address-only cycles and masters terminate the cycle without waiting for an acknowledgment

AM   Address modifier code

ANSI  American National Standards Institute

Arbiter  A functional module that accepts bus requests from requester modules and grants control of the DTB to one requester at a time

Arbitration  The process of assigning control of the DTB to a requester

Arbitration bus  One of the four buses provided by the backplane. This bus allows an arbiter module and several requester modules to coordinate use of the DTB

Arbitration cycle  An arbitration cycle begins when the arbiter senses a bus request. The arbiter grants the request to a bus requester, which signals
that the DTB is busy. The requester terminates the cycle by removing the bus busy signal, which causes the arbiter to sample the bus requests again.

**ASCII** American National Standard Code for Information Interchange

**Asynchronous** A system which is not synchronized to some periodic time base such as a clock signal. There is no predictable relationship between an asynchronous system and a time reference.

**ATE** Automatic test equipment

**ATR** Austin Trumbel Radio [box]. Air Transportable Rack is an alternative definition

**Backplane** A printed wiring board (PWB) with connectors and signal paths that bus the connector pins. Some systems have a single PWB, called the J1 backplane. It provides the signal paths needed for basic operation. Other systems also have a second PWB, called the J2 backplane. It provides additional connectors and signal paths needed for wider data and address transfers. Still others have a single PWB, called a J1/J2 backplane which provides the signal conductors needed for both the J1 and the J2 backplane.

**Backplane interface logic** Special logic that takes into account the characteristics of the backplane: its signal line impedance, propagation time, termination values, etc. The specification prescribes certain rules for the design of this logic based on the maximum length of the backplane and its maximum number of board slots.

**Bandwidth** In the context of communications systems the term refers to a quantity of the frequency spectrum available to, or used by, a system for transmitting information. In the context of micro-computer backplane buses, bandwidth refers to the highest frequency of data transfers of which a system is capable. For example, specifying a 40 Mbytes/sec bandwidth implies the maximum amount of data that can be moving across the bus continuously is 40 MB of data every second.

**BIT** Built-in test
**Block read cycle**  A DTB cycle used to transfer a block of 1 to 256 bytes from a slave to a master. This transfer is done using a string of 1, 2, or 4 byte data transfers. Once the block transfer is started, the master does not release the bus until all of the bytes have been transferred. It differs from a string of read cycles in that the master broadcasts only one address and address modifier code at the beginning of the cycle. Then the slave increments this address on each subsequent transfer so that data for the next cycle is retrieved from the next higher location.

**Block write cycle**  A DTB cycle used to transfer a block of 1 to 256 bytes from a master to a slave. This transfer is done using a string of 1, 2, or 4 byte data transfers. Once the block transfer is started, the master does not release the bus until all of the bytes have been transferred. It differs from a string of write cycles in that the master broadcasts only one address and address modifier code at the beginning of the cycle. Then the slave increments this address on each subsequent transfer so that data for the next cycle is stored to the next higher location.

**Board**  A printed wiring board (PWB), its collection of electronic components, bus connectors, and hardware necessary to form an assembly which will occupy one slot in the bus system.

**BTL**  Bus transceiver logic

**Bus**  A system composed of electrical conductors, signal definitions, and protocols which connects one or more electronic systems such that information can be transferred between them. Computer buses often use backplanes but some also use cables with two or more conductors.

**Bus timer**  A functional module that measures the time each data transfer takes on the DTB and to terminate a DTB cycle if a transfer takes too long. Without this module, if the master tries to transfer data to or from a nonexistent slave location it could wait forever for a slave to respond. The bus timer prevents this by terminating the cycle.

**Byte**  Eight bits of binary data

**C4I**  Command, control, communications, computers, and intelligence

**CFM**  Cubic feet per minute
CISC complex instruction set computers

COTS Commercial, off-the-shelf. Refers to products which can be purchased with relatively short delivery times and are normally available to commercial customers. These products are not customized or designed for any specific military project

CPI [clock] Cycles per instruction

CPU Central processing unit. This term is sometimes used synonymously with arithmetic logic unit, ALU. The term is used to refer to the component of a computer system which does the primary processing of a computer program. This term sometimes to refer to the entire microprocessor together with all of the components which allow the microprocessor to function as a computer

D08(O) (1) A slave that sends and receives data 8 bits at time over D00-D07, or (2) An interrupt handler that receives 8-bit status/ID over D00-D07, or (3) An interrupter that sends 8-bit status/ID over D00-D07

