

IEEE Standard for Futurebus+[®] Spaceborne Systems—Profile S

Sponsor

**Bus Architecture Standards Committee
of the
IEEE Computer Society**

Approved 20 March 1997

IEEE Standards Board

Abstract: In the Futurebus+[®] series of standards, tools with which high-performance bus-based systems may be developed are provided. This architecture provides a wide range of performance scalability over both cost and time for multiple generations of single- and multiple-bus multiprocessor systems. This document, a companion standard to the ISO/IEC 10857: 1994 (ANSI/IEEE Std 896.1, 1994 Edition) Futurebus+[®] Logical Layer Specification, builds on the logical layer by adding requirements for a spaceborne profile. It is to this profile that products will claim conformance. Other specifications may be required in conjunction with this standard.

Keywords: bus architecture, Futurebus+[®], spaceborne computers, spaceborne modules, spaceborne systems

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Copyright © 1997 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1997. Printed in the United States of America.

ISBN 1-55937-909-X

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

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (508) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

(This introduction is not part of IEEE Std 896.10-1997, IEEE Standard for Futurebus+[®] Spaceborne Systems—Profile S.)

Overview

Space systems have traditionally pursued point solutions for implementation of digital computers and controllers without regard for accepted commercial standards or component commonality. Stringent size, weight, power, performance factors, and environmental requirements have driven space system developers to these point solutions. The problem with the traditional approach is that the space qualification of multiple unique implementations is expensive and does not permit the reuse of common-function components on multiple missions.

This document provides a standard open architecture framework for implementing spaceborne computer systems while addressing spacecraft unique restrictions. It brings forth several issues not previously defined, namely: (1) establishing common printed circuit board form factors, (2) enabling the prototyping of flight hardware with relatively low-cost commercial equivalents, (3) permitting component reuse across multiple missions, and (4) allowing reduction in the time required to move state-of-the-art technology into existing and new applications. A low-power, high-performance standard backplane tailored specifically for space applications is defined.

This standard, along with the companion IEEE Std 1101.7-1995, *IEEE Standard for Space Applications Module, Extended Height Format E Form Factor (ANSI)*; IEEE Std 1101.3-1993, *IEEE Mechanical Standard for Conduction-Cooled and Air-Cooled 10 SU Modules (ANSI)*; and IEEE Std 1156.4-1997, *IEEE Standard for Environmental Specifications for Spaceborne Computer Modules*, comprise a complete set of standards for implementing spaceborne computer systems that will satisfy a broad range of applications. Compatibility with the Futurebus+ logical layer [ISO/IEC 10857: 1994 (ANSI/IEEE Std 896.1, 1994 Edition)] and provisions for the adoption of the fault tolerance extensions in IEEE Std 896.9-1994, *IEEE Standard for Fault Tolerant Extensions to the Futurebus+[®] Architecture*, were key elements in this development.

Futurebus+ is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

Working group members

The following members of the Futurebus+ Spaceborne Profile Working Group were involved in the development and review of this document during the time that it was in the working group:

William Schneider, Chair
Joseph R. Marshall, Technical Editor

Gary A. Gardner, Vice Chair
Conrad A. Laurvick, Secretary

Cory Ackerman
Darryl Adams
Bob Alchinger
Peter Alcivar
Ray Alderman
William Ashton
Paul Bartlett
Harrison Beasley
Mitch Beaumont
Andrew Beck
Stephen E. Belvin
Robert Bertin
Ronald Bettencourt
Andy Bonafini
Andy Brough
Jim Brown
Mark H. Bunker
Kim Burris
Don Caldwell
Doug Caldwell
Ken Campbell
Andy Capobianco
Charles Chestwood
Karen Churas
Tom Clark
Kim Clohessy
Tom Cohen
Bill Crnkovich
Tom Cronauer
David Dai
C. Alan Dennis
Paul Dixon
Robert Doran
Mike Dorsett
Frank Duffy
Bruce Dunlop
Jerry Edwards
Robert Fluher
Michael S. Foster
John Franz

Richard Fryer
Robert George
John Griffith
Darryl Gunn
Claue-Goran Gustausson
Tom Hillyer
Joe Hoffman
Pat Hoffman
Ken Hoffner
Gary Hokenson
Ron Hudson
Ed Ille
Fred Jackson
Ed Jacques
Gary Jewell
G. E. Johnson
Val Johnson
Jason Jones
Rangesh Kasturi
Bud Keith
Jerry Keller
Dennis Kern
Dave Kirk
James Kuo
Ralph Lachenmaier
Michael Lambrou
Dale Langston
Robert Lipp
David Maas
Ernest Mardage
Ed Marian
Fred Martin
Fenton McDonald
John W. McKinstry
Rusty Meier
Thanos Mentzelopoulos
Michael Munroe
Eric Nickerson
Mary Lynne Nielsen

Richard Normann
Peter Onufryk
Fred J. Orlando
Elwood T. Parsons
Gary Pash
Heinz Piorunneck
Paul Poehler
Tom Potyraj
Bill Ramsay
Thomas Reglein
Ken Ristow
Dave Roderick
Jack Rosenberg
John L. Rynearson
Bob Schethide
Gene Schramm
Christine Seliskar
Mark Sheets
Dan Sherick
Jim Simon
Dennis Smith
Keith Spence
William Stapor
Richard Straker
Zolton Stroll
Jack Stuebs
Lance Sundstrom
Bill Suras
John Theus
Matthew Ting
Tony Tomarchio
Joe Toy
Dean Van de Walker
David Wasler
Randy Wattlelet
Colin Whitby-Stevens
Joel Witt
Richard Wood
Mark Woodbury
Janusz Zalewski

Balloting committee

The following people were on the balloting committee:

Ghassan A. Abbas	Michael S. Foster	Granville Ott
Dharma P. Agrawal	Gary A. Gardner	Elwood T. Parsons
Malcolm J. Airst	Joseph D. George	Mira Pauker
Keith D. Anthony	Michael C. Hayward	Thomas Potyraj
Peter C. Bahrs	Richard W. Kuberry	Delia Rankin
James Beaumont	Conrad A. Laurvick	Steve Rankin
Stephen E. Belvin	Rollins Linser	David Robak
William P. Blase	Joseph R. Marshall	Frederick E. Sauer
John C. Brightwell	Stewart McDonald	Robert C. Tripi
Mark H. Bunker	John W. McKinstry	Joseph G. Tront
C. H. Chen	Thanos Mentzelopoulos	Eike G. Waltz
Dante Del Corso	Yoshiki Mitani	Thomas H. Wegmann
Sourav K. Dutta	Klaus-Dieter Mueller	James Wolffe
Roger D. Edwards	J. D. Nicoud	Bin Wu
Gordon Force	Fred J. Orlando	Janusz Zalewski

Ballot resolution committee

The following are the members of the Futurebus+ Spaceborne Profile Ballot Resolution Committee that responded to the ballot comments and finished this standard.

Tom Reglein, *Ballot Coordinator*

Stephen E. Belvin	Tony DiIulio	Conrad A. Laurvick
Robert Cherney	Gary A. Gardner	Joseph R. Marshall
Marlin Clark	Eric Johnson	William Schneider
	Henry Kay	

When the IEEE Standards Board approved this standard on 20 March 1997, it had the following membership:

Donald C. Loughry, *Chair*

Richard J. Holleman, *Vice Chair*

Andrew G. Salem, *Secretary*

Clyde R. Camp	Lowell Johnson	Louis-François Pau
Stephen L. Diamond	Robert Kennelly	Gerald H. Peterson
Harold E. Epstein	E. G. "Al" Kiener	John W. Pope
Donald C. Fleckenstein	Joseph L. Koepfinger*	Jose R. Ramos
Jay Forster*	Stephen R. Lambert	Ronald H. Reimer
Thomas F. Garrity	Lawrence V. McCall	Ingo Rüschi
Donald N. Heirman	L. Bruce McClung	John S. Ryan
Jim Isaak	Marco W. Migliaro	Chee Kiow Tan
Ben C. Johnson		Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Alan H. Cookson

Kristin M. Dittmann
IEEE Standards Project Editor

Contents

CLAUSE	PAGE
1. Overview.....	1
1.1 Scope	1
1.2 Applicability	1
1.3 Background to the standard.....	1
1.4 How to use this standard	2
1.5 Why use this standard	3
2. References.....	4
2.1 Referenced documents	4
2.2 Conflicting standards.....	5
3. Definitions.....	6
3.1 Special word usage.....	6
3.2 Bus line and signal conditions.....	6
3.3 Futurebus+ terminology	7
3.4 Conventions.....	10
3.5 Syntax.....	10
3.6 Futurebus+ Logo	11
4. Profile S reference specification	12
4.1 Introduction	12
4.2 Target applications	14
4.3 Reference tables	14
4.4 Profile S interoperability	18
4.5 Conformance testing	20
5. Detailed specification, Futurebus+ logical layer	21
5.1 Arbitration	21
5.2 Parallel protocol	21
6. Utility Signals	24
6.1 Trigger 0 (TR0*) and Trigger 1 (TR1*).....	24
6.2 Run (RUN*)	24
6.3 Nuclear Event Detect (NED*).....	24
6.4 Power Failure Imminent (PFI*)	25
6.5 Power Normal (PN).....	26
6.6 System Reset (SR*).....	26
6.7 Utility Spares (UPSP* and UWSP).....	26

CLAUSE	PAGE
7. Fault tolerance aspects of the parallel protocol and utility signals	27
7.1 Error detection.....	27
7.2 Error isolation, logging, and diagnostics.....	36
7.3 Error removal and recovery.....	37
7.4 Pinout summary.....	43
8. Serial Bus	45
8.1 Serial Bus physical layer	45
8.2 Serial Bus link layer	46
8.3 Serial Bus transaction layer.....	46
9. Bus/Node management and CSRs	48
9.1 Addressing.....	48
9.2 Byte-Lane wiring and byte ordering	48
9.3 CSRs.....	48
9.4 Futurebus+ CSRs.....	49
9.5 Unit space CSRs.....	81
9.6 Serial Bus management	89
9.7 Private space.....	90
9.8 Interrupts	91
9.9 Diagnostics and test.....	91
9.10 Monarch selection	91
10. Detailed specification, physical layer—SEM-E Stretch.....	92
10.1 Mechanical	92
10.2 Input/Output	92
10.3 Profile connector, power, and signal pin assignments	92
11. Detailed specification, Physical layer—10 SU.....	122
11.1 Mechanical	122
11.2 Input/Output	122
11.3 Profile connector, power, and signal pin assignments	122
12. Profile S electrical.....	159
12.1 Backplane characteristics and design requirements	159
12.2 Module electrical characteristics and design requirements.....	160
13. Profile S power	166
13.1 Module power	166
13.2 Host system power supplies	167

CLAUSE	PAGE
14. Profile S environmental specifications	170
14.1 System environmental requirements	170
14.2 Module environmental requirements	170
15. Recommended practices	171
15.1 Thermal considerations	171
15.2 Power consumption	171
15.3 Pinouts	172
15.4 Bus performance analysis.....	172
15.5 Bandwidth considerations	173
16. Profile S module selection criteria	176
16.1 Introduction	176
16.2 Profile S modules	176
16.3 Profile S systems	177
17. Alternate profile S module connector	178
17.1 Mechanical	178
17.2 Input/Output	178
17.3 Profile connector, power, and signal pin assignments	178
ANNEX	
Annex A (informative) Bibliography	208

IEEE Standard for Futurebus+[®] Spaceborne Systems—Profile S

1. Overview

1.1 Scope

This interface standard establishes a modular open architecture for spaceborne computing systems employing digital computers, processors, and other electronic modules. It provides for the integration and interoperability of diverse sets of electronic modules for the purpose of configuring various types of spaceborne computer systems. This standard specifies the electrical, logical, mechanical, thermal, and input/output interfaces necessary to allow modules developed to this standard to operate together in an integrated system or subsystem. In particular, the issues unique to spaceborne systems, such as size, weight, power, environment, reliability, and fault tolerance, are addressed. This common interface standard will allow the development of cost-effective, reliable, fault tolerant modules that will have broad application across the range of space vehicles.

1.2 Applicability

The applicability of this standard includes all spaceborne computing systems in spacecraft or launch vehicles of any mission duration or environment.

1.3 Background to the standard

This standard is the top-level document specifying a family of interoperable high-performance backplane bus connected modules. As such, this standard is built upon the collected work of many other standards. A decision was made early on in the creation of this standard not to repeat verbatim specifications carried in lower-level standards but instead to accurately reference them in this document. Most specifications in this document beyond references to other standards represent new or changed specifications. As much rationale as possible has been included throughout this standard; however, it will be next to impossible to fully understand this standard without a cursory understanding of the underlying standards.

To make full use of this standard, one needs to obtain or be familiar with the content of most if not all of the referenced IEEE standards. These are listed in 2.1.1. This standard is built upon the Futurebus+[®] family of standards and the Serial Bus (IEEE Std 1394-1995) standard. One should review the logical layer as defined in ISO/IEC 10857: 1994 and the physical layer as defined in IEEE Std 896.2-1991 to understand the basic Futurebus+ protocol. Since all Profile S modules will be fault tolerant at the bus level, one should next look

at the extensions to the Futurebus+ that are added by IEEE Std 896.9-1994. At about this time, one will probably need more understanding of the Control and Status Register (CSR) architecture as defined in ISO/IEC 13213: 1994. This is the basis of both the Futurebus+ and Serial Bus. Serial Bus is a mostly stand-alone document, and this standard specifies the backplane version for use on Profile S modules and should be looked at next. At this point, the only additional requirement is an understanding of the physical form factors and the environment. The environment is specified in IEEE Std 1156.4-1997. Two form factors have been specified for Profile S modules—SEM-E Stretch (IEEE Std 1101.7-1995) and 10 SU (IEEE Std 1101.3-1993). With at least a cursory understanding of the contents of these standards, one can now more fully appreciate and use this document.

