

Implementation of
IEC 823:1990

Microprocessor system bus (VMSbus) — Serial sub-system bus of the IEC 821 bus (VMEbus)

Committees responsible for this British Standard

The preparation of this British Standard was entrusted by the Information Systems Technology Standards Policy Committee (IST/-) to Technical Committee IST/6, upon which the following bodies were represented:

- Association for Payment Clearing Services
- British Computer Society
- British Telecommunications plc
- Business Equipment and Information Technology Association
- Department of Trade and Industry (Information Technology Standards Unit)
- Department of Trade and Industry (National Physical Laboratory)
- Electricity Supply Industry in England and Wales
- Electronic Engineering Association
- HM Treasury (Central Computer and Telecommunications Agency)
- Information Technology Users' Standards Association
- Institute of Data Processing Management
- Inter-Universities Computing Committee
- Joint Network Team
- National Computing Centre Ltd.
- National Health Service
- OFTEL (Office of Telecommunications)
- Post Office
- Telecommunication Engineering and Manufacturing Association
- Telecommunications Managers' Association

This British Standard, having been prepared under the direction of the Information Systems Technology Standards Policy Committee, was published under the authority of the Board of BSI and comes into effect on 28 February 1991

© BSI 05-2000

The following BSI references relate to the work on this standard:
Committee reference IST/6
Draft for comment 90/68117 DC

ISBN 0 580 19343 8

Amendments issued since publication

Amd. No.	Date	Comments

Contents

	Page
Committees responsible	Inside front cover
National foreword	ii
Foreword	vii
Text of IEC 823	1

National foreword

This British Standard reproduces verbatim IEC 823:1990 and implements it as the UK national standard.

This British Standard is published under the direction of the Information Systems Technology Standards Policy Committee whose Technical Committee IST/6 has the responsibility to:

- aid enquirers to understand the text;
- present to the responsible international committee any enquiries on interpretation, or proposals for change, and keep UK interests informed;
- monitor related international and European developments and promulgate them in the UK.

NOTE International and European Standards, as well as overseas standards, are available from BSI Sales Department, BSI, Linford Wood, Milton Keynes, MK14 6LE.

A British Standard does not purport to include all the necessary provisions of a contract. Users of British Standards are responsible for their correct application.

Compliance with a British Standard does not of itself confer immunity from legal obligations.

Summary of pages

This document comprises a front cover, an inside front cover, pages i and ii, the IEC title page, pages ii to viii, pages 1 to 108 and a back cover.

This standard has been updated (see copyright date) and may have had amendments incorporated. This will be indicated in the amendment table on the inside front cover.

**NORME
INTERNATIONALE
INTERNATIONAL
STANDARD**

**CEI
IEC
823**

Première édition
First edition
1990-10

**Bus système à microprocesseurs (VMSbus) -
Bus sous-système série du Bus CEI 821
(VMEbus)**

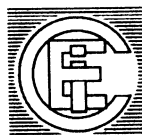
**Microprocessor system bus (VMSbus) -
Serial sub-system bus of the IEC 821 Bus
(VMEbus)**

© CEI 1990 Droits de reproduction réservés — Copyright — all rights reserved

Aucune partie de cette publication ne peut être reproduite ni
utilisée sous quelque forme que ce soit et par aucun pro-
cédé, électronique ou mécanique, y compris la photocopie et
les microfilms, sans l'accord écrit de l'éditeur.

No part of this publication may be reproduced or utilized in
any form or by any means, electronic or mechanical,
including photocopying and microfilm, without permission
in writing from the publisher.

Bureau Central de la Commission Electrotechnique Internationale 3, rue de Varembe Genève, Suisse



Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

CODE PRIX
PRICE CODE **XG**

*Pour prix, voir catalogue en vigueur
For price, see current catalogue*

Contents

	Page
Foreword	vii
Chapter 0: Scope	1
Chapter 1: Introduction to the IEC 823 VMSbus standard	
1.1 Serial bus objectives	2
1.1.1 IEC 823 VMSbus standard objectives	2
1.2 Serial bus interface system elements	2
1.3 Serial bus standard terminology	3
1.4 Standard organization	4
1.5 Standard relationship of the serial bus and a parallel backplane bus	4
Chapter 2: Serial bus overview	
2.1 Overview of the Physical Layer	6
2.1.1 Physical Layer modules and signals	6
2.1.2 Physical Layer signalling	7
2.2 Link Layer overview	9
2.2.1 Modules and interfaces	9
2.2.2 Module groups	10
2.2.3 Frame transmission protocol	10
2.3 Using the serial bus to transfer data	11
2.4 Using the serial bus to set and reset flags	11
2.5 Applications of serial bus module groups	11
2.6 Serial bus addressing capabilities	11
Chapter 3: Serial bus frames and subframes	
3.1 Basic frame types	13
3.2 The Header	15
3.3 The Frame Type subframe	16
3.4 The Data subframe	17
3.5 The Frame Status subframe	18
3.6 The Jam Detect subframe	19
3.7 Jam protocol	19
3.8 Jam on Reset	20
Chapter 4: The Physical Layer	
4.1 CLOCK SOURCE module	21
4.1.1 Medium interface	21
4.1.2 Physical and Link Layer interface	22
4.1.3 Operation	22
4.2 BRIDGE module	23
4.2.1 Media interfaces	23
4.2.2 Physical and Link Layer interfaces	25
4.2.3 Operation	25
4.3 BUS ACCESS module	26
4.3.1 Medium interface	26
4.3.2 Physical Layer interface	26
4.3.3 Link Layer interface	28
4.3.4 Operation	28
4.4 Electrical specifications	29
4.4.1 Input characteristics for the Backplane Medium	29
4.4.2 Driving and loading for SERCLK	29
4.4.3 Driving and loading for SERDAT*	30
4.4.4 Input characteristics for the Extended Medium	30

	Page
4.4.5 Driving the Extended Medium	30
4.4.6 Loading on the Extended Medium	32
4.4.7 Stress/damage resistance for the Extended Medium	32
Chapter 5: Link Layer modules	
5.1 State diagram notation	34
5.2 HEADER SENDER module	34
5.2.1 Physical Layer interface	35
5.2.2 Link Layer interface	35
5.2.3 Higher Layer interface	35
5.2.4 Initialization	36
5.2.5 Operation	37
5.3 HEADER RECEIVER module	39
5.3.1 Physical Layer interface	39
5.3.2 Link Layer interface	39
5.3.3 Higher Layer interface	39
5.3.4 Initialization	39
5.3.5 Programming the address	40
5.3.6 Operation	40
5.4 DATA SENDER module	43
5.4.1 Physical Layer interface	43
5.4.2 Link Layer interface	43
5.4.3 Higher Layer interface	44
5.4.4 Initialization	45
5.4.5 Programming the Data Port	45
5.4.6 Operation	45
5.5 DATA RECEIVER module	48
5.5.1 Physical Layer interface	48
5.5.2 Link Layer interface	48
5.5.3 Higher Layer interface	48
5.5.4 Initialization	49
5.5.5 Reading data from the Data Port	49
5.5.6 Operation	50
5.6 FRAME MONITOR module	52
5.6.1 Physical Layer interface	52
5.6.2 Link Layer interface	52
5.6.3 Higher Layer interface	53
5.6.4 Initialization	53
5.6.5 Operation	54
Chapter 6: Link Layer groups and protocol	
6.1 Simple groups	56
6.1.1 Simple Flag	56
6.1.2 Virtual Signal Controller	57
6.1.3 On-Demand Talker	58
6.1.4 Transaction Talker	60
6.1.5 On-Demand Listener	61
6.1.6 Transaction Listener	63
6.1.7 Multiaddress Flag	64
6.1.8 Multiaddress Talker	65

	Page
6.1.9 Multiaddress Listener	66
6.1.10 Variable Priority Controller	67
6.2 Compound groups	69
6.2.1 Writing Controller	69
6.2.2 Reading Controller	70
6.2.3 Virtual Bus Transceiver	72
6.2.4 Semaphore	73
6.2.5 Signature-Checking Semaphore	77
6.2.6 Token Passing Group	79
6.2.7 Handshaking Writing Controller	81
6.2.8 Handshaking Transaction Listener	82
6.2.9 Locking Transaction Listener	84
6.2.10 Locking Transaction Talker	85
Chapter 7: IEC 821 VMEbus backplane medium	
7.1 Additional electrical specifications	88
7.1.1 Terminating resistors	88
7.2 Mechanical specifications	88
7.3 Timing parameters	88
7.3.1 Testing of compliance	91
Chapter 8: Extended medium	
8.1 Additional electrical specifications	92
8.1.1 Termination networks	92
8.1.2 The BALANCE line	92
8.1.3 Cable parameters	94
8.1.4 Filtering and optocoupling	94
8.2 Mechanical specifications	94
8.2.1 Connectors and signal assignments	94
8.2.1.1 9-pole D-sub connector	94
8.2.1.2 10-pole flat cable connector and P2/J2 connector	95
8.2.1.3 Mixtures of all three connector types	96
8.2.2 Cable segment length	101
8.3 Extended Medium timing	102
8.3.1 Data rate versus length	102
8.3.2 Timing parameters	102
8.3.3 Terminology for Extended Medium signals	107
8.3.4 Testing of compliance	107
Figure 2-1 — Serial bus system structure and layering	5
Figure 2-2 — Layering and interfacing on a typical board	6
Figure 2-3 — Single-subrack Backplane Medium configuration	7
Figure 2-4 — Free-standing board on the Extended Medium	8
Figure 3-1 — Basic frame types	14
Figure 4-1 — CLOCK SOURCE module	21
Figure 4-2 — Extended bus waveforms	22
Figure 4-3 — Signals used by the BRIDGE module	23
Figure 4-4 — Serial bus waveforms	24
Figure 4-5 — Signals used by the BUS ACCESS module	27
Figure 4-6 — Backplane Medium waveforms	27
Figure 4-7 — Test load circuit for RULE 4.26 and RULE 4.27	31

