

WITHDRAWN

ISO/IEC 10861 : 1994
[ANSI/IEEE Std 1296, 1994 Edition]

Information technology— Microprocessor systems— High-performance synchronous 32-bit bus: MULTIBUS II

Sponsor
**Technical Committee on Microprocessors and Microcomputers
of the
IEEE Computer Society**

**Adopted as an International Standard by the
International Organization for Standardization
and by the
International Electrotechnical Commission**

**Published by
The Institute of Electrical and Electronics Engineers, Inc.**

WITHDRAWN

Abstract: The operation, functions, and attributes of a parallel system bus (PSB), called MULTIBUS II, are defined. A high-performance backplane bus intended for use in multiple processor systems, the PSB incorporates synchronous, 32-bit multiplexed address/data, with error detection, and uses a 10 MHz bus clock. This design is intended to provide reliable state-of-the-art operation and to allow the implementation of cost-effective, high-performance VLSI for the bus interface. Memory, I/O, message, and geographic address spaces are defined. Error detection and retry are provided for messages. The message-passing design allows a VLSI implementation, so that virtually all modules on the bus will utilize the bus at its highest performance—32 to 40 Mbyte/s. An overview of PSB, signal descriptions, the PSB protocol, electrical characteristics, and mechanical specifications are covered.

Keywords: high-performance synchronous 32-bit bus, MULTIBUS II, system bus architectures

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

Copyright © 1994 by the The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1994. Printed in the United States of America.

ISBN 1-55937-368-7

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.

WITHDRAWN

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and nongovernmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

In 1990, ANSI/IEEE Std 1296-1987 was adopted by ISO/IEC JTC 1, as draft International Standard ISO/IEC/DIS 10861. This draft was subsequently approved by ISO/IEC JTC 1 in the form of this edition, which is published as International Standard ISO/IEC 10861 : 1994.

International Organization for Standardization/International Electrotechnical Commission
Case postale 56 • CH-1211 Genève 20 • Switzerland

WITHDRAWN

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 which 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 once 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

IEEE standards documents may involve the use of patented technology. Their approval by the Institute of Electrical and Electronics Engineers does not mean that using such technology for the purpose of conforming to such standards is authorized by the patent owner. It is the obligation of the user of such technology to obtain all necessary permissions.

WITHDRAWN

Introduction

(This introduction is not a normative part of ISO/IEC 10861 : 1994 [ANSI/IEEE Std 1296, 1994 Edition], but is included for information only.)

In the last decade, the avalanche of new microcomputer technology, especially VLSI, threatened to obsolete products almost before they went into production. To buffer users from this onrush of technology, Intel helped develop standard interfaces. One of the most notable was the MULTIBUS I system bus, which was used as the basis for a standard by the IEEE in 1983 as IEEE Std 796-1983 (after going through a 5-year review and revision process).

In the early 1980s, Intel recognized that the trends toward multiprocessing and more sophisticated microcomputer-based systems called for an advanced 32-bit system bus architecture. Intel called this new bus MULTIBUS II. In continuing to pioneer the open systems technology, which included multiprocessing, four critical requirements were observed: technical credibility, processor independence, standardization, and openness to all levels of integration. Early in the development of the new bus, Intel established a “MULTIBUS II Development Consortium.” The consortium gave the new bus a technical credibility that few buses, especially those defined only among board vendors, can match. The companies in the consortium also represented all microprocessor families; included in the group were 68020, 32032, 80386, and Z8000 board and system users, thus ensuring that the bus is easily adaptable to virtually any manufacturer's processor.

The primary benefits being sought in the creation of this new bus were high-performance multiprocessing, high system reliability, ease-of-use by system designers, and improved cost/performance.

Specific bus features were developed in response to these objectives. The 32 Mbyte/s message passing of the bus provides a bus that acts like a very high-speed network connection for multiple processors (or processor equivalents). There is a recognition that the bus is no longer to interconnect a CPU with its memory and I/O; instead the bus is to interconnect whole stand-alone processors with each other and with intelligent “subsystems-on-a-board.”

System reliability is enhanced by the features of bus parity, synchronous operation, negative acknowledge, transfer retries, geographic addressing, and advanced backplane design. Ease-of-use by system designers is implemented primarily through the geographic addressing, which provides for dynamic system configuration. The bus encourages the use of software programmable configuration options (and discourages any use of mechanical jumpers). The standardization of the high-level message-passing protocol also gives the system designer an easy-to-use capability for interprocessor communication.

The cost/performance objective of the bus is delivered through its specification of a realizable 32 to 40 Mbyte/s bus bandwidth. Virtually all boards designed to the bus can achieve this bus utilization factor due to the high-level protocol called out in the specification, and thus the availability of standard, high-performance and cost-effective VLSI components to actually implement this level of performance. For example, this specification and the VLSI make it possible for eight concurrent 4 megabyte/second transfers to take place on the bus. This, or other combinations of transfers that add up to 32 Mbyte/s, demonstrate the real cost/performance advantages of the bus for multiprocessor applications.

In 1983 MULTIBUS II was introduced to the IEEE standards process as a part of the considerations for the P896 (Future Bus) working group activities. In the 1984/1985 time frame the MSC (Microcomputer Standards Committee, of the TCMM) formed an independent study group for MULTIBUS II. During this time the many active participants of the group proceeded to thoroughly review and make changes to the proposed draft. In early 1986 the group was assigned a formal project number P1296. During the remainder of 1986, the draft was passed by the Working Group and the MSC after thorough review, discussion, and changes. In 1987, the draft was presented for Sponsor ballot and, after passing, presented to the June 1987 meeting of the IEEE Standards Board.

The IEEE Standards Board calls attention to the fact that there are patents claimed and/or pending on many aspects of this bus by Intel Corporation. IEEE takes no position with respect to patent validity. Intel Corporation has assured the IEEE that it is willing to grant a license for these patents on reasonable and nondiscriminatory terms to anyone wishing

WITHDRAWN

to obtain such a license. The general terms of the license are a one-time administration fee of \$100 for a nonexclusive perpetual license. Intel Corporation's undertakings in this respect are on file obtained from the legal department of Intel Corporation whose address is Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124.

There were many contributors to the standards review process, but the following members deserve special mention for their active participation:

Task Force Coordinators: Jack Blevins
Maurice Hubert
Hubert
Kirmann
Jim Nebus

Secretary of Working Group: Steve Cooper

Original Study Group Chairman: Paul Borrill

Original Draft Editor: Scott Tetrick

The P1296 Working Group that prepared this standard had the following membership:

Richard W. Boberg, Chair

Web Augustine
Jack Blevins
Paul Borrill
Steve Cooper
Tom Crawford
Gene Freehauf

Maurice Hubert
Hubert Kirmann
Klaus Mueller
Jim Nebus
Don Nickel
Ken Smith

Michael Thompson
Scott Tetrick
Eike Waltz
Janusz Zalewski

The following members of the Technical Committee on Microprocessors and Microcomputers were on the balloting body:

Andrew Allison
Peter J. Ashenden
Matt Biewer
John Black
Jack Blevins
Richard Boberg
Paul Borrill
Bradley Brown
Clyde Camp
John D. Charlton
Steve Cooper
Randy Davis
J. Robert Davis
Shirish P. Deodhar
Jim Dunlay
Wayne Fischer
Jim Flournoy
Gordon Force
Martin Freeman

David Gustavson
Tom Harkaway
Dave Hawley
David James
Laurel Kaleda
Richard Karpinski
Hubert Kirmann
Doug Kraft
Tom Kurihara
Glen Langdon
Gerry Laws
Tom Leonard
Rollie Linser
Gary Lyons
James Nebus
Gary Nelson
Deene Ogden
Tom Pittman
Shlomo Pri-Tal

P. Reghunathan
Richard Rawson
Bill Shields
Michael Smolin
Robert Stewart
Subramanganesan
Michael Teener
Scott Tetrick
Eike Waltz
Carl Warren
George White
Fritz Whittington
Tom Wicklund
Andrew Wilson
Anthony Winter

WITHDRAWN

When the IEEE Standards Board approved this standard on June 11, 1987, it had the following membership:

Donald C. Fleckenstein, *Chair*
Marco W. Migliaro, *Vice Chair*
Andrew G. Salem, *Secretary*

James H. Beall
Dennis Bodson
Marshall L. Cain
James M. Daly
Stephen R. Dillon
Eugene P. Fogarty
Jay Forster
Kenneth D. Hendrix
Irvin N. Howell

Leslie R. Kerr
Jack Kinn
Irving Kolodny
Joseph L. Koepfinger*
Edward Lohse
John May
Lawrence V. McCall
L. Bruce McClung
Donald T. Michael*

L. John Rankine
John P. Riganati
Gary S. Robinson
Frank L. Rose
Robert E. Rountree
Sava I. Sherr*
William R. Tackaberry
William B. Wilkens
Helen M. Wood

*Member Emeritus

IEEE Std 1296-1987 was approved by the American National Standards Institute on February 8, 1987 and was reaffirmed by IEEE on March 17, 1994.

WITHDRAWN

CLAUSE	PAGE
1. General overview	1
1.1 Scope	1
1.2 Normative references	1
2. Definitions.....	2
3. Guide to notation.....	5
3.1 General	5
3.2 Signal notation	5
3.3 Figure notation	5
3.4 Notation in state-flow diagrams.....	6
3.5 Notation for multiple bit data representation	6
4. PSB overview.....	7
4.1 General.....	7
4.2 Address/data path and system control signals.....	8
4.3 Message-passing facility	8
4.4 Interconnect facility	8
4.5 Synchronous operation of the PSB	8
4.6 Bus operations on the PSB.....	8
4.7 Central services module.....	12
5. Signal descriptions	12
5.1 General	12
5.2 Signal groups.....	12
6. PSB protocol	20
6.1 General.....	20
6.2 Arbitration operation.....	21
6.3 Transfer operation	31
6.4 Exception operation	52
6.5 Central control functions.....	56
6.6 State-flow diagrams	63
7. Electrical characteristics	76
7.1 General.....	76
7.2 AC timing specifications.....	77
7.3 DC specifications for signals	84
7.4 Current limitations per connector	85
7.5 Pin assignments.....	86
8. Mechanical specifications	89
8.1 General.....	89
8.2 Board sizes and dimensions	90
8.3 Printed board layout considerations	91

WITHDRAWN

CLAUSE	PAGE
8.4 Front panel	91
8.5 Connectors	91
8.6 Backplanes	92
9. IEEE 1296 System Interface specification.....	100
9.1 Overview	100
9.2 Interconnect space operation.....	101
9.3 I/O space operation	116
9.4 Memory space operations	117
9.5 Message space operations	117
10. IEEE 1296 capabilities.....	128
10.1 Characteristic codes	128
Annex A (Informative) Recommended documentation practices.....	129

WITHDRAWN

Information technology—Microprocessor systems—High-performance synchronous 32-bit bus: MULTIBUS II

1. General overview

1.1 Scope

This International Standard defines the operation, functions, and attributes of the IEEE 1296 bus standard.

- a) This standard defines a high-performance 32-bit synchronous bus standard.
- b) The bus standard must have a design-in lifetime of 10 years with backward compatibility.
- c) The standard is intended for general purpose applications to optimize block transfers, including protocol for message passing. For real-time applications, the bus will provide a means of ensuring an upper limit to message delivery time.
- d) The standard is intended to be compatible with existing IEC mechanical standards (IEC Pub 297-1,¹ 297-3, and 603-2) with recognition of the need for special front panels to address ESD, EMI, and RFI requirements.
- e) Options within the standard will be clearly identified.
- f) The standard is intended to support multiple processor modules in a functionally partitioned configuration and heterogeneous processor types in the same system.
- g) The standard is intended to support heterogeneous processor types in the same system.
- h) Message-passing format and protocol is intended for future migration to a serial system bus.

1.2 Normative references

The following standards contain provisions which, through references in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent edition of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

DIN 41612, Two Part Connectors for Printed Board, GRIB, to 54 mm, Common Mounting Features, Survey of Types.²

¹Information on references can be found in 1.2.

²DIN publications are available from the Deutsches Institut für Normung, Burggrafenstrasse 6, D-1000 Berlin 30, Germany.

IEC 297-1 : 1986, Dimensions of mechanical structures of the 482,6 mm (19 in) series—Part 1.: Panels and racks.³

IEC 297-3 : 1984, Dimensions of mechanical structures of the 482,6 mm (19 in) series—Part 3.: Subracks and associated plug-in units.

IEC 603-2 : 1988, Connectors for frequencies below 3 MHz for use with printed boards—Part 2.: Two-part connectors for printed boards, for basic grid of 2,54 mm (0,1 in), with common mounting features.

IEEE Std 1101-1987, IEEE Standard for Mechanical Core Specifications for Microcomputers (ANSI).⁴

2. Definitions

The following definitions apply to all clauses of this International Standard.

2.1 acquisition phase: The final phase of the arbitration operation entered after determining that an agent has the highest priority and the bus is available. *See:* **arbitration operation; agent.**

2.2 address transfer: The passing of address information over the multiplexed address/data bus from the bus owner in order to select the replying agent(s). *See:* **bus owner; replying agent.**

2.3 address/data bus signal group: A set of thirty-six (36) signals, consisting of 32 address/data signals and four parity signals that are used for address and data transfers.

2.4 agent: A physical unit that has an interface to the parallel system bus, for example, a single-board computer.

2.5 agent error: An agent status that indicates an error condition in a replying agent.

2.6 agent status: The condition of the replying agent, transmitted during the reply phase of a transfer operation. *See:* **reply phase; transfer operation.**

2.7 arbitration operation: The bus operation in which agents attempt to gain exclusive access to the parallel system bus.

2.8 backplane: The physical mechanism by which signals are routed between agents.