D08(EO) (1) A master that sends or receives data 8-bits at a time over either D00-D07 or D08-D15, or (2) A slave that sends and receives data 8-bits at a time over either D00-D07 or D08-D15

D16 (1) A master that sends and receives data 16-bits at a time over D00-15, or, (2) A slave that sends and receives data 16-bits at a time over D00-D15, or, (3) An interrupt handler that receives 16-bit status/ID over D00-D15, or, (4) An interrupter that sends 16-bit status/ID over D00-D15

D32 (1) a master that sends and receives data 32-bits at time over D00-D31, or (2) a slave that sends and receives data 32-bit at a time over D00-D31, or (3) an interrupt handler that receives 32-bit status/ID over D00-D31, or (4) and interrupter that sends 32-bit status/ID over D00-D31

Daisy chain A special type of signal line that is used to propagate a signal level from board to board, starting with the first slot and ending with the last slot. There are four bus grant daisy-chains and one interrupt acknowledge daisy-chain on the backplane
Data transfer bus  One of the four buses provided by the backplane. The data transfer bus allows masters to direct the transfer of binary data between themselves and slaves

Data-transfer-bus cycle  A sequence of level transitions on the signal line of the DTB that result in the transfer of an address and data between a master and a slave. There are 34 types of data transfer bus cycles. Newer revisions of ANSI/IEEE 1014 due for release in 1993 will add several more DTB cycle types

DEC  Digital Equipment Corporation

DMAC  Direct memory access controller

DOD  Department of Defense

DSP  Digital signal processing

DTB  Acronym and mnemonic for data transfer bus

ECL  Emitter coupled logic

EISA  Extended industry standard architecture. Refers to the 32-bit microcomputer bus used in IBM-AT clones

EMC  Electromagnetic compatibility

ETL  Enhanced Transceiver Logic

Eurocard  A printed wiring board which is defined in size and connector use by a European industry standard

FDDI  Fiber distributed data interface. Refers to a bus system which uses optical fibers to provide a data bus connection between computer systems

Firmware  Software which is programmed into non-volatile memory such as EPROM. Firmware is distinguished from software because changes to firmware require removing ICs from circuit boards for implementation of the revisions
IACK daisy-chain driver A functional module that activates the interrupt acknowledge daisy-chain whenever an interrupt handler acknowledges an interrupt request. This daisy-chain ensures that only one interrupter will respond with status/ID when more than one has generated an interrupt request on the same level.

IC Integrated circuit

IEEE Institute of Electrical and Electronic Engineers. Largest professional association of electrical engineers. Also, primary standards body for many electrical and computer standards, including VMEbus

IF Intermediate frequency

I/O Input/output. Refers to moving data into or out of a computer system

Interrupt acknowledge cycle A DTB cycle, initiated by an interrupt handler, that reads a status/ID from an interrupter. An interrupt handler generates this cycle when it detects an interrupt request from an interrupter and it has control of the DTB

Interrupt handler A functional module that detects interrupt requests generated by interrupters and responds to those requests by asking for status/ID information

Interrupt vector Following an interrupt request, in most computer systems, the interrupt requestor provides to the main processor a number, or interrupt vector, which is used by the processor to determine the appropriate routine to execute in response to the interrupt request. The interrupt vector is used to identify the source of the interrupt as well as to pass additional information to the processor

LAN Local area network

LED Light emitting diode

Location monitor A functional module that monitors data transfers over the DTB to detect accesses to the locations it has been assigned to watch. When an access occurs to one of these assigned locations, the location monitor generates an on-board signal
**Long word**  Four bytes

**Master**  A functional module that initiates DTB cycles to transfer data between itself and a slave module

**MB**  Megabyte

**MBLT**  Multiplexed block transfer

**MIPS**  Millions of instructions per second

**MFLOPS**  Millions of floating point operations per second

**MTBF**  Mean time between failures

**MTTR**  Mean-time-to-repair

**Microprocessor**  An integrated circuit which contains a central processing unit and often many other functions which are basic to a computer system. Microprocessors integrate on a single chip what used to take several circuits boards to build in the 1970s

**Monolithic**  In the context of backplanes, refers to a backplane structure which is one continuous PWB with no breaks

**MOPS**  Millions of operations per second

**PCB**  Printed circuit board

**Power monitor**  A functional module that monitors the status of the primary power source to the system and signals when the power has strayed outside the limits required for reliable system operation. Since most systems are powered by an ac source, the power monitor is typically designed to detect drop-out or brown-out conditions on ac input lines