1.4 Structure of the standard

Clause 4 presents in table form a summary of most of the content of this standard. After this, Clause 5 details the Futurebus+ logical layer, Clause 6 the utility signals, Clause 7 the fault tolerance aspects of the Futurebus+ and utility signals, and Clause 8 the Serial Bus. Clause 9 maps the CSRs from all the underlying standards (both Futurebus+ and Serial Bus) into a consistent group to be used across all Profile S modules. The majority of the new information here introduces an error correcting configuration of Futurebus+. Clause 10 provides the physical layer detail for a SEM-E Stretch module, and Clause 11 gives the same for a 10 SU module. Clause 17 provides a second connector variation for the SEM-E Stretch form factor. These sub-clauses provide the interoperable pinout definitions on each of these form factors. Since the electrical characteristics are different (lower power) in this standard than the standard backplane transceiver logic (BTL) called out in IEEE Std 896.2-1991, Clause 12 gives the specifications for compatible operation. Clause 13 provides the detail of the power distribution. Clause 14 details the environmental requirements and mostly references the user to IEEE Std 1156.4-1997. Clause 15 collects in one place the recommended practices for the use of the Space Profile. Clause 16 provides a guide for manufacturers of Profile S modules, giving required and recommended data that should be provided for the best use of a Profile S module. Finally, a bibliography of works of interest is included in Annex A.

1.5 How to use this standard

This document may be used in different ways by different users for specific purposes. If one is a silicon implementor, ISO/IEC 10857: 1994, IEEE Std 896.2-1991, IEEE Std 896.9-1994, and IEEE Std 1394-1995 provide most of the information one needs. This standard provides two new major areas of interest. The first is the new electrical requirements, which are applicable to all Profile S modules. These have been defined these so that protocol chips and functions that meet the above bus standards will work with parts made to the new electrical standards as a direct replacement for BTL in order to leverage off much higher volumes that drive the development of the protocol chips. Clause 12 provides the areas of the most interest. The second new area of interest is the Error Correcting configuration definition. Much of the new function in this standard is directed to this bus implementation. It has been blended, wherever possible, with equivalent parts of the existing standards. Someone trying to implement silicon to support the Error Correcting configuration should pay particular attention to the logical layer extensions and the CSRs to decide where to implement these signals.

To the module or system builder, this document provides the interconnect in either Clause 10, 11, or 17. While SEM-E Stretch provides what most users need today with its 32-bit bus capabilities, the forward-thinking system builder may want to start with 10 SU to have many more I/Os today and a more expandable form factor for the future. A system builder should review the excellent Failure Modes and Effects Analysis (FMEA) in IEEE Std 896.9-1994 as a starting point for performing the various analyses required to determine the overall fault tolerance capability of the system. A functional fault analysis (i.e., an analysis of error classes and how they are handled at each error containment region) is a recommended front-end system engineering activity. A Profile S module is a natural error containment region. The error handling scenarios provided in IEEE Std 896.9-1994 provide most of the potential interface error classes. These may then be extended to include the actual complete view of the error classes at each bus's level in the system.

To the device driver writer, the detailed CSR definitions in Clause 9 as well as those referenced in IEEE Std 896.9-1994, ISO/IEC 10857: 1994, and ISO/IEC 13213: 1994, are important. Error scenarios in IEEE Std 896.9-1994 indicate what shall be logged in each error case and provide a structure for handling the many types that may be injected. This standard adds spare signals to the Error Correcting configuration that provide another recovery method for this bus. The various registers provided for diagnostics and fault injection may be used to create those services. These CSRs were developed based on collective contributions from telecommunications, military, commercial, and space users. Wherever possible, CSRs were reused from other profiles rather than invented anew. The device driver writer should check upon availability of device drivers for these profiles as a way to reuse much of their content and then personalize it to Profile S.

Finally, the user of modules built to this standard should specify exactly what is required but leave as much optional as possible. This will enable a smaller set of interoperable modules to serve a larger application space. One area where this is especially true is in pinouts. Space users have always created unique solutions in the past. Because of this heritage, the working group could not specify just one pinout or one I/O configuration and expect anyone to use it. But having 44 different module classes identified introduces complexity in interoperability between modules. Many of these are naturally interoperable. But unique requirements can be placed on a module that will make it interoperable only with its own class. A good way to look at this is to lay out requirements over more than one application or over more than one time period. Look for commonality and look at the areas where growth is desired. Try to specify what will work for the first and what could be used as at least a foundation for the second. Specifying individual needs with a global view should enable more reuse. In the future, one can take a collection of previous application modules and put together a quick prototype similar to what is possible today with buses like VME.

1.6 Why use this standard

This standard was developed so that a scalable high-performance commercial standard (Futurebus+) could be applied and used in spacecraft applications. The major problems considered to make this possible are those that apply to any space system: size, weight, power, fault tolerance, and radiation.

The commercial Futurebus+ is a strong foundation on which to build this spaceborne standard. It provides high performance that is scalable; significant configuration options in message passing, arbitration, and transaction variety; a standardized register interface, backplane interoperability, and cache coherency; and a commercial base upon which to build.

In the size and weight area, two standard form factors and a matching high density connector for each that pack a maximum amount of pins into the smallest area possible have been specified. A variety of pinout configurations to enable this to be used in small 1–5 slot applications equally as well as in large 15+ slot applications has been provided. Using Futurebus+ and its built in scalability and personalization features, one may build a range of backpanels that either emphasize bus bandwidth, external I/O control and status, or a combination of both.

A major challenge in bringing Futurebus+ to space was to reduce the power of each module's transceivers. A new electrical specification has been developed that maintains the incident wave switching properties of BTL yet cuts the power required by over 80% per slot with only a small degradation in performance. Although not electrically compatible with BTL, it is expected that there will be transceivers built that are compatible with the commercial protocol chips available today. Space profile boards may be separately powered due to the cold-spares properties of the electrical interface.

In the fault tolerance area, all extensions provided by IEEE Std 896.9-1994 have been incorporated, and three different levels of bus fault tolerance have been defined. An application can use the one that best matches the rest of its subsystem, detection, and latency requirements. A Level 0 sublevel is provided to match current silicon. A Profile S module's Futurebus+ interface will be single point fault tolerant. It will also have the diagnostic capability of any telecommunication system due to the strong influence of the tele-

communications profile on this work. Therefore, one may build a system that has full access to telemetry data on the module. One may also build a system that autonomously attempts recovery from errors.

Finally, all Profile S modules shall be built to meet environmental performance levels for the environments specified in IEEE Std 1156.4-1997. This should greatly reduce the uniqueness of each program in the environmental area. Radiation is one of the major categories addressed by IEEE Std 1156.4-1997, and three levels of radiation tolerance are specified.

This standard has been written to provide applicable specifications today that will go a long way toward standardizing the space computer industry at the board interface level, just as VME boards are standard for many ground applications.

2. References

2.1 Referenced documents

There are several standards that are referenced by the work being done on Futurebus+ documents. While the companion standards are stand-alone, i.e., can be referenced by other bus specifications, these standards must be referenced to ensure intermateable and interoperable systems. This standard shall be used in conjunction with the following standards. When the following standards are superseded by an approved revision, the revision shall apply. These standards are listed below:

2.1.1 IEEE standards

IEEE Std 896.2-1991, IEEE Standard for Futurebus+®—Physical Layer and Profile Specification (ANSI) (includes IEEE Std 896.2a-1994).¹

IEEE Std 896.3-1993, IEEE Recommended Practices for Futurebus+® (ANSI).

IEEE Std 896.4-1993, IEEE Standard for Conformance Test Requirements for Futurebus+® (ANSI) (includes IEEE Std 896.4a-1995).

IEEE P896.6, Draft Standard for Futurebus+®—Telecommunications Profile (D3, Dec. 1994).²

IEEE Std 896.9-1994, IEEE Standard for Fault Tolerance Extensions to the Futurebus+® Architecture (ANSI).

IEEE Std 1101.3-1993, IEEE Mechanical Standard for Conduction-Cooled and Air-Cooled 10 SU Modules (ANSI).

IEEE Std 1101.4-1993, IEEE Standard for Military Module, Format E Form Factor (ANSI).

IEEE Std 1101.7-1995, IEEE Standard for Space Applications Module, Extended Height Format E Form Factor (ANSI).

IEEE Std 1156.1-1993, IEEE Standard for Microcomputer Environment Specifications for Computer Modules (ANSI).

¹IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

²Numbers preceded by P are IEEE authorized standards projects that were not approved by the IEEE Standards Board at the time this publication went to press. For information about obtaining drafts, contact the IEEE.

IEEE Std 1156.4-1997, IEEE Standard for Environmental Specifications for Spaceborne Computer Modules.

IEEE Std 1212.1-1993, IEEE Standard for Communicating Among Processors and Peripherals Using Shared Memory (Direct Memory Access—DMA) (ANSI).

IEEE Std 1394-1995, IEEE Standard for a High Performance Serial Bus (ANSI).

ISO/IEC 10857: 1994 (ANSI/IEEE Std 896.1, 1994 Edition), Information technology—Microprocessor systems—Futurebus+[®]—Logical protocol specification (*includes IEEE Std 896.1a-1991*).³

ISO/IEC 13213: 1994 (ANSI/IEEE Std 1212, 1994 Edition), Information technology—Microprocessor systems—Control and Status Registers (CSR) Architecture for microcomputer buses.

ISO/IEC 14536: 1995 (ANSI/IEEE Std 896.5, 1995 Edition), Information technology—Microprocessor systems—Futurebus+[®]—Profile M (Military) (*includes 896.5a-1994*).

2.1.2 Military standards

MIL-STD-1540C, Military Standard Test Requirements for Launch, Upper-Stage, and Space Vehicles, United States Air Force, 15 September 1994.⁴

MIL-STD-1547, Electronic Parts, Materials, and Processes for Space and Launch Vehicles, United States Air Force, 1 Dec. 1987.

MIL-STD-1553B, Aircraft Internal Time Division Command Response Multiplex Data Bus, Notice 2-8 Sep. 1986.

MIL-STD-1773, Fiber Optic Mechanization of an Aircraft Internal Time Division Command/Response Multiplex Data Bus, Notice 1-2 Oct. 1989.

2.1.3 EIA standards

EIA IS E700A, 396 Pin Connector Keying.⁵

EIA IS 22, 372 Pin Connector Keying.

EIA PN 2933A, 556 Pin Connector Keying.

2.2 Conflicting standards

Should any of the above standards conflict as to the implementation of CSRs, the precedence shall be resolved with the following priority: this standard, IEEE Std 896.9-1994, ISO/IEC 10857: 1994, IEEE Std 896.2-1991, IEEE Std 1394-1995, and ISO/IEC 13213: 1994.

³ISO/IEC (ANSI/IEEE) publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA. They are also available from IEC Sales Department, Case Postale 131, 3 rue de Varembe, CH-1211, Genève 20, Switzerland/Suisse.

⁴Military standards and handbooks are listed in the Department of Defense Index of Specifications and Standards (DODISS). Copies of the DODISS are available on a yearly subscription basis from the Director, Navy Publications and Printing Office, 700 Robbins Avenue, Philadelphia, PA 19111-5093, USA.

⁵EIA publications are available from Global Engineering, 1990 M Street NW, Suite 400, Washington, DC, 20036, USA.

3. Definitions

3.1 Special word usage

3.1.1 may: A keyword that indicates flexibility of choice with no implied preference.

3.1.2 not supported: Profile-Compliant systems or modules may not respond, or may respond inappropriately, to the existence or assertion of the feature under discussion.

3.1.3 optional, option: Refers to the implementation or use of a function that is not required. Options may be Futurebus+-specific, profile-specific, vendor-specific, or unit-specific. The latter two are not discussed or specified in profiles, and may be implemented provided they do not conflict in any way with profiles or other standards, have no effect on required levels of interoperability, and are fully documented in product specifications. Futurebus+-specific options are those that are specified in Futurebus+ documents. Reference shall be made to the appropriate standard, with wording such as “if implemented, this option shall meet the requirements specified in (document and subclause).” An example of a Futurebus+-specific option is distributed arbitration. Similarly, a profile-specific option is one that is specified in detail in the profile. Any profile-compliant design implementing that function shall do it as specified in the profile. Availability of an option shall not be required for the correct operation of a profile-compliant module or system; modules implementing an option should be able to default to operation with the option disabled. Availability of an option should be specified in a documented capability read-only memory (ROM) bit in Control and Status Register (CSR); a corresponding read/write bit should be available to turn the function on or off.

3.1.4 prohibited: Profile-Compliant systems or modules shall not exhibit or assert the behavior or function under discussion. Modules compliant with multiple profiles may implement a function prohibited by a profile provided that function is disabled in systems compliant with that profile, or provided they do not direct prohibited transactions at modules compliant with that profile.

3.1.5 required or mandatory: All profile-compliant modules or systems shall implement the specified feature or function.

3.1.6 shall: A keyword that indicates a mandatory requirement. Designers must implement all such mandatory requirements to ensure interoperability with other IEEE Std 896.10-1997 compliant designs.

3.1.7 should: A keyword that indicates flexibility of choice with a strongly preferred implementation. The phrase “it is recommended” is used interchangeably with the keyword *should*.

3.1.8 supported: Profile-Compliant systems or modules shall respond appropriately to the existence or assertion of the specified feature or function. The profile or a referenced Futurebus+ standard shall specify the appropriate functional behavior or design.

3.2 Bus line and signal conditions

3.2.1 activate: (A) The action of applying signals to a group of bus lines. (B) The state of a group of bus lines when they carry signals.

3.2.2 assert: (A) The action of applying a logic 1 signal to a bus line. (B) The state of a bus line when the signal it carries represents a logic 1.

3.2.3 * (asterisk): When appended to a signal’s name, the suffix “*” (asterisk) indicates that the logic 1 state of the signal is represented by a less positive voltage than the logic 0 state.

3.2.4 bus line: The medium for the transmission of signals. Futurebus+ Spaceborne Profile uses both open collector and push-pull drivers to match whether a signal is expected to be driven by more than one board simultaneously. A bus line is driven by only one board for those signals that are push-pull. A bus line may be driven by several boards simultaneously for those signals that are open collector. Therefore, the signal carried by the bus line is the combination of signals applied to that line from each board in the open collector case or from the one board in the push-pull case.

3.2.5 release: (A) The action of applying a logic 0 signal to a bus line. (B) The state of a bus line when the signal it carries represents a logic 0.