	Page
Figure 4-8 — Test load circuit for RULE 4.28 and RULE 4.29	31
Figure 5-1 — HEADER SENDER with FRAME MONITOR	35
Figure 5-2 — HEADER SENDER state diagram	38
Figure 5-3 — Signals used by a HEADER RECEIVER	40
Figure 5-4 — HEADER RECEIVER state diagram	41
Figure 5-5 — DATA SENDER with HEADER RECEIVER	44
Figure 5-6 — DATA SENDER state diagram	47
Figure 5-7 — DATA RECEIVER with HEADER RECEIVER	49
Figure 5-8 — DATA RECEIVER state diagram	51
Figure 5-9 — FRAME MONITOR state diagram	55
Figure 6-1 — Simple Flag	56
Figure 6-2 — Virtual Signal Controller	57
Figure 6-3 — On-Demand Talker	59
Figure 6-4 — Transaction Talker	60
Figure 6-5 — On-Demand Listener	62
Figure 6-6 — Transaction Listener	64
Figure 6-7 — Multiaddress Flag	65
Figure 6-8 — Multiaddress On-Demand Talker	66
Figure 6-9 — Multiaddress Transaction Listener	67
Figure 6-10 — Variable Priority Controller	68
Figure 6-11 — Writing Controller	69
Figure 6-12 — Reading Controller	71
Figure 6-13 — Virtual Bus Transceiver	72
Figure 6-14 — Semaphore	74
Figure 6-15 — Signature-Checking Semaphore	77
Figure 6-16 — Token Passing Group	79
Figure 6-17 — Handshaking Writing Controller	81
Figure 6-18 — Handshaking Transaction Listener	83
Figure 6-19 — Locking Transaction Listener	84
Figure 6-20 — Locking Transaction Talker	86
Figure 7-1 — Backplane Medium waveforms and timing	89
Figure 8-1 — EXTCLK termination network	92
Figure 8-2 — EXTDAT termination networks	93
Figure 8-3 — Connection to the BALANCE line	93
Figure 8-4 — Twist-and-flat cable for Extended Medium within a cabinet, showing all the possible connector combinations	98
Figure 8-5a — Piggyback D-sub connector allows PCB removal while maintaining galvanic continuity of the Extended Medium	99
Figure 8-5b — Double cable in single D-sub connector maintaining galvanic continuity	100
Figure 8-6 — Using a PCB to link the Extended Medium outside and inside a cabinet	101
Figure 8-7 — Waveforms and timing for the Extended Medium	103
Figure 8-8 — Extended and Backplane Media timing	105
Figure 8-9 — States of an Extended bus signal	107
Table 2-1 — Physical Layer/Link Layer interface signals	9
Table 2-2 — XONE/XSTART/XJAM versus bit driven on medium	10
Table 2-3 — Bit received from medium versus RONE/RSTART state	10
Table 3-1 — Subframe usage by modules	15

	Page
Table 3-2 — Frame Type codes	17
Table 3-3 — Frame Status codes	19
Table 4-1 — SERDAT* output protocol for BUS ACCESS module	29
Table 4-2 — SERDAT* input protocol for BUS ACCESS module	29
Table 7-1 — Timing RULES for single-subrack CLOCK SOURCE*BRIDGE group	89
Table 7-2 — Timing RULES and OBSERVATIONS for BUS ACCESS modules	90
Table 7-3 — Parameter measurement points	91
Table 8-1 — Extended Medium data rate versus propagation versus length	102
Table 8-2 — Timing RULES for CLOCK SOURCE in extended configurations	103
Table 8-3 — Timing RULES and OBSERVATIONS, Stand-alone BRIDGE*BUS ACCESS group	104
Table 8-4 — Timing RULES and OBSERVATIONS for Backplane/Extended BRIDGE	106
Table 8-5 — Parameter measurement points	108

Foreword

- 1) The formal decisions or agreements of the IEC on technical matters, prepared by Technical Committees on which all the National Committees having a special interest therein are represented, express, as nearly as possible, an international consensus of opinion on the subjects dealt with.
- 2) They have the form of recommendations for international use and they are accepted by the National Committees in that sense.
- 3) In order to promote international unification, the IEC expresses the wish that all National Committees should adopt the text of the IEC recommendation for their national rules in so far as national conditions will permit. Any divergence between the IEC recommendation and the corresponding national rules should, as far as possible, be clearly indicated in the latter.
- 4) The IEC has not laid down any procedure concerning marking as an indication of approval and has no responsibility when an item of equipment is declared to comply with one of its recommendations.

Preface

This standard has been prepared by JTC 1/SC 26¹⁾: Microprocessor systems.

The text of this standard is based on the following documents:

Six Months' Rule	Report on Voting	Two Months' Procedure	Report on Voting
47B(CO)28	JTC 1/SC 47B N 14	JTC 1/SC 47B N 15	JTC 1/SC 47B N 20

Full information on the voting for the approval of this standard can be found in the Voting Reports indicated in the above table.

The following IEC publication is quoted in this standard:

Publication No. 821:1987 IEC 821 *BUS* — *Microprocessor system bus for 1 to 4 byte data.*

¹⁾ Formerly Sub-Committee 47B of IEC Technical Committee No. 47.

Chapter 0: Scope

This standard describes an interfacing method used to connect boards within a subrack, and an extended interfacing method used to connect subracks and stand-alone boards over limited distances. The interconnection of these boards and subracks is provided by a serial data medium.

Chapter 1: Introduction to the IEC 823 VMSbus standard

1.1 Serial bus objectives

The serial bus has been conceived with the following objectives:

- a) To support distributed information systems in both tightly coupled (multiprocessor) and loosely coupled configurations.
- b) To provide for the prompt delivery of short, urgent messages among boards and subracks in such systems.
- c) To support fault-tolerant systems by providing a control path that is complementary to a parallel backplane bus.
- d) To provide generalized control and message-transfer capabilities including broadcast, group addressing and group polling operations.
- e) To provide flexible data flow including read and move functions as well as the more common write function.
- f) To maximize medium availability and hardware simplicity by including an arbitration and allocation mechanism in the frame transmission process.
- g) To minimize hardware costs for boards that do not need the data rate of a parallel backplane bus, by eliminating the bus interface.
- h) For the extended bus, to further minimize hardware costs for functions that can be accomplished on a single remote board, by eliminating backplanes and subracks.

1.1.1 IEC 823 VMSbus standard objectives

This standard has the following objectives:

- a) To specify the characteristics required to design boards and integrated circuits that will reliably and unambiguously communicate with other devices attached to the serial bus.
- b) To specify protocols that precisely define the interaction between the serial bus and devices interfaced to it.
- c) To provide terminology and definitions that describe system protocol.

1.2 Serial bus interface system elements

This section defines some basic terms used to describe the serial bus. The following terms describe *mechanical* elements of serial bus systems:

Connector

A physical device which can be mated with another similar device to conduct one or more signal(s).

Backplane

A rigid electronic assembly including connectors and conductive paths that bus the signals among them.

Board

A rigid electronic assembly including active components, conductive paths, and one or more connectors that can be plugged into a backplane.

Slot

A position where a board can be plugged into a backplane.

Subrack

A rigid framework that provides mechanical support for boards inserted into a backplane, helping to ensure that the connectors mate properly and that adjacent boards do not contact each other.

The following terms describe *functional* elements of serial bus systems:

Functional module

A conceptual collection of electronic circuitry that resides on one board and works together to accomplish a specific function. The size and complexity of modules is chosen to permit clear and relatively simple specification of their function and operation.

Module group

A collection of functional modules that resides on one board and works together to accomplish a more extensive function.

Link Layer

The part of the serial bus interface system which is independent of the physical configuration of the bus and of the way that signals are modulated on the bus. The Link Layer provides functional and procedural means to transfer frames. The Link Layer stands between the Physical Layer and higher layers of the system.

Physical layer

The part of the serial bus interface system which varies with the way signals are modulated on the bus. The Physical Layer provides functional and procedural means to transfer bits. It includes electrical specifications for the bus drivers and receivers. This standard describes two Physical Layers. The Backplane Physical Layer uses TTL-compatible signal levels to transfer bits among boards within a subrack. The Extended Physical Layer uses dual-differential signalling to transfer bits among such subracks and/or free-standing boards.

Medium

The parts of the serial bus interface system which vary with the physical configuration of the bus. The Medium interconnects the Physical Layers on various boards. This standard defines an IEC 821 VMEbus Backplane Medium which interconnects Physical Layers on boards within a subrack, and an Extended Medium which interconnects Physical Layers between subracks and free-standing boards. The Medium dictates mechanical and performance specifications.

The following terms describe elements of the serial bus *protocol*:

Subframe

A series of consecutive bits on the serial bus in a specified arrangement or format.

Frame

A series of consecutive subframes on the serial bus in a specified arrangement or format.

Semaphore

In general, a mechanism that controls access to a resource which can be shared among several users (e.g. several processors). Used in this specification to indicate a module group that implements a single-bit mechanism which restricts access to the resource to a single user and does not indicate how many users are waiting to use the resource.

Token

In another control method for a shared resource, a token is a right to use the resource, which is passed around among users of the resource. The serial bus can be used to implement tokens, but is not itself controlled by this method.

1.3 Serial bus standard terminology

A standard like this one includes a variety of types of information. On one hand, the requirements for compliance must be spelled out, while on the other hand various types of useful data must be included which are not directly related to compliance. To differentiate these types of information, many of the paragraphs in this standard are labelled with the following KEYWORDS:

RULE

RECOMMENDATION

SUGGESTION

OBSERVATION

PERMISSION

Any text not labelled with one of these KEYWORDS describes the serial bus structure or operation. It is easily recognizable by its narrative style and exclusive use of the present tense.

These KEYWORDS are used as follows:

RULE chapter. number:

RULES form the basic framework of this standard. They are sometimes expressed in text form and sometimes in the form of figures, tables, or state diagrams. All serial bus rules **MUST** be followed to ensure compatibility among serial bus designs. Rules are characterized by an imperative style. The upper-case words **MUST** and **MUST NOT** are reserved exclusively for stating rules in this standard and are not used for any other purpose.

RECOMMENDATION chapter. number:

Whenever a **RECOMMENDATION** appears, designers would be wise to take the advice offered. Doing otherwise may result in certain awkward problems or poor performance.

SUGGESTION chapter. number:

In this standard, a **SUGGESTION** contains advice which is helpful but not vital. The reader is encouraged to consider the advice before discarding it. Whether or not a **SUGGESTION** is useful often depends on the application of the serial bus that the designer is pursuing.

OBSERVATION chapter. number:

OBSERVATIONS do not offer any specific advice. They usually follow naturally from what has been discussed previously. They spell out the implications of certain rules and bring attention to things that might otherwise be overlooked. They also give the rationale behind certain **RULES**, so that the reader understands why the **RULE** must be followed.

PERMISSION chapter. number:

In some cases, a serial bus **RULE** does not specifically prohibit a certain design approach, but the reader might be left wondering whether that approach might violate the spirit of the **RULE**, or whether it might lead to some subtle problem. **PERMISSIONS** reassure the reader that a certain approach is acceptable, and will cause no problems. The upper-case word **MAY** is used exclusively for stating **PERMISSIONS**.

1.4 Standard organization

The following chapters are arranged to proceed from general principles to specific details. Chapter 2 gives an overview of the subject. Chapter 3 describes the types of frames that appear on the serial bus, and contains specific details of the subframes from which frames are built.