2.9 bit (b): A binary digit.

2.10 broadcast message: A sequence of one or more data transfers from the bus owner to all replying agents, with uninterrupted bus ownership.

2.11 bus clock cycle: An amount of time equal to one bus clock period, nominally 100 nanoseconds.

2.12 bus loss: The amount of time required for a valid signal transition to occur at every point on the backplane. This value is equivalent to two bus propagation delays plus the clock skew.

2.13 bus operation: The basic unit of processing whereby digital signals effect the transfer of data across an interface by means of a sequence of control signals and an integral number of bus clock cycles.

2.14 bus owner: The agent that enters the acquisition phase of the arbitration operation and initiates one or more transfer operations. *See:* **acquisition phase; arbitration operation; transfer operation.**

2.15 bus request sequence: A set of one or more arbitration operations in which all agents that simultaneously request the bus become the bus owner, one at a time. *See:* **arbitration operation; bus owner.**

2.16 byte (B): A group of eight adjacent bits operated on as a unit.

³IEC standards are available from the IEC Sales Department, Case Postale 131, 3 rue de Varembe, CH-1211, Genève 20, Switzerland/Suisse.

⁴IEEE publications are available from the Institute of Electrical and Electronics Engineers, Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

2.17 central services module (CSM): A specific module that is required in all systems using the parallel system bus. Its services, such as starting certain bus operations and guaranteeing uniform initialization of all agents, are required by all agents on the parallel system bus. It is always located in a specific slot in the system backplane. *See:* **parallel system bus**.

2.18 client: An agent that requests services of a server. *See:* **server**.

2.19 cold-start: A sequence of events performed on the application of power that ensures a uniform initialization period for all agents, giving them the ability to begin operation from a known state.

2.20 command transfer: The passing of command information over the system control signal group, from the bus owner to the replying agent(s), during the request phase of a transfer operation. Command information includes parameters for the impending transfer operation, as well as additional address space information not transmitted with the address transfer. *See:* **request phase; system control signal group**.

2.21 data transfer: The passing of data over the multiplexed address/data bus, between the bus owner and the replying agent(s), during the reply phase of a transfer operation.

2.22 driving agent: The agent that is permitted to assert or negate a signal on the bus.

2.23 dual-mode agent: An agent that supports both memory-mode and message-mode communication on the parallel system bus. *See:* **memory-mode agent; message-mode agent**.

2.24 dual-mode system: A system that supports both memory-mode and message-mode communication. A mixture of both communication types is used on the parallel system bus.

2.25 end of transfer (EOT) status: A handshake status that indicates the last data transfer of the transfer operation. *See:* **handshake status**.

2.26 exception: An abnormal condition on the bus caused by either a bus parity error, a bus time-out, a protocol violation, or a bus owner reply phase termination.

2.27 exception operation: A bus operation in which an agent places an error indication on the parallel system bus. The error indication causes all bus agents to terminate arbitration and transfer operations.

2.28 handshake status: A status transfer that indicates the exchange of data between bus owner and replying agent(s).

2.29 I/O space: The address space used for accessing peripheral devices such as communication controllers and mass storage devices.

2.30 interconnect space: The address space used for board identification, system configuration, and board specific functions such as testing and diagnostics.

2.31 interconnect template: A definition of the contents of the interconnect space of an agent.

2.32 interface: A shared boundary between modules or agents of a computer system, through which information is conveyed.

2.33 locked: A condition of the bus that guarantees exclusive access to the parallel system bus and to resources on the replying agent(s). This inhibits transfer operations between the replying agent and any other bus interface.

2.34 memory space: The address space used for accessing physical memory devices for storage and retrieval of code and data.

2.35 memory-mode agent: An agent that communicates with others by using memory and/or I/O space on the parallel system bus.

2.36 memory-mode system: A system in which the agents communicate with one another with data structures in memory and/or I/O space.

2.37 message space: The address space used for packet based communications ranging from interrupts to negotiated data movement. *See:* **packet**.

2.38 message-mode agent: An agent that exclusively uses message space for communication with other agents.

- 2.39 message-mode system:** A system in which communication between agents is via blocks of data transmitted in the message space.
- 2.40 module:** A basic functional unit within an agent.
- 2.41 nibble:** A group of four adjacent bits operated on as a unit.
- 2.42 packet:** A block of information that is transmitted within a single transfer operation in message space. *See: message space; transfer operation.*
- 2.43 parallel system bus (PSB):** The signals, media and protocol used to interconnect agents in the IEEE 1296 system.
- 2.44 parking:** The state of the bus owner where, after the completion of the current transfer operation, ownership is retained until there is a request by another agent for the use of the bus.
- 2.45 power failure recovery:** A sequence of events that provides orderly control of system shutdown during a temporary power failure and start-up after power is restored.
- 2.46 protocol:** The set of signaling rules used to convey information between agents.
- 2.47 read data transfer:** One or more data transfers from a replying agent to a bus owner, with uninterrupted bus ownership.
- 2.48 receiver:** An agent that is the recipient of the data during a solicited message. *See: solicited message.*
- 2.49 recovery phase:** The final phase of an exception operation in which the parallel system bus is allowed to sit idle for a defined amount of time. *See: exception operation.*
- 2.50 reply phase:** The final phase of a transfer operation that consists of one or more consecutive data and/or status transfers on the parallel system bus.
- 2.51 replying agent:** An agent that participates in a transfer operation with the bus owner.
- 2.52 request phase:** The initial phase of a transfer operation in which the bus owner places command and address information on the parallel system bus.
- 2.53 requesting agent:** An agent that has entered arbitration for bus access. *See: arbitration operation.*
- 2.54 resolution phase:** The initial phase of an arbitration operation in which all agents requesting access to the bus drive an arbitration ID onto the parallel system bus. Agents mutually resolve requests and allow the agent with the highest priority to gain access to the bus. *See: arbitration operation.*
- 2.55 sender:** The agent that supplies the data for a solicited message. *See: solicited message.*
- 2.56 sequential transfer:** A transfer operation with multiple data transfers during the reply phase. *See: reply phase; transfer operation.*
- 2.57 server:** An agent that performs a service for clients. *See: client.*
- 2.58 signal phase:** The initial phase of an exception operation in which all agents are notified of an error condition. *See: exception operation.*
- 2.59 skew:** The time difference between signals because of timing differences for logic and backplane delays.
- 2.60 solicited messages:** A negotiated data transfer in message space. *See: data transfer; message space.*
- 2.61 status transfer:** The passing of information over the system control signal group, between the bus owner and the replying agent, during the reply phase of a transfer operation. *See: agent status.*
- 2.62 system control signal group:** A set of ten (10) signals, including two parity bits, which supply command and status information between the bus owner and the replying agent.
- 2.63 transfer operation:** The bus operation in which a bus owner transfers data on the parallel system bus. *See: bus operation; bus owner.*

2.64 warm-start: A sequence of events performed to reset a running system.

2.65 write data transfer: One or more data transfers from the bus owner to a replying agent or agents, with uninterrupted bus ownership.

3. Guide to notation

3.1 General

A consistent convention on all signals, figures, and state-flow diagrams is used throughout this International Standard. The conventions used are described in the following subclauses.

3.2 Signal notation

Throughout this International Standard, the following convention is used for signal notation. An asterisk (*) following a signal name indicates that the signal is active when it is at a low voltage level (as described in clause 4.). A signal name that is not followed by an asterisk indicates that the signal is active when it is at a high voltage level. Additionally, a high signal voltage level may be indicated by an uppercase h (H), and a low signal voltage level may be indicated by an uppercase l (L). This convention is summarized in table 3.2-1.

Table 3.2-1 —Signal notation

Signal name	Electrical level	Signal notation
BREQ	Low or L	Inactive, deasserted, negated
	High or H	Active, asserted
BREQ*	Low or L	Active, asserted
	High or H	Inactive, deasserted, negated

Many signals on the parallel system bus (PSB) are more easily or conveniently discussed as a group. Names for these signals follow a decimal radix numbering convention. When discussed as an individual signal, the decimal number is simply appended to the signal name, e.g., AD15*. A group of signal lines with the same signal group name are collectively referred to by listing the group name and enclosing the decimal numbers within brackets, e.g., AD<31,23,7,0>*. A range of consecutive signals is referred to by using a double period to list the beginning and ending signals, inclusively, e.g., AD<31 . . 24>*. Consecutive and disjoint signals are referred to by using both methods, e.g., SC<9,3 . . 0>*. A bus signal group name with no brackets and digits afterward refers to the entire group, e.g., AD* is equivalent to AD<31 . . 0>*.

Signal lines on the PSB are always shown in capital letters, e.g., BREQ*. In the case of open-collector drivers, where more than one agent may drive the line at any time, the signal level driven by an agent may not be the logic level currently on the PSB signal line. The signal an agent drives on the corresponding bus line is designated by the same signal name, but in lowercase letters, e.g., breq*.

3.3 Figure notation

An example of the notation used in figures is shown in figure 3.3-1. All solid vertical lines in the figure represent the sampling point for signals (the falling edge of the BCLK* signal). Dashed vertical lines indicate phase transitions. Groups of signals used for phases and transfers are enclosed in polygons indicating simultaneous high and low logic

levels. All bus operations (arbitration, transfer, and exception) are enclosed in rectangles. Curved lines and arrows indicate cause-and-effect relationships, not asynchronous operation. Double-slashed lines in the timing diagrams (//) indicate the occurrence of an indefinite number of bus clock cycles, unless otherwise noted.

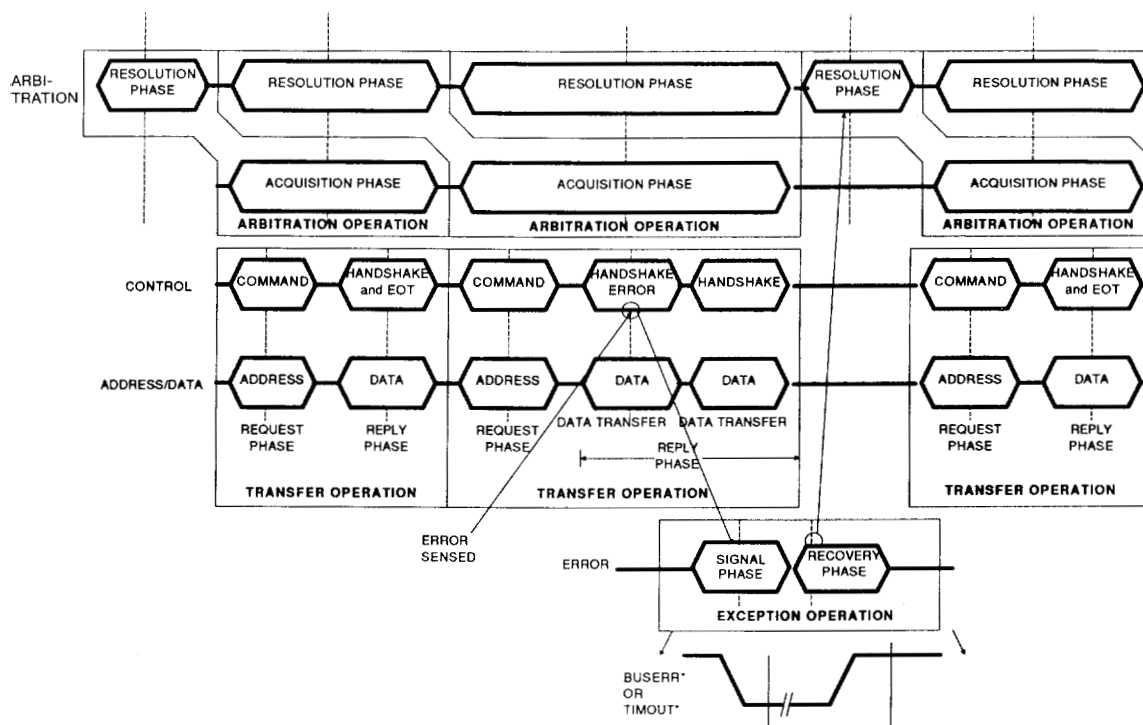


Figure 3.3-1 —Figure notation example

3.4 Notation in state-flow diagrams

The state-flow diagrams use a consistent notation system in describing the transitions between states during an operation.

Within each diagram there are various components. Each component describes, either graphically or in words, some aspect of the operation of an agent. The various states that an agent may assume are represented as circles. Transitions to other states are shown as arrowed lines and are labeled with a number that corresponds to the numbered paragraph that defines their function. Bold type is used to name conditions that are derived from bus signals or name conditions that describe the internal status of an agent. Generally, such conditions are associated with transitions occurring from one state to another. A condition that causes a transition is shown next to the arrowed line for that transition. Equals signs (=) are used to indicate the state of individual lines or the components of a condition. The state equations and conditions use “AND” to indicate when the AND operator is required and “OR” to indicate when the OR operator is used.

3.5 Notation for multiple bit data representation

Data values for spanning multiple bits are represented in a consistent manner. The value for the multiple bit quantity is shown as a decimal number, except where noted. The value is shown as high-true, making it independent of bus level inversions. For byte values, the highest numbered bit is the most significant bit of the byte. This International Standard does define the format of multiple data byte specification, except when noted.

If the data is in ASCII format, the data consists of eight consecutive bits, with the value of the eighth (most significant bit) always 0.

If the data is BCD format, each nibble within the byte represents a decimal digit. The value of bits {3 . . 0} is used for the units digit of the number and the value of bits {7 . . 4} is used for the tens digit of the number.

4. PSB overview

4.1 General

The PSB is a general purpose interface for multiple agents. Figure 4.1-1 shows a functional diagram of the PSB. As shown, the PSB consists of five signals groups that requesting and replying agents use to communicate with one another.