3.2.6 signal names: The simplified alphanumeric notation used to represent a group of lines or signals. For example, where a group of bus lines are represented by the same letters, the lines within the group are numbered, e.g., **AD0**, **AD1**, **AD2**, etc. In order to represent a group of lines or signals in more convenient form, the notation **AD[31..0]** is used. Also, the notation **AD[]** is used to refer to all of the lines within the group.

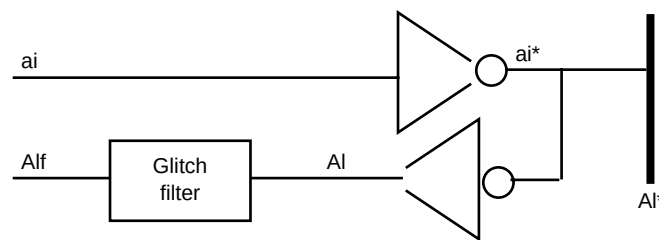


Figure 3-1—Signal naming convention

The signal that a module applies to the module-side input of its driver is designated by a lowercase label, e.g., **ai**. The signal that a specific module applies to a bus line is designated by a lowercase label with an asterisk, e.g., **ai***. The signal that appears on a bus line as the result of the combined signals applied to it by all modules is designated by an uppercase label with an asterisk, e.g., **AI***.

When appended to a signal name, the suffix “f” (filtered) refers to the bus line signal after it has passed through a receiver and a wire-or glitch filter (integrator). For example, **AIf** refers to the signal on the **AI*** line, after it has passed through an inverting receiver to become **AI** and the wire-or glitch filter to become **AIf**.

3.3 Futurebus+ terminology

3.3.1 Application Environment Profile (AEP): A document that describes functional requirements and points to existing standards, selecting and binding options within those standards. An implementor who then designs a specific module and/or system should be reasonably assured that another designer’s (manufacturer’s or supplier’s) modules will properly function within the same system. This includes all aspects of definition: mechanical, electrical, protocol, environmental, and system considerations.

3.3.2 backplane: An electronic circuit board and connectors used to interconnect modules together electrically. The backplane connects selected pins of the connectors, thus providing the medium for the transfer of signals needed for the operation of the bus.

3.3.3 backplane bus: A means to connect circuit modules using common signal traces on a backplane and a standard set of rules.

3.3.4 big endian: A method of storing multibyte data in a byte-addressable memory such that the most significant byte of the data is stored at the lowest address.

3.3.5 bus bridge: A bus bridge is an interconnect between two or more buses that provides signal and protocol translation from one bus to another. The buses may adhere to different bus standards for mechanical, electrical, and logical operation (such as a bus bridge from Futurebus+ to VMEbus or to Multibus II).

3.3.6 command_reset: An initialization event that is initiated by a write to the **RESET_START** CSR.

3.3.7 CSR: Control and Status Register.

3.3.8 geographical address: A unique identifier statically assigned to each slot by the backplane.

3.3.9 glitch filter: As explained in ISO/IEC 10857: 1994, this filters out a fundamental transmission line effect found in bused backplane implementations. The effect is commonly called the *wire-OR glitch*.

3.3.10 immediate entry: A read-only memory (ROM) entry that provides a 24-bit immediate data value.

3.3.11 intermateable: Mechanical compatibility between modules with the backplane, card cage, and system into which they are inserted. Sometimes includes compatibility in the assignment of power and ground pins.

3.3.12 interoperate, interoperability: Compatibility of modules with each other or with the system/backplane into which they are inserted. When a pair of such units is said to *interoperate*, (a) they cannot suffer damage as a consequence of being powered and functioning in the same system; (b) the modules and the system will each be able to perform the basic function for which they were designed; (c) the modules will be able to communicate with each other using specified Futurebus+ transactions.

Some modules or backplanes will have optional features not shared by others in the same system, so they need to be able to default to the smaller common set in order to communicate with each other. A module or system working in such a reduced-capability mode may suffer performance degradation that may be severe, but will still be interoperable in that system.

3.3.13 key: When used in the context of a read-only memory (ROM) entry, a key is an 8-bit field whose value identifies a ROM location as an immediate entry, offset entry, leaf entry, or subdirectory entry. This is a term used (but not defined) in ISO/IEC 13213: 1994.

3.3.14 leaf entry: A read-only memory (ROM) entry that specifies the address of a variable-length data block.

3.3.15 little addressian: A description of the physical location of data-byte addresses on a multiplexed address/data bus. On such a bus, the data-byte with the smallest address is multiplexed (in time or space) with the least significant byte of the address.

3.3.16 little endian: A method of storing multibyte data in a byte addressable memory such that the least significant byte of the data is stored at the lowest address.

3.3.17 live insertion: The process of inserting boards into a backplane when power is on. Insertion of boards into a backplane can be performed when power is off or when power is on.

3.3.18 LSB: Least significant bit.

3.3.19 monarch processor: The processor that is selected to partially initialize the local-bus resources and fetch the initial boot code.

3.3.20 MSB: Most significant bit.

3.3.21 node: A set of Control and Status Register (CSR) addresses (including identification read-only memory and reset command registers) that are initially defined in a 4 kB (minimum) initial node address space. Each node can be reset independently (a reset of one node has no effect on other nodes).

3.3.22 node ROM: A range of register addresses that are mapped to address offsets 1024 to 2047. In ISO/IEC 13213: 1994 this is known as ROM_WINDOW.

3.3.23 offset entry: A read-only memory (ROM) entry that provides a 24-bit offset value. The offset values specifies the location of a Control and Status Register (CSR) that provides a 32-bit parameter.

3.3.24 Profile S module: The assembled unit, containing a Futurebus+ interface and one or two nodes complying with Profile S, which is inserted into a compatible Futurebus+ slot.

3.3.25 Profile S system: The assembly of hardware elements made up of, at a minimum, the Profile S compliant backplane, card cage, and a power supply. Modules complying with other profiles may operate compatibly with Profile S systems and modules if they meet Profile S mechanical requirements, and if their features constitute an identity or a superset of those mandated in this profile.

3.3.26 root directory: A region in read-only memory (ROM) specified in ISO/IEC 13213: 1994 whose size is identified in the first location of the root directory and whose contents include ROM entries that may be identified using the ROM key.

3.3.27 SEM-E Stretch: A mechanical module equivalent in design to the module defined by IEEE Std 1101.4-1993. The height of this module has been extended beyond the measurement given in IEEE Std 1101.4-1993. The complete mechanical specifications for the SEM-E Stretch can be found in IEEE Std 1101.7-1995.

3.3.28 serial bus: A peripheral interconnect and an alternate diagnostic and control path.

3.3.29 slot: A mechanical location on a backplane.

3.3.30 subdirectory entry: A read-only memory (ROM) entry that specifies the address of another ROM subdirectory.

3.3.31 system configuration process: The software that initializes the system. The monarch processor executes the system configuration process.

3.3.32 system reset: As specified in ISO/IEC 10857: 1994, this is an initialization event initiated when any module asserts the **re*** signal for 100 to 200 ms. The equivalent term for this event, found in ISO/IEC 13213: 1994, is *power_reset*.

3.3.33 unit: A subcomponent of a node that provides a processing, memory, or I/O functionality. After the node has been initialized (typically by generic software), the unit provides the register interface that is accessed by I/O driver software. The units normally operate independently of each other, and do not affect the operation of the node upon which they reside. Note that one node could have multiple units (for example, processor, memory, and SCSI controller).

3.3.34 write one to clear: A method used to clear specific bits in a register. For example, if a write one to clear register contained 0xFFFFFFFF and the value 0x00800000 was written into it, the contents of the register would become 0xFF7FFFFFFF. (In 4.3 this method is spelled out as R/C.)

3.4 Conventions

- Hexadecimal numbers in this specification are represented by the notation 0x####, where each “#” is an integer from 0–9 or a letter from A–F.
- Binary numbers in this specification are represented by a sequence of 1s and 0s with subscript 2 (e.g., 10 = 0xA = 1010₂).
- All other numbers in the specification are decimal numbers.
- Superscript (e.g., 2³²) indicates “to the power of.”
- All register names and bit names are uppercase.
- Futurebus+ is little endian.
- CSRs are big endian.
- For all CSRs in this specification, the bit position of each data bit when reading or writing the CSR from the Futurebus+ may be determined by overlaying Figure 3-2 on the CSR figure.

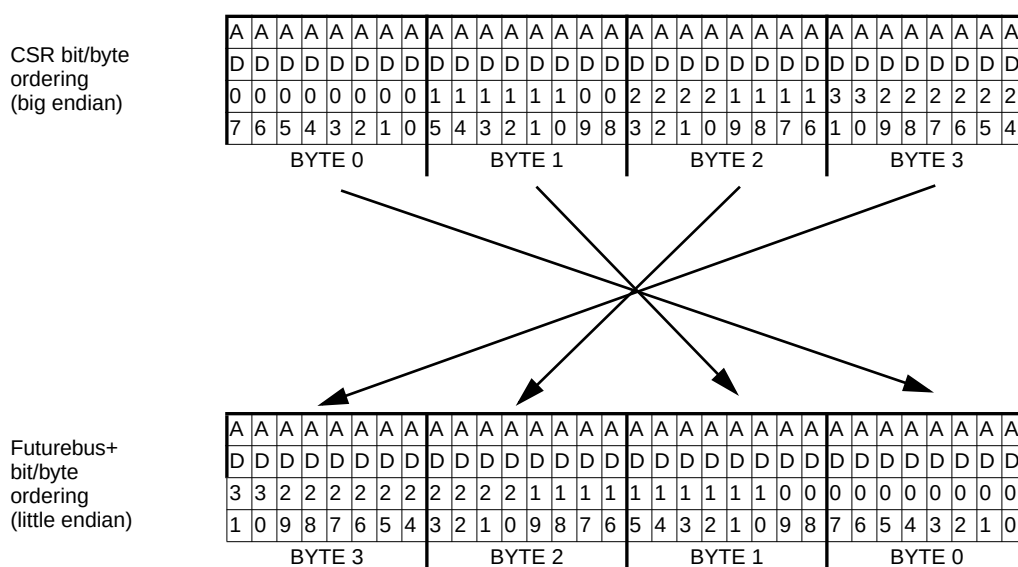


Figure 3-2—CSR bit positions

3.5 Syntax

In Clause 9, fields within the CSRs are denoted in uppercase, bold notation using the following convention:

OWNING-MODULE.CSR-NAME.CSR-FIELD where the optional **OWNING-MODULE** attributes may be one of MASTER, if the MASTER attribute is validly true in the context; or SLAVE, if the MASTER attribute is validly false in the context; or MODULE, if the MASTER attribute is irrelevant in the context. **CSR-NAME** is a valid CSR. **CSR-FIELD** is defined as contiguous bits within **CSR-NAME**.

Symbols used in equations obey the syntax and precedence noted in 2.4 of ISO/IEC 10857: 1994. In addition, the change in the state of fields within CSRs is noted with the following convention:

OWNING-MODULE.CSR-NAME.CSR-FIELD<-new-state where the left-hand side of the expression denotes the field that shall be updated with the right-hand side of the expression as a result of execution of the expression.

3.6 Futurebus+ logo

The 896 and 896.10 logos are copyrighted by the IEEE Computer Society and shall only be used on products that fully conform to all mandatory requirements of an IEEE approved 896 profile document, adhere to ISO/IEC 10857: 1994 or IEEE Std 896.2-1991. The 896.10 logo shall only be used on products that fully conform to all mandatory requirements of this document. Figure 3-3 shows the 896.10 logo; Figure 3-4 shows the 896 logo.



Figure 3-3—896.10 logo



Figure 3-4—896 logo

4. Profile S reference specification

4.1 Introduction

This clause specifies those aspects of a class of modules, backplanes, and chassis required to guarantee interoperability at the physical (mechanical, electrical, and environmental) layer. This profile also specifies the logical layer and guarantees interoperability between modules with common features. Two form factors are specified for Profile S modules, SEM-E Stretch and 10 SU, as shown in Table 4-1. SEM-E Stretch modules are further subdivided between two connector sizes. The major difference between the three form factor options is the number of pins available in each backplane connector. The logical layers are almost identical: the few differences that occur in module and backplane address and data path width are pinout related. All form factors support a 32-bit bus width. Higher-Performance bus widths are available based on the number of connector pins.

Table 4-1—Space profile form factors

Form factor	Standard	Connector pins	Bus width
SEM-E Stretch	IEEE Std 1101.7-1995	396	32, (64)
SEM-E Stretch	IEEE Std 1101.7-1995	372	32, (64)
10 SU	IEEE Std 1101.3-1993	556	32, (64), (128)

SEM-E Stretch and 10 SU modules may be designed to work in other profile systems that utilize similar logical and the same physical (mechanical and electrical) definitions. When designing a module that is compliant with multiple profiles, one must carefully compare the logical functions.

All Profile S modules are fault tolerant as specified by IEEE Std 896.9-1994. However, this profile specifies three levels of fault tolerance centered around their recovery latency for handling hard errors detected on the bus, as shown in Table 4-2. A subset of the first two levels is also specified (Level 0) that matches current silicon and is more fault tolerant than ISO/IEC 10857: 1994 and IEEE Std 896.2-1991 yet not as complete as the levels specified herein.

Table 4-2—Space profile fault tolerance levels

Fault tolerance level	Bus error removal	Hard error replacement
Single Futurebus+	Detect, Retry	Entire subsystem
Dual Futurebus+	Detect, Retry, Swap Bus	Entire bus
Error Correcting Futurebus+	Detect, Correct, Retry, Swap Signal	Signal

The Single Futurebus+ will detect all single point errors and support retry in order to remove transient bus errors. If the bus is found to have a hard error, then that bus subsystem is either no longer capable of using the Futurebus+ or must use it in a degraded mode.

The Dual Futurebus+ level provides two identical Futurebus+ buses on the same physical backplane. One is normally active at a time although the use of both simultaneously is also permitted. If the detection and retry is unsuccessful in removing an error from a bus, the subsystem (through its monarch or through some form

of distributed decision-making capability) may decide to switch to the second bus. As in the Single Futurebus+, either bus could be utilized with errors in various degraded modes depending on the types of hard errors encountered.