**Priority interrupt bus**  One of the four major buses provided by the backplane. The priority interrupt bus allows interrupter modules to send interrupt requests to interrupt handler modules, and interrupt handler modules to acknowledge these interrupt requests
PWB  Printed wiring board. Referred to in commercial industry as a PCB, printed circuit board

RAM  Random access memory. In a general purpose computer main memory is always composed of RAM. Therefore RAM is often synonymous with main memory

Read cycle  A DTB cycle used to transfer 1-, 2-, 3-, or 4-bytes from a slave to a master. The cycle begins when the master broadcasts an address and address modifier code. Each slave captures the address and address modifier code and checks to see if it is to respond to the cycle. If so, it retrieves data from its internal storage, places it on the data bus, and acknowledges the transfer. Then the master terminates the cycle

Read-modify-write cycle  A DTB cycle that is used to read from, and write to, a slave's byte location(s) without permitting any other masters to access that location during that cycle. This cycle is most useful in multi-processing systems where certain memory locations are used to control access to certain systems resources, for example, semaphore locations

Register  An electronic circuit, which holds the value of a finite number of bits. In VMEbus applications, registers are on boards and usually hold 1-, 2-, or 4-bytes of data. A register is a small memory device. Registers are used to hold board configuration commands. Registers are sometimes used on I/O boards where there is little need for buffering data. In certain contexts, a register access can be synonymous with a memory access

Requester  A functional module that resides on the same board as a master or interrupt handler and requests use of the DTB whenever its master or interrupt handler needs it

RMW  Read-modify-write

ROAK  Release on acknowledge

ROR  Release on-request

RORA  Release on register access.

RWD  Release-when-done.
**SCSI**  Small Computer System Interface.

**Serial clock driver**  A functional module that provides a periodic timing signal that synchronizes the operation of the IEEE 1132 serial bus.

**Slave**  A functional module that detects DTB cycles initiated by a master and, when those cycles specify its participation, transfers data between itself and the master.

**Slot**  A position where a board can be inserted into a backplane.

**SPAWAR**  Space and Naval Warfare Systems Command.

**Standard**  1) A set of rules and design requirements defined by an organization which is nationally recognized as a standards authority. 2) A set of rules and design requirements which may or may not be formally defined by an organization but is used by many organizations or manufacturers because of market forces (a de facto standard).

**Standard backplane bus**  A computer bus which is defined by a standards organization and is recognized as such by the computer industry. It also be a computer bus which is defined by a manufacturer and is promoted to the public by that manufacturer for general use.

**Status/ID**  The VMEbus term for what is normally referred to as an interrupt vector. This data is supplied to an interrupt handler by the interrupt requester during an interrupt acknowledge cycle.

**Subrack**  A rigid framework that provides mechanical support for boards inserted into the backplane, ensuring that the connectors mate properly and that adjacent boards do not contact each other. It also guides the cooling airflow through the system and ensures that inserted boards do not disengage themselves from the backplane due to vibration or shock.

**SYSRESET***  System reset signal.

**System clock driver**  A functional module that provides a 16 MHz timing signal on the utility bus.
**System controller board** A board that resides in slot 1 of the backplane and has a system clock driver, a DTB arbiter, and IACK daisy-chain driver, and a bus timer. Some also have a serial clock driver, a power monitor, or both.

**U** When used in the context of VMEbus (e.g. 3U, 6U) refers to a unit of height, 1.75 inches, for a panel in a 19-inch rack. This measurement is not actually used in the mechanical design of VMEbus boards and is not included in the VMEbus specification. The VMEbus specification provides the necessary board dimensions directly.

**UAT** Unaligned transfer. Also, a mnemonic used in the VMEbus specification to refer to a master that sends or receives data in an unaligned fashion.

**Utility bus** One of the four buses provided by the backplane. This bus includes signals that provide periodic timing and coordinate the power-up and power-down sequence of the system.

**VME** VERSA Module Eurocard.

**VSB** VMEbus subsystem bus. This is a standard bus which uses the user defined pins of VMEbus to create a secondary 32-bit high-speed bus.

**WAN** Wide area network.

**Word** Two bytes.

**Write cycle** A DTB cycle used to transfer 1-, 2-, 3-, or 4-bytes from a master to a slave. The cycle begins when the master broadcasts an address and address modifier. Each slave on the bus checks the address to see if is to respond to the cycle. If so, the slave stores the data and then acknowledges the transfer. The master then terminates the cycle.