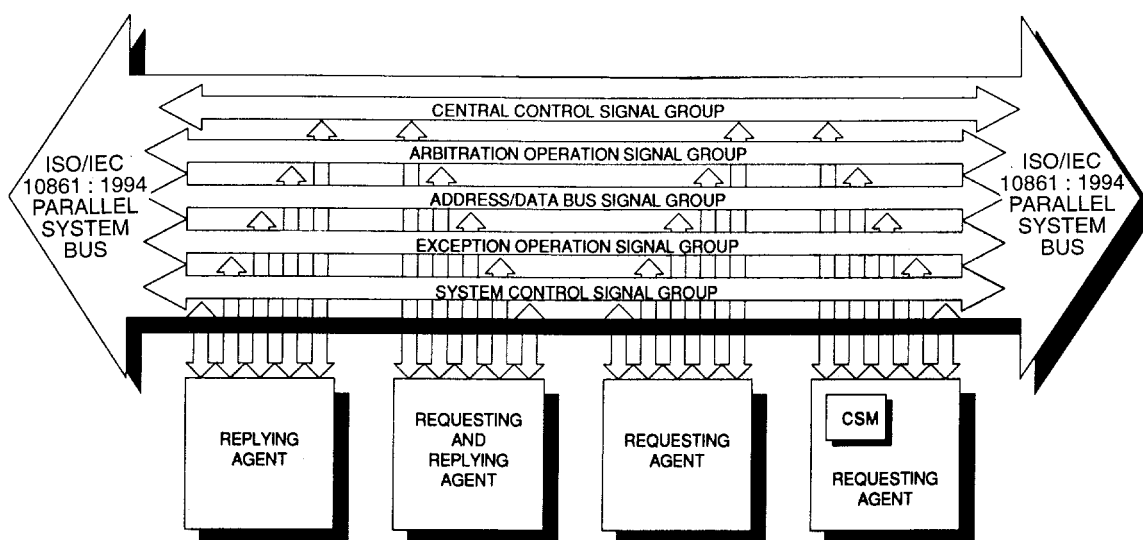


Figure 4.1-1 —Block diagram of the PSB interface

The PSB has several specific attributes, as follows:

- The address/data path is 32-bit plus four parity bits, providing a 4-Gbyte address range and capable of 8-, 16-, 24-, or 32-bit transfers. Ten system control signals, including nibble parity, provide command, status, and handshake information for the address/data path.
- The message-passing facility provides interagent communication.
- The interconnect facility allows configuration of agents and modules.
- The bus operations are synchronous; the bus uses a handshaking protocol that is synchronous with the basic clock rate (10 MHz) for the system resulting in a 40 Mbyte/s maximum transfer rate.
- The bus operates in defined bus operations. The three types of bus operation are arbitration operation, transfer operation, and exception operation.
- The bus uses a central services module to provide some centralized system functions for up to 21 agents.

Each of these attributes is explained further in the following subclauses.

4.2 Address/data path and system control signals

The address/data path on the PSB consists of 36 signal lines, including four parity lines, that are time-multiplexed. During the request phase of a transfer operation, the signal lines provide address information; during the reply phase of a transfer operation, the signal lines contain data. Byte parity is provided for all address and data transfers on the address/data path.

Ten system control signals, including two parity signals, provide command, status and handshake information for the address/data path. These signals are also time-multiplexed. During the request phase of a transfer operation, the system control signals provide command information; during the reply phase of a transfer operation, the signals are used for status and handshake information. Nibble parity is provided for the system control lines at all times.

4.3 Message-passing facility

The PSB defines a dedicated address space for use by agents in passing messages. The message-passing facility provides a standardized method for performing direct transfers of messages (command and data) from one agent to another. Message passing on the PSB provides for interagent communications (interrupts and data movement) through the use of the dedicated message space.

4.4 Interconnect facility

The interconnect facility within the PSB defines geographic addressing for ease of configuration, initialization, and diagnosis of an agent on the PSB. This allows software to identify, diagnose, configure, and initialize an agent within a system.

4.5 Synchronous operation of the PSB

The PSB is referred to as being “synchronous” in that all operations on the PSB occur relative to the active edge of a bus clock that is distributed to all agents on the bus. The synchronous nature of the bus does not place rigid time constraints on the length of a transfer operation. It requires that agents sample and drive signals at a specific time with respect to the bus clock.

The synchronous nature of the PSB allows each operation on the bus interface to be defined as a sequence of bus states. Agents on the bus proceed from one bus state to the next by monitoring and asserting specific status and control signals in a state control sequence defined by the operation. The specific meaning or function of some status or control signals can vary and is dependent on the bus operation and state.

Bus operations and associated state sequences are diagrammed later in this International Standard.

4.6 Bus operations on the PSB

Agents perform three types of bus operation on the PSB: an arbitration operation, a transfer operation, or an exception operation. Each operation has a definite event that signals the start and end of the operation. The three types of bus operation can occur simultaneously, maximizing bus utilization.

A typical read or write sequence on the PSB consists of a series of two operations. First, the requesting agent participates in the arbitration operation for access to the bus. When the requesting agent becomes the bus owner, it performs one or more transfer operations to read or write data. If any agent senses an exception (error) on the bus, that agent places an exception indication on the bus. The exception indication initiates an exception operation, which terminates the current arbitration and transfer operations.

Figure 4.6-1 shows how the two types of operation form a typical read or write sequence on the bus. The arbitration operation is a time period in which agents decide which will be the next to control the bus; the transfer operation is a subsequent time period in which that agent performs the address, command, data, and status transfers. Note that an exception operation is an error reporting time period that occurs only when an error is sensed.

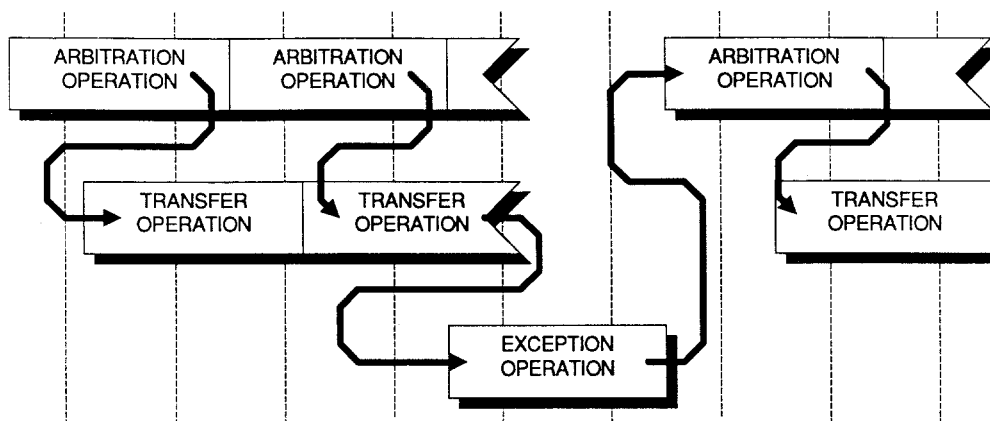


Figure 4.6-1 —Bus operation relationships

Each of the three types of bus operation is described further in the following paragraphs.

4.6.1 Arbitration operation

The arbitration operation is the bus operation in which agents attempt to gain exclusive access to the PSB. For each agent that requests access, the arbitration operation includes two phases: the resolution phase and the acquisition phase.

The resolution phase is the time period during which agents that desire the bus collectively arbitrate for access rights to the bus. The agents decide among themselves which one of them is going to control the bus on completion of the current transfer operation.

The arbitration method used during the resolution phase for the PSB is referred to as a parallel contention or self-selecting method. In this method, each agent on the bus is assigned an encoded bit pattern that gives it a priority level when it makes a request for bus use.

The agent that has the highest priority request during the resolution phase obtains access rights to the bus and begins the acquisition phase while the remaining agents begin the resolution phase of the next arbitration operation. The agent in the acquisition phase is referred to as the bus owner.

Once in the acquisition phase, the bus owner begins its transfer operation while the other requesting agents conduct another resolution phase (resolving for next access rights). Figure 4.6-2 shows the end-of-a-transfer operation (EOT) initiating an acquisition phase of the arbitration operation to allow the next bus owner to acquire control of the bus.

The arbitration scheme is such that no agent is prevented from gaining access to the bus. Agents that enter arbitration enter collectively as a group, and each agent within the group receives bus access before any new agent with normal priority is allowed to participate in the bus arbitration process. A bus request sequence is a period consisting of at least one arbitration operation in which all requesting agents (at least one) within a group arbitrate for and receive bus access. When all agents within the group have received bus access, a new bus request sequence is entered and a new

group of requesting agents begin arbitration for bus access. An agent may enter the bus request sequence if it has a high priority request as defined in 6.2.2.2.

Figure 4.6-2 shows bus operations that include three consecutive, errorless transfer operations on the bus. This type of operation sequence is typical of a system with more than one requesting agent. The first operation consists of a single data transfer, the second consists of three data transfers, and the third consists of one data transfer. The diagram shows the relationship between the arbitration operation and the transfer operation and shows the two phases of each arbitration operation: the resolution phase and the acquisition phase.

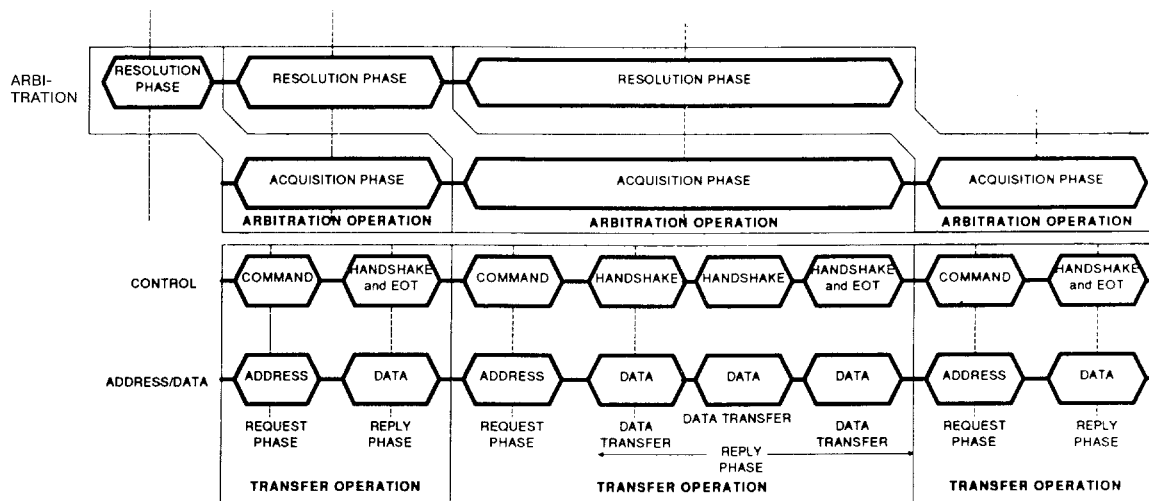


Figure 4.6-2 —Block diagram of bus operations

Note that the resolution phase of the next arbitration operation overlaps the acquisition phase of the current arbitration operation.

4.6.2 Transfer operation

After arbitrating for and winning control of the bus, the bus owner performs transfer operations to move data to or from another agent.

Figure 4.6-2 shows three consecutive transfer operations on the bus. The diagram shows the two phases of the transfer operation, the request phase and the reply phase, and shows the transfers within each phase, including the command, the address, the data, the handshake, and the EOT.

The request phase is controlled by the bus owner. During the request phase, the bus owner places address and control information onto the bus. The address and control information defines the replying agent(s), the type of operation, and the type of address space involved in the transfer operation.

After the bus owner transmits the address and control information, the reply phase of the transfer operation begins, in which the replying agent(s) satisfies the request.

During the reply phase, the bus owner and replying agent engage in a handshake that synchronizes the data transfer. In the case of a read transfer, the handshake status signifies that the replying agent has placed the data on the bus and the bus owner has received it. In the case of a write transfer, the handshake status signifies that the bus owner has placed the data on the bus and that the replying agent has received it. The bus owner and replying agent may perform one or more data transfers in a reply phase. The final data transfer is accompanied by an EOT indication. With the EOT, the

bus owner releases ownership of the bus if other agents request access to the bus. Otherwise, the bus owner retains ownership of the bus.

4.6.3 Exception operation

The exception operation is an error-reporting and recovery tool. If an agent detects an error (exception), that agent provides an exception indication on the bus and all agents begin an exception operation. The exception operation terminates any arbitration and transfer operations.

Figure 4.6-3 shows the same operation sequence presented in figure 4.6-2, except that in figure 4.6-3 an agent detects and signals an exception, thus initiating an exception operation. (Any agent can detect and signal an exception.) The diagram shows the two phases of the exception operation, the signal phase and the recovery phase, and shows how the exception operation terminates transfer and arbitration operations.