The Error Correcting Futurebus+ corrects all single errors immediately upon detection. If an error continues to be detected after the initial correction, then it may be removed by sparing the pin in error as indicated by the syndrome bits. As in the case of a Dual Futurebus+, this sparing shall be directed by the Bus Monarch or by some form of distributed decision-making among bus masters.

Table 4-3 shows the relationships between the three fault tolerance variations, bus widths, and form factors.

Table 4-3—Space profile bus widths

Bus width	Single Futurebus+	Dual Futurebus+	Error Correcting Futurebus+
32 bits	SEM-ES/10 SU	SEM-ES/10 SU	SEM-ES/10 SU
64 bits	SEM-ES/10 SU	10 SU	10 SU
128 bits	10 SU	Not supported	Not supported
256 bits	Not supported	Not supported	Not supported

The four variants of the three levels of fault tolerance have been combined into a single footprint for each form factor. However, for a given subsystem backpanel, only one of these fault tolerance levels may be implemented. The Profile S Futurebus+ signal set common to all configurations is designated FB+1. The Profile S Futurebus+ signal set providing the second bus on a given backpanel is designated FB+2. The Profile S Futurebus+ signal set providing the error correcting bus on a given backpanel is designated as the normal signals, with A and B copies.

The Profile S Serial Bus signal set common to all configurations is designated SB-1. The Profile S Serial Bus signal set providing the second bus on a given backplane is designated SB-2.

All Profile S modules shall provide signals in addition to the base Futurebus+. These include various utility signals and one or two Serial Buses. The Serial Buses provide an alternate access path in the event of an error as well as a diagnostic and test interface. One set of functional utility and Serial Bus signals accompanies a Single Futurebus+. A second set accompanies the Dual Futurebus+. In the Error Correcting configuration, the functional utility signals and the Serial Bus signals are simply duplicated due to their non-functional usage. Only one copy of the test signals is present in all configurations.

Profile S User I/O should be through the backplane. Since most I/O in spacecraft is point to point and not bused between backpanel modules, and due to the large number of unique I/O solutions in existing space systems, this profile has not attempted to lay out all possible options for I/O. In cases when a bus such as MIL-STD-1553B is identified as having widespread use throughout the space community, it will be defined as an option in this profile. Otherwise I/O are left to the discretion of the module and system designers with the caveat that I/O modules must be interchangeable for a class of Profile S modules as defined in Table 4-4. Each class will have its own unique keying as defined in 10.3.5, 11.3.5, and 17.3.5. There are 44 possible module types with the options shown in Table 4-4. It is strongly suggested that implementors concentrate on a subset of module types to maximize the interchangeability of solutions and use new types only when absolutely necessary.

ISO/IEC 10857: 1994 and IEEE Std 896.2-1991 incorporate features required for fault tolerant, real-time, secure, testable, reliable, and maintainable systems. IEEE Std 896.3-1993 and IEEE Std 896.9-1994 provide

additional guidelines and insight into the implementation of these features. Many of these features are required to complete a well-defined system. When implementing a specific feature included in IEEE Std 896.3-1993 and not overridden in IEEE Std 896.9-1994 or this document, that feature shall be implemented as defined in IEEE Std 896.3-1993.

Table 4-4—Profile S module classes

Profile S class	Form factor	Fault tolerance	Width (bits)	Special functions
0–15	SEM-E Stretch (either connector)	Single Futurebus+ Dual Futurebus+ Error Correcting Futurebus+	32 or 64 32 32	None, 1553B, central arbiter, or both
100–127	10 SU	Single Futurebus+ Dual Futurebus+ Error Correcting Futurebus+	32, 64, or 128 32 or 64 32 or 64	None, 1553B, central arbiter, or both

4.2 Target applications

Profile S is specifically designed for spaceborne applications; hence, modules are strictly conduction cooled. Minimum size, weight, and power as well as the ability to tolerate harsh radiation, shock, and vibration environments are primary concerns for implementors of spaceborne computing systems. Logical functionality is specified to ensure interoperability.

4.3 Reference tables

Table 4-5 lists the features of the SEM-E Stretch and 10 SU logical layer and indicates whether the referenced specification is required (mandatory) or optional. Table 4-6 lists the physical features of SEM-E Stretch and 10 SU in a similar manner. Fault Tolerant requirements not part of the fault tolerant subset (Level 0) are indicated by “(L0-opt)”.

Table 4-5—Profile S logical layer requirements

Function	Single Futurebus+	Dual Futurebus+	Error Correcting Futurebus+
Arbitration^a Central/Distributed Central Arbiter	Required Optional	Required Optional	Required Optional
Arbitrated Messages	Optional	Optional	Optional
SEM-E Stretch form factor module address width			
FB+ 1 32	Required	Required	Required
64	Optional	Not supported	Not supported
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Not supported	Not supported
10 SU form factor module address width			
FB+ 1 32	Required	Required	Required
64	Optional	Optional	Optional
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Optional	Not supported

Table 4-5—Profile S logical layer requirements (*continued*)

Function	Single Futurebus+	Dual Futurebus+	Error Correcting Futurebus+
SEM-E Stretch form factor module data path width			
FB+ 1 32	Required	Required	Required
64	Optional	Not supported	Not supported
128	Not supported	Not supported	Not supported
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Not supported	Not supported
128	Not supported	Not supported	Not supported
10 SU form factor module data path width			
FB+ 1 32	Required	Required	Required
64	Optional	Optional	Optional
128	Optional	Not supported	Not supported
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Optional	Not supported
128	Not supported	Not supported	Not supported
SEM-E Stretch form factor backplane width			
FB+ 1 32	Required	Required	Required
64	Optional	Not supported	Not supported
128	Not supported	Not supported	Not supported
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Not supported	Not supported
128	Not supported	Not supported	Not supported
256	Not supported	Not supported	Not supported
10 SU form factor backplane width			
FB+ 1 32	Required	Required	Required
64	Optional	Optional	Optional
128	Optional	Not supported	Not supported
FB+ 2 32	Not supported	Required	Not supported
64	Not supported	Optional	Not supported
128	Not supported	Not supported	Not supported
256	Not supported	Not supported	Not supported
Triplicated control lines	Not supported	Not supported	Required
Split transactions			
Requesters	Optional	Optional	Optional
Responders	Optional	Optional	Optional
Locked operation	Optional	Optional	Optional
Lock commands			
Mask & Swap	Required ^b	Required ^b	Required ^b
Compare & Swap	Required ^b	Required ^b	Required ^b
Fetch & Add Big	Optional	Optional	Optional
Fetch & Add Little	Optional	Optional	Optional
Broadcast			
Masters	Optional	Optional	Optional
Slaves	Optional	Optional	Optional
Broadcall			
Masters	Not supported	Not supported	Not supported
Slaves	Not supported	Not supported	Not supported
Cache coherence	Optional	Optional	Optional

Table 4-5—Profile S logical layer requirements (continued)

Function	Single Futurebus+	Dual Futurebus+	Error Correcting Futurebus+
Message passing	Optional	Optional	Optional
Packet mode	Optional	Optional	Optional
Transaction types^c			
Read Unlocked	Required	Required	Required
Write Unlocked	Required	Required	Required
Address-Only Unlocked	Required	Required	Required
Locked			
Read Locked	Optional	Optional	Optional
Write Locked	Optional	Optional	Optional
Address Only Locked	Optional	Optional	Optional
Read partial	Optional	Optional	Optional
Write partial	Optional	Optional	Optional
Read Partial Locked	Optional	Optional	Optional
Write Partial Locked	Optional	Optional	Optional
Cache Coherent: ^d			
Write Response	Optional*	Optional*	Optional*
Read Response	Optional*	Optional*	Optional*
Write No-acknowledge	Optional*	Optional*	Optional*
Read Invalid	Optional*	Optional*	Optional*
Write Invalid	Optional*	Optional*	Optional*
Read Shared	Optional*	Optional*	Optional*
Copyback	Optional*	Optional*	Optional*
Read Modified	Optional*	Optional*	Optional*
Invalidate	Optional*	Optional*	Optional*
Shared Response	Optional*	Optional*	Optional*
Modified Response:	Optional*	Optional*	Optional*
Tag bits^e			
Modules	Not supported	Not supported	Not supported
Backplanes	Not supported	Not supported	Not supported
Serial Bus			
Modules			
SB-1	Required	Required	Required
SB-2	Optional	Required	Required
Backplanes			
SB-1	Required	Required	Required
SB-2	Optional	Required	Required
Utility signals			
Modules	Optional	Optional	Optional
Backplanes	Optional	Optional	Optional

Table 4-5—Profile S logical layer requirements (continued)

Function	Single Futurebus+	Dual Futurebus+	Error Correcting Futurebus+
Error detection^f			
BP: Even Parity	Required	Required	Not supported
Error Correcting Code ^g	Not supported	Not supported	Required
CP: Even Parity	Required	Required	Not supported
Error Correcting Code	Not supported	Not supported	Required
ABP: Even Parity	Optional	Optional	Not supported
Error Correcting Code	Not supported	Not supported	Optional
Protocol Monitoring ^h	Required (L0-opt)		
Parallel Protocol Handshake	Optional (L0-opt)		
Arbitrated Messages	Optional (L0-opt)	Required (L0-opt)	Required
Synchronization	Required (L0-opt)	Optional (L0-opt)	Optional
Preemption	Required (L0-opt)	Optional (L0-opt)	Optional
Request and Grant	Required (L0-opt)	Required (L0-opt)	Required
End of Tenure	Not supported	Required (L0-opt)	Required
GC: Error Correcting Code	Required	Required (L0-opt)	Required
Geographical Addr Parity ⁱ		Not supported	Required
Duplicated Signals ^j		Required	Required
TED	Required (L0-opt)	Required (L0-opt)	Required
SLD	Required (L0-opt)	Required (L0-opt)	Required
RED	Required (L0-opt)	Required (L0-opt)	Required
Reflected Protection ^k			
CM	Required (L0-opt)	Required (L0-opt)	Required
ST	Required (L0-opt)	Required (L0-opt)	Required
RF	Required (L0-opt)	Required (L0-opt)	Required
Timeout: ^l			
Transaction	Required (L0-opt)	Required (L0-opt)	Required
Split Transaction ^m	Optional (L0-opt)	Optional (L0-opt)	Optional
Arbitration	Required (L0-opt)	Required (L0-opt)	Required
Central Arbiter	Required (L0-opt)	Required (L0-opt)	Required
Distributed Arbitration	Optional (L0-opt)	Optional (L0-opt)	Optional
Bus Activity	Required (L0-opt)	Required (L0-opt)	Required
Error isolation, logging, and diagnostics			
Logging all detected errors ⁿ	Required (L0-opt)	Required (L0-opt)	Required
Enable/Disable transceivers	Required (L0-opt)	Required (L0-opt)	Required
MAINT_MONITORS	Required (L0-opt)	Required (L0-opt)	Required
Diagnostic data registers	Required (L0-opt)	Required (L0-opt)	Required
Critical reg write protection	Required (L0-opt)	Required (L0-opt)	Required
Error removal and recovery			
Hardware Retry ^o	Required (L0-opt)	Required (L0-opt)	Required
Software Retry ^p	Optional	Optional	Optional
Spare Lines ^q	Not supported	Not supported	Optional
Dual Buses	Not supported	Required	Not supported
CSRs per Clause 9	Required	Required	Required

^aArbitration is not necessary for slave-only modules.

^bIf Locked Operation is implemented, these lock commands are required.

^cCSRs must be accessible while waiting for a split response, i.e., CSR access should not return busy.

^d* —For Cache Coherent Implementations.

^eNo pinout location defined in this profile.

^fIf Distributed Arbitration is used, ABP, arbitrated message, distributed arbitration timeout, and synchronization options are required except those marked “L0-opt”.

^gError correcting codes are defined in this profile.

^hProtocol monitoring is defined in IEEE Std 896.9-1994.

ⁱGeographical Address Parity is defined in IEEE Std 896.9-1994.

^jDuplicated Signals are defined in IEEE Std 896.9-1994.

^kReflected protection is defined in IEEE Std 896.9-1994.

^lTimeouts are defined in ISO/IEC 10857: 1994, IEEE Std 896.2-1991, and IEEE Std 896.9-1994.

^mIf split transactions are used, timeout is also required.

ⁿDefined in IEEE Std 896.9-1994.

^oHardware Retry is defined in IEEE Std 896.9-1994.

^pSoftware Retry is defined in IEEE Std 896.2-1991.

^qSpare lines and dual buses are defined in this profile.

4.4 Profile S interoperability

SEM-E Stretch and 10 SU modules may be designed to work in other profile systems that utilize similar logical and the same physical (mechanical and electrical) definitions. When designing a module that is to be compliant with multiple profiles, one must carefully compare the logical functions.

Table 4-6—Profile S physical layer requirements

Function	10 SU	SEM-E Stretch	Mandate
Module Height Depth	IEEE Std 1101.3-1993 215 mm (8.465 in) 162 mm (6.375 in)	IEEE Std 1101.7-1995 150 mm (5.880 in) 218 mm (8.570 in)	Required Required
Module pitch Slot width	IEEE Std 1101.3-1993 20 mm (0.787 in)	IEEE Std 1101.7-1995 19 mm (0.750 in)	Required
Connector type^a Power Signal Keying	IEEE Std 1101.3-1993	IEEE Std 1101.7-1995	Required Required Required
Module connector pinout^b Single Futurebus+ Dual Futurebus+ Error Correcting Futurebus+ Digital Power IEEE 1394 Serial Bus Utility signals MIL STD-1553B Local interconnects User-Defined I/O Fiber Central arbitration Distributed arbitration 28 V	IEEE Std 896.10-1997	IEEE Std 896.10-1997	Required Optional Optional Required Required Required Optional Optional Optional Optional Required Optional Optional ^c
Backplane connector pinouts Single Futurebus+ Dual Futurebus+ Error Correcting Futurebus+ Digital Power IEEE 1394 Serial Bus Utility MIL-STD-1553B Local interconnects User-Defined I/O Fiber Central arbitration Distributed arbitration 28 V	IEEE Std 896.10-1997	IEEE Std 896.10-1997	Required Optional Optional Required Required Required Required Optional Optional Optional Optional Required Optional ^d Required

Table 4-6—Profile S physical layer requirements (continued)

[illegible]

Table 4-6—Profile S physical layer requirements (continued)

Function	10 SU	SEM-E Stretch	Mandate
Module voltages and powerⁱ 5 V 3.3 V 2.1 V 28 V	IEEE Std 896.10-1997	IEEE Std 896.10-1997	Optional Optional Optional Optional
Backplane voltages and powerⁱ 5 V 3.3 V 2.1 V 28 V	IEEE Std 896.10-1997	IEEE Std 896.10-1997	Required Required Required Required
Live insertion	IEEE Std 896.2-1991	IEEE Std 896.2-1991	Not supported

^a1 A max. per pin.

