

IEEE Standard for Spaceborne Fiber-Optic Data Bus

Sponsor

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

Approved 4 February 1999

IEEE-SA Standards Board

Abstract: The design requirements for a fiber-optic serial interconnect protocol, topology, and media is established. The application target for this standard is the interconnection of multiple aerospace sensors, processing resources, bulk storage resources, and communications resources onboard aerospace platforms. This standard is for subsystem interconnection, as opposed to intra-backplane connection.

Keywords: data link layer, fiber optic, fiber-optic bus, fiber-optic network, intranet, management layer, network, physical layer, space, space environment

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

Copyright © 1999 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 23 April 1999. Printed in the United States of America.

Print: ISBN 0-7381-1549-5 SH94713
PDF: ISBN 0-7381-1550-9 SS94713

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

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

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

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

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating 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-SA Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

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

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (978) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

(This introduction is not part of IEEE Std 1393-1999, IEEE Standard for Spaceborne Fiber-Optic Data Bus.)

Application perspective

The bandwidth and resolution of aerospace remote-sensing payloads continue to advance, placing ever increasing demands on onboard data-handling networks. At the same time, economics and rapid development requirements are driving onboard data-handling networks toward flexible, nonproprietary architectures and interface standardization. These characteristics are extensively addressed in existing ground-based networks. However, unlike most ground-based networks designed primarily to support the transfer of non-real-time data between computers, an onboard data-handling network must support the real time needs of the aerospace remote-sensing environment.

Network perspective

Aerospace remote-sensing data is characterized by synchronous components, common to continuous mode sensors, and asynchronous components, common to event driven sensors. Both sensor types have real time data-handling requirements, and sensor performance is driving data bandwidth requirements into multiple gigabit per second range. Even at these higher data rates, aerospace systems will continue to be constrained by size, weight, and power limitations. In order to achieve high data-rate performance while maintaining low size, weight, and power, onboard data-handling subsystems must employ highly integrated components and must avoid the higher layer protocol features commonly found in ground-based networks. Furthermore, space-based data-handling networks must be fault tolerant and able to withstand the rigors of launch and the harsh space environment.

Adaptation of existing ground-based network standards is not practical because of insufficient bandwidth and high power dissipation. Also, conversion of current implementations to space qualifiable processes is not feasible. Therefore, a new onboard data-handling network standard for aerospace remote-sensing applications is necessary.

Technology perspective

Fiber-optic and ASIC technologies have matured to the point that multiple gigabit data-handling networks for aerospace applications are practical. Fiber-optic technology is especially appealing because of its extremely high bandwidth capacity. Multimode fibers have been demonstrated to be radiation hard with negligible increase in signal loss over a ten year life in space radiation environments. Space qualifiable connectors have been designed and built specifically for fiber cable. Techniques for low loss termination and coupling of fibers have been perfected. High power laser diodes capable of operation over a wide temperature range and low noise optical receivers are now available. State-of-the-art semiconductor materials and processes, as well as packaging techniques, are leading towards smaller, faster, and lower power devices that will enable high-speed data-handling systems for space applications.

Participants

The following is a list of participants in the Spaceborne Fiber-Optic Data Bus (SFODB) Working Group.

Fred J. Orlando, *Acting Chair*

Pete Ayotte
Charles Chalfant

John DeRuiter
Kaye Rankin
Steven L. Rankin

William Schneider, *Chair*

Ron Smith
Bruce Urbon

The following individuals have made significant contributions to this standard:

Dennis Andrucyk
Drew Bertagna
Cheryl Dale
Mark Flannagan
Gary Gardner
Steve Gross

Feiling Jia
Conrad A. Laurvick
Paul Marshall
Lisa McMurray
Diana Orlando

Tom Reglein
Joe Rich
Ali Senol
Tim Spinney
Eric Swanson
Darby Terry

The following members of the balloting committee voted on this standard:

Malcolm J. Airst
Charles Chalfant
Christopher Dailey

Gary Gardner
Conrad A. Laurvick
Fred J. Orlando
Steven L. Rankin

Ed Rezek
Marc Sawyer
Jim Toy

The final conditions for approval of this standard were met on 4 February 1999. This standard was conditionally approved by the IEEE-SA Standards Board on 8 December 1998, with the following membership:

Richard J. Holleman, *Chair*

Donald N. Heirman, *Vice Chair*

Judith Gorman, *Secretary*

Satish K. Aggarwal
Clyde R. Camp
James T. Carlo
Gary R. Engmann
Harold E. Epstein
Jay Forster*
Thomas F. Garrity
Ruben D. Garzon

James H. Gurney
Jim D. Isaak
Lowell G. Johnson
Robert Kennelly
E. G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Jim Logothetis
Donald C. Loughry

L. Bruce McClung
Louis-François Pau
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Hans E. Weinrich
Donald W. Zipse