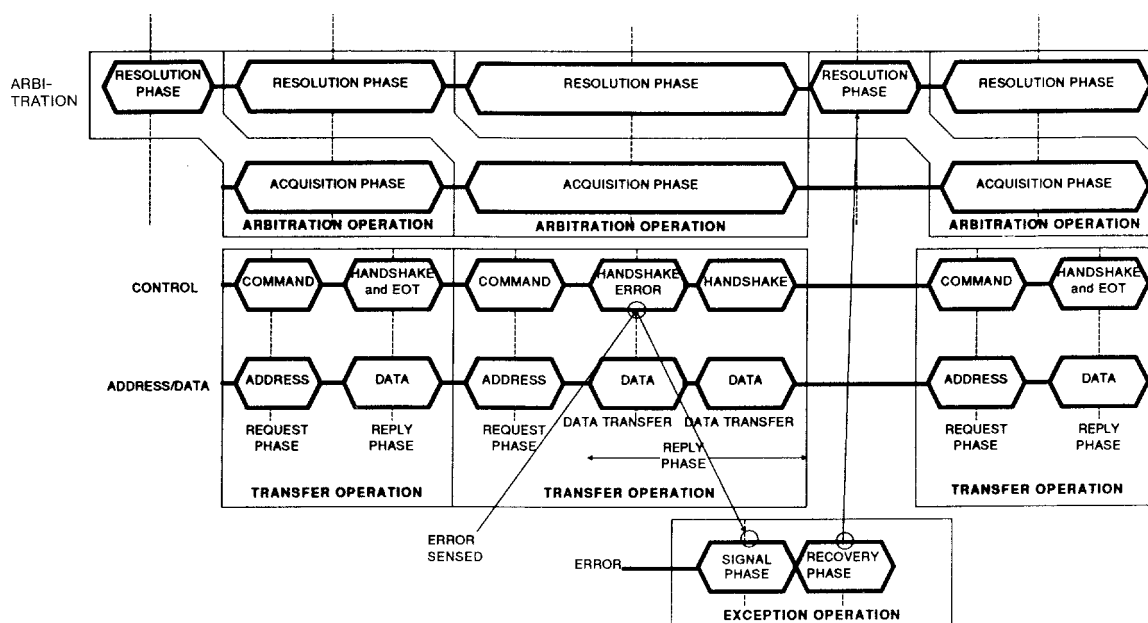


Figure 4.6-3 —Block diagram of bus operations with errors

During the signal phase, the agent that detected the exception condition indicates the problem to all agents on the bus by asserting an exception line. As a result of the signal phase, all agents terminate any arbitration and transfer operation that may be in progress.

After the exception line is negated by the agent that signalled the error, the recovery phase begins. One bus clock cycle following the negation of the exception line, arbitration operations may begin. Four bus clock cycles following the negation of the exception line, transfer operations may begin.

As figure 4.6-3 shows, the exception operation has a specific time relationship with respect to sensing an exception in the transfer operation. That is, the current exception operation reflects an exception condition sensed during the previous transfer operation.

4.7 Central services module

The PSB uses a single central services module (CSM) to provide central control of the bus functions. The CSM is responsible for clock and time out generation, monitoring the state of the power supplies and initialization of the bus interface logic. One and only one CSM is active in the PSB backplane.

5. Signal descriptions

5.1 General

This clause lists and defines the five signal groups that make up the PSB. Each signal is discussed in detail.

5.2 Signal groups

The PSB contains five groups of signals over which requesting and replying agents enact the bus protocol. All signals are sampled and driven relative to the falling edge of the bus clock unless otherwise noted. The signal groups, signal names, and their functions are listed in table 5.2-1. Each signal is discussed in the following text. Refer to 7.5.1 for information regarding physical signal position on the connector.

Table 5.2-1 —Signal groups on the PSB

Signal group name	Signals in group	Number of signals	Description
Arbitration operation signal group	BREQ*, ARB*	7	Provides the signals to request the bus and resolve arbitration for the next bus owner.
Address/data bus signal group	AD*, PAR*	36	Provides the address, data, and parity signals for read and write sequences.
System control signal group	SC*	10	Provides the control signals needed to transfer address and data on the address/data signal group.
Exception operation signal group	TIMOUT*, BUSERR*	2	Provides the error indication signals to stop transfer operations.
Central control signal group	RST*, RSTNC*, DCLOW*, PROT*, BCLK*, CCLK*, LACHn*	7	Provides system-wide services such as reset and initialization control.

Refer to clause 7. for electrical specifications of the signals.

5.2.1 Arbitration operation signal group

The arbitration signals on the PSB provide for the requesting, arbitrating, and granting of bus ownership to bus agents. All agents that require access to bus resources must arbitrate for use of the bus.

The seven signals within the arbitration operation signal group that implement the arbitration operation protocol on the PSB are bus request (BREQ*) and the arbitration ID signals ARB<5 . . 0>*. These signals are described in the following subclauses.

5.2.1.1 BREQ* (bus request)

BREQ*, when asserted, indicates that a bus request sequence is in progress. Any bus agent that requires access to the bus asserts its breq*. All breq* signals are ORed to become the BREQ* signal. Each requesting agent negates breq* as it receives bus access. As the last requesting agent in the current bus request operation is granted the bus, it negates breq*, which causes the negation of BREQ* on the bus. The negation of BREQ* indicates to all bus agents that the bus request sequence has ended. BREQ* is used only in arbitration operations; once an agent has ownership of the bus, it negates breq* and performs the transfer operation. Other agents wishing access to the bus must sample BREQ* to determine if arbitration is already underway. If arbitration has not begun, an agent may assert its breq* and enter arbitration.

5.2.1.2 ARB* (arbitration)

The arbitration signals are ARB<5 . . 0>*. The arbitration signals provide three functions:

- a) Carry cardslot ID assignment during reset
- b) Carry arbitration ID assignment during reset
- c) Arbitration

During reset, the CSM uses the ARB* lines to assign the agent in each cardslot a cardslot ID and an arbitration ID. The CSM drives the cardslot ID when ARB5* is asserted and the arbitration ID when ARB5* is negated. The cardslot and arbitration IDs are placed on ARB<4 . . 0>*. The cardslot ID gives the agent a geographical address that is used in addressing interconnect space. The arbitration ID defines the arbitration priority level for the agent. During reset, all agents are required to latch their cardslot ID and requesting agents are required to latch their arbitration ID from the ARB<4 . . 0>* lines.

The ARB* lines are used to resolve priorities for agents requesting access to the bus. During the resolution phase, a requesting agent places its arbitration ID on its arb* lines. Then, the agent with an arbitration ID that matches the ORed arbitration code present on ARB* gains access to the bus. As an agent gains bus access, it stops asserting its arb* and breq* lines and leaves arbitration to perform transfer operations as the bus owner. The remaining agents in the bus request sequence successively gain access to the bus in the same fashion as described.

An agent uses the ARB5* line to request high-priority access to the bus during arbitration. If two or more agents assert ARB5* at the same time, their bus access is determined by the priority of their arbitration IDs on ARB<4 . . 0>*.

5.2.2 Address/data bus signal group

Only the bus owner and the selected replying agent(s) use the multiplexed address/data bus of the PSB. The address/data bus signal group includes two sets of signals: address/data signals and parity signals. The address/data signals and the parity signals of the address/data bus signal group are described in the next subclauses.

5.2.2.1 AD* (address/data bus)

The address/data signals on the PSB are AD<31 . . 0>*. These 32 signals are multiplexed and serve a dual purpose depending on the phase of the transfer operation.

During the request phase of the transfer operation, AD* signals contain the address for the ensuing data transfer (AD0* is the least significant and AD31* is the most significant address bit). This address corresponds to a location in memory or I/O space, a processing agent in message space, or a cardslot/register location in interconnect space. The bus owner drives these lines during the request phase.

During the reply phase of the transfer operation, AD* signals provide for a variable-width data transfer, carrying 8, 16, 24, or 32 bits of data. The replying agent drives the AD* lines during the reply phase of a read transfer. The bus owner drives the AD* lines during the reply phase of a write transfer.

5.2.2.2 PAR* (parity)

The parity signals on the address/data bus signal group are PAR<3 . . 0>*. The parity signals help ensure the integrity of address and data information transferred on the AD* lines. Each parity signal carries an even-parity bit for one byte of a 4-byte (32-bit) address/data bus. An agent that drives less than 4 bytes need not place parity for the unused bytes onto the bus. The parity signals are assigned to bytes as follows:

- PAR0* —parity for AD<7 . . 0>*
- PAR1* —parity for AD<15 . . 8>*
- PAR2* —parity for AD<23 . . 16>*
- PAR3* —parity for AD<31 . . 24>*

Even parity means that when the number of asserted bits within a byte on the bus (not including the parity bit) is even, the corresponding parity signal is negated. If the number is odd, the parity signal is asserted.

5.2.3 System control signal group

The system control signal group on the PSB consists of ten signals, SC<9 . . 0>*, that provide control between agents during transfer operations. Agents use the SC* signals to define commands or report status, depending on the phase of the transfer operation.

During the request phase of a transfer operation, the bus owner drives the SC* signals providing command information to the replying agent(s).

During the reply phase of a transfer operation (not broadcast messages), the bus owner drives the SC<9,3 . . 0>* signals and the replying agent drives the SC<8 . . 4>* signals. The SC* signals provide for handshake and the transfer of status information between the bus owner and the replying agent(s). During the reply phase of broadcast messages, the bus owner continues to drive all SC* signals.

The following subclauses describe the functions of each SC* signal during the request and reply phases. Table 5.2-2 summarizes the functions of the SC* signals. Tables 5.2-3 and 5.2-4 list the request phase and reply phase functions for each of the SC* signals.

5.2.3.1 SC0* (request phase or reply phase)

SC0* is always driven by the bus owner and serves the same function during the request and reply phases of a transfer operation. When asserted, SC0* indicates to all agents that the bus owner is in the request phase of a transfer operation and that the address/data bus and the SC* lines contain valid request phase information. When negated, SC0* indicates to all agents that the bus owner is in the reply phase of the transfer operation.

5.2.3.2 SC1* (lock)

SC1* is always driven by the bus owner and serves the same function during the request and reply phases of a transfer operation. When the bus owner asserts SC1*, pending bus requests are unheeded and the bus owner retains ownership of the bus. Agents in the resolution phase remain there until the current bus owner relinquishes the bus. Additionally, when SC1* is asserted, all multiported resources on the PSB that are accessed during the time that the bus owner asserts SC1* are locked from access by other devices. The locked resources are dedicated to the current bus owner until SC1* is negated.

Table 5.2-2 —Summary of functions of SC* signals

Signal	Function during request phase	Function during reply phase
SC0*	Request phase	Reply phase
SC1*	LOCK	LOCK
SC2*	Data width	End-of-transfer
SC3*	Data width	Bus owner ready
SC4*	Address space	Replying agent ready
SC5*	Address space	Agent status
SC6*	Read/write data transfer	Agent status
SC7*	Reserved	Agent status
SC8*	Even parity on SC<7 . . 4>*	Even parity on SC<7 . . 4>*
SC9*	Even parity on SC<3 . . 0>*	Even parity on SC<3 . . 0>*

Table 5.2-3 —Functions of SC* during request phase

Signal	Driving agent	Functions during request phase	Meaning
SC0*	Bus owner	Identify transition between request phase and reply phase of the transfer operation.	L = request phase
SC1*	Bus owner	Locked access	L = lock active H = lock inactive
SC<3 . . 2>*	Bus owner	Identify the width of the data during the transfer operation.	SC3* H H L L SC2* H = 8-bit transfers L = 16-bit transfers H = 24-bit transfers L = 32-bit transfers
SC<5 . . 4>*	Bus owner	Identify the address space for the transfer operation.	SC5* H H L L SC4* H = memory space L = I/O space H = message space L = interconnect space
SC6*	Bus owner	Identify the type of data transfer in the reply phase.	L = write data transfer H = read data transfer
SC7*	Bus owner	Not used; must be high.	
SC8*	Bus owner	Provide even parity for SC<7 . . 4>*:	L = odd number of lows on SC<7 . . 4>* H = even number of lows on SC<7 . . 4>*
SC9*	Bus owner	Provide even parity for SC<3 . . 0>*:	L = odd number of lows on SC<3 . . 0>* H = even number of lows on SC<3 . . 0>*

Table 5.2-4 —Functions of SC* during reply phase

Signal	Driving agent	Functions during request phase	Meaning																																				
SC0*	Bus owner	Identify transition between request phase and reply phase of the transfer operation.	H = not request phase																																				
SC1*	Bus owner	Locked access	L = lock active H = lock inactive																																				
SC2*	Bus owner	Place an end-of-transfer indication onto the bus.	L = end-of-transfer for transfer operation H = not end-of-transfer																																				
SC3*	Bus owner	Provide a bus owner ready indication on the bus (part of the reply phase handshake).	L = agent ready for data transfer operation H = agent not ready																																				
SC4*	Replying agent	Provide a replying-agent-ready indication on the bus (part of the reply phase handshake).	L = agent ready for data transfer operation H = agent not ready																																				
SC<7 .. 5>*	Replying agent	Place an agent status indication onto the bus.	<table> <tr> <th>SC7*</th><th>SC6*</th><th>SC5*</th><th>Description</th></tr> <tr> <td>L</td><td>L</td><td>L</td><td>Reserved.</td></tr> <tr> <td>L</td><td>L</td><td>H</td><td>Reserved.</td></tr> <tr> <td>L</td><td>H</td><td>L</td><td>Agent data error.</td></tr> <tr> <td>L</td><td>H</td><td>H</td><td>NACK; Replying agent cannot respond because of a temporarily unavailable resource.</td></tr> <tr> <td>H</td><td>L</td><td>L</td><td>Transfer not understood error; multiple errors sensed.</td></tr> <tr> <td>H</td><td>L</td><td>H</td><td>Continuation error; replying agent is unable to continue.</td></tr> <tr> <td>H</td><td>H</td><td>L</td><td>Width error; width of transfer is not compatible with replying agent.</td></tr> <tr> <td>H</td><td>H</td><td>H</td><td>No error; transfer completed without errors.</td></tr> </table>	SC7*	SC6*	SC5*	Description	L	L	L	Reserved.	L	L	H	Reserved.	L	H	L	Agent data error.	L	H	H	NACK; Replying agent cannot respond because of a temporarily unavailable resource.	H	L	L	Transfer not understood error; multiple errors sensed.	H	L	H	Continuation error; replying agent is unable to continue.	H	H	L	Width error; width of transfer is not compatible with replying agent.	H	H	H	No error; transfer completed without errors.
SC7*	SC6*	SC5*	Description																																				
L	L	L	Reserved.																																				
L	L	H	Reserved.																																				
L	H	L	Agent data error.																																				
L	H	H	NACK; Replying agent cannot respond because of a temporarily unavailable resource.																																				
H	L	L	Transfer not understood error; multiple errors sensed.																																				
H	L	H	Continuation error; replying agent is unable to continue.																																				
H	H	L	Width error; width of transfer is not compatible with replying agent.																																				
H	H	H	No error; transfer completed without errors.																																				
SC8*	Replying agent	Provide even parity for SC<7 .. 4>*:	L = odd number of lows on SC<7 .. 4>* H = even number of lows on SC<7 .. 4>*																																				
SC9*	Bus owner	Provide even parity for SC<3 .. 0>*:	L = odd number of lows on SC<3 .. 0>* H = even number of lows on SC<3 .. 0>*																																				

5.2.3.3 SC2* (data width or end-of-transfer operation)

SC2* changes functions depending on the phase of the transfer operation.