^bUser shall implement exactly one of the following: A Single Futurebus+, a Dual Futurebus+, or an Error Correcting Futurebus+.

^cPins reserved.

^dRecommended.

^eGoal: 5 pF.

^fFolded backpanel.

^g15 pF max. load.

^hConnector Sections C through H.

ⁱProduct specification shall state voltages and currents required under all specified environmental conditions.

4.5 Conformance testing

The conformance test requirements of IEEE Std 896.4-1993 shall be used for the conformance test requirements of this standard wherever applicable.

IEEE Std 896.4-1993 defines the conformance test requirements for the protocols identified in ISO/IEC 10857: 1994 and IEEE Std 896.2-1991 for Futurebus+ modules, backplanes, and systems. It does not address any of the extensions in the IEEE Std 896.9-1994 Fault Tolerance Extensions or those in this standard. Similar conformance testing to cover these extensions should be used.

5. Detailed specification, Futurebus+ logical layer

This clause provides the detailed explanations and requirements of the Futurebus+ logical layer summarized in Table 4-5.

5.1 Arbitration

Arbitration applies only to modules that are not slave-only.

5.1.1 Central arbitration

All Profile S modules and backplanes that are not slave-only shall support the central arbitration mechanism specified in Clauses 4 and 5 of ISO/IEC 10857: 1994. Modules may implement up to 256 different priority levels. A Profile S backplane may contain wiring for one or more central arbiters, but is only required to reserve the central arbitration pins at the module level. A Profile S system is not required to contain a central arbiter.

Modules that modify their arbitration priority “on the fly” via arbitrated messages shall do so in a way that does not delay their ability to arbitrate. Use of any arbitrated messages is optional.

5.1.2 Distributed arbitration

In some applications, distributed arbitration is preferred. Distributed arbitration as defined in Clause 5 of ISO/IEC 10857: 1994 may be implemented optionally. The mechanism for switching between central and distributed arbitration is controlled by enabling and disabling the arbitration control registers in the CSRs and/or via arbitrated messages.

At initialization, systems that have a central arbiter shall assert the preemption signal (**PE***), otherwise arbitration is distributed. Systems capable of both central and distributed arbitration may be configured by the monarch during the configuration process or at run time to use either arbitration mechanism. System integrators shall ensure only one mechanism is employed at a time.

5.2 Parallel protocol

Clause 6 of ISO/IEC 10857: 1994 specifies the Futurebus+ parallel protocol. The protocol supports 32 or 64 bits of address and 32, 64, 128, or 256 bits of data. SEM-E Stretch modules shall implement 32-bit addressing and may optionally implement 64-bit addressing. SEM-E Stretch modules shall implement 32-bit data capability and may optionally implement 64-bit data capability. One hundred twenty-eight or 256 bit data widths are not supported by SEM-E Stretch modules. 10 SU modules shall implement 32-bit addressing and may optionally implement 64-bit addressing. All 10 SU modules shall implement 32-bit data capability and may optionally implement 64-bit or 128-bit data capability. Two hundred fifty-six bit data widths are not supported by 10 SU modules.

SEM-E Stretch and 10 SU modules shall declare their supported bus widths in the **MODULE_LOGICAL_CAPABILITY** register as defined in ISO/IEC 10857: 1994, 7.1.2.1.1. For compatibility with other profiles and each other, SEM-E Stretch and 10 SU modules shall default to 32-bit data path width.

SEM-E Stretch backplanes shall be bused and terminated for data path widths of 32-bits. SEM-E Stretch 64-bit backplanes are optional for a Single Futurebus+ system. 10 SU backplanes shall be bused and terminated for data path widths of 64-bits. 10 SU 128-bit backplanes are optional for a Single Futurebus+ system.

Clause 2 of ISO/IEC 10857: 1994 describes the Futurebus+ signal requirements. Conformance to this profile requires implementation of the full signal set (i.e., signals listed in Table 2-1 of ISO/IEC 10857: 1994) with the exception of Tag Bits and Tag Parity.

5.2.1 Transaction types

Profile S modules can support any combination of the parallel transaction types needed for a specific module's functions. A module shall assert beat error (**BE***) and record a capability error in the CSR error register when specifically addressed with a transaction for which it has no capability.

Following is the list of possible transactions:

- Read Unlocked
- Write Unlocked
- Address Only Unlocked
- Read Locked
- Write Locked
- Address Only Locked
- Read Partial
- Write Partial
- Read Partial Locked
- Write Partial Locked
- Write Response
- Read Response
- Write No Acknowledge
- Read Invalid
- Write Invalid
- Read Shared
- Copyback
- Read Modified
- Invalidate
- Shared Response
- Modified Response

Use of busy: Modules should use **busy** status only to indicate that resources are full or in use. A module shall not assert **busy** to postpone a read response. It should limit the number of outstanding read requests so its return-data buffer space cannot overflow.

Use of splits: Transactions may be split. Split transaction responses shall not be split. All CSRs shall be accessible while waiting for a split response. Splits shall not be initiated by masters.

Packet mode: Profile S modules may optionally use packet mode.

5.2.2 Cache coherence

Cache coherence is an option for Profile S modules. If a module is designed to support cache coherence, it may operate in 32-bit data width or 64-bit data width. Modules that do not support cache transactions shall lie outside the cache coherence domain of the system and not take part in transactions involving cache coherence protocols. All Profile S modules that implement backplane cache coherence shall implement the full cache coherence protocols and all relevant transaction types. System implementors are responsible for correct interoperability between the modules within and outside the cache coherence domain. Only one cache line width should be used per cache domain.

5.2.3 Message passing

Message passing is optional for Profile S modules and systems. If modules with and without message passing are mixed in a system, modules not supporting message passing shall not take part in transactions involving message passing protocols. System implementors are responsible for correct interoperation between modules with and without message passing capability.

5.2.4 Tag bits

Profile S modules are not required to implement, use, or respond to Tag Bits or Tag Parity. Profile S systems or modules shall not rely on Tag Bits for correct operation.

NOTE—If Tag Bits are used, their function may be incompatible with that of the same signals from another module that is still compliant with Profile S. System implementors should ensure that modules employ Tag Bits in a compatible and interoperable manner.

6. Utility Signals

Utility Signals are discrete lines used to distribute information where there is a unique requirement that cannot be satisfied by use of the Futurebus+ or the Serial Bus. Examples include low latency requirements or support for special radiation hardened hardware. Reservation of the pin assignments of the Utility Signals is required and, if the functions are implemented, they shall be implemented in accordance with this standard.

Functional Utility Signals are duplicated in the Dual Futurebus+ and triplicated in the Error Correcting configurations. Test Utility signals are not duplicated in any configuration. The Utility Signals are specified below.

6.1 Trigger 0 (TR0*) and Trigger 1 (TR1*)

Trigger 0 (**TR0***) and Trigger 1 (**TR1***) are discretes that may be programmed to be driven by any module(s). Any module may monitor these triggers. The action that a module takes upon receipt of a trigger signal is programmed via the Futurebus+ (**TR0*** only) or the Serial Bus (**TR0*** and **TR1***). Typical functions accomplished with a trigger are stopping or starting a collection of processors in a precise time-synchronized fashion and relating events in different modules by triggering the trace stack resources.

TR0* and **TR1*** signals shall be active low for a minimum of 50 ns.

TR0* and **TR1*** are not duplicated in the Dual Futurebus+ or the Error Correcting configurations.

6.2 Run (RUN*)

Run (**RUN***) is a discrete signal that shall cause the processor to go to a run mode only when it transitions from an inactive to an active level. Holding the **RUN*** signal at an active level shall not keep the processor in run mode if a Stop is generated during this time. The **RUN*** discrete has the same effect as a write to the **CONTINUE_EXECUTION** bit in the **UCIF_CONTROL** register.

RUN* shall be active low for a minimum of 50 ns.

RUN* is not duplicated in the Dual Futurebus+ or the Error Correcting configurations.

6.3 Nuclear Event Detect (NED*)

The assertion of **NED*** indicates that a nuclear event has occurred and that circumvention of TREE (Transient Radiation and its Effects on Electronics) should be initiated. A nuclear event may be natural (solar flare) or man-made. It is intended that the circumvention be accomplished by (1) draining all power supplies and other modules, if necessary, (2) having the Nuclear Event Detection module initiate hardened timers and set hardened CSRs for the purpose of restarting power, (3) restarting the system hardware, (4) restarting the system software, and (5) distinguishing between power and input/output glitches that are associated with the nuclear event. In systems implementing **NED*** shutdown, system start-up software should poll a hardened standard unit address to determine if the start-up is due to recovery from an **NED*** or for other reasons. The location of these various hardened elements is user-configurable and beyond the scope of this standard. If any elements of the TREE circumvention are on a Profile S module, it is expected they will use the **NED*** signal in their operation. Typically non-TREE circumvention logic on Profile S modules may use an active **NED*** to disable all outputs or hold all logic in reset.

NED* shall be an active low signal. **NED*** shall be asserted only by a Nuclear Event Detection module for a minimum of 50 ns. It shall use TTL levels instead of FB+ to enable it to be driven from a source that may not have access to FB+ transceivers.

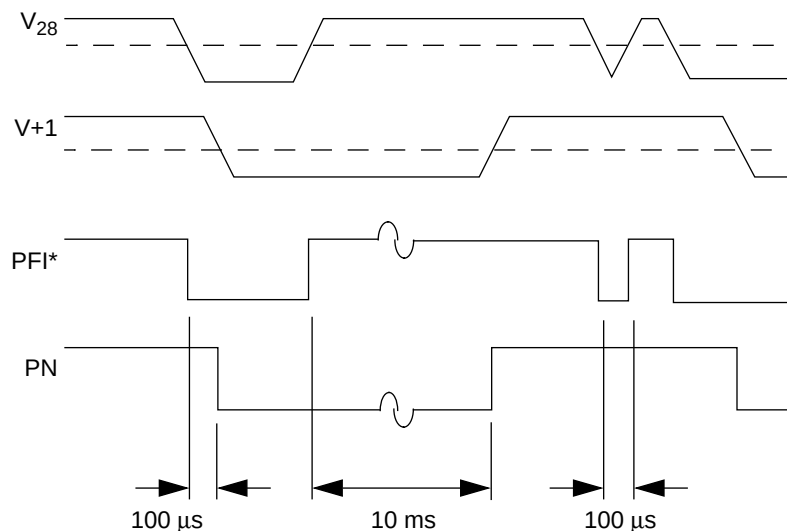
All logic associated with detecting, generating, and distributing this signal in a Nuclear Event Detection module should be hardened against radiation upset to the expected level of the Nuclear Event (most likely level IVa of IEEE Std 1156.4-1997), otherwise this signal may be invalid.

NED* is duplicated in the Dual Futurebus+ configuration. These signals are designated **1-NED*** and **2-NED***. When the Dual Futurebus+ is used, the module shall be configurable to accept either or both **1-NED*** or **2-NED***, depending upon which bus is active.

NED* is triple modular redundant in the Error Correcting configuration. These signals are designated **NED***, **NEDA***, and **NEDB***. At least two of these three signals must be active to indicate that a nuclear event has occurred.

6.4 Power Failure Imminent (PFI*)

The assertion of **PFI*** indicates that input power to the power supply or to the power conditioning unit is not within acceptable ranges and that the power supply output may consequently fall out of regulation (see Figure 6-1). Regulated power supply output voltage at rated current shall be available for at least 100 μ s after the assertion of **PFI***. **PFI*** shall be released upon recovery of nominal power supply input power, provided that **PFI*** has been asserted for a minimum of 100 μ s. **PFI*** shall not be asserted for less than 100 μ s. It shall use TTL levels instead of FB+ to enable it to be driven from a power supply that may not have access to FB+ transceivers.



NOTE—Dotted line represents the power supply output threshold voltage.

Figure 6-1—Power supply monitoring signal

PFI* shall be an active low signal that is usually asserted by the power supply.

PFI* is duplicated in the Dual Futurebus+ configuration. These signals are designated **1-PFI*** and **2-PFI***. When the Dual Futurebus+ is used, the module shall be configurable to accept either or both **1-PFI*** or **2-PFI***, depending upon which bus is active.

PFI* is triple modular redundant in the Error Correcting configuration. These signals are designated **PFI***, **PFIA***, and **PFIB***. At least two of these three signals must be active for a module to recognize **PFI*** as active.

6.5 Power Normal (PN)

This discrete is an active high wire or signal. It shall use TTL levels instead of FB+ to enable it to be driven from a power supply that may not have access to FB+ transceivers. Power Normal is active when dc power has stabilized and stays within the voltage tolerances (see Figure 6-1). Power Normal is inactive while dc power is coming up and until dc power has stabilized for at least 10 ms. At that time, the Power Normal signal shall become active. The Power Normal signal shall remain active until a voltage-out-of-tolerance condition occurs and that causes Power Normal to go inactive. Power Normal shall provide the Futurebus+ modules with a constant reset while power is coming up and holds a constant reset while power is going down.

The Power Normal discrete is active high and is normally generated by the power supply that monitors the dc regulators and the input power. If no regulated power is supplied on a backplane, **PN** shall remain inactive.

PN is duplicated in the Dual Futurebus+ configuration. These signals are designated **1-PN** and **2-PN**. When the Dual Futurebus+ is used, the module shall be configurable to accept either or both **1-PN** or **2-PN**, depending upon which bus is active.

PN is triple modular redundant in the Error Correcting configuration. These signals are designated **PN**, **PNA**, and **PNB**. At least two of these three signals must be active for a module to recognize **PN** as active.