*Member Emeritus

Yvette Ho Sang
IEEE Standards Project Editor

Contents

1. Overview	1
1.1 Scope	1
1.2 Purpose.....	1
1.3 Organization of document map.....	2
2. References.....	2
3. Definitions, acronyms, abbreviations, and conventions	2
4. Network description.....	7
4.1 Introduction.....	7
4.2 Operation.....	12
4.3 Sample startup scenario	33
5. Physical plant.....	33
5.1 Topology	33
5.2 Fiber-optic cable	33
5.3 Connectors	37
6. Fiber-optic bus interface unit (FBIU)	37
6.1 FBIU Physical Layer.....	39
6.2 FBIU Data Link Layer.....	46
6.3 FBIU Management Layer processes.....	58
7. Control fiber-optic bus interface unit (CFBIU).....	77
7.1 CFBIU Physical Layer.....	78
7.2 CFBIU Data Link Layer	82
7.3 CFBIU Management Layer.....	85
8. Operating Environment.....	117
8.1 Temperature	118
8.2 Pressure	118
8.3 Outgassing requirements.....	118
8.4 Random vibration.....	119
8.5 Radiation	119
Annex A (informative) Bibliography	120

IEEE Standard for Spaceborne Fiber-Optic Data Bus

1. Overview

1.1 Scope

This standard establishes the design requirements for a fiber-optic serial interconnect protocol, topology, and media. The application target for this standard is the interconnection of multiple aerospace sensors, processing resources, bulk storage resources, and communications resources on board aerospace platforms. This standard is for subsystem interconnection, as opposed to intrabackplane connection.

1.2 Purpose

The purpose of this standard is the specification of a highly reliable and fault tolerant fiber-optic network that is compatible with the harsh thermal, mechanical, and radiation environments of aerospace applications requiring small size and low power dissipation. Current standards do not cover data rate needs above 200 Mbps. This standard will specify a node-to-node operating data rate range of 200 Mbps through 1 Gbps. The standard will define physical and logical protocol, and subnetwork management functions that allow mixing high-speed synchronous and asynchronous distributed access for deterministic communications.

1.3 Organization of document map

Figure 1.1 maps out the organization of this standard.

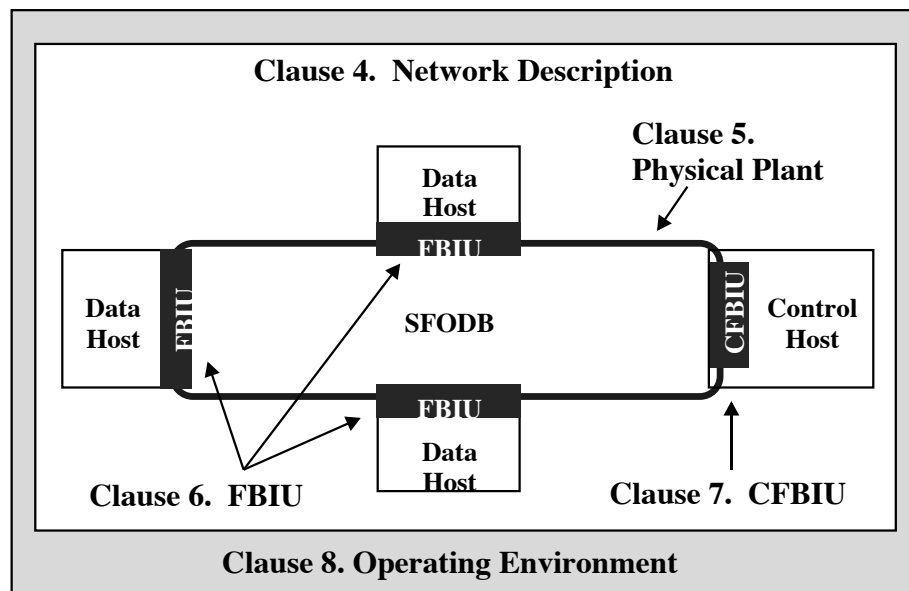


Figure 1.1—Document map

2. References

This standard shall be used in conjunction with the following publications. When the following standards are superseded by an approved revision, the revision shall apply.

ANSI T1.627-1993, American National Standard Telecommunications—Broadband ISDN—ATM Layer Functionality and Specification.¹

ANSI X3.230-1994, American National Standard Information Technology—Fibre Channel Physical and Signalling Interface (FC-PH).

IEEE Std 1156.4-1997, IEEE Standard for Environmental Specifications for Spaceborne Computer Modules.²

3. Definitions, acronyms, abbreviations, and conventions

3.1 Definitions

3.1.1 8B/10B encoding: A byte-oriented encoding scheme developed by IBM which encodes 8-bit data into “dc balanced” 10-bit symbols. 8B/10B encoding guarantees a minimum of four (4) non-return-to-zero (NRZ) transitions per 10-bit symbol and, with “running disparity,” produces a bit stream with zero dc offset.

¹ANSI publications are available from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA (<http://www.ansi.org/>).

²IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (<http://www.standards.ieee.org/>).

3.1.2 bandwidth reuse: A ring segmentation feature that multiplies the overall data throughput capacity of the spaceborne fiber-optic data bus (SFODB) network by allowing independent ring segments to use the same dedicated data bandwidth.