During the request phase, the bus owner uses SC2* (with SC3*) to tell the replying agent(s) the data-width format for the ensuing data transfer. The bus owner codes SC<3 . . 2>* to select one of four possible data widths: 8-, 16-, 24-, or 32-bits. Table 5.2-3 shows the four codes and the associated data widths.

During the reply phase, the bus owner asserts SC2* to inform all agents that the current data transfer is the last in the transfer operation, the end-of-transfer (EOT).

5.2.3.4 SC3* (data width or bus owner ready)

SC3* changes functions depending on the phase of the transfer operation.

During the request phase, the owner uses SC3* (with SC2*) to tell the replying agent(s) the data-width format for the ensuing data transfer. The bus owner drives SC<3 . . 2>* to select one of four possible data widths: 8-, 16-, 24-, or 32-bits. Table 5.2-3 shows the encoding of each data width.

During the reply phase, the bus owner asserts SC3* to notify the replying agent when it is ready to send or receive data. SC3* and SC4* provide a two-directional handshake during the reply phase of the transfer operation.

5.2.3.5 SC4* (address space or replier ready)

SC4* changes functions depending on the phase of the transfer operation.

During the request phase, the bus owner drives SC4* (with SC5*) to select which address space will be used for the impending data transfer. For a read transfer, the address space is a source of data; for a write sequence, the address space is a destination. The bus owner has four address space options from which to choose: memory, I/O, message, or interconnect. Table 5.2-3 shows the four codes available using SC4* and SC5* and the associated address space selection.

During the reply phase, the replying agent asserts SC4* to indicate to the bus owner that it is ready to send or receive data. SC4* and SC3* provide a two-directional handshake during the reply phase of the transfer operation.

5.2.3.6 SC5* (address space or agent status)

SC5* changes functions depending on the phase of the transfer operation.

During the request phase, the bus owner drives SC5* and SC4* to select which address space will be used for the impending data transfer. For a read transfer, the address space is a source of data; for a write sequence, the address space is a destination. Table 5.2-3 shows the four codes available and the associated address space selection.

During the reply phase, SC5* is driven (with SC<7 . . 6>*) by the replying agent to provide agent status information. Table 5.2-4 shows the assigned agent status codes.

5.2.3.7 SC6* (read-write or agent status)

SC6* changes functions depending on the phase of the transfer operation.

During the request phase, the bus owner uses SC6* to indicate the direction of data flow for the transfer. When the bus owner negates SC6*, the reply phase of the transfer operation is a read data transfer. When the bus owner asserts SC6*, the transfer operation is a write data transfer.

During the reply phase, SC6* is driven with SC<7,5>* by the replying agent to provide agent status information. Table 5.2-4 shows the assigned agent status codes.

5.2.3.8 SC7* (reserved or agent status)

SC7* changes functions depending on the phase of the transfer operation.

During the request phase, SC7* is not used and must be negated by the bus owner.

During the reply phase, SC7* is driven with SC<6 . . 5>* by the replying agent to provide agent status information. Table 5.2-4 shows the assigned agent status codes.

5.2.3.9 SC8* (parity on SC<7 . . 4>*)

The function of SC8* remains the same in both the request phase and the reply phase of the transfer operation. Agents use SC8* to provide even parity for the SC<7 . . 4>* signals. If SC<7 . . 4>* contain an odd number of asserted signals on the bus, then the agent driving SC<7 . . 4>* must also assert SC8* on the bus.

5.2.3.10 SC9* (parity on SC<3 . . 0>*)

The function of SC9* remains the same in both the request phase and the reply phase of the transfer operation. Agents use SC9* to provide even parity for the SC<3 . . 0>* signals. If SC<3 . . 0>* contain an odd number of asserted signals on the bus, then the agent driving SC<3 . . 0>* must also assert SC9* on the bus.

5.2.4 Exception operation signal group

The PSB provides the exception operation signal group for passing indications of exception errors to all agents. This group contains two signals: bus error (BUSERR*) and bus time out (TIMOUT*). Each of these signals is on a dedicated line. They are defined in the following subclauses.

5.2.4.1 BUSERR* exception

An agent asserts bus error to indicate the detection of a data integrity problem during a transfer. Problems reported through the BUSERR* signal are: detection of a parity error on the valid bytes of the AD* lines or the SC<7 . . 0>* lines, a bus owner reply phase termination, or a protocol violation occurring during the reply phase. BUSERR* is received by all agents and can be asserted by any agent capable of detecting transfer integrity problems.

5.2.4.2 TIMOUT* exception

An agent monitors TIMOUT* to determine when a bus time-out condition occurs. Bus time out on the PSB is a result of the CSM determining that an agent is taking too much time to assert a handshake signal on the bus. TIMOUT* may also be generated by the CSM after determining that the bus owner has maintained ownership for an excessive length of time, which is user defined. TIMOUT* is received by all active agents on the bus and is generated from time-out circuitry located on the CSM.

5.2.5 Central control signal group

The central control signal group provides bus status and control information for devices operating on the PSB. This signal group contains seven signals. They are as follows:

- Reset (RST*)
- Reset not complete (RSTNC*)
- DC power low (DCLOW*) Protect (PROT*)
- Bus clock (BCLK*)
- Central clock (CCLK*)
- ID latch (LACHn*)

Each of the signals is described in more detail in the following subclauses. Electrical specifications of the signals are contained in clause 7.

5.2.5.1 Reset (RST*)

The CSM asserts RST* as a system-level initialization signal to all agents. Agents may monitor RST* along with DC power low (DCLOW*) and protect (PROT*) to distinguish between the types of reset: warm-start, cold-start or power failure recovery (see 6.5.2).

5.2.5.2 Reset not complete (RSTNC*)

Agents may assert RSTNC* during reset to extend the initialization time period provided by the CSM. RSTNC* is an OR-tied signal that is low when one or more agents on the PSB have not completed their reset requirements. Agents cannot perform bus operations while RSTNC* is asserted. The rstnc* signal must be driven by any agent that needs more initialization time than the RST* signal provides. The rstnc* signal should be negated by the agent as soon as possible, as no bus operations can be performed until the RSTNC* signal is negated. RSTNC* is received by all bus agents.

5.2.5.3 DC power low (DCLOW*)

The CSM asserts DCLOW* to all agents as a warning of an imminent power failure. The CSM monitors the power level from the power supply and holds DCLOW* high during normal system operation. When the power supply identifies a power failure for the CSM, the CSM asserts DCLOW* to indicate to all agents that a power failure is imminent. Systems use DCLOW* as the control signal for enabling back-up power and storage. DCLOW* is asynchronous to the bus clock.

5.2.5.4 Protect (PROT*)

The CSM asserts PROT* as a power-failure protection signal to other system agents. PROT* is a delayed result of the CSM sensing a power failure and indicates that system-supplied voltages are not within tolerance. Agents in the system that contain battery-backed resources receive PROT*. The CSM negates PROT* after ac line voltage is stable and system voltages are available. PROT* is asynchronous to the bus clock.

5.2.5.5 Bus clock (BCLK*)

Only the CSM generates BCLK* and all agents in a system receive BCLK*. An agent uses BCLK* to drive the PSB state machines. BCLK* has a fixed frequency of 10 MHz on the bus for normal operation, though the frequency can be reduced to dc for testing of the PSB. Agents should be designed to operate at BCLK* frequencies from 10 MHz to dc. The falling edge of BCLK* provides all system timing references.

5.2.5.6 Central clock (CCLK*)

CCLK* is generated by the CSM and may be used for additional timing among agents on the PSB. CCLK* operates with a specific relationship to BCLK* and at twice the frequency.

5.2.5.7 ID latch (LACHn*)

An agent uses its ID latch signal only during a reset. The LACHn* signal is driven by the CSM and serves two functions: it tells the agent when to latch the arbitration ID and when to latch the cardslot ID from the ARB<4 . . 0>* lines, depending on the state of ARB5*.

The ID latch signal is called LACHn* (where “n” is the cardslot to which the ID is assigned). Each of the cardslots, except 0, contains a LACHn* signal that is activated when the CSM drives one of the address/data lines (AD<20 . . 1>*).

For example, the agent in cardslot 2 accepts its ID when the CSM asserts AD2* which is connected on the backplane to LACHn* at cardslot 2. This is shown in figure 5.2-1.

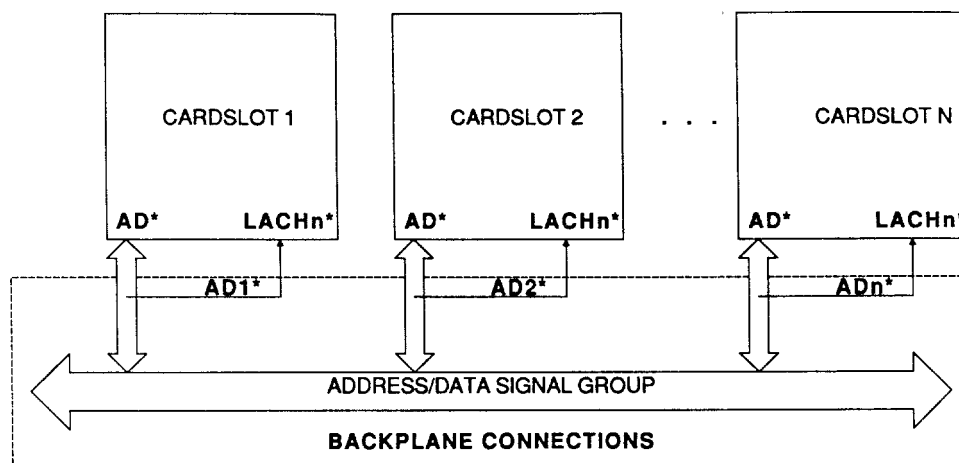


Figure 5.2-1 —Backplane connection of LACHn*

5.2.6 Power

The PSB provides power for bus agents. The system power includes dc power at +5 V, +12 V, -12 V, ground, and facilities for +5 V battery backup.

6. PSB protocol

6.1 General

This clause describes the communication protocol for the PSB. The communication protocol among agents on the PSB is consistent for all bus operations. All agents implement the protocol by monitoring and controlling the signals within the five signal groups described in clause 5. The description of the protocol is organized according to the three general bus operations that occur: the arbitration operation, the transfer operation, and the exception operation. Following the discussion of bus operations, the systemwide functions of the PSB are presented.

The protocol for all bus operations is presented in two stages, first in timing diagrams and then in state-flow diagrams. The timing diagrams provide a time-ordered representation of bus operations by showing the transition relationships among the signals that define the bus protocol. The timing diagrams illustrate the synchronous nature of the PSB by showing the bus clock cycles as vertical lines. The bus clock provides a common signal for all bus agents and is used to synchronize bus operations.

State-flow diagrams present the lowest level of detail in defining the protocol responsibilities of each agent in an operation. The state-flow diagrams list signal conditions required for each state-transition in an operation.

Figure 6.1-1 shows the three types of bus operation and the signal groups that create the protocol for each operation. The three types of bus operation are as follows:

- a) Arbitration operation
- b) Transfer operation

c) Exception operation

These three bus operations are discussed in detail in the following subclauses.

6.2 Arbitration operation

An agent that wishes to transfer data on the PSB must participate in a bus request sequence and participate in at least one arbitration operation. During an arbitration operation, one or more agents arbitrate for control of the bus. As a result of a single arbitration operation, one agent gains control of the bus and becomes the bus owner. Agents must gain control of the PSB before they can transfer data.

The arbitration operation performs two functions: first, it allows all requesting agents equal opportunity to access the bus, and second, it eliminates the possibility of more than one agent trying to transfer data on the bus at any one instant. In a case where more than one agent requests access to the bus at the same instant (or during the same bus request sequence), those agents that are refused access to the bus engage in additional arbitration operations until all requesting agents are granted bus access, in a sequential fashion, based on their priority.