Chapter 4 defines the Physical Layer, including the functional modules for both the IEC 821 VMEbus Backplane Medium and the Extended Medium. Chapter 5 formally defines the Link Layer by specifying the operation of Link Layer modules. Chapter 6 defines the serial bus protocol of various module groups.

Chapters 7 and 8 contain the mechanical and performance specifications for the IEC 821 VMEbus Backplane Medium and the Extended Medium, respectively.

1.5 Relationship of the serial bus and a parallel backplane bus

When the serial bus is used with a system bus, there are no prescribed timing constraints between any of its signals and any of the other signals on the backplane.

Compliance with this standard is independent of compliance with any other standard such as that for the IEC 821 VMEbus. A board or product which is built to comply with this standard should not claim compatibility with the IEC 821 VMEbus unless it also contains an IEC 821 VMEbus interface.

Chapter 2: Serial bus overview

This standard covers four parts of the serial bus interface system, the Link Layer, the Physical Layer, the Backplane Medium, and the Extended Medium. Figure 2-1, page 5, shows the general configuration of an Extended serial bus, with the layers arranged vertically, and the boards and subracks that are interconnected by the serial bus arranged horizontally.

In the Link Layer, functional modules are arranged in groups, which are the entities that exchange information on the serial bus. Physical Layer modules are more independent and are grouped only in two special circumstances. Since the Medium simply represents conductive paths, there are no functional modules in the Medium.

OBSERVATION 2.1:

The representation of this standard in a layered Form is similar to that put forth by the Open System Interconnection (OSI) model of the International Organization for Standardization (ISO). The serial bus layers correspond to the lowest two functional layers of OSI plus its physical medium.

Figure 2-1 depicts the various *interfaces* in the serial bus system as broken lines. Figure 2-2, page 6, shows further details of these interfaces, for a typical serial bus board connected to the Backplane Medium.

The interfaces between the Link Layer and higher layers of control are described in Chapter 5, as part of the formal definition of the various Link Layer modules. Chapter 5 also defines the interface between the Link Layer modules in a group. The interface between the Link Layer and the Physical Layer, and the interface between the Physical Layer and the Medium, are described in general terms below and in greater detail in Chapter 4.

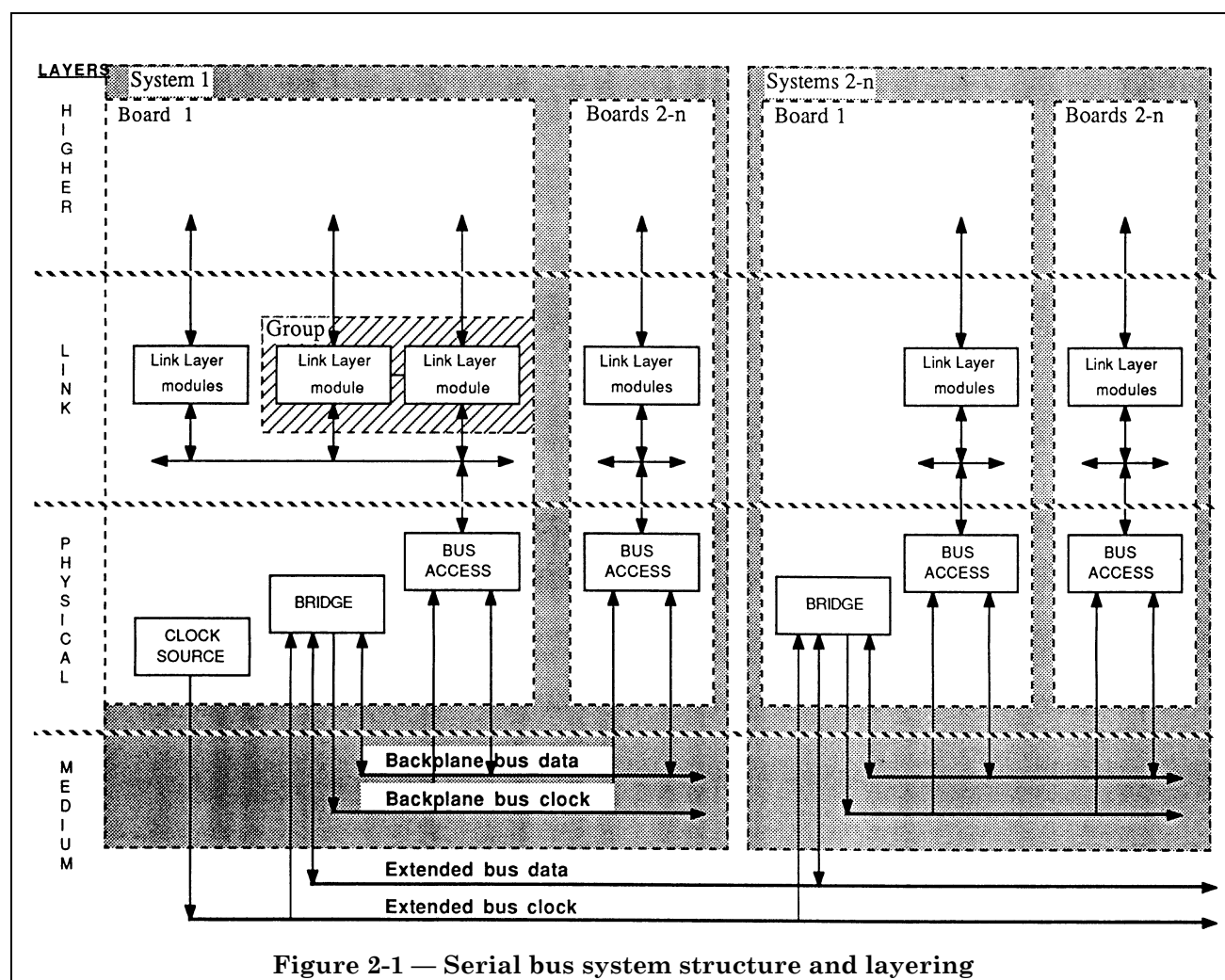


Figure 2-1 — Serial bus system structure and layering

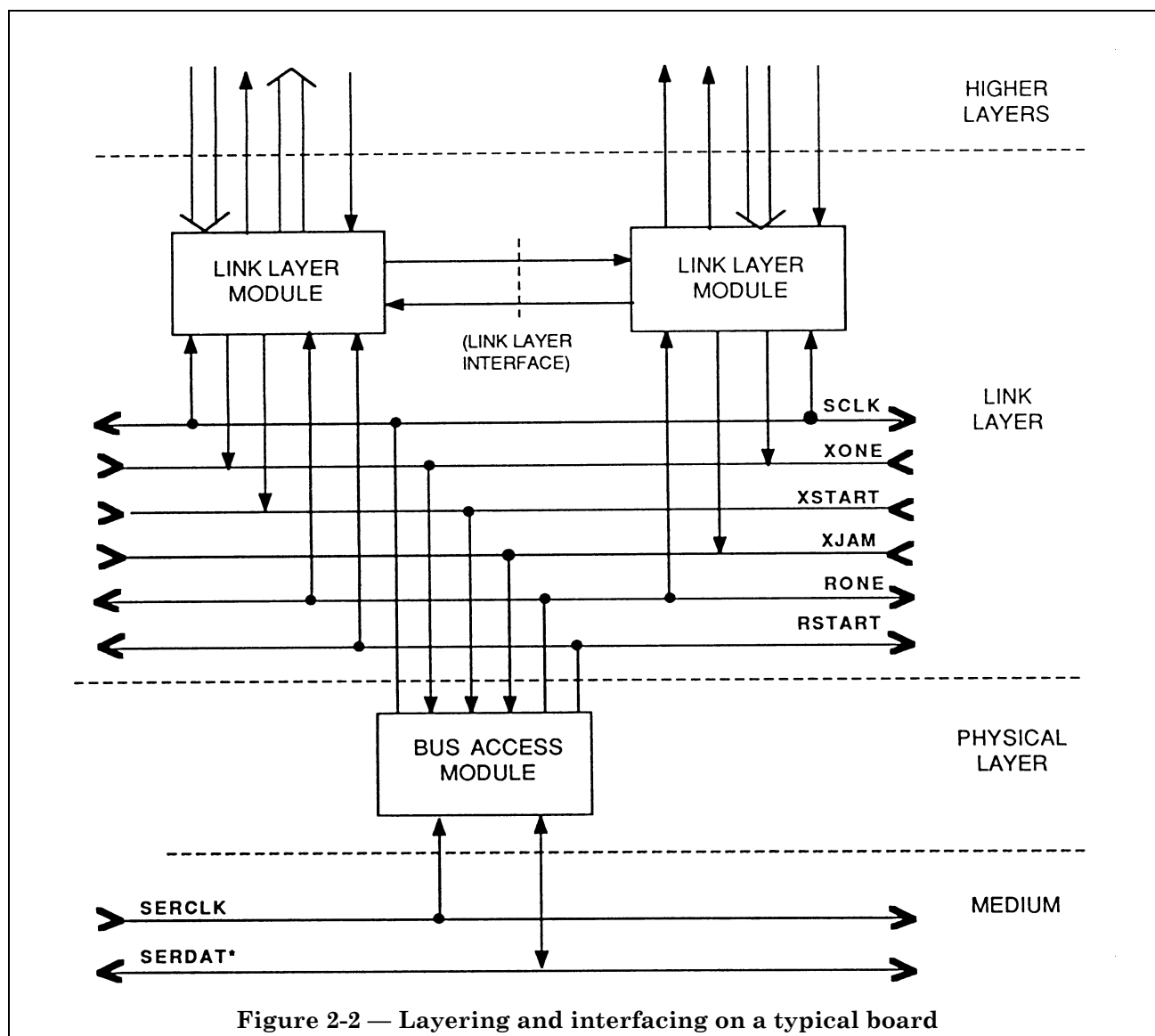


Figure 2-2 — Layering and interfacing on a typical board

2.1 Overview of the Physical Layer

2.1.1 Physical Layer modules and signals

The Physical Layer includes three types of modules called **CLOCK SOURCE**, **BRIDGE** and **BUS ACCESS**. The **CLOCK SOURCE** drives the Extended Medium signal pair called **EXTCLK⁺** and **EXTCLK⁻**. There is only one (active) **CLOCK SOURCE** on an Extended Medium.

The **BRIDGE** stands between the Extended Medium and a Backplane Medium. There is one (active) **BRIDGE** for each subrack or stand-alone board connected to the Extended Medium. It receives **EXTCLK⁺** and **EXTCLK⁻** from the Extended Medium; based on these it derives and drives the **SERCLK** signal on the Backplane Medium. The **BRIDGE** also passes data bidirectionally between the Extended Medium signal pair called **EXTDAT⁺** and **EXTDAT⁻**, and the Backplane Medium signal **SERDAT***.