3.1.3 bit clock: A clock recovered from the input Rx Serial Symbol stream which is bit synchronous with less than 5 degrees of single-sided phase noise.

3.1.4 byte clock: A clock derived from the bit clock with a period equal to 10 clock cycles.

3.1.5 center wavelength: The wavelength that is the arithmetic mean of the half-maximum spectral intensity points of the transmitter. If the spectral intensity distribution is symmetric and singly peaked, the center wavelength is at maximum intensity.

3.1.6 control host: The spaceborne fiber-optic data bus (SFODB) network management node.

3.1.7 data host: The spaceborne fiber-optic data bus (SFODB) source and destination nodes.

3.1.8 dc level unmodulated dc power: Average dc level of an optical signal. For example, if 100% effective power is 3 mW (4.8 dBm) and 0% effective power is 1 mW (0 dBm) the dc level is 2 mW (3 dBm).

3.1.9 effective power: The difference, expressed in dBm, between the absolute optical power, measured in milliwatts, at the midpoint in time of a high optical signal vs. the midpoint in time of a low optical signal.

3.1.10 encoded symbol: A 10-bit symbol created from 8-bit data using the 8B/10B encoding scheme.

3.1.11 fiber-optic receiver operating range: The range of optical power over which the fiber-optic receiver will meet the specified bit error rate (BER).

3.1.12 frame: A set of consecutive time slots in which the position of each time slot can be identified by reference to a framing signal.

3.1.13 K28.5: The 8B/10B control (K) symbol used as one of the spaceborne fiber-optic data bus (SFODB) Frame Sync characters.

3.1.14 K28.7: The 8B/10B control (K) symbol used as the other spaceborne fiber-optic data bus (SFODB) Frame Sync character.

3.1.15 latency adjust buffer (LAB): A first-in first-out (FIFO) function within the CFBIU with the ability to store one full frame of Rx Data. The LAB function insures an integral number of frames are rotating on the SFODB ring.

3.1.16 optical fall time: The time it takes for optical power to fall from 90% effective power to 10% effective power. When expressed as a percentage, the fall time is specified as a percentage of an encoded bit period.

3.1.17 optical rise time: The time it takes for optical power to rise from 10% effective power to 90% effective power. When expressed as a percentage, the rise time is specified as a percentage of an encoded bit period.

3.1.18 standard test fiber: A silica graded index multi-mode optical fiber with a core diameter of 100 μm , an outside diameter of 140 μm , a numerical aperture of 0.29, a bandwidth of 400 MHz-km at 1300 nm and terminated in a standard subminiature assembly (SMA) connector.

3.1.19 service data unit (SDU): The 48-byte data payload of an asynchronous transfer mode (ATM) Cell.

3.1.20 subframe overhead (OH): 12-byte SFODB Overhead structure used to transfer configuration and status information between the control fiber-optic bus interface unit (CFBIU) and fiber-optic bus interface units (FBIUs) on the network.

3.1.21 symbol: A 10-bit, 8B/10B encoded byte.

3.1.22 token: The 3-bit field of authority that is passed between data hosts using a token access method to indicate which data host is currently in control of the medium.

3.1.23 token access: The asynchronous data transmission process utilizing tokens to indicate which fiber-optic bus interface unit (FBIU) is currently in control of a token group.

3.1.24 token group: There are four token groups. Each fiber-optic bus interface unit (FBIU) can be a member of up to four token groups. Members of each token group share access to the same Tx Data Slots defined by the Token Arbitrated Transmit Slot Masks of FBIU configuration and status registers 16 through 19 [FCSR-(16–19)]. That is, FCSR-(16–19) for each member of a token group must be identical. This shared access is controlled using a simple token passing protocol.

3.2 Acronyms and abbreviations

ATM	asynchronous transfer mode
BER	bit error rate
CCSR	CFBIU configuration and status register
CCIR	CFBIU command interface register
CFBIU	control fiber-optic bus interface unit
CLP	cell loss priority (ATM)
CRC	cyclic redundancy check
Demux	demultiplex
DLL	data link layer
DMA	direct memory access
DPRAM	dual port random access memory
EDAC	error detection and correction
FBIU	fiber-optic bus interface unit
FCSR	FBIU configuration and status register
FWHM	full width half maximum
HEC	header error check
I&T	integration and test

LAB	latency adjust buffer
MDPL	media dependent physical layer
ML	management layer
NMDPL	non-media dependent physical layer
NRZ	non-return to zero
OH	overhead
OSI	open system interconnection
Rx	receiver (when referring to the fiber-optic receivers) received (when referring to data or Subframe OH)
SDU	service data unit
SEU	single event upset
SFODB	spaceborne fiber-optic data bus
SMA	subminiature assembly
TDMA	time division multiple access
TRT	token rotation time
Tx	transmitter (when referring to the fiber-optic transmitters) transmitted (when referring to data or Subframe OH)
VCi	virtual channel identifier
VPI	virtual path identifier

3.3 Language Conventions

3.3.1 Structured English statements

Throughout this standard, processes will be defined by the use of Structured English statements. The following conventions have been followed:

- Action words are ***bold italics***.
- Signal names are **bold**.
- Conditional words and joining words are *italics*.
- Signal state words are ALL CAPITALS.
- Punctuation: The first requirement in the statement is left justified with additional requirements indented and followed by a semicolon. The first condition is left justified with additional conditions and/or alternate conditions indented. A period is at the end.
- Subprocesses designated by a., b., c., etc. are intended to be executed in order.