6.2.1 Agent arbitration ID and cardslot ID assignment

During reset, the CSM in each system assigns cardslot IDs (1 through 20) to each agent and assigns arbitration IDs to each agent on the bus. The sequence involves the use of the arbitration ID lines

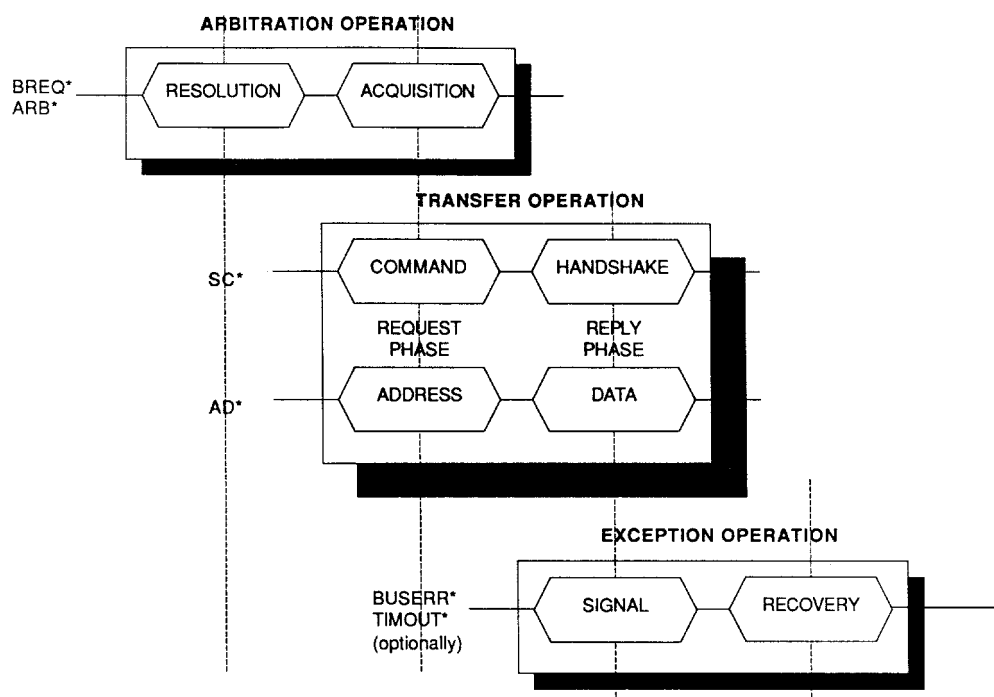


Figure 6.1-1 —Bus operation details

(ARB*), the address/data bus lines (AD<20 .. 1>*), the ID latch signal (LACHn*), and the reset signal (RST*). When ARB5* is asserted, the CSM drives cardslot IDs on ARB<4 .. 0>*; when ARB5* is negated, the CSM drives arbitration IDs on ARB<4 .. 0>*.

With each arbitration/cardslot ID, the CSM drives a corresponding address/data line (connected to the LACHn* signal for each cardslot) to identify the destination for the ID. The CSM asserts one and only one of the AD* lines at any time during the assignment sequence. Each agent uses its LACHn* line (connected to one of the address/data lines) to identify the ID from the ARB<4 . . 0>* lines. Whenever an agent detects RST*, LACHn* and ARB5* asserted, the agent must latch the values on ARB<4 . . 0>* for use as its cardslot ID. Whenever an agent detects RST* and LACHn* asserted and ARB5* negated, the agent must latch the values on ARB<4 . . 0>* for use as its arbitration ID.

Figure 6.2-1 details the relationship among RST*, ARB*, and LACHn*. The LACHn* signal remains asserted for at least one bus clock cycle. The arbitration or slot ID is guaranteed to be stable for at least one clock cycle before and after the LACHn* signal is active. Refer to 6.5 for more information on the assignment of cardslot and arbitration IDs.

6.2.2 Bus arbitration priority

All agents in a system share access to six signal lines on the bus, ARB<5 . . 0>*. These signals provide the mechanism that allows agents to decide among themselves which agent has first access rights to the bus. At the beginning of a resolution phase, each agent desiring access to the bus places its arbitration ID and priority onto its arb* lines. The arbitration operation signal group supports two different priorities at which an agent can request use of the bus: normal priority (order of access determined by the arbitration ID on ARB<4 . . 0>*) and high priority (access based on condition of ARB5* and arbitration ID on ARB<4 . . 0>*). It is possible that the implementations could utilize different values of ARB<4 . . 0>* for high and normal priority arbitration. The protocol for each priority method is discussed in the next subclauses.

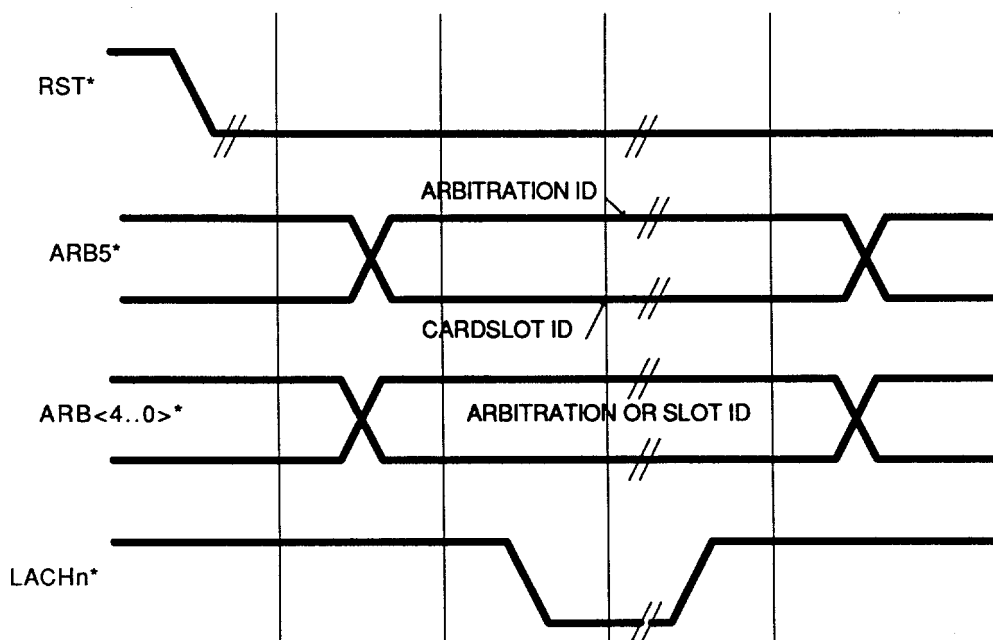


Figure 6.2-1 —Reset sequence

6.2.2.1 Normal priority

An agent places a normal priority request by asserting its breq*, holding arb5* inactive, and placing its arbitration ID onto arb<4 . . 0>*. Then each agent that requests access to the bus determines whether it is making the highest priority request (higher arbitration IDs have higher priority—see table 6.5-1) by monitoring the condition of all arbitration lines on the bus.

At the beginning of the resolution phase, each agent requesting the bus must place its arbitration ID on its arb(4 . . 0)* lines. If, for the given agent, the arbitration ID that it asserts does not match the ID present on the ARB* lines, the agent stops driving the arbitration ID lines below the bit that doesn't match (toward the LSB). The agent continues to compare its arb* lines to the ARB* lines throughout the resolution phase. The settling time for the ARB* lines is a maximum of three bus clock cycles, but the driving of the arb* lines by requesting agents continues until the bus ownership can be exchanged. At the end of the resolution phase, the agent whose arbitration ID matches the ARB* lines becomes the bus owner and removes its breq* and arb* signals. Agents whose arbitration ID lines do not match the ARB* lines continue into the resolution phase of the next arbitration operation. One arbitration operation is executed for each agent that receives access to the bus. To service all agents that have requested bus access, a series of arbitration operations are executed. This series of arbitration operations, during which all requesting agents receive bus ownership access sequentially, based on their priority, is called a bus request sequence.

Agents making a normal priority request may request bus access and enter a bus request operation only when no bus request sequence is in progress, as indicated by the negation of BREQ*. All agents engaged in the current bus request sequence must be given bus access, and the current bus request sequence must end before a new bus request sequence can begin. This ensures that no normal priority agent in a given bus request sequence is locked out of bus access by another normal priority agent.

6.2.2.2 High priority

If an agent requires urgent bus operations, that agent uses the high priority feature. An agent's high priority request does not wait for the negation of BREQ* before driving its arb* lines. However, if BREQ* is asserted, an agent with a high priority request must wait until the beginning of the next resolution phase to drive its arb* lines. In this case, the beginning of the next resolution phase immediately follows an EOT status transfer without a lock indication. If BREQ* is not asserted at the time of a high priority request, the requesting agent enters a resolution phase in the same manner as agents with normal priority requests.

Once having entered the resolution phase, an agent places a high priority request by asserting arb5* while placing its arbitration ID onto its arb(4 . . 0)* lines. By asserting arb5*, an agent guarantees itself access to the bus before any simultaneous or pending requests from other agents asserting a normal priority request. When more than one agent simultaneously places a high priority request, (asserts arb5*) the agent with the higher priority specified by its arbitration ID gains first access.

Table 6.2-1 summarizes the interaction between agents of different priority levels on the bus.

Table 6.2-1 —Arbitration priority protocol

Priority level of Agent A's request	Priority level of Agent B's request	Next bus owner	Description
Normal	Normal	Either	Arbitration based on ARB(4 . . 0)* to select next bus owner.
Normal	High	Agent B	Allow the agent making the high priority request to be the next bus owner.
High	Normal	Agent A	Allow the agent making the high priority request to be the next bus owner.
High	High	Either	Arbitrate among high priority agents based on ARB(4 . . 0)*

6.2.3 Bus ownership

If, at the end of the resolution phase, an agent recognizes that its arbitration ID matches the ID on the bus, the agent begins an acquisition phase.

During the acquisition phase, the bus owner performs transfer operations and all agents with pending requests begin arbitrating again to determine the next bus owner. During the last handshake of the transfer operation, the bus owner asserts SC2*. If another agent has won access rights to the bus, then the current bus owner releases control of the bus and the new bus owner immediately enters the bus acquisition phase.

A bus owner can perform several consecutive transfer operations by asserting lock via SC1*. By asserting SC1*, the bus owner guarantees itself exclusive use of the bus until it negates SC1*; other agents are prohibited from gaining ownership of the bus while SC1* is asserted.

Figure 6.2-2 shows the timing sequence for an operation during which the bus owner has locked the bus. While SC1* is asserted, the bus owner may perform several consecutive transfer operations without interruption from any other device.

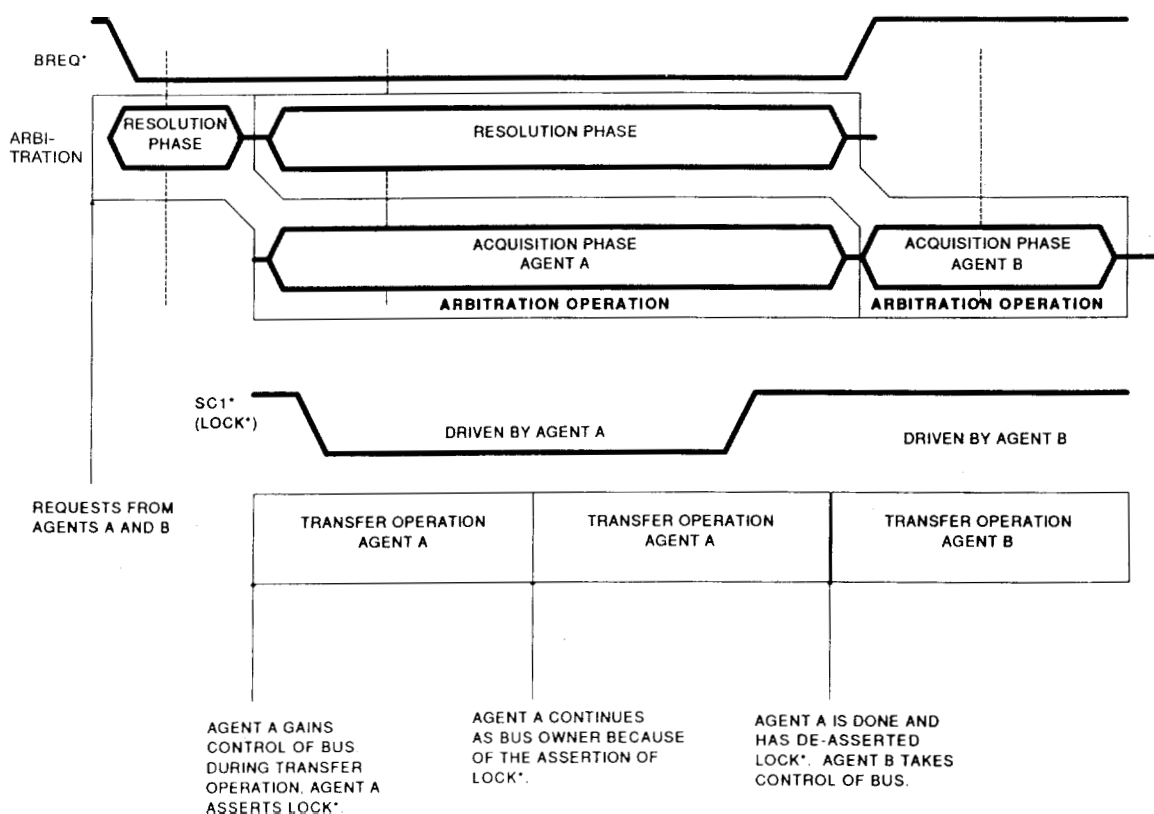


Figure 6.2-2 —Timing sequence with bus locked

6.2.4 Parking and bus release

A bus owner releases control of the bus if it has completed its transfer operation, SC1* (lock) is not asserted, and another agent has entered the resolution phase of an arbitration operation. If the current bus owner is the last agent in the bus request sequence to gain access to the bus, then as it negates breq*, the BREQ* signal goes inactive on the bus. The negation of BREQ* allows all agents to participate in the next bus request sequence. As the bus owner completes its operation, the EOT status transfer on the bus allows the next bus owner to assume ownership of the bus.

If no other agents request access to the bus, the last agent to be the bus owner retains ownership of the bus. As such, this agent can perform another transfer operation without arbitrating for access to the bus. This condition is known as parking.

If a bus owner performs multiple data transfers while other agents are in the resolution phase of an arbitration operation, the resolution phase is extended until the EOT status transfer occurs. The occurrence of the EOT status transfer permits the new bus owner to enter the acquisition phase and perform transfer operations.