The **BUS ACCESS** module stands between a Backplane Medium and the Link Layer. There is one **BUS ACCESS** module on each board that communicates on the serial bus. It receives **SERCLK** and drives and receives **SERDAT***.

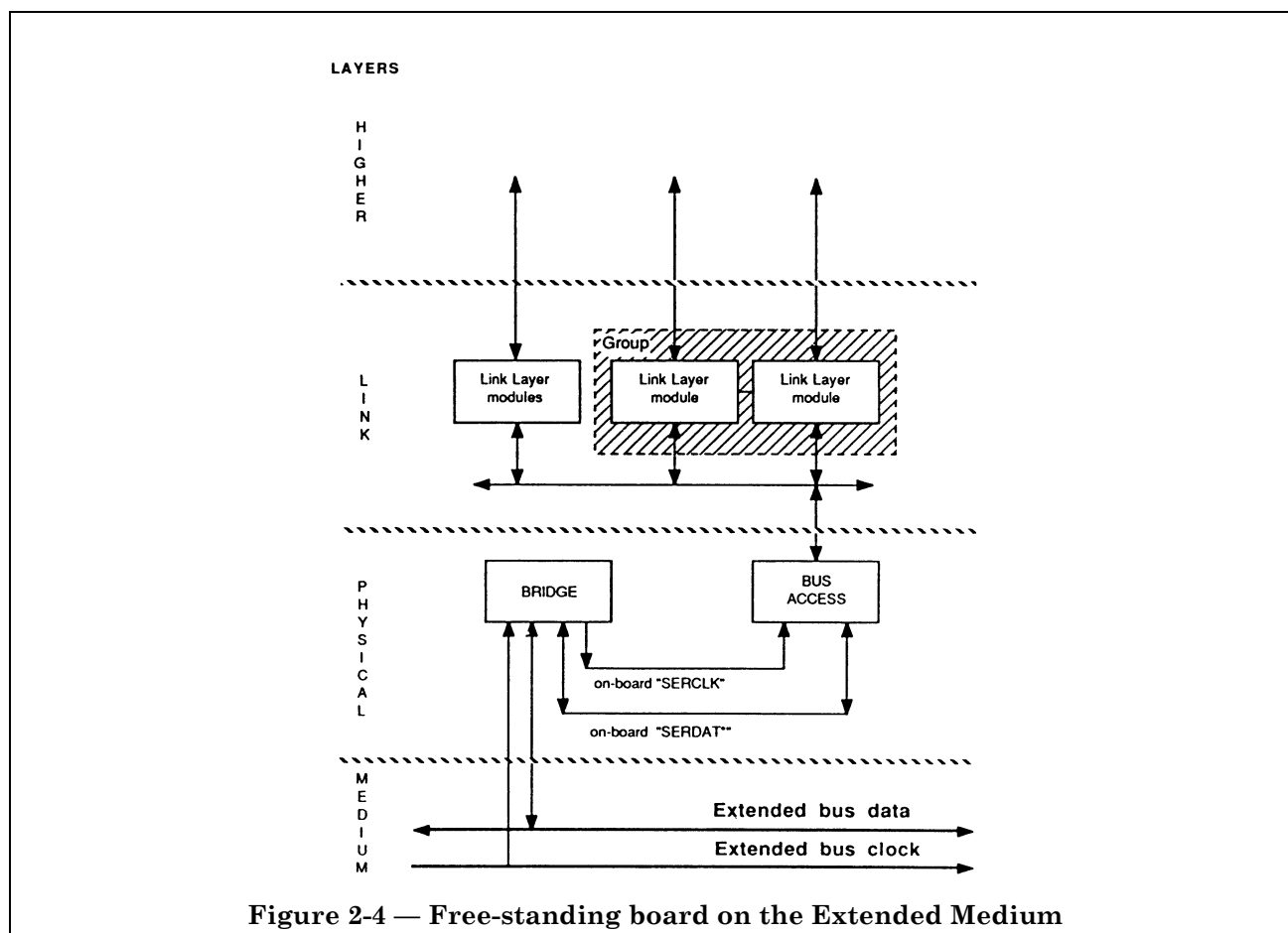


Figure 2-4 — Free-standing board on the Extended Medium

The various BUS ACCESS modules on a serial bus are allowed to present different types of bits on SERDAT* at the same time. When the Extended Medium is used, the various BRIDGE modules thus present different types of bits on EXTDAT. When this occurs, the resulting bit received by all the BUS ACCESS modules on the bus is the same and is determined as follows:

RULE 2.1:

BUS ACCESS modules and BRIDGES MUST be designed so that, when any BUS ACCESS module on a serial bus presents a one_bit, all the BUS ACCESS modules on the bus receive a one_bit.

RULE 2.2:

BUS ACCESS modules and BRIDGES MUST be designed so that, when no BUS ACCESS module on a serial bus presents a one_bit, but one or more present a start_bit, all BUS ACCESS modules on the bus receive a start_bit.

RULE 2.3:

BUS ACCESS modules and BRIDGES MUST be designed so that, when all BUS ACCESS modules on a serial bus present zero_bits, all BUS ACCESS modules on the bus receive a zero_bit.

Chapter 4 defines the Physical Layer modules for the Backplane Medium, which implement the above rules using TTL-compatible voltage levels. It also defines the Physical Layer modules for the Extended Medium, which implement the rules using differential signalling similar to that used in EIA (Electronic Industries Association) standards RS-422 and RS-485.

2.2 Link Layer overview

2.2.1 Modules and interfaces

The Link Layer includes five types of modules called HEADER SENDERS, FRAME MONITORS, HEADER RECEIVERS, DATA SENDERS and DATA RECEIVERS. Any number and combination of these modules can be present on any board on a serial bus.

OBSERVATION 2.3:

Module names are intended to indicate the major function of each module. They are not meant to constrain the modules' operation. For example, both HEADER RECEIVERS and DATA RECEIVERS can also *send* information on the serial bus in the Frame Status field, as described in subsequent chapters.

The interface between Link Layer modules and the Physical Layer BUS ACCESS module consists of six signals as shown in Table 2-1. The SCLK, RONE, and RSTART signals are driven by the BUS ACCESS module and received by the Link Layer modules, while XONE, XSTART, and XJAM are driven by the Link Layer modules and received by the BUS ACCESS module.

The SCLK signal is derived from the SERCLK signal by the BUS ACCESS module. It serves to inform the Link Layer modules of bit timing so that they may present bits on XONE and XSTART and interpret received bits on RONE and RSTART. The BUS ACCESS module uses XONE, XSTART, and XJAM to present bits on SERDAT*, and presents the information received from SERDAT* on RONE and RSTART. The XONE, XSTART, XJAM, RONE, and RSTART signals are driven to True or False corresponding to one_bits, zero_bits, and start_bits on SERDAT*, as shown in Table 2-2 and Table 2-3.

OBSERVATION 2.4:

The information on the XONE, XSTART, XJAM, RONE, and RSTART lines is equivalent to that on SERDAT*. Link Layer modules could be described as using SERDAT* directly, or an identical single signal in the Physical Layer/Link Layer interface. The multi-line scheme is used because it simplifies the formal description of Link Layer modules.

The serial bus allows the various Link Layer modules on a board to present different states on XONE, XSTART and XJAM at the same time.

RULE 2.4:

Link Layer modules **MUST** be designed so that, when any Link Layer module on a board presents XONE, XSTART, or XJAM True, the BUS ACCESS module receives that signal True, even if another Link Layer module on the board presents the signal False.

Table 2-1 — Physical Layer/Link Layer interface signals

Signal	Driven by	Use
SCLK	Physical Layer	Signals bit timing to the Link Layer
XONE	Link Layer	Commands the Physical Layer to send a one_bit
XJAM	Link Layer	Commands the Physical Layer to send a one_bit
XSTART	Link Layer	Commands the Physical Layer to send a start_bit
RONE	Physical Layer	Indicates a received one_bit to the Link Layer
RSTART	Physical Layer	Indicates a received start_bit to the Link Layer

Table 2-2 — XONE/XSTART/XJAM versus bit driven on medium

XJAM	XONE	XSTART	Bit driven on medium
True	x	x	One_bit
False	True	x	One_bit
False	False	True	Start_bit
False	False	False	Zero_bit

NOTE The “x” in the above table indicates that the signal has no effect on the bit driven on the medium.

Table 2-3 — Bit received from medium versus RONE/RSTART state

Bit received from medium	RONE	RSTART
One_bit	True	False
Start_bit	False	True
Zero_bit	False	False

2.2.2 Module groups

Link Layer modules are deliberately restricted in size and functionality in this standard, so that the formal descriptions of their operation remain straightforward. However, the overall operation of the serial bus is easier to describe in terms of larger entities called *module groups*.

The following are the most common groups:

Controller: A HEADER SENDER and a FRAME MONITOR

Flag: A HEADER RECEIVER and a latch. Flag groups are further classified according to their operational protocol, into Simple_Flag, Semaphore and Token Passing groups

Talker: A HEADER RECEIVER and a DATA SENDER

Listener: A HEADER RECEIVER and a DATA RECEIVER

2.2.3 Frame transmission protocol

All communication between Link Layer module groups is done through the Physical Layer, by sending frames on the serial bus. These frames are composed of “subframes” which are sent by various groups. A frame is initiated when a Controller group sends a “Header subframe”. Other groups then respond by sending subframes according to a prescribed protocol until the end of the frame is reached.

The FRAME MONITOR in each Controller group receives every bit from the serial bus and follows the progress of each frame. A HEADER SENDER can thus determine whether there is a frame in progress, from an on-board signal generated by its FRAME MONITOR. Whenever there is no frame in progress, it can initiate one by sending a Header.

There are no separate signal lines or time period dedicated to determining which Link Layer modules may send information on the serial bus. Rather, an arbitration mechanism is built into the procedure for transmitting frames.

When a HEADER SENDER module transmits a Header, it samples each bit on the RONE line while it is sending. If the HEADER SENDER detects RONE True when it presented XONE False, it stops sending. This allows other HEADER SENDER(S) to finish sending the Header subframe without interference.

OBSERVATION 2.5:

This method of arbitration allows several HEADER SENDERS to start sending frames simultaneously, without affecting each other’s transmissions. (One of the transmissions will be successful while the others will be tried again later.)

The “winning” Header specifies what other modules will participate in the frame transmission, by means of its ten-bit “S address” and ten-bit “R address”. Each HEADER RECEIVER on the bus, be it in a Talker, Listener, or Flag group, has an address which it compares with these two fields in the Header. If it finds a match in either of these fields, it responds by signalling the match to the other modules in the group. Depending on which type of group the HEADER RECEIVER is in, an address match may have different effects:

- 1) in a Flag group, it may change the state of the latch;
- 2) in a Talker group, it may signal the DATA SENDER to send, or
- 3) in a Listener group, it may signal the DATA RECEIVER to receive.