3.3.2 Specific action words

The following defines the word conventions used for specific actions:

- **Set** defines a static condition and is always followed by the definitive signal condition.
- **Reset** defines a temporary or pulsed condition that causes a signal to be set to a False or Logic “0” state. When used to define an action to a register or counter, it defines a temporary or pulsed condition that causes the register or counter to be set to the all zeros state.
- **Assert** defines a temporary or pulsed signal state of TRUE.

3.3.3 Signal names

Signal names are in **bold type** and have been structured to help describe the direction of data flow between interfaces. Each signal name has a prefix that defines its origin followed by an underscore symbol. The syntax of the signal naming convention is defined as follows:

- External inputs-to-SFODB signals: External input signals will be preceded by its origin followed by an underscore symbol. For Example: **Data Host_Host Transmit Select**.
- Process-to-process signals: Signals that flow between processes within an FBIU or CFBIU will be preceded by its protocol layer of origin abbreviation followed by an underscore symbol and its process origin followed by another underscore symbol. For Example: **ML_Rx OH Cmd Execution_Normal Mode** designates a signal named Normal Mode originating from the Rx OH Cmd Execution process within the Management Layer (ML).

NOTE—Inter-OS Layer signals represent requirements. Intra-OS Layer signals are for descriptive purposes only.

- CFBIU configuration and status register (CCSR), FBIU configuration and status register (FCSR), and CFBIU command interface register (CCIR) fields: When referencing a field within a CCSR, FCSR or CCIR, the name will be preceded by “CCSR-N_,” “FCSR-N_,” or “CCIR-N_,” respectively, where N is the register number. For example: **FCSR-8_Tx Select** refers the Tx Select field of FBIU Command & Status Register number eight. When referencing multiple CCSRs, FCSRs or CCIRs, “CCSR-(x-y),” “FCSR-(x-y),” or “CCIR-(x-y)” will be used to represent registers “x” through “y.” For Example: CCSR-(0–4) represents the contents of CCSR-0 through CCSR-4.
- Received subframe overhead (Rx Subframe OH) fields: When referencing a field within the Received Subframe Overhead Bytes, the name will be preceded by “Rx Subframe OH_.” For example: **Rx Subframe OH_Frame Type**.
- Transmitted subframe overhead (Tx Subframe OH) fields: When referencing a field within the Transmitted Subframe Overhead Bytes, the name will be preceded by “Tx Subframe OH_.” For Example: **Tx Subframe OH_Frame Type**.
- Brackets: Brackets within a signal name or register field name represent a multiple bit signal or multiple bits of a field. For example: **Data Host_Host Rx Select [2:0]** represents a 3-bit input signal from the Data Host with the most significant bit (msb) defined by “2” and the least significant bit (lsb) defined by “0.”
- Big Endian/Little Endian: Unless otherwise stated, bytes and words are in Little Endian bit format, ATM Cells and Subframe Overhead are in Big Endian byte format.

4. Network description

4.1 Introduction

This clause introduces the spaceborne fiber-optic data bus (SFODB) functional elements, maps the relevant open system interconnection (OSI) protocol layers to the SFODB functional elements and highlights the key characteristics of the SFODB. No binding requirements are specified in this subclause. SFODB protocol layers and processes are described in their entirety for each functional element in Clause 5, Clause 6, and Clause 7.

4.1.1 Topology

The physical topology of the SFODB network is a ring of FBIUs interconnected by a fiber-optic Physical Plant with a CFBIU for network configuration and control (see Figure 4.1).

The Physical Plant contains the passive elements of the SFODB network; i.e., the fiber-optic cables and connectors.

The FBIU is intended to be an embedded module within each Data Host. It provides the primary data-handling functions required to both send and receive data over the SFODB network.

The CFBIU is intended to be an embedded module within a Control Host. It provides the primary configuration, control, and status monitoring interface to the SFODB network.