6.6 System Reset (SR*)

The System Reset signal provides a way to tell a module to run an abbreviated reset instead of a full reset such as after the receipt of an active Power Normal (**PN**). It shall use TTL levels instead of FB+ to enable it to be driven from a power supply or other source that may not have access to FB+ transceivers. If System Reset is desired, the System Reset discrete shall be held active for a minimum of 10 μ s.

SR* shall be active low.

SR* is duplicated in the Dual Futurebus+ configuration. These signals are designated **1-SR*** and **2-SR***. When the Dual Futurebus+ configuration is used, the module shall be configurable to accept either or both **1-SR*** or **2-SR***, depending upon which bus is active.

SR* is triple modular redundant in the Error Correcting configuration. These signals are designated **SR***, **SRA***, and **SRB***. At least two of the these three signals must be active for a module to recognize **SR*** as active.

6.7 Utility Spares (UPSP* and UWSP)

Two spare pins are provided in the Error Correcting configuration to cover the following utility signals: **NED***, **NEDA***, **NEDB***, **PN**, **PNA**, **PNB**, **PFI***, **PFIA***, **PFIB***, **SR***, **SRA***, and **SRB***. **UPSP*** covers the push-pull TTL signals (**NED***, **NEDA***, **NEDB***, **PFI***, **PFIA***, **PFIB***, **SR***, **SRA***, and **SRB***). **UWSP** covers the wire-OR TTL signals (**PN**, **PNA**, and **PNB**). These pins are selected to replace an active line using the **SPARE_UTILITY_PP** and **SPARE_UTILITY_WW** bits in **SP_ENABLE_SPARES_I** CSR 680. **UPSP*** and **UWSP** are implemented only for Error Correcting configurations.

7. Fault tolerance aspects of the parallel protocol and utility signals

A common requirement for any space system is some level of fault tolerance. This is usually allocated from the top down as the system is designed and defined. Since fault tolerance is but one aspect of a system that is traded at each level of the hierarchy, by the time one reaches the backplane bus such as Futurebus+, allocations and strategies may already be in place to handle errors at a higher level. On the other hand, recovery latency requirements may be such that an error must be removed rapidly and thus require some form of error removal at the bus level. Profile S provides the bus with mechanisms to support a spectrum of fault tolerance applications from simply detecting and logging errors to operating through errors. An application designer must decide which is appropriate for his or her system and then specify and implement that level of fault tolerance. See Clause 1 of IEEE Std 896.9-1994 for a detailed discussion of the various fault tolerance applications supported by this profile. A subset of this capability of the Single Futurebus+ and Dual Futurebus+ configuration named Level 0 is allowed. These are marked throughout this clause.

7.1 Error detection

All of the aforementioned fault tolerance applications rely on near-100% detection of errors in order to know when an error has occurred so that appropriate action can be taken. A system that does not detect its errors cannot be termed fault tolerant of those errors. IEEE Std 896.9-1994 guarantees near-100% detection of errors on a Futurebus+ and is used as the basis for this document's fault tolerance applications. The following subclauses describe the Futurebus+ error detection mechanisms that are part of one or more fault tolerance applications.

7.1.1 Even parity

In all fault tolerance applications except the Error Correcting configuration, Profile S modules shall use parity to protect the Address/Data lines (**AD[31..0]***, **AD[63..0]***, or **AD[127..0]***) and Command lines (**CM[7..0]***) as defined in IEEE Std 896.9-1994.

In all fault tolerance applications implementing distributed arbitration except the Error Correcting configuration, Profile S modules shall use even parity to protect the arbitrated message lines (**AB[7..0]***) as defined in IEEE Std 896.9-1994.

All Profile S modules shall use even parity to protect the Geographic Address (**GA[5..0]***) as defined in IEEE Std 896.9-1994.

All Profile S Error Correcting configuration modules shall use even parity to protect the Bus Grant Code lines (**GC[5..0]***).

7.1.2 Error correcting codes

The error correction codes and techniques used provide for the detection of all single line errors (as defined in 7.1.3 and 7.1.4) and their correction with no software intervention. The techniques and codes used provide at least single error detection and correction for each of the lines that make up Futurebus+.

These techniques assume that only one bit covered by a checker can fail and be corrected. These techniques place no additional electrical requirements on the Futurebus+ specification. To ensure that there are no single points of failure from the input of the bus driver to the output of the bus receiver, partitioning to driver and receiver should be made such that a single enable line controls only one driver or receiver in a check group. All mechanisms are initialized in the disabled state. Each mechanism is enabled by its corresponding CSR. Upon detection of an error, bits in the corresponding CSR are set indicating failure type and syndrome of the error. The error CSRs can be cleared by the module software.

All Error Correcting configuration Profile S modules shall use error correcting codes to protect the 32-bit Address/Data **AD[31..0]*** with the Bus Code lines **BC[7..0]***, and the Command lines **CM[7..0]*** with the Command Code lines **CC[4..0]***.

Additionally, if a 64-bit Address/Data module is implemented, the module shall use the Bus Code lines **BC[7..0]*** to protect **AD[63..0]*** (10 SU only).

If distributed arbitration is implemented in an Error Correcting configuration Profile S module, the Arbitrated Message **AB[7..0]*** shall be protected by the Arbitrated Bus Code **ABC[4..0]***.

7.1.2.1 Sixty-Four-Bit SEC/DED/SBD

To provide single fault detection and correction for data and detection of b-bit symbol faults (b is the variable length of the symbol detection capability), single error correction, and double error detection with single b-bit symbol detection (SEC-DED-SbBD) codes, are used as the standard methods. The (72,64) code has been widely used throughout industry since the number of check bits required equal the number of parity bits typically used for this size data structure. The (72,64) code used here also provides detection of 3-bit byte or 4-bit byte failures. In addition, this code seems to provide the optimal balance between signal lines and internal logic complexity. For this reason, even with wider data structures, the code is generated on 8-byte boundaries.

The parity check matrix is given by Tables 7-1 and 7-2. Correctable and uncorrectable errors shall be indicated by storing a code in the corresponding CSR (**SP_TMR_HOLD_I** (CSR 456), or **SP_TMR_HOLD_II** (CSR 460)) for each type of failure. The syndrome of the failure shall be stored in a CSR (**SP_ECC_HOLD** (CSR 452)).

This code does not detect all triple or four bit errors, only those that occur within symbols. From the three-bit point of view **D63..D61**, **D60..D58**, ..., **BC2..BC0** should be wired as symbols on a module. All three-bit errors within a symbol will be detected. From the four-bit point of view **D63..D60**, **D59..D56**, ..., **BC3..BC0** should be wired as symbols on a module. All four-bit errors within a symbol will be detected. In a 32- or 64-bit system, it is not possible to get full 8- or 9-bit symbol detection.

Note that Profile S does not support an Error Correcting 128- or 256-bit configurations. In all cases, controls and other common logic should be divided into separate signals at the highest level of full symbol coverage as early as possible to have the highest detection probabilities.

Table 7-1—Parity check matrix of Odd-Weight-Column (72,64) SEC-DED code

Syndrome bit	D63 ... D56	D55 ... D48	D47 ... D40	D39 ... D32	D31 ... D24
7	11111111	11111111	+++1+++1	+++1+++1	++1+++1+
6	1+++1+++	1+++1+++	11111111	11111111	+++1+++1
5	+1+++1++	+1+++1++	1+++1+++	1+++1+++	11111111
4	++1+++1+	++1+++1+	+1+++1++	+1+++1++	1+++1+++
3	+++1+++1	+++1+++1	++11+++1	++11+++1	+111+111
2	++++1111	++++1111	++++1111	++++1111	++++1111
1	+++++++	11111111	+++++++	11111111	+++++++
0	1111++++	++++1111	111++++1	+++1111+	11++++11

Table 7-2—Parity check matrix of Odd-Weight-Column (72,64) SEC-DED code

Syndrome bit	D23 ... D16	D15 ... D08	D07 ... D00	BC7 ... BC0
7	++1+++1+	+++1+++1	+++1+1++	1+++++++
6	+++1+++1	1+++1+++	1+++++1+	+1++++++
5	11111111	+1+++1++	+1+++++1	++1+++++
4	1+++1+++	++1+++1+	++1+1111	+++1++++
3	+111+111	+++++++	++++1111	++++1+++
2	++++1111	11111111	+++++++	+++++1++
1	11111111	1111++++	11111111	++++++1+
0	++1111++	++++1111	1111+111	++++++1

7.1.2.2 Eight-Bit SEC/DED

For the 8-bit wide buses over which Error Correcting configuration (ECC) codes are to be used (such as the arbitrated message bus), a full 5-bit Odd-Weight-Column single error correction/double error detection code shall be used.

A full 5-bit Odd-Weight-Column single error correction/double error detection code will also provide detection of a full 8-bit symbol error over the data or two 4-bit symbol errors over the data, along with a 5-bit symbol error over the ECC code. The parity check matrix is given by Table 7-3. Correctable and uncorrectable errors shall be indicated by storing a code in the corresponding CSR [**SP_TMR_HOLD_I** (CSR 456), or **SP_TMR_HOLD_II** (CSR 460)] for each type of failure. The syndrome of the failure shall be stored in a CSR [**SP_ECC_HOLD** (CSR 452)].

Table 7-3—Parity check matrix of Odd-Weight-Column (13,8) SEC-DED code

Syndrome bit	D07 ... D00	xC4 ... xC0
4	111+1+1+	1++++
3	1111+++1	+1+++
2	++1111++	++1++
1	1+++1111	+++1+
0	+1+1+111	++++1

7.1.3 Triple modular redundant (TMR) lines

Triple modular redundancy is guaranteed to detect all single bit errors on lines that are covered by it. In this method, comparisons are made in groups of two of the three lines by sampling the signals after the end of the glitch filter period and after the skew between lines has settled out. The value agreeing on two of the three lines is assumed to be the correct value for those lines.

Profile S modules that choose to implement the Error Correcting configuration shall protect the following signals through the use of two extra copies of each signal:

- Status **ST[7..0]*** shall be protected with **STA[7..0]*** and **STB[7..0]***.
- Capability **CA[2..0]*** shall be protected with **CAA[2..0]*** and **CAB[2..0]***.
- Address Handshakes **AS***, **AK***, and **AI***, shall be protected with **ASA***, **ASB***, **AKA***, **AKB***, **AIA***, and **AIB***.
- Data Handshakes **DS***, **DK***, and **DI***, shall be protected with **DSA***, **DSB***, **DKA***, **DKB***, **DIA***, and **DIB***.
- End of Tenure **ET*** shall be protected with **ETA*** and **ETB***.
- Reset **RE*** shall be protected with **REA*** and **REB***.
- Requests **RQ[1..0]*** shall be protected with **RQA[1..0]*** and **RQB[1..0]***.
- Preemption **PE*** shall be protected with **PEA*** and **PEB***.
- Nuclear Event Detect **NED*** shall be protected with **NEDA*** and **NEDB***.
- Power Normal **PN** shall be protected with **PNA** and **PNB**.
- Power Fail Interrupt **PFI*** shall be protected with **PFIA*** and **PFIB***.
- Warm Start **SR*** shall be protected with **SRA*** and **SRB***.

Profile S modules that choose to implement the Error Correcting configuration and distributed arbitration shall protect the following signals through the use of two extra copies of each signal.

- Arbitrated Message Sync **AP***, **AQ***, and **AR***, shall be protected with **APA***, **APB***, **AQA***, **AQB***, **ARA***, and **ARB***.
- Arbitration Condition **AC[1..0]*** shall be protected with **ACA[1..0]*** and **ACB[1..0]***.

A 2-bit code shall be set in **SP_TMR_HOLD_I** (CSR 456) or **SP_TMR_HOLD_II** (CSR 460) indicating which bit, if any, was in error.

7.1.3.1 TMR module/system considerations

TMR triples the number of signal lines per control signal over a noncorrected implementation. Although capable of detecting and correcting all single bit failures, this method is guaranteed to select the wrong lines on multiple line errors of a given signal without detecting the failure. Thus the module's software needs to take note of a consistently failing line and spare around it if sparing mechanisms (at the bit or bus level) are available, and/or call for replacement of the component causing the error on the bus if they are not.

Implementing an Error Correcting configuration may cost more in unique silicon, I/O, and reduced performance. It is also important that a module that uses the Error Correcting configuration increase its own fault detection and masking capabilities so that the true value of the corrected data handling may be realized throughout the system.

To ensure proper sampling of the signals covered by TMR, the following relation must be satisfied:

$$t_{\text{total}} = t_{\text{norm}} + t_{\text{comp}} + t_{\text{sel}} + t_{\text{skew}}$$

where

- t_{total} is the total delay that must be accounted for before a signal covered by TMR may be sampled;
- t_{norm} is the maximum delay associated with the signal in a non-TMR implementation;
- t_{comp} is the maximum propagation delay through the comparison logic used in the TMR;
- t_{sel} is the maximum propagation delay through the logic to select one of the three lines;
- t_{skew} is the time difference between the earliest arrival of the fastest TMR line and the latest arrival of the slowest TMR line. Before using any line covered by TMR, this delay must be accounted for.

7.1.3.2 ECC and TMR registers

The following provides a description of the extensions required for an Error Correcting configuration. These are organized by the signals that are covered.

7.1.3.2.1 Address/Data and bus codes

Two bits (1–0), **ECC_ADDR_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), control enabling of ECC for these lines as follows:

- 00 - No ECC - **BC[7..0]*** used for parity as in IEEE Std 896.9-1994. This is the default/reset state.
- 01 - 32-bit mode: **AD[31..0]*** and **BC[7..0]*** are active, **AD[63..32]*** are forced inactive. Together, **AD[63..0]*** and **BC[7..0]*** form one (72,64) code with a format of FFFFFFFFxxxxxxx.
- 11 - 64-bit mode: **AD[63..0]*** and **BC[7..0]*** form one (72,64) code.

Eight bits (7–0) of **BYTE_LANE_IN_ERROR** (CSR 400) are used to store the error syndrome from a 32- or 64-bit bus implementation.

When an error is detected, **BYTE_LANE_IN_ERROR** (CSR 400) shall be frozen and **ECC_TMR_ERROR_DETECTED** shall be set. Writing 1 to clear this bit in **ERROR_HI** (CSR 384) shall unfreeze these syndrome bits.