2.3 Using the serial bus to transfer data

To transfer data, the HEADER SENDER sends an S Address that corresponds to a Talker on the bus, and an R Address that corresponds to a Listener. (The actual addresses used to select these groups depend on how the system is configured. For example, there is no standard block of addresses assigned specifically for selecting Listeners.)

Each HEADER RECEIVER on the serial bus compares these addresses against its own. In a Data Transfer frame, at least one of the Talkers’ HEADER RECEIVERS finds a match with the S address. It tells its DATA SENDER to send data. Similarly, one or more of the Listeners’ HEADER RECEIVERS find a match with the R address and tells its DATA RECEIVER to receive data.

The actual number of bytes transferred is left up to the DATA SENDER. After the HEADER SENDER has sent the Header, the DATA SENDER sends a three bit subframe indicating the number of bytes it intends to send to the DATA RECEIVER, followed by the data bytes. After this, the DATA RECEIVER responds with a status code, to indicate that it was selected and whether it was able to accept the data.

2.4 Using the serial bus to set and reset flags

When a HEADER SENDER is used to set or reset a Flag group, it sends a Header as in the case described above. If the frame is intended to *set* a flag, it sends the Flag’s address in the S field. If the frame is intended to *reset* a flag, it sends the Flag’s address in the R field.

When the Header is sent, each HEADER RECEIVER on the serial bus checks its S and R addresses to see if its module group is selected. One (or more) HEADER RECEIVER(S) matches the S (or R) address and sets (or resets) its latch.

Further information on how frames are used to control Flag groups is given in Chapter 6.

2.5 Applications of serial bus module groups

Chapter 6 explains how groups of Link Layer modules such as those described above can be used as building blocks for a wide variety of system configurations. For example, the combination of a Controller, Talker and Listener, on each of two boards, can be used by one board to pass an address and read the contents of a memory location on the other. Flag groups can be used to re-initialize one or more of the boards in a system, or to selectively disconnect a failed board from the system bus. Flags can be also be used to provide semaphores that can have operational advantages over semaphores in a common memory, or to provide token passing functions.

2.6 Serial bus addressing capabilities

As indicated above, the S or R address in a frame can select more than one HEADER RECEIVER, and thus more than one module group. This fact provides some of the more powerful capabilities of the serial bus.

When multiple Listeners are selected, “broadcast” or “group address” operations can be done. For example, a Listener on a processor card might respond to frames addressed to all processors, processors of a specific type or its own unique address.

When multiple Talkers are selected, “group poll” operations can be done. For example, a Talker connected to a temperature sensor might send data in response to Headers which select “all temperature sensors”, “all sensors in zone N” or this specific sensor.

For such group poll operations, Talkers can be commanded whether to use serial bus arbitration (like that used by HEADER SENDERS) while sending the data. If data field arbitration is used, the resultant data is the largest binary value among the Talkers' data (e.g. the highest temperature). If data field arbitration is not used, the resultant data is the OR of the Talkers' data, which is useful for status-type information (e.g. for "alarm" bits).

Broadcast and group address operations can also be applied to Flag groups. For example, if Flag groups are used to allow disconnection of failed boards from the system bus, a frame could be sent to disconnect "all bus masters", a particular bus master or "all boards made by manufacturer XYZ".

Chapter 3: Serial bus frames and subframes

3.1 Basic frame types

As noted in Chapter 2, a frame is composed of subframes which are sent by several module groups in sequence whenever a Controller group sends a Header. Depending on the types of module groups that the Header selects, three basic frame types can occur on the serial bus, as shown in Figure 3-1, page 14.

A Cancelled frame indicates that a selected module group is not ready for the frame.

If all of the selected groups are ready, and the S address in the Header selects a Talker group, a Data Transfer frame results. Data Transfer frames send 1–32 bytes of data from Talker to Listener groups, and may also affect Flag groups that are used to “track” data transfers.

A Control frame results if all of the selected groups are ready and the S address has not selected a Talker group. Control frames do not contain any data, and are used only to set and reset Flag groups.

Table 3-1 defines which modules drive and receive the various subframes in each of these basic frame types. Since each frame transmission on the serial bus involves interaction among several functional modules, there are two ways to describe the activity on the serial bus:

- 1) Overall serial bus operation can be described for each frame, noting the actions of the various module types in each subframe. (This is the approach used in this chapter.)
- 2) The operation of each functional module can be described separately, in terms of the interaction between it and the serial bus. (This is the approach used in chapter 5.)

All frames start with a Header and a Frame Type subframe, and end with a Jam Detect subframe. The Header determines which modules will participate in the rest of the frame, which in turn determines the type of frame that is sent. The Frame Type subframe makes the type of frame known to all serial bus modules, and specifies the length of the frame. The Jam Detect subframe “validates” the complete frame and ensures “frame synchronization” among all serial bus modules.

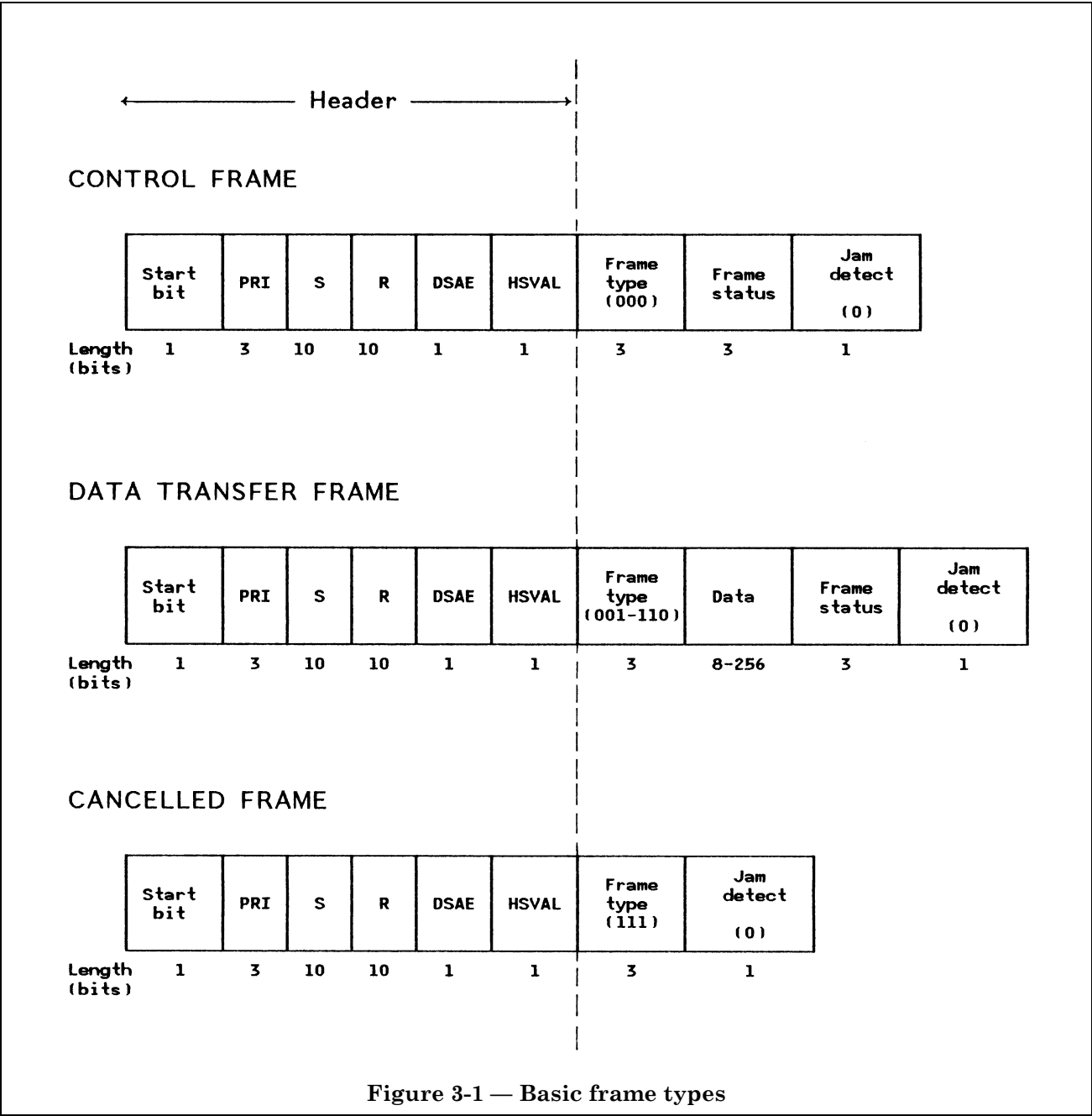


Table 3-1 — Subframe usage by modules

Frame	Subframe	Sent by	Received by
Control	Header	HEADER SENDER	HEADER RECEIVER, FRAME MONITOR
	Frame type	(No module, defaults to 000)	HEADER RECEIVER, FRAME MONITOR
	Frame status	HEADER RECEIVER	FRAME MONITOR
	Jam bit	(No module, defaults to 0)	HEADER RECEIVER, DATA SENDER, DATA RECEIVER, FRAME MONITOR
Data transfer	Header	HEADER SENDER	HEADER RECEIVER, FRAME MONITOR
	Frame type	DATA SENDER (001–110)	DATA RECEIVER, FRAME MONITOR
	Data	DATA SENDER	DATA RECEIVER
	Frame status	DATA SENDER, DATA RECEIVER	FRAME MONITOR, DATA SENDER, DATA RECEIVER
	Jam bit	(No module, defaults to 0)	HEADER RECEIVER, DATA SENDER, DATA RECEIVER, FRAME MONITOR
Cancelled	Header	HEADER SENDER	HEADER RECEIVER, FRAME MONITOR
	Frame type	HEADER RECEIVER (111)	FRAME MONITOR, DATA SENDER, DATA RECEIVER
	Jam bit	(No module, defaults to 0)	HEADER RECEIVER, DATA SENDER, DATA RECEIVER, FRAME MONITOR

3.2 The Header

A Header is composed of 26 bits arranged in six fields:

Start bit	Frame priority	S	R	DATA SENDER Arb enable	HSVAL
1	3	10	10	1	1

The contents of each of these fields are as follows:

- 1) A single start_bit. When the start_bit is transmitted, it signals serial bus modules that a frame is beginning.
- 2) A three-bit “Frame Priority” field. If two or more Talker groups attempt to send a frame at the same time, this field is used to arbitrate control of the serial bus so that the frame with higher priority is sent first.
- 3) A ten-bit S address, which may select one or more serial bus module group(s). This address may select Flags that are to be set, and/or Talkers that are to send data in the Data subframe which follows. (The S address may select several groups if their HEADER RECEIVERS have all been configured to recognize it.)