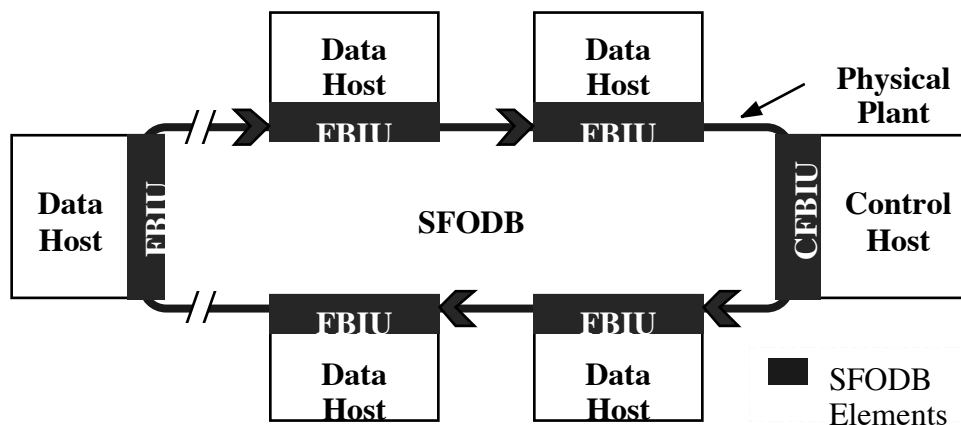


Figure 4.1 – SFODB network topology

4.1.2 OSI protocol layer allocation

In an effort to standardize the protocol structure, the International Standards Organization (ISO) created the 7-layer OSI model to define and partition the basic network processes (see Figure 4.2). The SFODB network incorporates the three layers of this 7-layer model associated with data transfer, the Physical Layer, Data Link Layer, and Network Layer. The SFODB also provides a limited set of Network Management functions. These Network Management functions have been combined with the Network Layer functions to form the SFODB ML. Figure 4.3 illustrates, in a simplified form, the allocation of these protocol layers to the SFODB functional element.

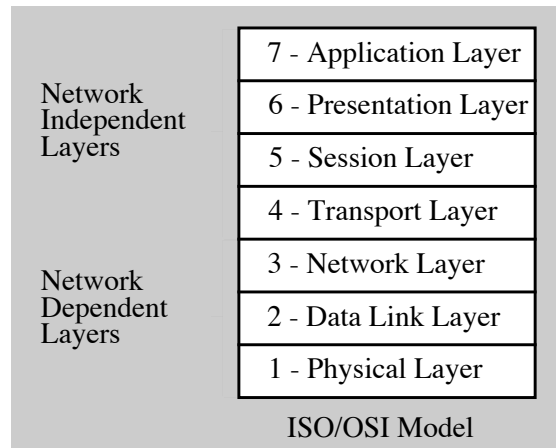


Figure 4.2—ISO/OSI model

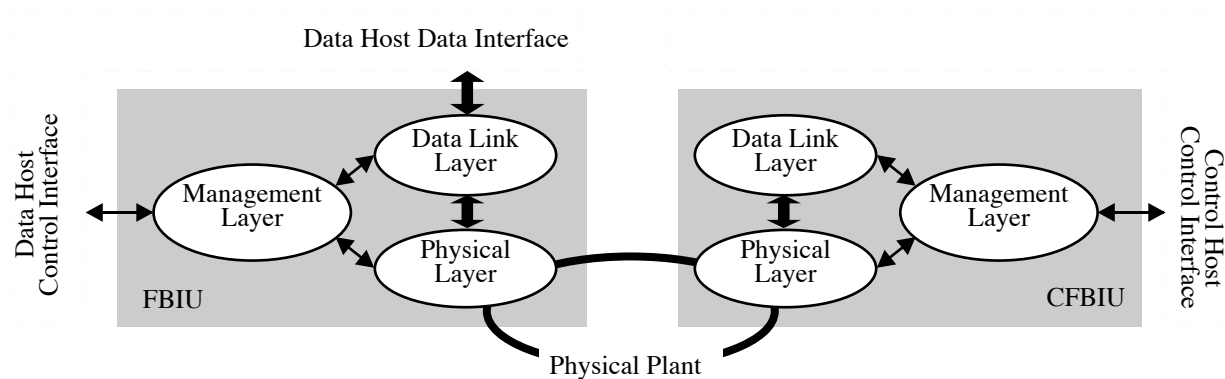


Figure 4.3—Protocol layer allocation

- Physical Layer:** In the SFODB the Physical Layer is divided into two parts, the Media Dependent Physical Layer (MDPL) and the Non-Media Dependent Physical Layer (NMDPL). The passive elements of the MDPL, the network cable and connectors, are contained within the Physical Plant. The active elements of the MDPL associated with signaling and the NMDPL processes associated with symbol encoding and timing are contained within the FBIU and CFBIU.
- Data Link Layer (DLL):** The DLL functions associated with data formatting, data transfer, and the Data Host interface are provided by the FBIU. The DLL functions associated with network synchronization and frame formatting are provided by the CFBIU. No Network Independent Layers are provided by the SFODB.
- Management Layer (ML):** In the SFODB, the Network Layer functions associated with network bandwidth allocations and data routing are considered part of the ML. These non-real-time ML functions are provided by the Control Host with the CFBIU serving as the command and status interface between the Control Host and the SFODB network.

4.1.3 Network characteristics

4.1.3.1 Physical layer characteristics

The baseline configuration for the SFODB network is a redundant, cross-strapped, serial ring with a passive optical bypass feature (see Figure 4.4). The network is capable of supporting up to 127 FBIUs and a CFBIU with a maximum node-to-node spacing of 50 m.