Bits 6–5, **ECC_72_64_ERROR** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), indicate an uncorrectable error on the (72,64) word. Writing 1 to this bit in **SP_ECC_HOLD** (CSR 452) shall clear this bit.

Bits 7–0, **ECC_BC** of R/W **SP_MAINT_MONITOR_ECC** (CSR 664) (see Figure 9-15), are put on **BC[7..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to a released state.

7.1.3.2.2 Command bits and command codes

Bit 2, **ECC_COMMAND_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of ECC for these lines as follows:

- 0 - No ECC - **CC[0]*** used for parity as in IEEE Std 896.9-1994. This is the default/reset state.
- 1 - **CM[7..0]*** and **CC[4..0]*** form (13,8) code word.

Five bits (20–16), **ECC_COMMAND_SYNDROME** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), are used to store the error syndrome. When an error is detected, **SP_ECC_HOLD** (CSR 452) shall be frozen and **ECC_TMR_ERROR_DETECTED** shall be set. Writing 1 to clear **ECC_TMR_ERROR_DETECTED** in **ERROR_LOW** (CSR 388) shall unfreeze these syndrome bits. The error syndrome bits shall be cleared by writing a 1 into them in **SP_ECC_HOLD** (CSR 452).

Two bits (25–24), **ECC_COMMAND_ERROR** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), indicate the results of the correction. These bits shall be cleared by writing a 1 into them in **SP_ECC_HOLD** (CSR 452).

Bits 18–14, **ECC_CC** of R/W **SP_MAINT_MONITOR_ECC** (CSR 664) (see Figure 9-15), are placed on **CC[4..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to a released state.

7.1.3.2.3 Status and capability lines

Bit 3, **TMR_ST_CAP_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - **ST[7..0]*** and **CA[2..0]*** are used as single lines. **STB[7]*** becomes the **RF*** signal, **STB[6]*** becomes **TED***, and **STB[5]*** becomes **SLD*** as defined in IEEE Std 896.9-1994. This is the default/reset state.
- 1 - **STA[7..0]***, **STB[7..0]***, **CAA[2..0]*** and **CAB[2..0]*** provide copies of **ST[7..0]*** and **CA[2..0]*** as described above.

Twenty-two bits (23–2), of R/C **SP_TMR_HOLD_I** (CSR 456) (see Figure 9-10), are used to indicate the status and capability bits in error. A 2-bit code is assigned to each signal line that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_I** (CSR 456) shall release these bits.

Bits 7–0, **TMR_ST[7..0]_A** of R/W **SP_MAINT_MONITOR_ST_CAP** (CSR 676) (see Figure 9-18), are placed on **STA[7..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted.

Bits 23–16, **TMR_ST[7..0]_B** of R/W **SP_MAINT_MONITOR_ST_CAP** (CSR 676) (see Figure 9-18), are placed on **STB[7..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted.

Bits 10–8, **TMR_CA[2..0]_A** of R/W **SP_MAINT_MONITOR_ST_CAP** (CSR 676) (see Figure 9-18), are placed on **CAA[2..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted.

Bits 26–24, **TMR_CA[2..0]_B** of R/W **SP_MAINT_MONITOR_ST_CAP** (CSR 676) (see Figure 9-18), are placed on **CAB[2..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted.

These are initially reset to an inactive state.

7.1.3.2.4 Synchronization lines

The “A” and “B” copies of these signals (**AIA***, **AKA***, **ASA***, **DIA***, **DKA***, **DSA***, **ETA***, and **AIB***, **AKB***, **ASB***, **DIB***, **DKB***, **DSB***, **ETB***) are used in the TMR voting.

Bit 4, **TMR_SYNCH_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - These lines are used as single lines. This is the default/reset state.
- 1 - The replica copies of the synchronization lines are provided as described above.

Fourteen bits (13–0) of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11) are used to indicate the synchronization bits in error. A 2-bit code is assigned to each signal line that describes the error.

The codes given above result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing a 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) will release these bits.

Bits 10–4, **TMR_AS/AK/AI/DS/DK/DI/ET_A** of R/W **SP_MAINT_MONITOR_SYNCH_A** (CSR 668), are placed on the A copy of the Synchronization lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. Bits 10–4, **TMR_AS/AK/AI/DS/DK/DI/ET_B** of R/W **SP_MAINT_MONITOR_SYNCH_B** (CSR 672) (see Figure 9-17), are placed on the B copy of the Synchronization lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted.

7.1.3.2.5 Arbitrated message bus and codes

Bit 5, **ECC_ARB_MESSAGE_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of ECC for these lines as follows:

- 0 - No ECC - **ABC[0]*** used for parity as in ISO/IEC 10857: 1994. This is the default/reset state.
- 1 - **AB[7..0]*** and **ABC[4..0]*** form (13,8) code word.

Five bits (4–0), **ECC_ARB_MESSAGE_SYNDROME** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), are used to store the error syndrome. These bits shall be cleared by writing a 1 into these bits in **SP_ECC_HOLD** (CSR 452).

Two bits (27–263), **ECC_ARB_MESSAGE_ERROR** of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11), indicate the results of the correction. Writing 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) shall clear these bits.

Bits 23–19, **ECC_ABC** of R/W **SP_MAINT_MONITOR_ECC** (CSR 664) (see Figure 9-15), are placed on **ABC[4..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to an inactive state.

7.1.3.2.6 Arbitration handshake signals

Bit 6, **TMR_ARB_SYNCH_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - Synchronization lines are used as single lines. This is the default/reset state.
- 1 - **APA***, **AQA***, **ARA***, and **APB***, **AQB***, **ARB*** provide copies of **AP***, **AQ***, **AR*** as described above.

Six bits (19..14) of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11) are used to store the arbitration synchronization bits in error. A 2-bit code is assigned to each signal line that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) will release these bits.

Bits 15–13, **TMR_AP/AQ/AR_A** of R/W **SP_MAINT_MONITOR_SYNCH_A** (CSR 668), are placed on the A copy of the Arbitration Synchronization lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. Bits 15–13, **TMR_AP/AQ/AR_B** of R/W **SP_MAINT_MONITOR_SYNCH_B** (CSR 672) (see Figure 9-17), are placed on the B copy of the Arbitration Synchronization lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to an inactive state.

7.1.3.2.7 Arbitrated condition lines

Bit 7, **TMR_ARB_COND_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - **AC[1..0]*** are used as single lines. This is the default/reset state.
- 1 - **ACA[1..0]*** and **ACB[1..0]*** provide copies of **AC[1..0]*** as described above.

Four bits (23–20) of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11) are used to indicate the arbitrated condition bits in error. A two bit code is assigned to each signal line that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) shall release these bits.

Bits 12–11, **TMR_AC_0/1_A** of R/W **SP_MAINT_MONITOR_SYNCH_A** (CSR 668), are placed on the A copy of the Arbitrated Message Condition lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. Bits 12–11, **TMR_AC_0/1_B** of R/W **SP_MAINT_MONITOR_SYNCH_B** (CSR 672) (see Figure 9-17), are placed on the B copy of the Arbitrated Message Condition lines when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to an inactive state.

7.1.3.2.8 Reset lines

Bit 8, **TMR_RESET_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14) controls enabling of TMR for these lines as follows:

- 0 - no correction - **RE*** and **RED*** are used as dual lines. This is the default/reset state.
- 1 - **REA*** and **REB*** provide copies of **RE*** as described above.

Two bits (25–24), **TMR_RESET_ERROR** of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11), are used to store the reset bits in error. A 2-bit code is assigned that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) shall release these bits.

7.1.3.2.9 Arbitration requests

Bit 9, **TMR_RQ_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - **RQ[1..0]*** are used as single lines. This is the default/reset state.
- 1 - **RQA[1..0]*** and **RQB[1..0]*** provide copies of **RQ[1..0]*** as described above.

For each set of request lines entering the central arbiter, the following CSRs must be supported.

Four bits (31–28) of R/C **SP_TMR_HOLD_II** (CSR 460) (see Figure 9-11) are used to indicate the request bits in error. A 2-bit code is assigned to each signal line that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_II** (CSR 460) shall release these bits.

Bits 3–2, **TMR_RQ_0/1_A** of R/W **SP_MAINT_MONITOR_SYNCH_A** (CSR 668), are placed on **RQA[1..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. Bits 3–2, **TMR_RQ_0/1_B** of R/W **SP_MAINT_MONITOR_SYNCH_B** (CSR 672) (see Figure 9-17) are placed on **RQB[1..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to an inactive state.

7.1.3.2.10 Grant lines and grant codes

Bit 10, **ECC_GRANT_ERROR_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), in all nodes controls enabling of correction for these lines as follows:

- 0 - no correction - the **GR*** to each node is solely used for providing grants during central arbitration. This is the default/reset state.
- 1 - A special code, **GC[5..0]*** is generated that specifies the master on the line according to the above description. **GCP*** is even parity over this code.

Seven bits (15–9), **GRANT_CODE_HOLD** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), at each receiving node are used to store the **GC[5..0]*** and **GCP*** as a result of a miscompare while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_ECC_HOLD** (CSR 452) shall release these bits.

Three bits (23–21), **ECC_GRANT_ERROR_CORRECTION** of R/C **SP_ECC_HOLD** (CSR 452) (see Figure 9-9), at each receiving node indicate the results of the correction.

These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_ECC_HOLD** (CSR 452) shall release these bits.

Bit 12, **ECC_GRANT_EVEN_PARITY_ENABLE** of R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), in the Central Arbiter, will force even parity codes to be generated on **GCP*** when active. Bits 13–8, **ECC_GRANT_GC** of R/W **SP_MAINT_MONITOR_ECC** (CSR 664) (see Figure 9-15), in the Central Arbiter, are placed on **GC[5..0]*** when the **MAINT_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. These are initially reset to an inactive state.

7.1.3.2.11 Preempt lines

Bit 11, **TMR_PE_ENABLE** from R/W **SP_ENABLE_ECC_TMR** (CSR 660) (see Figure 9-14), controls enabling of TMR for these lines as follows:

- 0 - no correction - **PE*** is used as a single line. This is the default/reset state.
- 1 - **PEA*** and **PEB*** provide copies of **PE*** as described above.

Two bits (1–0), **TMR_PE_ERROR** of R/C **SP_TMR_HOLD_I** (CSR 456) (see Figure 9-10), are used to store the arbitrated condition bits in error. A 2-bit code is assigned that describes the error.

The codes given result from comparisons made while correction is enabled. These bits are initially reset to an inactive state. Writing 1 into these bits in **SP_TMR_HOLD_I** (CSR 456) shall release these bits.

Bit 0, **TMR_PE_A** of R/W **SP_MAINT_MONITOR_SYNCH_A** (CSR 668), is placed on **PEA*** when the **MAINT_MONITOR_SP_MAINT_MONITOR_SYNCH_B** (CSR 672) (see Figure 9-17), is placed on **PEB*** when the **MAINT_ENABLE** bit in the **MAINT_CONTROL** CSR is asserted. Bit 0, **TMR_PE_B** of R/W **_MONITOR_ENABLE** bit in the **MAINT_CONTROL** CSR, is asserted. These are initially reset to an inactive state.

7.1.4 Duplicated signals

Profile S modules not in the Error Correcting configuration shall duplicate the Reset (**RE***), Select (**SL***), and Transaction Error (**TE***) signals with **RED***, **SLD***, and **TED***, respectively, for each implemented Futurebus+ as defined in IEEE Std 896.9-1994. These duplicated signals are optional in Level 0.

7.1.5 Reflected protection

All Profile S modules shall implement reflected protection using the Status and Capability fields of each Futurebus+ as defined in IEEE Std 896.9-1994. All Profile S modules not in the Error Correcting configuration shall also be required to implement the Reflected Field (**RF***) signal as defined in IEEE Std 896.9-1994. This reflected protection is optional in Level 0.

7.1.6 Protocol monitoring

Profile S modules shall implement parallel protocol handshake protection as defined in IEEE Std 896.9-1994.

Profile S modules that implement distributed arbitration shall implement protocol monitoring on each bus as defined in IEEE Std 896.9-1994 to protect the arbitrated condition and arbitration synchronization signals.

Profile S modules shall implement protocol monitoring on each bus as defined in IEEE Std 896.9-1994 to protect the preemption and end of tenure lines.

All Profile S central arbiter modules that do not implement the Error Correcting configuration shall use protocol monitoring on the grant lines as defined in IEEE Std 896.9-1994.

Protocol monitoring is optional in Level 0.

7.1.7 Timeouts

All Profile S modules shall implement Transaction, Arbitration, Central Arbitration, and Bus Activity timeouts as defined in ISO/IEC 10857: 1994, IEEE Std 896.2-1991, and IEEE Std 896.9-1994. If Split Transactions are implemented, a Profile S module shall implement the Split Transaction timeout as defined in ISO/IEC 10857: 1994 and IEEE Std 896.2-1991. If Distributed Arbitration is implemented, a Profile S module shall implement the Distributed Arbitration Timeout as defined in IEEE Std 896.9-1994. Timeouts are optional in Level 0.

7.2 Error isolation, logging, and diagnostics

Once an error is detected, some action must be taken. At the simple end of the spectrum, bus activity is stopped, the error is logged, and the bus is marked as bad to all the participants. As applications require more information to diagnose the error or to aid in the recovery, various diagnostic activities of increasing complexity may be utilized making use of the alternate access bus. At the far end of this spectrum, enough information is generated with the error to instantly mask the error obviating the need for immediate repair and recovery. As in error detection, the minimum error logging and capability to shut down the bus is required in all Profile S modules.

7.2.1 Error handling

Errors detected by parity, duplicated signals, reflected protection, protocol monitoring, and timeouts shall be handled by Profile S modules as defined in IEEE Std 896.9-1994, Clause 4.

Single errors detected by error correcting codes or triple modular redundancy shall be logged in the appropriate ECC or TMR **SP_xyz_HOLD** CSR location with sufficient information to locate the error. These will cause the **ECC_TMR_ERROR_DETECTED** bit to be set in the **ERROR_LO** CSR and are further defined in Clause 9. As long as the error is masked by the ECC/TMR circuitry and no other error is detected or reported, this will not cause a break in the usage of the Error Correcting configuration. It is strongly rec-

ommended that software is notified through a low-level interrupt or through polling that error masking has occurred.