4) A ten-bit R address, which may select one or more serial bus module group(s). This address may select Flags that are to be reset, and/or Listeners that are to capture the data in the Data subframe which follows. In a frame intended to set a Semaphore group, this address is a “Requester code” which does not select a group, but rather guarantees that only one Controller group wins the serial bus arbitration in the Header. (The R address may select several groups if their HEADER RECEIVERS have all been set up to recognize it.)

OBSERVATION 3.1:

Chapter 6 gives further information about the use of the R address for “Requester codes”.
A frame might have no other purpose than to set or reset a Flag, with no requirement that this be done by a unique Controller. In such a case, one of the two addresses in the Header is effectively “unused”. The serial bus reserves the “all ones” value (decimal 1023) for use in an “unused” address field.

RULE 3.1:

Higher layers of serial bus management MUST NOT configure any serial bus group to respond to the “all ones” address (decimal 1023).

OBSERVATION 3.2:

This standard does not define the operation of serial bus module groups when they are selected by both the S and R addresses in a Header.

RECOMMENDATION 3.1:

Except for diagnostic purposes, do not send Headers with the same address in the S and R fields. Similarly, do not send Headers with S and R addresses that are different but which, due to group addressing, both select the same module group.

5) A DATA SENDER arbitration enable bit (DSAE). When two or more Talkers share a common address, they will be selected to send at the same time. In this case, the resulting Data subframe can contain either the logical OR of their values, or the largest value among the Talkers’ data. To specify that a logical OR is required, the Controller sends a zero_bit in this bit position. To specify that the largest value is required, it sends a one_bit. (A one_bit commands each Talker to use serial bus arbitration while sending the data, that is, to stop sending data if it samples a one_bit in a bit in which it sent a zero_bit. A zero_bit tells the Talkers to omit arbitration, and keep sending regardless of the sampled data.)

OBSERVATION 3.3:

The DSAE bit has no defined function in a Control frame.
6) A HEADER SENDER validation bit (HSVAL). This bit is always sent as a one_bit by a Controller at the conclusion of the Header. This bit is specified as part of the Header, to ensure that some error on the serial bus would not cause all the Controllers to think they have lost the arbitration and retire from the bus. If this occurs, the Controller leaves a string of zero_bits in the rest of the Header. This could be misinterpreted by other modules as a valid Header.
The validation bit solves this problem because if all of the Talkers retire it will be a zero_bit, causing all serial bus modules to ignore the Header.

3.3 The Frame Type subframe

The second subframe present in all frames is the Frame Type subframe. It is composed of three bits:

Most significant bit (MSB)	bit 2	bit 1	bit 0
-------------------------------	-------	-------	-------

The code in these bits differentiates the three basic types of serial bus frames: Control frames, Data Transfer frames and Cancelled frames. Table 3-2 shows the meaning of the various Frame Type codes.
When a Talker is selected by the S address, it sends a three bit Frame Type code in this subframe. This code indicates how many bytes of data the Talker is planning to send.
When a Talker, Listener or Flag group is selected by either the S or the R address, it may cancel the frame by sending 111 in this subframe, if it is signalled to do so by a higher layer of serial bus management.

Table 3-2 — Frame Type codes

Code	Meaning
000	Control frame
001	1 byte data transfer frame
010	2 byte data transfer frame
011	4 byte data transfer frame
100	8 byte data transfer frame
101	16 byte data transfer frame
110	32 byte data transfer frame
111	Cancelled frame

Since data is OR'ed on the serial bus, a Cancellation (111) predominates over the length code (001-110) sent by Talkers. If no Talker, Listener or Flag cancels the frame, and no Talker sends a length code, a Control frame (000) results.

In general, cancelling a frame indicates that a group is “not ready” for the frame. This standard does not define all of the local/on-board conditions which may cause a group to cancel a frame, but the most common conditions are as follows:

1) If a Listener group has received data in a previous frame and has not “disposed” of that data yet. In order to keep from losing the data from the previous frame, the group’s DATA RECEIVER signals its HEADER RECEIVER that it is not ready for another frame. If a Controller sends another Header with the Listener’s address in the R field, before the previous data has been disposed of, the HEADER RECEIVER cancels the new frame by sending 111 in the Frame Type. Talkers sample the Frame Type subframe as they send it, and stop sending if they sample 111. (A Cancelled frame ends after the Frame Type bits.)

2) If a Talker group has not been provided with data by higher layers of serial bus management. As in the Listener group in the previous case, the Talker’s DATA SENDER signals its HEADER RECEIVER of this fact via an on-board signal. If a Controller sends a frame with the Talker’s address in the S field while the Talker “has no data”, its HEADER RECEIVER cancels the frame by sending 111 in the Frame Type subframe. Selected Listeners sample this code and ignore the frame.

A system can be set up so that more than one Talker is selected by a particular S address. If multiple Talkers are selected by the S address, and one or more of them is not ready to send, those which are ready sample the 111 code and do not send any data.

3) A semaphore or token passing group may also send 111 in the Frame Type subframe to cancel a frame. If it is selected by the S address and its latch is already set, or (in a token passing group) if it is selected by the R address and its latch is already reset, it cancels the frame.

OBSERVATION 3.4:

As described in Chapters 5 and 6, Talker and Listener groups are not required to cancel frames as described above. Some types of Talkers and Listeners are always ready for a frame.

In summary, the HEADER RECEIVER in a Talker, Listener or Flag group sends 111 to cancel a frame if it is selected by either the S or the R address, and its DATA SENDER, DATA RECEIVER, or other on-board logic has signalled that it should cancel such a frame.

When a frame is cancelled, this fact is captured by the Controller group that initiated the frame, and is reported to higher layers of serial bus management.

3.4 The Data subframe

This subframe occurs only in Data Transfer frames. Its length is specified by the selected Talker(s) in the Frame Type subframe. There may be 1, 2, 4, 8, 16 or 32 bytes in a Data subframe. As described earlier, there may be one or more Talker(s) sending in this subframe, depending on how many are configured to respond to the S address. If several Talkers are selected, the Data subframe will either contain the logical OR of their data, or the largest value among them. (This is determined by whether the Talkers’ arbitration logic was disabled or enabled by the Data Sender Arbitration Enable bit sent in the Header.)

3.5 The Frame Status subframe

While the cancellation of a frame can be determined from the Frame Type field, other problems can arise when a frame is sent. Like cancellations, such problems are reported to higher layers of serial bus management, by the Controller group that initiated the frame. The Frame Status subframe is included at the end of every frame except a Cancelled frame, and is used to signal whether module groups were selected by the S and R addresses. It is also used to signal any problems (exceptional conditions) that might have arisen during the frame's transmission. The Frame Status subframe is composed of three bits.

Most significant bit (MSB)	bit 2	bit 1	bit 0
-------------------------------	-------	-------	-------

Whenever a HEADER RECEIVER receives a Header with an S or an R address that matches its own, it makes one of two on-board outputs True. Depending on whether the HEADER RECEIVER is part of a Talker, Listener or Flag group, these outputs may or may not enable an on-board DATA SENDER or DATA RECEIVER.

The HEADER RECEIVER then samples the Frame Type subframe. If a Talker group is selected, its DATA SENDER sends one of the values 001 through 110 in the Frame Type. If the HEADER RECEIVER samples 001–110, it passes over the Data subframe and samples these three Frame Status bits and the following Jam Detect bit. (In this case the HEADER RECEIVER is simply acting as a “tracker” to reflect how DATA SENDERS and DATA RECEIVERS responded to the frame.) If the Frame Status is 011 and the Jam Detect bit is a zero_bit, the HEADER RECEIVER pulses one of two on-board outputs. Depending on what type of module group the HEADER RECEIVER is in, this output may set or clear an on-board “tracking” latch.

If the HEADER RECEIVER samples 000 in the Frame Type, this indicates that no Talker was selected by the frame. The HEADER RECEIVER samples the MSB of the (immediately following) Frame Status field, and then sends either a 10 or a 01 in the next 2 bits to indicate that it was selected by the S or the R address, respectively. It then samples the Jam Detect bit. If the MSB of the Frame Status and the Jam Detect bits are both zero_bits, the Control frame is considered successful, and the HEADER RECEIVER pulses one of two outputs to set or clear an on-board latch.

If two or more Talkers are selected by the S address in the Header, the serial bus protocol requires that they agree about how much data is to be sent. As each Talker sends its code in the Frame Type field to signal how much data it has to send, it also samples these bits to detect a conflict with other Talkers. If a Talker samples a one_bit in any of the 3 bits while it is sending a zero_bit, it stops sending any further bits in the Frame Type subframe or Data subframe. However, in this case it does sample all the bits in the Frame Type. If the sampled value is not 111 (i.e. not a cancellation), the Talker uses the sampled value to determine the length of the Data subframe, passes over the data, and thereafter sends 110 during the Frame Status subframe to signify a conflict among selected Talkers as to the size of the Data subframe.

If a Talker samples the same value in the Frame Type that it is sending, it sends its data. It then samples the MSB of the Frame Status, which indicates a data-size problem between other Talkers and Listeners. If the MSB is a zero_bit, the Talker sends 10 in the next two bits of the Frame Status to indicate that it survived any arbitration in the Data subframe.

Suppose that a Talker is selected by the S address in the Header, and a Listener is selected by the R address. Let us further suppose that the Listener is only capable of receiving Data subframes up to a certain length (less than 32 bytes). The Listener samples the Frame Type subframe. If the Frame Type code indicates that the Talker is trying to send more data than it can handle, the Listener counts bits across the size of data indicated by the Frame Type, and then sends 101 in the Frame Status subframe to signal the problem.

If a selected Listener samples 000 in the Frame Type field, there is no Talker selected to send data to it. It handles this situation by sending 101 in the Frame Status subframe.

Otherwise (for Frame Types 001–110) the Listener samples the number of bytes of data indicated by the Frame Type. It then samples the MSB of the Frame Status subframe. If the MSB is a zero_bit, it sends 01 in the next two bits of the Frame Status to show that it was selected. It then samples the Jam Detect bit. If the MSB of the Frame Status and the Jam Detect bit were both zero_bits, it presents the data it sampled to higher layers of serial bus management, along with an output that indicates the arrival of data to higher layers.

In summary, the Frame Status value is interpreted differently for a Data Transfer frame versus a Control frame. In each, case there are eight possible status values, as shown in Table 3-3.