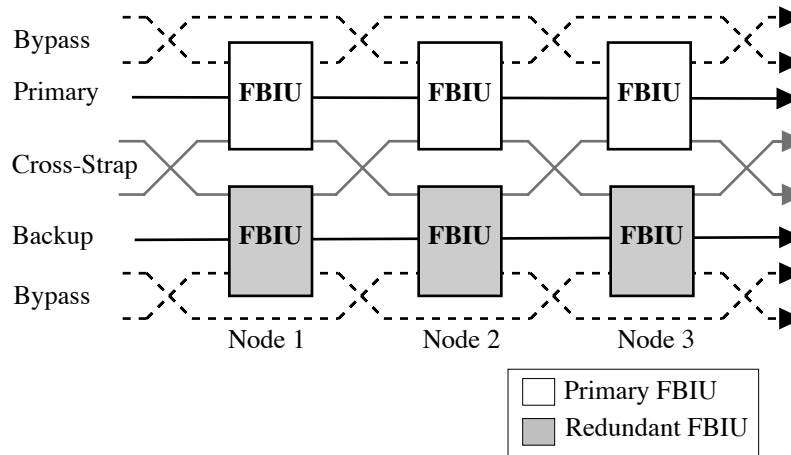


Figure 4.4—SFODB redundancy configuration

The MDPL is characterized by multimode, graded index fiber with laser diode transmitters and PIN diode receivers operating at an optical frequency in the 1300 nm range. Although multifiber cable, multipin connectors, and specific termini are recommended for interoperability, the user is free to select a different cable interconnect scheme.

The NMDPL is characterized by a continuous transmission, serial communications mode. Data are encoded using 8B/10B symbol encoding, and frame level synchronization is maintained using the unique K28.5 and K28.7 8B/10B command codes. The encoded symbols are transmitted serially using direct modulation digital On/Off signaling. The SFODB supports a selectable data throughput rate of 200 Mbps to 1 Gbps, which equates to an encoded bit rate of 300 Mbps to 1.458 Gbps, including ATM Headers and SFODB overhead.

4.1.3.2 Data Link Layer characteristics

The DLL services are provided by the protocol processing elements of the FBIU and CFBIU. These services include the Data Host-to-FBIU and FBIU-to-Data Host data transfer services, the ATM Cell formatting services, the SFODB media access services, and the CFBIU Latency Adjust Buffer (LAB) services.

Transfer of data between Data Hosts on the SFODB ring is accomplished using a simple fixed length 32-Slot, time division multiple access (TDMA) frame format. Each of the 32-Slots are a fixed 56 bytes in length and contain a standard 53-byte ATM Cell and 3 bytes of SFODB control overhead (see Figure 4.5). There is a deterministic delay around the ring equal to one, two, or three frames, depending on the number of FBIUs and the distance between the FBIUs in a particular network configuration. This deterministic delay is maintained by a LAB within the CFBIU that compensates for FBIU quantity and spacing variations and insures that an integral number of frames is rotating on the ring.

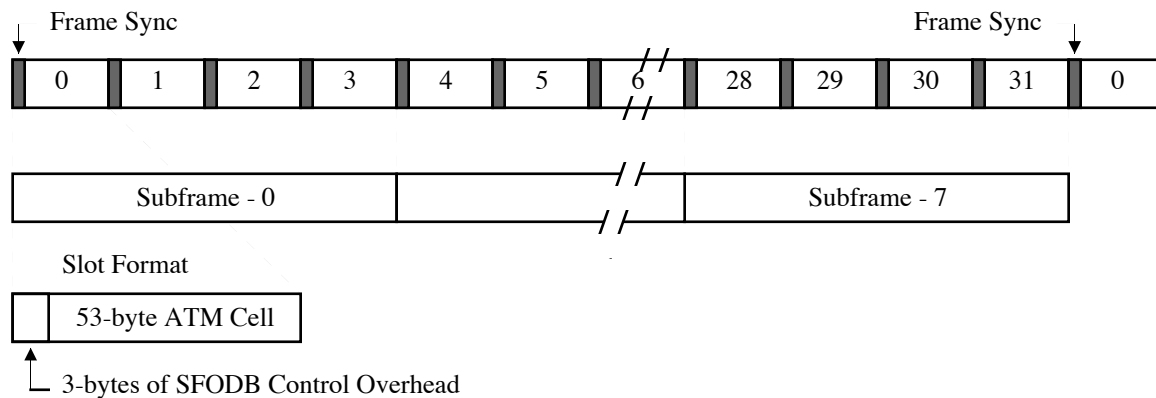


Figure 4.5—SFODB Frame format

Network bandwidth can be dedicated to an FBIU using the Dedicated Transmit Slot process or shared between multiple FBIUs using the Token Arbitrated process. Both processes transmit data in a broadcast mode. Data delivery can be connection oriented using the Dedicated Receive Slot process or packet oriented using the ATM Header Address process. Further explanation of these processes is provided later in this sub-clause.