Figure 6.2-3 shows the three conditions that must be satisfied to exchange ownership of the bus: a pending request for the bus, no transfer operation in progress, and SC1* (lock) negated.

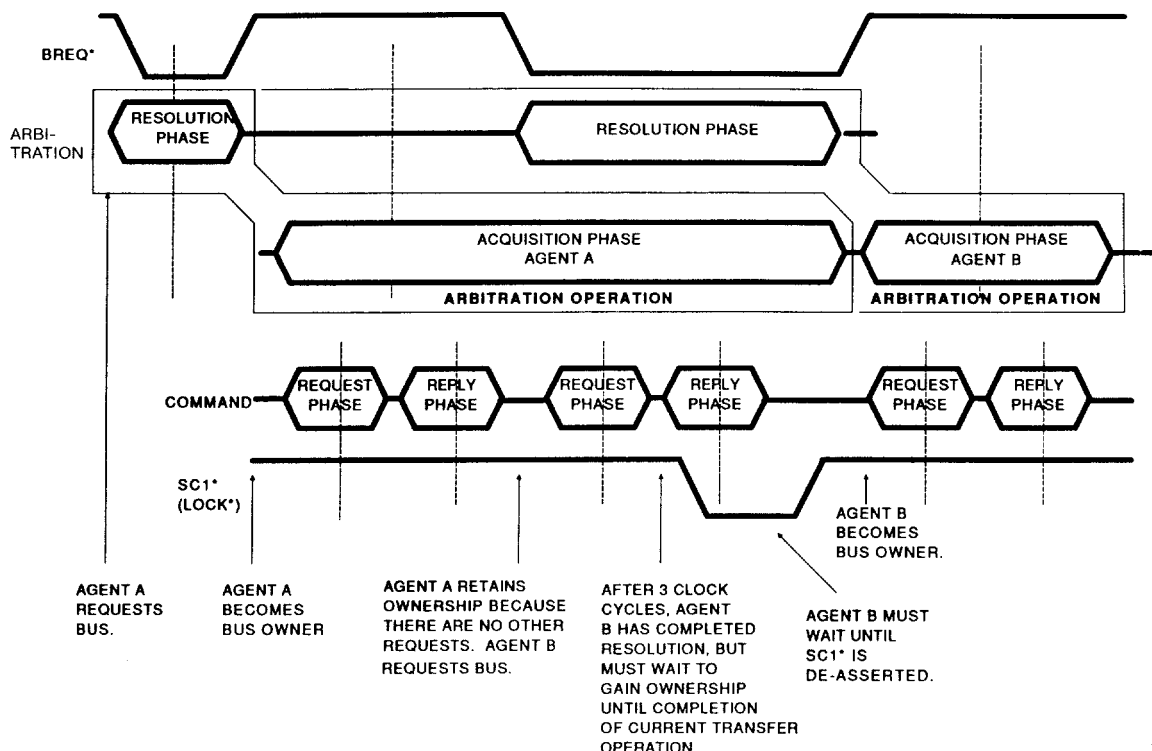


Figure 6.2-3 —Timing sequence for bus release

6.2.5 Arbitration priority example

Figure 6.2-4 shows two consecutive bus request sequences during which four agents (Agent A, Agent B, Agent C, and Agent x) make normal priority requests for bus access assuming an idle bus. These two bus request sequences occur as described in the following steps:

- Agents A, B, and C make normal priority requests for access to the bus by asserting their breq* lines and placing their arbitration IDs onto their arb4 . . 0)* lines with arb5* negated.
- The breq* signals from the three agents are ORed together and placed on the bus as BREQ*, thus closing the current bus request sequence to any additional normal priority requests.
- The first arbitration operation begins. During the resolution phase, Agent A is determined to have the next bus ownership by virtue of its arbitration ID and is given access rights to the bus.
- Agent A now negates its breq* and arb*, enters the acquisition phase, and begins the transfer operation over the PSB. As the first arbitration operation enters its acquisition phase, the second arbitration operation begins its resolution phase with Agents B and C. Note that the acquisition phase of the first arbitration operation occurs coincident with the resolution phase of the second arbitration operation.
- During the resolution phase of the second arbitration period, agent X desires to make a normal priority bus request. However, a bus request sequence is currently in progress, and this is by the assertion of BREQ*.

Agent X is not permitted to make the normal priority bus request or engage in arbitration until the current bus request sequence is completed and all current requesting agents have been satisfied.

- f) At the end of the resolution phase of the second arbitration operation, Agent B is determined to have next bus ownership, and is given access rights to the bus. Agent B negates its $breq^*$ and arb^* after Agent A has relinquished bus ownership. Agent B then enters the acquisition phase and begins the transfer operation over the PSB.

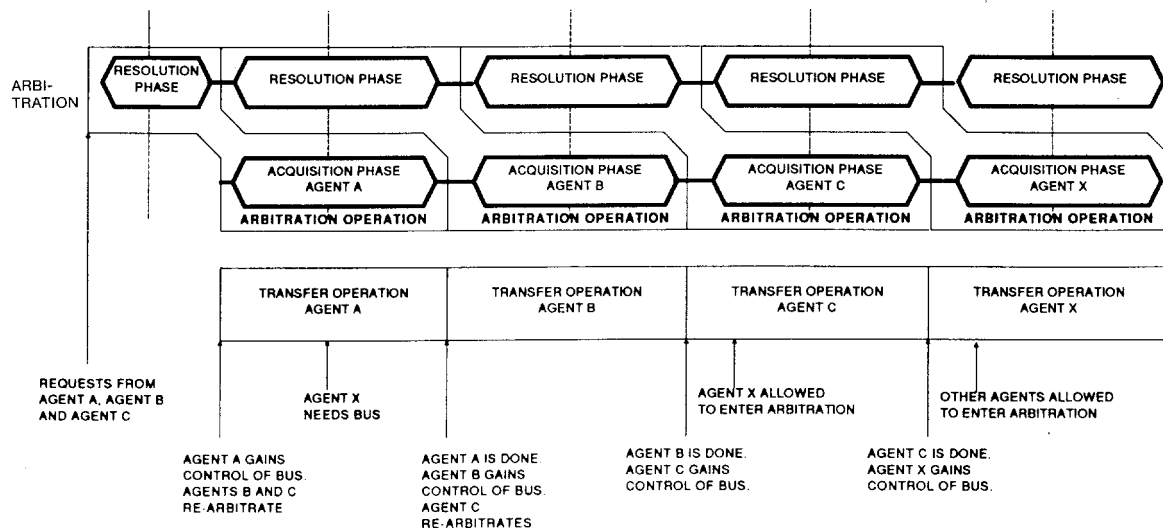


Figure 6.2-4 —Typical arbitration operation sequence

- g) As Agent B enters the acquisition phase described in the last step, Agent C enters the resolution phase of the third arbitration operation.
- h) At the end of the resolution phase, Agent C wins access rights to the bus. As Agent B finishes its transfer operation, it relinquishes control of the bus and Agent C enters the acquisition phase of the third arbitration operation and negates its $breq^*$ and arb^* .
- i) The negation of $breq^*$ by Agent C results in the negation of $BREQ^*$ on the bus. This indicates to all bus agents that the current bus request sequence has ended. At this point, any agent requiring access to the PSB (excluding Agent C, which currently has access) can make a bus request in the normal fashion described previously. In this example, Agent X makes a normal priority bus request by placing its arbitration ID on $arb\langle 4 \dots 0 \rangle^*$ and asserts $breq^*$.
- j) Agent X enters the resolution phase of the first arbitration operation of the new bus request sequence. Agent C is performing its transfer operation over the PSB.
- k) Agent X wins the resolution phase of the arbitration operation and is given access rights to the bus. As Agent C completes its transfer operation, Agent X enters the acquisition phase and negates its $breq^*$ and arb^* . Agent X then begins the transfer operation.

In the previous example, if Agent X had entered a high priority bus request, it would have entered the very next resolution phase of the first bus request sequence. Agent X would have been granted bus access in place of Agent C, and Agent C would have been the last agent to receive bus access. In this case, there would have been only one bus request sequence, containing four arbitration operations, providing bus access to four bus agents, although only three bus agents requested the bus at the beginning of the bus request period.

6.2.6 Resolution algorithm example

As described earlier, a bus agent uses its arbitration for bus access. The function of the arbitration ID is described further in the following paragraphs.

When an agent attempts to gain control of the bus, the agent places its arbitration ID onto the bus and compares its own ID, bit-by-bit, with the ID present on the bus. At the bit location where the agent ID does not match the ID on the bus, the agent uses a back-off algorithm. If an agent asserts an arbitration line, that signal level is dominant in forming the bus ID; the bus ID consists of the “wired OR” function performed on all agent IDs. Figure 6.2-5 shows an example of the arbitration circuit on a requesting agent. Note that the value of the ARB* lines affect the value driven by the agent on its arb* lines.

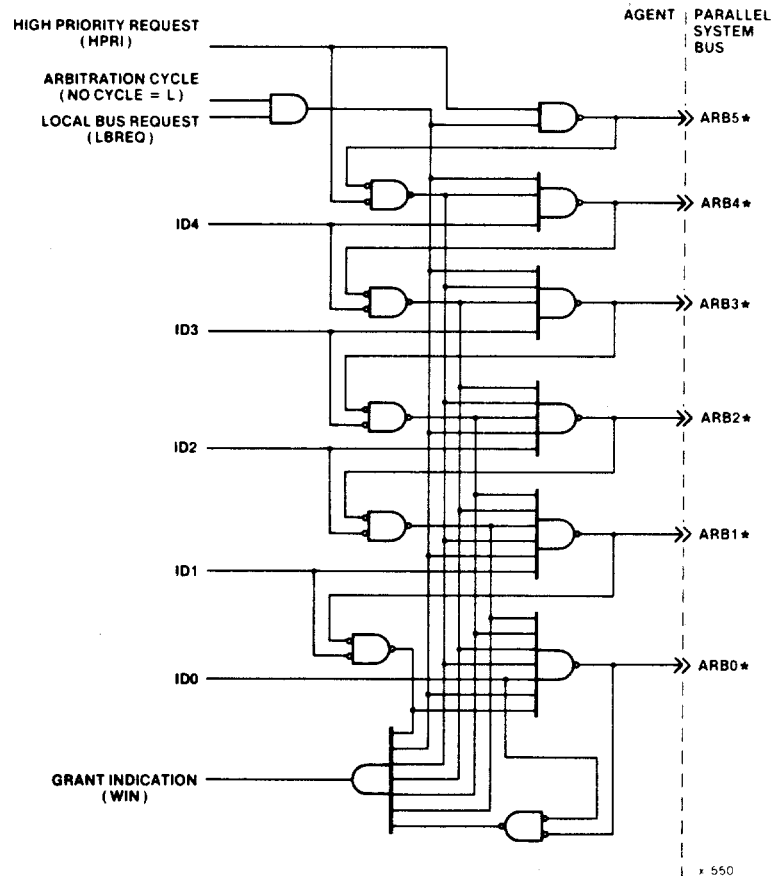


Figure 6.2-5 —Arbitration ID interface example (at each requesting agent)

6.2.6.1 Example 1

To understand how the back-off algorithm works, consider the following example. This example describes the resolution phase of an arbitration operation. Table 6.2-2 shows the values of the ARB* signals during this example. It is important to note that the periods in the table represent the settling time for the ARB* lines, not bus clock cycles. Three agents simultaneously make normal priority requests for bus access. At the start of the resolution phase, Agent 1 places a value of HLLLHL on arb<5..0>*, Agent 2 places HLLLHH, and Agent 3 places HLLHLL. The IDs for all three agents are listed in the Start column of table 6.2-2. The arbitration process is described in the following steps:

- Since these are all normal priority requests, ARB5* is negated. The arbitration ID present on the bus is the wired OR of the IDs being placed on the bus by the three agents.
- The agents begin comparing their IDs with the bus ID. During Period 1, agents move from the high-order to low-order bits comparing, each in turn. When an agent detects a match, it moves to the next lower bit (LSB) and compares.

- c) If the agent detects the assertion of a bit which it is not asserting, it immediately stops driving all bits below it (that is, from the mismatched bit toward the LSB, all bits are negated, denoted in the table by 'h'). This may cause an agent to drive its arb* lines with a value differing from its arbitration ID, which is referred to as a pseudo-ID. Each arbitrating agent creates its pseudo-ID in this fashion. The pseudo-IDs are driven onto the bus and compared throughout the resolution phase.

Table 6.2-2 —Arbitration ID comparison example

Settling periods are not related to bus clock cycles.			
Settling periods	Participants	Initial value	Resultant value
Period 1	Initial ARB* Agent 1 ID* = HLLLHL Agent 2 ID* = HLLLHH Agent 3 ID* = HLLHLL Resulting ARB*	HHHHHH	HLLLHL HLLLHH HLLHLL HLLLLL
Period 2	Previous ARB* Agent 1 arb* Agent 2 arb* Agent 3 arb* Resulting ARB*	HLLLLL	HLLHh Pseudo-ID lose HLLHh Lose HLLHh Pseudo-ID lose HLLLHH
Period 3	Previous ARB* Agent 1 arb* Agent 2 arb* Agent 3 arb* Resulting ARB*	HLLLHH	HLLLHL Win HLLLHH Win HLLHh Pseudo-ID lose HLLLHL
Period 4— Steady state, does not cause ARB* transitions	Previous ARB* Agent 1 arb* Agent 2 arb* Agent 3 arb* Resulting ARB*	HLLLHL	HLLLHL Win HLLLHH Lose HLLHh Pseudo-ID lose HLLLHL
NOTE — h = value in the corresponding bit position is deasserted by agent because it must drive a pseudo-ID.			