7.2.2 Enable/Disable transceivers

Profile S modules shall implement the **MAINT_CONTROL** and **MAINT_MONITOR** CSR bits necessary to enable or disable the transceivers for each Futurebus+ and Serial Bus through at least two separate paths. These are defined in Clause 9.

7.2.3 Force errors

Profile S modules shall implement the **MAINT_CONTROL** CSR and **MAINT_MONITOR** CSR bits necessary to force errors to implemented error detection mechanisms and verify error detection. These are defined further in Clause 9.

7.2.4 Diagnostic data register

Profile S modules shall implement the **MAINT_UTILITY** CSR as defined in IEEE Std 896.9-1994 to provide a diagnostic register to test Futurebus+ transaction timing and error detection. These are included in Clause 9.

7.2.5 Critical register write protection

Profile S modules shall implement the **WRITE_PROTECT** and **WRITE_PROTECTED** CSRs as defined in IEEE Std 896.9-1994. This protects critical Futurebus+ CSRs from spurious writes.

7.3 Error removal and recovery

Once an error is detected, isolated, and logged, it is possible to attempt to remove the error or recover from the error in other ways. With the exception of simple hardware retry, all of these mechanisms are optional.

7.3.1 Hardware retry

Profile S modules shall implement immediate hardware retry as defined in IEEE Std 896.9-1994 in response to all detected bus errors that are not masked through error correction. Hardware retry is optional in Level 0.

7.3.2 Software retry

Profile S modules that implement software retry shall do so as defined in IEEE Std 896.2-1991 and IEEE Std 896.9-1994.

7.3.3 Spare lines

Several spare lines shall be provided in the Error Correcting configuration in support of hard fault recovery.

Once a line is identified that has a hard error on it, the TMR or ECC covering that line is greatly reduced. In the case of TMR, a second error in the three lines would result in an incorrect vote. In the case of ECC, errors can no longer be corrected and only single bit additional errors can be detected. For this reason, it is imperative that a failing line be replaced as soon as possible for the proper continuing operation of software. It was decided that having a complete second Error Correcting Futurebus+ would use too much power and too many pins. Since the TMR and ECC inherently take longer and thus have a lower performance, it was felt that the slight performance penalty to carry spare lines multiplexed into the main Error Correcting Futurebus+ could be absorbed by the longer TMR/ECC bus characteristics. This is not the case with the

Dual Buses as these are actually two independent high performance Futurebus+ buses as the next subclause will define.

Table 7-4 shows the spare lines grouped with the lines they cover. The number of signals covered by some spares is much smaller than for others because of performance and signal grouping issues. The address, data, and code spares (**ADS[15..0]*** and **BCS[1..0]***) each only cover four signals. The other spares cover between 6 and 15 signals. The address, data, and code lines are in the critical path and use ECC coding for every bus transaction, so any degradation will significantly affect system performance. This degradation comes from the larger, slower multiplexers that are required if more signals are to be spared combined with the larger checking circuits required for the ECC syndromes. The other signals, however, are covered by TMR. The process of TMR voting only involves three signals per vote and this combined with larger multiplexor delays for the larger number of lines to be covered by each spare signal should balance out with those signals covered by ECC in the error correcting bus configuration.

The **ENABLE_SPARE_LINES** in the **SP_ENABLE_SPARES_I** (CSR 680), **SP_ENABLE_SPARES_II** (CSR 684), **SP_ENABLE_SPARES_III** (CSR 688), and **SP_ENABLE_SPARES_IV** (CSR 692) CSRs shall be provided to enable the spare lines. Each spare line has an N-bit code in the CSRs given above that defines the bit that it replaces (see Table 7-5). Any forced error condition produced by the **MAINT_MONITOR** CSRs shall make use of any spare lines configured in place of the original bus lines. The spare lines shall be given for both 32-bit and 64-bit Error Correcting configurations. These are summarized in Table 7-4.

For complete bus pinouts, see Clauses 10,11, and 17.

In Error Correcting configurations, the following spare lines shall be implemented:

Command Spares **CMS[1..0]*** shall be capable of sparing any of the Command (**CM[7..0]***) or Command Code (**CC[4..0]***) signals per Table 7-4.

Status, Capability, and Handshake Spares **TMRS[4..0]*** shall be capable of sparing any of the Status (**ST[7..0]***, **STA[7..0]***, or **STB[7..0]***), Capability (**CA[2..0]***, **CAA[2..0]***, or **CAB[2..0]***), Address Handshakes (**AS***, **ASA***, **ASB***, **AK***, **AKA***, **AKB***, **AI***, **AIA***, or **AIB***), Data Handshakes (**DS***, **DSA***, **DSB***, **DK***, **DKA***, **DKB***, **DI***, **DIA***, or **DIB***), or End of Tenure (**ET***, **ETA***, or **ETB***) signals per Table 7-4.

Reset Spare **RES*** shall be capable of sparing any of the Reset (**RE***, **REA***, **REB***) signals per Table 7-4.

Central Arbitration Spares **GRS*** and **GCS*** shall be capable of sparing any of the Requests (**RQ[1..0]***, **RQA[1..0]***, or **RQB[1..0]***), Grants (**GR***), Grant Codes (**GC[5..0]***), Grant Code Parity (**GCP***), or Pre-emption (**PE***, **PEA***, or **PEB***) signals per Table 7-4.

Utility Spare **UPSP*** shall be capable of sparing any of the Nuclear Event Detect (**NED***, **NEDA***, or **NEDB***), Power Fail Imminent (**PFI***, **PFIA***, or **PFIB***), or System Reset (**SR***, **SRA***, or **SRB***) signals per Table 7-4. **UWSP** shall be capable of sparing any of the Power Normal (**PN**, **PNA**, or **PNB**) signals per Table 7-4.

Address/Data Spares **ADS[7..0]*** & **BCS[1..0]*** shall be capable of sparing any of the Address/Data (**AD[31..0]***), or Bus Code (**BC[7..0]***) signals per Table 7-4.

In 64-bit Error Correcting configurations, Address/Data Spares **ADS[15..0]*** & **BCS[1..0]*** shall be capable of sparing the respective Address/Data (**AD[63..0]***), or Bus Code (**BC[7..0]***) signals per Table 7-4.

In Error Correcting configurations that implement distributed arbitration, Distributed Arbitration Spares **ARBS*** and **ABS[1..0]*** shall be capable of sparing any of the Arbitrated Message Bus (**AB[7..0]***) Arbi-

trated Message Bus Codes (**ABC[4..0]***), Arbitration Handshakes (**AR***, **ARA***, **ARB***, **AP***, **APA***, **APB***, **AQ***, **AQA***, or **AQB***), or Arbitrated Message Condition (**AC[1..0]***, **ACA[1..0]***, or **ACB[1..0]***) signals per Table 7-4.

Table 7-4—Error Correcting configuration spare pin definition

Signal name	Signal description	Signal coverage by spare
ABS0*	Arbitrated Message Spare 0	AB0*, AB2*, AB4*, AB6*, ABC0*, ABC2*, ABC4*
ABS1*	Arbitrated Message Spare 1	AB1*, AB3*, AB5*, AB7*, ABC1*, ABC3*
CMS0*	Command Spare 0	CM0*, CM2*, CM4*, CM6*, CC0*, CC2*, CC4*
CMS1*	Command Spare 1	CM1*, CM3*, CM5*, CM7*, CC1*, CC3*
TMRS0*	Status, Capability, & Handshake Spare 0	ST0*, ST4*, STB1*, STB5*, AS*, ASA*, ASB*, CA0*, CA1*
TMRS1*	Status, Capability, & Handshake Spare 1	ST1*, ST5*, STA0*, STA4*, STB2*, STB6*, CAA0*, CAA1*, DI*, DIA*, DIB*
TMRS2*	Status, Capability, & Handshake Spare 2	ST2*, ST6*, STA1*, STA5*, STB3*, STB7*, CAB0*, CAB1*, DK*, DKA*, DKB*
TMRS3*	Status, Capability, & Handshake Spare 3	ST3*, ST7*, STA2*, STA6*, AI*, AIA*, AIB*, DS*, DSA*, DSB*
TMRS4*	Status, Capability, & Handshake Spare 4	STA3*, STA7*, STB0*, STB4*, AK*, AKA*, AKB*, ET*, ETA*, ETB*
ARBS*	Arbitration Condition & Handshake Spare	AP*, APA*, APB*, AQ*, AQA*, AQB*, AR*, ARA*, ARB*, AC0*, AC1*, ACA0*, ACA1*, ACB0*, ACB1*
RES*	Reset Spare	RE*, REA*, REB*
GRS*	Arbitration Request and Grant Spare	GR*, RQ0*, RQ1*, RQA0*, RQA1*, RQB0*, RQB1*
GCS*	Grant Code & Preemption Spare	GC0*, GC1*, GC2*, GC3*, GC4*, GC5*, GCP*, PE*, PEA*, PEB*
ADS0*	Address, Data, and Code Spare 0	AD0*, AD2*, AD4*, AD6*
ADS1*	Address, Data, and Code Spare 1	AD1*, AD3*, AD5*, AD7*
ADS2*	Address, Data, and Code Spare 2	AD8*, AD10*, AD12*, AD14*
ADS3*	Address, Data, and Code Spare 3	AD9*, AD11*, AD13*, AD15*
ADS4*	Address, Data, and Code Spare 4	AD16*, AD18*, AD20*, AD22*
ADS5*	Address, Data, and Code Spare 5	AD17*, AD19*, AD21*, AD23*
ADS6*	Address, Data, and Code Spare 6	AD24*, AD26*, AD28*, AD30*
ADS7*	Address, Data, and Code Spare 7	AD25*, AD27*, AD29*, AD31*
ADS8*	Address, Data, and Code Spare 8	AD32*, AD34*, AD36*, AD38*
ADS9*	Address, Data, and Code Spare 9	AD33*, AD35*, AD37*, AD39*
ADS10*	Address, Data, and Code Spare 10	AD40*, AD42*, AD44*, AD46*
ADS11*	Address, Data, and Code Spare 11	AD41*, AD43*, AD45*, AD47*
ADS12*	Address, Data, and Code Spare 12	AD48*, AD50*, AD52*, AD54*
ADS13*	Address, Data, and Code Spare 13	AD49*, AD51*, AD53*, AD55*
ADS14*	Address, Data, and Code Spare 14	AD56*, AD58*, AD60*, AD62*
ADS15*	Address, Data, and Code Spare 15	AD57*, AD59*, AD61*, AD63*
BCS0*	Address, Data, and Code Spare 16	BC0*, BC2*, BC4*, BC6*
BCS1*	Address, Data, and Code Spare 17	BC1*, BC3*, BC5*, BC7*
UPSP*	Utility Signal Push Pull TTL Spare	NED*, NEDA*, NEDB*, PFI*, PFIA*, PFIB*, SR*, SRA*, SRB*
UWSP	Utility Signal Wire OR TTL Spare	PN, PNA, PNB

7.3.3.1 Spare line enables

The following shall select the signal that a spare line replaces in the Error Correcting configuration (see Table 7-5). A code of all zeroes disables that spare line. All spare lines shall be initially zero upon power up.

Bits 3–0, **SPARE_UTILITY_PP** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **UPSP***.

Bits 5–40, **SPARE_UTILITY_WO** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **UWSP**.

Bits 19–16, **SPARE_BUS_CODE_EVEN** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **BCS0***.

Bits 23–20, **SPARE_BUS_CODE_ODD** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **BCS1***.

Bits 27–24, **SPARE_ARB_MESSAGE_EVEN** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **ABS0***.

Bits 31–28, **SPARE_ARB_MESSAGE_ODD** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **ABS1***.

Bits 8–6, **SPARE_COMMAND_EVEN** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **CMS0***.

Bits 11–9, **SPARE_COMMAND_ODD** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **CMS1***.

Bits 15–12, **SPARE_ST_CAP_HNDSHK_0** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **TMRS0***.

Bits 19–16, **SPARE_ST_CAP_HNDSHK_1** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **TMRS1***.

Bits 23–20, **SPARE_ST_CAP_HNDSHK_2** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **TMRS2***.

Bits 27–24, **SPARE_ST_CAP_HNDSHK_3** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **TMRS3***.

Bits 31–28, **SPARE_ST_CAP_HNDSHK_4** of R/W **SP_ENABLE_SPARES_I** (CSR 680), selects which signal to spare with **TMRS4***.

Bits 3–0, **SPARE_ARB_CONDITION_HNDSHK** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **ARBS***.

Bits 7–4, **SPARE_PREEMPT_GRANT_CODE** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **GCS***.

Bits 11–8, **SPARE_RESET** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **RES***.

Bits 15–12, **SPARE_GRANT_ARB_REQ** of R/W **SP_ENABLE_SPARES_II** (CSR 684), selects which signal to spare with **GRS***.

Bits 3–0, **SPARE_ADDR_DATA_0** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS0***.

Bits 7–4, **SPARE_ADDR_DATA_1** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS1***.

Bits 11–8, **SPARE_ADDR_DATA_2** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS2***.

Bits 15–12, **SPARE_ADDR_DATA_3** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS3***.

Bits 19–16, **SPARE_ADDR_DATA_4** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS4***.

Bits 23–20, **SPARE_ADDR_DATA_5** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS5***.

Bits 27–24, **SPARE_ADDR_DATA_6** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS6***.

Bits 31–28, **SPARE_ADDR_DATA_7** of R/W **SP_ENABLE_SPARES_III** (CSR 688), selects which signal to spare with **ADS7***.

Bits 3–0, **SPARE_ADDR_DATA_8** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS8***.

Bits 7–4, **SPARE_ADDR_DATA_9** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS9***.

Bits 11–8, **SPARE_ADDR_DATA_10** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS10***.

Bits 15–12, **SPARE_ADDR_DATA_11** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS11***.

Bits 19–16, **SPARE_ADDR_DATA_12** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS12***.

Bits 23–20, **SPARE_ADDR_DATA_13** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS13***.

Bits 27–24, **SPARE_ADDR_DATA_14** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS14***.

Bits 31–28, **SPARE_ADDR_DATA_15** of R/W **SP_ENABLE_SPARES_I** (CSR 692), selects which signal to spare with **ADS15***.