The transfer of an ATM Cell between the FBIU and the Data Host forms the DLL boundary. The Data Host is expected to provide all message layer and higher layer OSI protocol services. In the special case of continuous unformatted data transfer between the FBIU and its Data Host, the FBIU can perform the data segmentation and reassembly function normally attributed to the Message Layer.

The following items summarize the key characteristics of the SFODB DLL.

- a) **Data format:** The SFODB network is capable of transferring continuous, unformatted data and formatted data packets between FBIUs. Both data types can be transferred synchronously or asynchronously. The format of the packetized data can be either simple 48-byte blocks or fully formatted ATM Cells.
- b) **Data delivery:** An SFODB network supports both circuit-switched and packet-switched data delivery services. These services may be mixed on the same SFODB network. Both services use multicast as the primary means of data transfer so that data destined for multiple FBIUs are transmitted only once.
- c) **Data throughput:** The SFODB network is scalable to allow the network to be optimized to meet specific spacecraft data throughput requirements. The network elements are scalable to accommodate a maximum node-to-node data throughput rate of 200 Mbps to 1 Gbps.
- d) **Bandwidth reuse:** In a structured spacecraft architecture, the data-handling capacity of an SFODB network can be multiplied several times by utilizing the SFODB's Bandwidth Reuse feature. Bandwidth Reuse simply means that once data reaches the destination FBIU the bandwidth used to transport that data can be immediately reused by the destination FBIU or any succeeding FBIU.
- e) **Deterministic latency:** Another key feature of an SFODB network is the deterministic nature of the data transfer latency. The data transfer latency on an SFODB network is deterministic and stable to within 1 byte. This is a product of the fixed-length frame format and the LAB within the CFBIU.
- f) **Data transfer options:** The SFODB network offers the user several methods of transferring data between FBIUs. In fact, the ability to transfer both synchronous and asynchronous data, using either dedicated or token arbitrated bus bandwidth, in either a circuit-switched or packet-switched network

configuration, can present a confusing set of options. However, all combinations of data-transfer methods available on an SFODB network can be simplified into two methods of sending data and two methods of receiving data.

- 1) Sending data: Network bandwidth is allocated by assigning one or more of the 32-Slots of an SFODB Frame to each of the FBIUs. These Slots can be either dedicated to a specific FBIU or shared between multiple FBIUs. When Slots are dedicated, the FBIU can insert data onto the network each time the assigned Slots rotate pass the FBIU. When Slots are shared, the FBIUs utilize token passing to arbitrate access to the assigned Slots with other FBIUs. The Dedicated Transmit Slot process is primarily intended for the transmission of continuous, high rate sensor data. The Token Arbitrated process allows multiple sources to efficiently share one or more Slots on the bus. Both transmit processes are broadcast in nature. Slot assignment is dynamically managed by the Control Host using the SFODB Subframe OH allowing real time bandwidth management without reducing data throughput.
 - 2) Receiving data: The broadcast data is accepted or rejected using either a Dedicated Receive Slot process or an ATM Header Addressed process. The Dedicated Receive Slot process supports the circuit switched method of data transfer. The ATM Header Addressed process supports the packet switched method of data transfer. The latter allows an SFODB network to function in a limited capacity as a distributed ATM switch. The data acceptance criteria is based on receive masks that define the Slots to be examined and virtual path identifier/virtual channel identifier (VPI/VCI) addresses to be accepted. Like the transmit Slot assignments, the FBIU receive masks are dynamically managed by the Control Host using the SFODB Subframe OH.
- g) Data Host/FBIU interface: Each FBIU provides its Data Host with full SFODB send and receive data access through two independent interface ports. Both ports are capable of transferring data in either ATM Cell format, 48-byte format, or as continuous, unformatted data. The use of the ATM Cell format as the data transfer format on an SFODB network insures compatibility with ATM ground-based communications networks and ATM compatible test equipment. However, since sensor data are not normally in ATM format, the FBIU send data port accepts continuous, unformatted data and builds an ATM Cell for transfer on the SFODB network. The FBIU receive data port can either output data in ATM format or strip the ATM Cell header and output the data as continuous, unformatted data.

4.1.3.3 Management Layer characteristics

The SFODB ML services require the participation of both the Control Host and Data Hosts. The primary element of control resides with the Control Host. The Data Host is limited to FBIU redundant element selection.

The CFBIU serves as the control and status interface between the Control Host and the SFODB network. The CFBIU provides the services required for network configuration, network synchronization, and status reporting. These services are controlled and managed by the Control Host and executed by the CFBIU.

The SFODB Network Layer functions are considered part of the ML services. Network bandwidth allocation and data routing services are established by the CFBIU based on configuration commands from the Control Host. In other words, the allocation of network bandwidth to specific FBIUs and the data routing map that defines the transfer of data between FBIUs is established by the Control Host using the FBIU configuration tables. These tables are generated by the Control Host, downloaded to the CFBIU, and distributed by the CFBIU to the FBIUs. All network connections are established and broken using this methodology.

The following provide a simplified overview of the SFODB network control services by briefly describing a power-up initialization sequence.