3.6 The Jam Detect subframe

This “subframe” consists of a single bit. It can be regarded as a subframe or as “the bit after a frame”. None of the modules which send other subframes ever drives this bit as a one_bit, but Talkers, Listeners and Flags all sample it to be sure it is a zero_bit. If the Jam Detect bit is ever sampled by these modules as other than a zero_bit, they ignore the preceding frame transmission.

3.7 Jam protocol

The procedure described above is necessary because, if a high level of noise is induced onto the serial bus, Controllers may become desynchronized. That is, they may lose track of where frames begin and end. This condition has to be detected, and the modules resynchronized, before frame information is distorted and misinterpreted. If Controllers become desynchronized, one of them begins to send a frame during the transmission of another frame.

The FRAME MONITORS in Controller groups constantly check for start_bits. When a FRAME MONITOR detects a start_bit when *no* frame is in progress, it tracks the transmission to the end of the frame. If it samples another start_bit before it has finished counting out the frame, then it “jams” the frame transmission by sending a sequence of 512 one_bits, or a Start_bit followed by 511 one_bits.

Thus, a “jammed” frame may look like any of the basic frame types shown in Figure 3-1, page 14. It differs from them in that (at least) the Jam Detect bit contains something other than the zero_bit shown in Figure 3-1.

Since all serial bus frames are shorter than 512 bits, the Jam Sequence ensures that no serial bus module receives a zero_bit in the Jam Detect bit position of the current frame, regardless of how it may be tracking frames.

Table 3-3 — Frame Status codes

Frame	Status value	Meaning
Control	000	No module group selected by either S or R
	001	Flag group selected by R only
	010	Flag selected by S only
	011	Flag selected by S and R
	100	RESERVED: should not occur
	101	Listener selected by R, nothing selected by S
	110	RESERVED: should not occur
	111	Listener selected by R, Flag (only) selected by S
Data transfer	000	No Listener selected by R, all Talkers selected by S have lost Data field arbitration
	001	Listener selected by R, all Talkers selected by S have lost Data field arbitration
	010	No Listener selected by R, Talker(s) selected by S have completed the Data subframe normally
	011	Data Transfer completed correctly
	100	RESERVED: should not occur
	101	Listener(s) unable to handle data size
	110	Data size conflict among Talkers
	111	(Both 101 and 110 conditions)

Because the frame in progress was damaged by the misplaced start_bit (and typically also by the Jam Sequence itself), a frame with something other than a zero_bit in the Jam Detect bit is ignored by all serial bus modules. In this case, no Flag is set or cleared, no Talker considers itself to have sent data, nor does any Listener consider itself to have received data.

The Jam Sequence also serves to resynchronize all the Controller groups on the bus. Typically, the frame which was jammed will be re-sent after the serial bus is resynchronized.

The FRAME MONITOR in each Controller group tracks frames by sampling the Frame Type Code in each one, and using the sampled value to determine the remaining number of bits in the frame. After counting off that number of bits, it then signals its associated HEADER SENDER that a frame is no longer in progress. Since even a 32-byte data transfer frame has only 260 bits after the Frame Type field, all HEADER SENDERS will be informed that no frame is in progress, before the end of the 512-bit Jam Sequence.

Whenever a HEADER SENDER is signalled that a frame is not in progress, it may initiate one by sending a start_bit, i.e. by presenting XSTART True to its BUS ACCESS module. However, the HEADER SENDER also monitors RONE from the BUS ACCESS module before it sends the start_bit. RONE stays True through the end of the Jam Sequence, and the HEADER SENDER waits to send the start_bit until it sees RONE False. At this point it sends the start_bit and proceeds to send the rest of the Header.

OBSERVATION 3.5:

The above procedure serves to resynchronize all the active Controller groups on the serial bus.

RECOMMENDATION 3.2:

When implementing a serial bus circuit that does not include the (complete) FRAME MONITOR function, design it to detect start_bits within a frame and jam the serial bus, as described in Chapter 5 for a FRAME MONITOR.

OBSERVATION 3.6:

This design practice provides greater security and protection against the effects of signal noise that might be sensed at one point on the serial bus, but not at another.

3.8 Jam on Reset

A Controller group is also allowed to send a Jam Sequence of at least 512 one_bits when it receives an on-board Reset signal. In this way, the serial bus can be used in extended configurations in which all the modules do not share a common Reset signal.

So, when a Controller group is Reset, it may be joining an already operating serial bus. At the point when the Reset is released, the Controller's FRAME MONITOR has no tracking of whether a serial bus frame is in progress. By sending a Jam Sequence, it ensures frame synchronization with other Controllers.

This type of Jam Sequence can also serve as an indication to higher layers of serial bus management, to reconfigure the serial bus system.

Chapter 4: The Physical Layer

The Physical Layer includes the CLOCK SOURCE, BRIDGE and BUS ACCESS modules. CLOCK SOURCE and BRIDGE modules connect to the signals in the Extended Medium, namely Extended Clock (EXTCLK⁺, EXTCLK[−]) and Extended Data (EXTDAT⁺, EXTDAT[−]). BRIDGES and BUS ACCESS modules connect to the signal lines in the Backplane Medium, namely Serial Clock (SERCLK) and Serial Data (SERDAT^{*}).

4.1 CLOCK SOURCE module

The CLOCK SOURCE module is shown in Figure 4-1, page 21. It drives the EXTCLK⁺ and EXTCLK[−] lines in the Extended Medium.

RULE 4.1:

Regardless of whether a serial bus is used in a single backplane or extends among several subracks and stand-alone boards, it MUST contain one and only one (active) CLOCK SOURCE.

4.1.1 Medium interface

RULE 4.2:

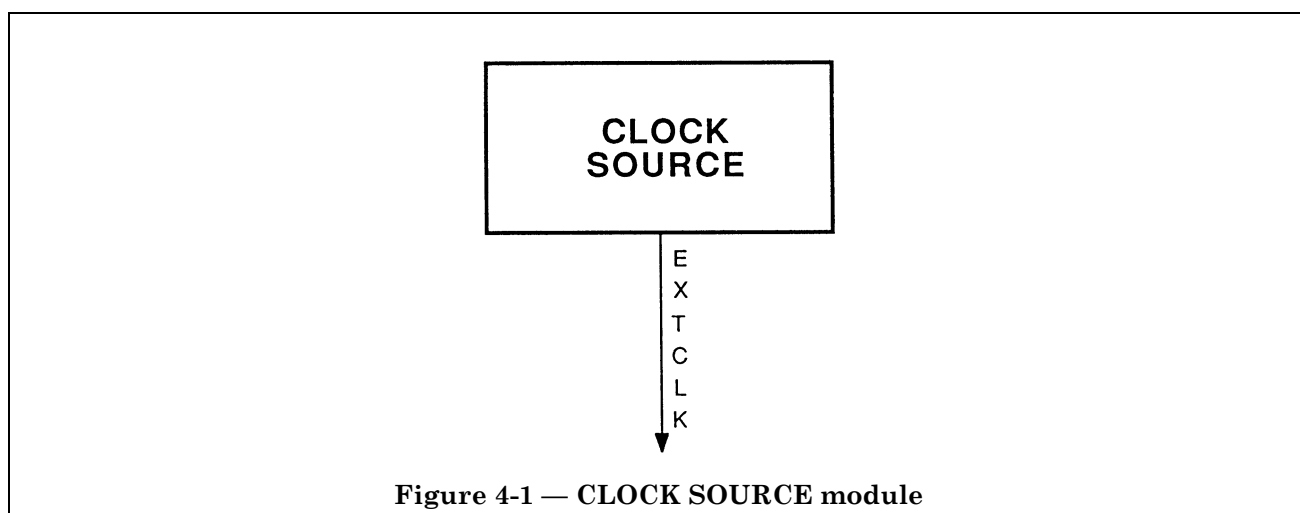
IF a board is connected to an Extended Medium,
AND it contains the (active) CLOCK SOURCE,
THEN it MUST be located at one end of the Extended Medium.

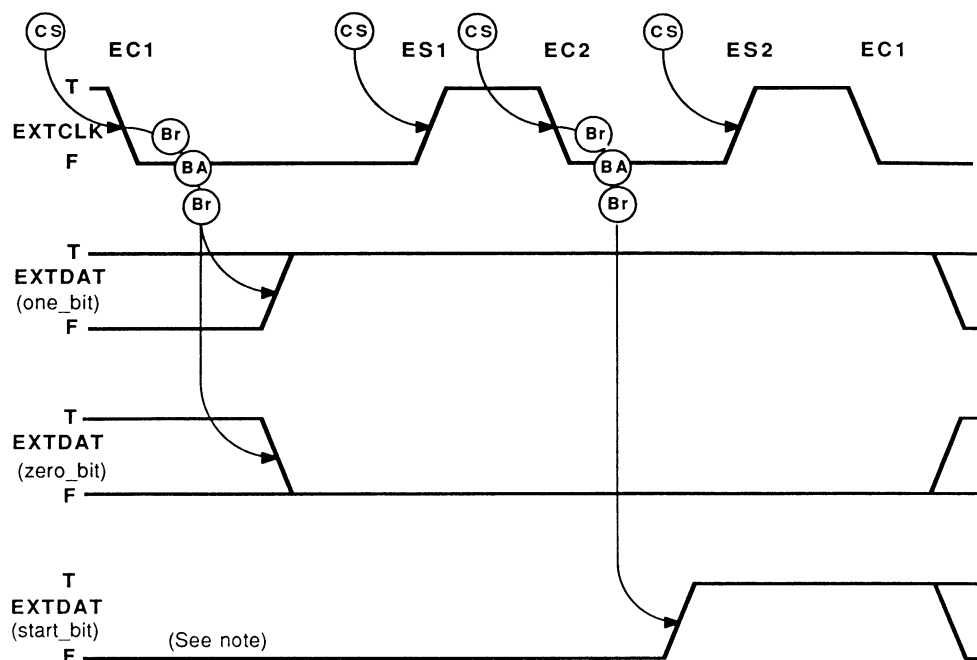
RULE 4.3:

IF a board is connected to an Extended Medium,
AND it contains the (active) CLOCK SOURCE,
THEN it MUST drive the EXTCLK⁺ and EXTCLK[−] lines with a full-time differential driver with the electrical characteristics given in Section 4.4.

RECOMMENDATION 4.1:

To allow multiple boards containing CLOCK SOURCE hardware to be used on the same serial bus, design them so that the EXTCLK driver can be disconnected from the Extended Medium.





NOTE As described in Chapter 5, a start_bit is always preceded by a zero_bit.

Figure 4-2 — Extended bus waveforms

4.1.2 Physical and Link Layer interfaces

The CLOCK SOURCE has no interface with the Link Layer, nor with other elements of the Physical Layer, except via the Extended Medium.