- d) After the first settling time for the ARB* lines agents on the bus have determined pseudo-IDs to be driven onto the PSB.
- e) After the pseudo-IDs calculated in Period 1 have settled on the bus, the ARB* lines have a new value. This causes all agents to determine new pseudo-IDs and drive them onto the bus.
- f) After the pseudo-IDs calculated in Period 2 have settled on the bus, the ARB* lines again have a new value. This process continues until the ARB* lines are stable (at most three bus clock cycles). Agent 1 has now determined that at the end of the resolution phase, it becomes the bus owner, and Agents 2 and 3 have determined that they are not to be the next bus owner.

- g) After Agent 1 takes possession of the bus, Agents 2 and 3 arbitrate for the next ownership. Note that the higher the value of the arbitration ID (lower value after inversion on the ARB* lines) the higher the priority.

6.2.6.2 Example 2

This example illustrates the timing sequence for normal priority bus acquisition (see figure 6.2-6). In this example there are three agents: A, B, and C. In terms of their arbitration IDs (those assigned during reset), Agent A obtains bus ownership first, followed by Agent B, then Agent C (last). The sequence is detailed in the following steps:

- a) During the same bus clock cycle both Agents A and B desire normal priority use of the bus. Since no bus request sequence is in progress ($BREQ^* = H$), both agents enter arbitration by asserting $breq^*$ and placing their arbitration IDs onto the $arb\langle 4 \dots 0 \rangle^*$. In both cases, $arb5^*$ is negated, indicating that both are normal priority requests.
- b) During this bus clock cycle and the next two, the ARB* lines settle according to the back-off algorithm described previously, resulting in a valid arbitration ID by the end of the third bus clock cycle.
- c) Meanwhile, during the current resolution phase just described, Agent C desires normal priority use of the bus. At the end of the third bus clock cycle in the current resolution phase, Agent A's arbitration ID is stable on ARB* indicating that it is the next bus owner.
- d) On the next bus clock cycle (fourth bus clock cycle), Agent A gets ownership of the bus. Agent A removes its arb^* and $breq^*$ signals and begins transfer operations. Agent B remains in the resolution phase, continues to assert $breq^*$, and drives its arbitration ID. Agent C is prevented from entering into arbitration with Agent B because Agent C has a normal priority request and a bus request sequence is currently active.
- e) By the end of the third bus clock cycle of the second resolution phase, Agent B's arbitration ID is stable on ARB* (indicating it is the next bus owner). However the second resolution phase is extended because the current transfer operation is not complete. When the current transfer operation completes, Agent A relinquishes bus ownership. Agent B becomes the bus owner and removes arb^* and $breq^*$ and begins transfer operations.
- f) Since all requesting agents participating in the bus request operation have obtained bus ownership, $BREQ^*$ is negated. Agent C, sensing that no bus request sequence is currently in progress, enters arbitration. Agent C progresses through a resolution phase and an acquisition phase as described above.

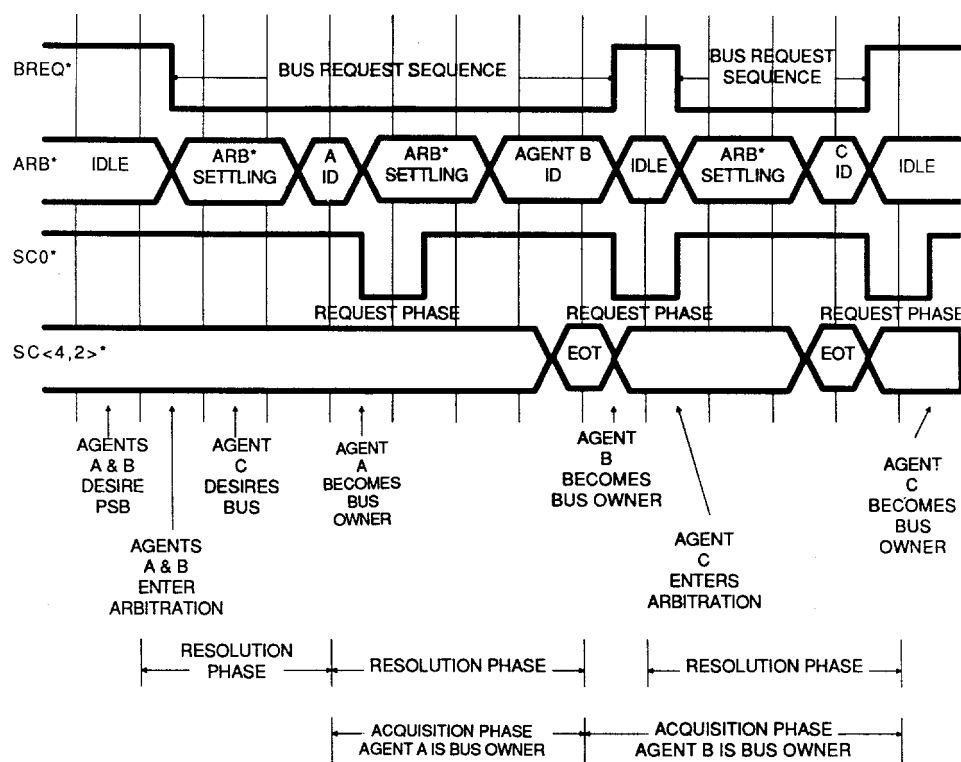


Figure assumes SC1* is negated

Figure 6.2-6 —Timing sequences for bus acquisition

6.2.6.3 Timing sequence for high priority bus acquisition

Figure 6.2-7 shows the timing sequence for high priority bus acquisition. In this example there are three agents: A, B, and C. In terms of their arbitration IDs (those assigned during reset) Agent A has the highest priority, Agent B next highest, and Agent C the lowest. The sequence of events for this bus acquisition (assuming an idle bus) is given in the following steps:

- In the same bus clock cycle, both Agents A and B desire normal priority use of the bus. Since no bus request sequence is in progress (BREQ* is negated), both agents place normal priority bus requests by asserting breq*, placing their IDs on arb<4..0>* with arb5* negated.
- During the first bus clock cycle and the next two, the ARB* lines settle according to the back-off algorithm described previously, resulting in steady-state ARB* lines by the end of the third bus clock cycle.
- Meanwhile, during the resolution phase described in the last step, Agent C desires high priority use of the bus. At the end of the third bus clock cycle, Agent A's arbitration ID is stable on the ARB* lines indicating that it is the next bus owner. During the next bus clock cycle, Agent A gets ownership of the bus. Agent A removes its arbitration ID from arb*, negates breq*, and begins transfer operations.
- Agent B remains in the resolution phase and continues to assert breq* and drive its arbitration ID. Agent C enters into arbitration with Agent B because Agent C has a high priority request and a new arbitration operation (and a new resolution phase) has begun. This is true even though the current bus request sequence has not ended. When Agent C enters the arbitration, it asserts its breq*, drives its arbitration on the arb<4..0>* lines with arb5* asserted, indicating a high priority request.
- By the end of the third bus clock cycle of the second arbitration operation, Agent C's arbitration ID is stable on the ARB* lines, indicating it is the next bus owner. This is true because Agent C placed a high priority request while Agent B placed a normal priority request. The resolution phase of the second arbitration

operation is extended because the current transfer operation is not complete. When the current transfer operation completes, Agent A relinquishes bus ownership and Agent C, entering the acquisition phase, becomes the new bus owner. As Agent C leaves the resolution phase, it negates its arbitration ID from the arb* (including arb5*) and breq*, and begins transfer operations.

- f) Agent B remains in the resolution phase with breq* asserted and its arbitration ID driven on the arb* lines. It then progresses through an arbitration operation (a resolution phase and an acquisition phase) as described previously.

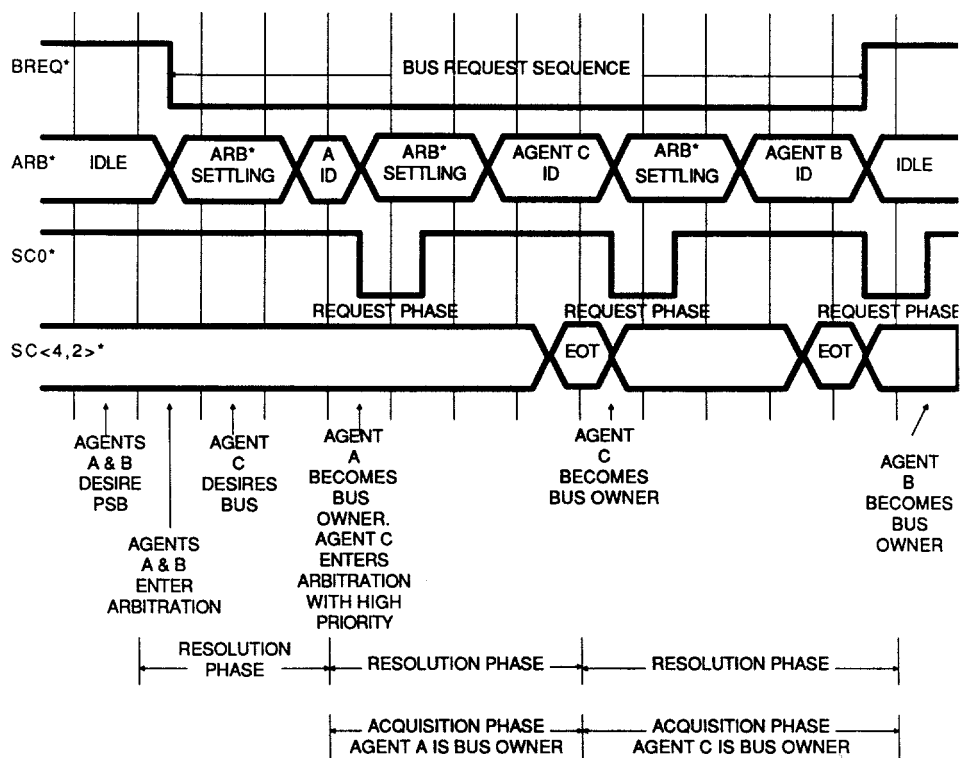


Figure assumes SC1* is negated

Figure 6.2-7 —Timing sequence for high priority bus acquisition

6.3 Transfer operation

An agent must complete a successful arbitration operation or be parked before it can initiate a transfer operation. On completing an arbitration operation, one agent gains ownership of the bus and starts a transfer operation immediately. Only the agent that owns the bus may conduct transfer operations on the PSB.

An agent initiates a transfer operation by manipulating signals within two signal groups on the PSB: the address/data bus signal group and the system control signal group.

The address/data bus signal group (AD* and PAR* signal lines) is a set of 36 signal lines that contains address information during the request phase and data information during the reply phase of a transfer operation. The function of the AD* lines change depending on the phase of the transfer operation.

The system control signal group (SC*) is a set of 10 signals that defines the specific operation to be performed within the transfer operation. As with the address/data bus signal group, the function performed by each of the SC* signals is dependent on the phase of the transfer operation.

In the request phase, the bus owner uses the system control signal group to specify for the replying agent(s) the parameters of the impending transfer operation. These parameters are the specific command or operation to be performed, the address space of the command, and the width of the data transfer. The bus owner also uses the address/data bus signal group to transfer the address for the operation during the request phase.

During the reply phase, the functions performed by the system control signal group are redefined. The protocol divides the system control signal group into two smaller groups (except for broadcast messages), one driven by the bus owner and one driven by the replying agent. These two signal groups permit both agents to successfully complete the transfer operation. The handshake signals synchronize the data transfer operation. The bus owner identifies the last data transfer by asserting SC2* (EOT), which concludes the transfer operation. The replying agent returns agent status to the bus owner with its part of the handshake. During the reply phase of broadcast messages, the bus owner drives all SC* lines.

6.3.1 Types of transfer operation

There are three types of data transfers: the single data transfer, the sequential data transfer, and the broadcast message. Each type implements a precise handshake sequence. Each is described further in the following subclauses.

6.3.1.1 Single data transfer

The single data transfer supports one data transfer.

The transfer may be 8-, 16-, 24- or 32-bits wide and is completed with one handshake. During the request phase, the bus owner places address information on the AD* lines. Parity for the address word is placed on PAR<3 . . 0>*. The bus owner uses the system control signal group to specify the data width, the address space, the phase (request), the transfer direction (read or write) and provide parity for the binary value present on the SC* lines.

During the reply phase, the bus owner negates SC0* to indicate that the request phase has completed and the reply phase is underway. The bus owner also asserts SC3* (bus owner ready) to indicate that it is ready to receive or send data (depending on whether the transfer operation is a read or write, respectively) and SC2* (EOT) to indicate that the current transfer is the last transfer. Additionally, the bus owner places parity for SC<3 . . 0>* on SC9*.

During the reply phase, the replying agent asserts SC4* (replying agent ready) to indicate that it is ready to send or receive data. The replying agent also places agent status on SC<7 . . 5>*, as well as parity for bits SC<7 . . 4>* on SC8*.

When SC<4 . . 2>* are asserted, the data transfer occurs and the transfer operation completes.

A single data transfer operation may be either a read (data transfer is from replying agent to bus owner) or a write (data transfer is from bus owner to replying agent) operation. The supplier of data, either the replying agent for a read operation or the bus owner for a write operation, must signal its part of the handshake status (SC3*) without waiting for the other agent.

During the reply phase of a read data transfer, the replying agent provides data on the AD* lines. The replying agent asserts SC4* (replying agent ready) when it has placed valid data on the AD* lines. The bus owner asserts SC2* (EOT) and SC3* (bus owner ready) when it is ready to accept data from the bus. When both SC3* and SC4* are asserted, the handshake occurs and the data transfer occurs synchronous with the falling edge of the bus clock.

Figure 6.3-1 shows a block diagram of the read data transfer and figure 6.3-2 shows the signal sequences for the read data transfer handshake.