Paragraph 4.2.2 discusses single-subrack serial bus configurations, in which the “Extended Medium” consists only of on-board signals between a CLOCK SOURCE and a BRIDGE.

4.1.3 Operation

The waveform with which the CLOCK SOURCE drives EXTCLK⁺ and EXTCLK⁻ is shown in Figure 4-2. The specific timing of this signal is a function of whether or not the Extended Medium is used and, if so, of its length, and is specified in Chapters 7 and 8.

Figure 4-2, Figure 4-4 and Figure 4-6, pages 22, 24 and 27, include cause-and-effect arrows to indicate how the various Physical Layer modules interact to produce signal transitions on the serial bus. One of the symbols:



on an arrow indicates that the CLOCK SOURCE, BRIDGE or BUS ACCESS module, respectively, responds to the transition at the start of the arrow by producing the transition at the end of the arrow. More than one of these symbols on an arrow means that several modules respond (sequentially and in the order shown) to produce the indicated transition.

RECOMMENDATION 4.2:

To allow for use of Extended Media of various lengths, design CLOCK SOURCE hardware so that it can be jumpered to operate at the various speeds cited in Chapters 7 and 8.

4.2 BRIDGE module

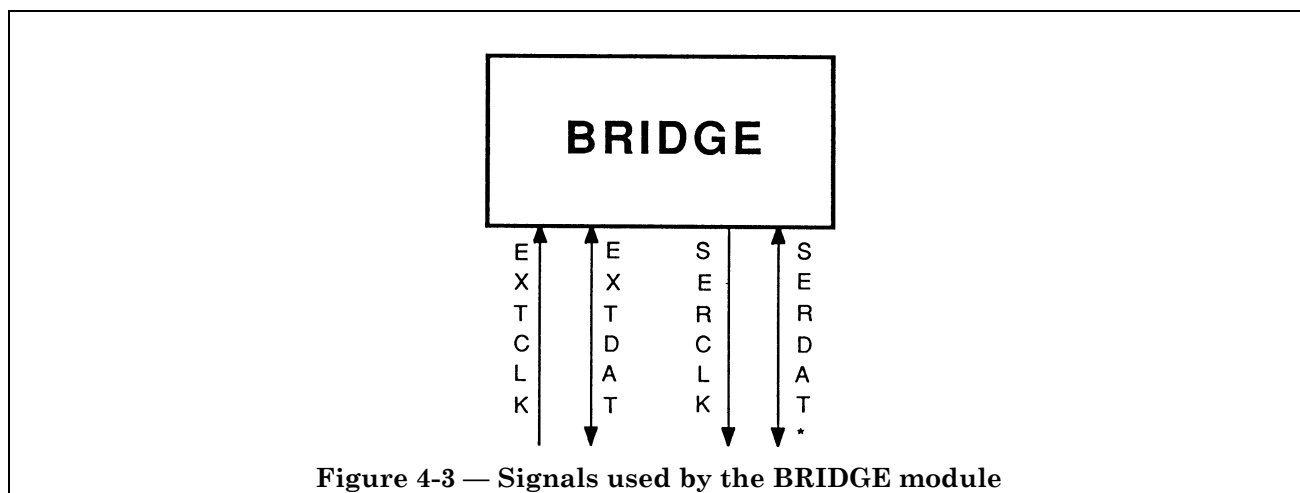
The BRIDGE module is shown in Figure 4-3, page 23. It interconnects an Extended Medium and a Backplane Medium, and translates the electrical levels between them. As described in this section, it also derives the backplane timing from that on the Extended Medium.

RULE 4.4:

Each subrack or free-standing board on an Extended Medium **MUST** have one and only one (active) BRIDGE.

4.2.1 Media interfaces

The BRIDGE receives the EXTCLK⁺ and EXTCLK[–] signals from the Extended Medium, and drives the SERCLK signal in the Backplane Medium. It drives and receives the EXTDAT⁺ and EXTDAT[–] lines in the Extended Medium, and drives and receives the SERDAT* line in the Backplane Medium.



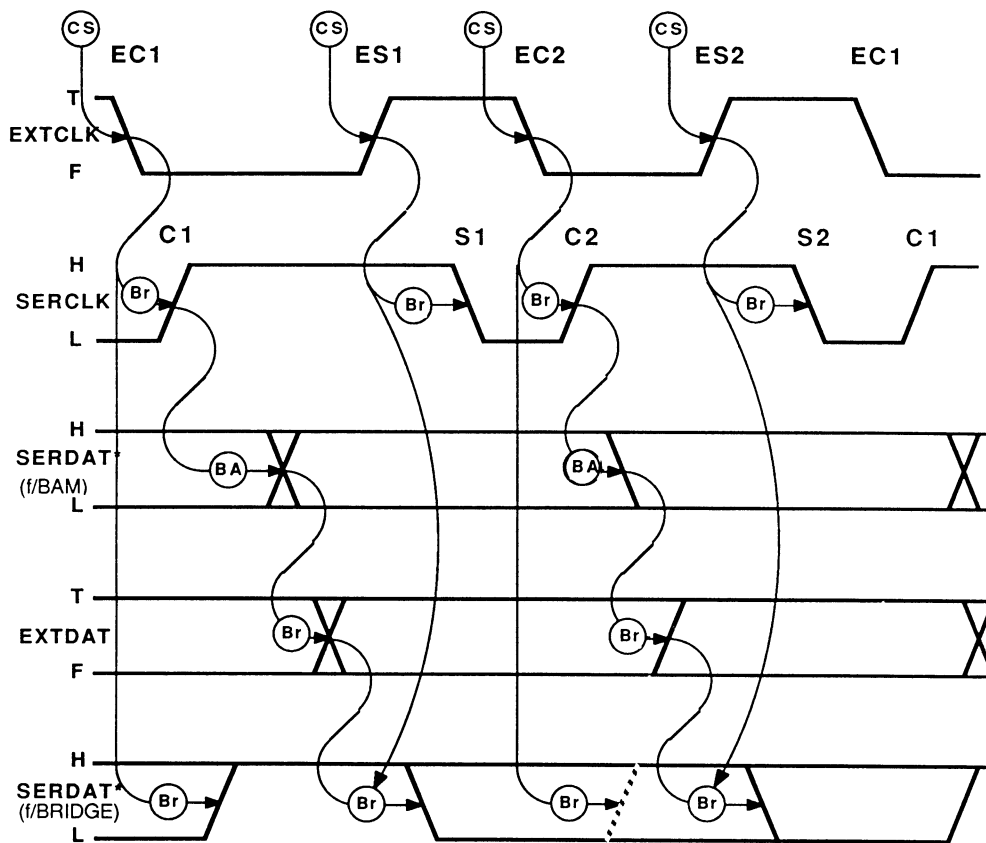


Figure 4-4 — Serial bus waveforms

RULE 4.5:

IF a board provides for Extended Medium connection
 AND it contains an (active) BRIDGE,
 THEN it MUST drive EXTDAT with a part-time differential driver with the electrical characteristics given in Section 4.4.

Any BRIDGE may drive EXTDAT True, so that the voltage on EXTDAT⁺ is higher than the voltage on EXTDAT⁻. When no BRIDGE drives the lines in this way, the Extended Medium's terminating resistors pull EXTDAT False, so that EXTDAT⁺ has a lower voltage than EXTDAT⁻.

RULE 4.6:

BRIDGE modules MUST NOT drive EXTDAT False, so that the voltage on EXTDAT⁺ is lower than that on EXTDAT⁻.

SUGGESTION 4.1:

To meet the above RULES, design BRIDGE modules to drive EXTDAT with an "RS 485" driver that has an Enable input. Connect the on-board serial data to Enable. Connect the driver's normal data input to a logic high or low level, or to the on-board serial data, as appropriate.

RULE 4.7:

IF a board is connected to a Backplane Medium,
 AND it contains the (active) BRIDGE,
 THEN the board MUST be located at one end of the Backplane Medium.

RULE 4.8:

IF a board is connected to a Backplane Medium,
 AND it contains the (active) BRIDGE,
 THEN it MUST drive SERCLK with a TTL-compatible high-current totem-pole driver with the electrical characteristics given in Section 4.4.

RULE 4.9:

IF a board is connected to a Backplane Medium,
 AND it contains the (active) BRIDGE,
 THEN it MUST drive SERDAT* with a TTL-compatible open-collector driver with the electrical characteristics given in Section 4.4.

4.2.2 Physical and Link Layer interfaces

The BRIDGE has no interface with the Link Layer, nor with other elements of the Physical Layer, except via the Extended and Backplane Media.

Paragraph 4.3.2 discusses free-standing serial bus boards, in which the Backplane Medium consists only of on-board signals between a BRIDGE and a BUS ACCESS module.

PERMISSION 4.1:

A board that is intended for use only in single-subrack configurations MAY include the CLOCK SOURCE function and the SERCLK—driving function of the BRIDGE module, without providing an Extended Medium connection. On such a board, the logical signal EXTCLK MAY be implemented without regard for the RULES for the Extended Medium, provided that the RULES for the Backplane Medium are observed.

RECOMMENDATION 4.3:

Design boards that drive SERCLK so that they include a connection to the Extended Medium. Provide a separate CLOCK SOURCE that can be disconnected from the Extended Medium.

OBSERVATION 4.1:

Designing boards in this way allows them to be used in both single-subrack and extended applications. Making the CLOCK SOURCE disconnectable allows the design to be used in all the subracks in an extended bus.

RULE 4.10:

IF a board provides for Extended Medium connection,
 AND it includes both a CLOCK SOURCE and a BRIDGE,
 THEN the BRIDGE MUST receive EXTCLK from the Extended Medium rather than directly from the CLOCK SOURCE via an on-board signal.

4.2.3 Operation

Each “bit cycle” on EXTCLK is used to send a single bit on EXTDAT and includes four transitions designated EC1, ES1, EC2 and ES2, as shown in Figure 4-4, page 24. Throughout the bit cycle, the BRIDGE module drives EXTDAT True if it receives SERDAT* low, and releases EXTDAT if it receives SERDAT* high. The four EXTCLK transitions are used by the BRIDGE as follows:

<i>Edge Designation</i>	<i>Use</i>
EC1	From this transition of EXTCLK, a BRIDGE drives SERCLK high (making the C1 transition of SERCLK) and releases its SERDAT* driver.
ES1	On this transition of EXTCLK, a BRIDGE samples the state of EXTDAT. If it samples EXTDAT true, it drives SERDAT* low. Regardless of the state of EXTDAT, the BRIDGE then delays a specified time to allow SERDAT* to settle, and then drives SERCLK low, making the S1 transition.
EC2	From this transition of EXTCLK, a BRIDGE drives SERCLK high, making the C2 transition of SERCLK. In addition, it may release its SERDAT* driver.