- a) **Redundant element selection:** Prior to power up, each Control Host and Data Host selects either the primary, cross-strap, or bypass transmitter and receiver (Tx and Rx) pair. As spacecraft power is applied to each Control Host and Data Host the host supplies power to one of the redundant CFBIU or FBIU elements.
- b) **Power up and synchronization:** Upon application of power the CFBIU automatically initiates the network synchronization sequence by the continuous transmission of a Synchronization Frame. As each FBIU in the ring establishes bit and frame synchronization, the FBIU allows the Synchronization Frame to pass through to the next FBIU. The receipt of the Synchronization Frame by the CFBIU signifies that network synchronization is complete. All FBIUs have established bit and frame synchronization and all FBIUs are online and ready to receive configuration commands from the CFBIU. If synchronization is lost during normal operation, the CFBIU automatically initiates this process in an attempt to reestablish network synchronization.
- c) **FBIU configuration:** Once SFODB network synchronization is complete, the CFBIU sets flags indicating to the Control Host that it is ready to accept the FBIU bandwidth allocation and data routing tables. The CFBIU is capable of accepting these tables as individual commands from the Control Host or automatically sequencing through these commands in direct memory access (DMA) mode.

As the CFBIU receives the FBIU configuration tables it transfers the data to the appropriate FBIU using the SFODB Subframe Overhead built into the network's 32-Slot frame. Each time a configuration command is sent to an FBIU, the CFBIU automatically polls the FBIU to verify that the command was received and implemented correctly.

- d) **Normal mode:** Once network initialization and the configuration of all active FBIUs are completed, the CFBIU notifies all FBIUs that data transfer may commence. During this normal data transfer mode, the CFBIU monitors the operation of the network and generates statistical network performance reports. While in this mode the configuration tables for any FBIU can be modified. This allows the Control Host to dynamically reconfigure the SFODB network at any time.

4.2 Operation

4.2.1 Control Host control interface

This subclause defines the interface between the CFBIU and the Control Host. The operation of each functional interface signal listed is defined in detail in Clause 7.

4.2.1.1 Discrete command and status interfaces

The discrete command and status interfaces listed in Table 4.1 allow the Control Host to select the CFBIU configuration, control the SFODB data rate, and execute selected commands.

- a) **Control Host_Bit Reference Clock**
Used by the CFBIU as the Tx Bit Clock and therefore sets the Symbol bit rate of the network.

— Frequency	292 Mbps to 1.458 Gbps (1.458 times the desired node-to-node data throughput rate)
— Initial stability	± 10 ppm
— Temperature stability	± 50 ppm
— Harmonics and subharmonics	−20 dBc
— Phase noise	−90 dBc/Hz

Table 4.1—Discrete command and status interface

CFBIU discrete command and status interfaces	I/O	Bit width
Control Host_Bit Reference Clock	I	1
Control Host_Host Tx/Rx Control	I	1
Control Host_Tx Enable [2:0]	I	3
Control Host_Rx Enable [2:0]	I	3
ML_Rx Enable & Monitor_Rx Select Status [2:0]	I	1
Control Host_Reset	O	3
ML_Cmd Execution & Status Reporting_Control Host Interrupt	O	1
ML_Failure Reporting_CFBIU Fault	O	1
ML_Control Host Interface Mgt_Control Host Interrupt (optional)	O	1

- b) **Control Host_Host Tx/Rx Control**
Defines whether the Control Host discrete commands or the contents of CCSR-0 will control the selection of the optical transmitters and receivers.
- c) **Control Host_Tx Enable [2:0]**
Controls the selection of the optical transmitter, when **Control Host_Host Tx/Rx Control** is TRUE (see Table 6.3).
- d) **Control Host_Rx Enable [2:0]**
Controls the selection of the optical receiver, when **Control Host_Host Tx/Rx Control** is TRUE (see Table 6.4).
- e) **ML_Rx Enable & Monitor_Rx Select Status [2:0]**
Indicates which of the optical receivers is selected.
- f) **Control Host_Reset**
Resets the CFBIU internal processes and causes the CFBIU to initiate the network Synchronization process.
- g) **ML_Cmd Execution & Status Reporting_Control Host Interrupt**
Request to the Control Host to read the contents of CCIR-(XX06–XX09) in response to the detection of an Error Frame, the detection of an FBIU Poll for Status Request, or a failed command execution.
 - In the case of an Error Frame, CCIR-(XX06–XX09) will contain the Error Frame Response defined in 4.2.1.8.
 - In the case of an FBIU Poll for Status Request, CCIR-(XX06–XX09) will contain the FBIU Address and the contents of FCSR-19 from the requesting FBIU.
 - In the case of a failed command execution, CCIR-(XX06–XX09) will contain one of the Command Failed Responses defined in 4.2.1.4.

- h) **ML_Failure Reporting_CFBIU Fault**
Indicates the CFBIU has loss Frame synchronization, a CCSR parity error has been detected or one of the CFBIU error counters has reached its maximum value. The Control Host must poll the contents of CCSR-11 to determine the cause of the CFBIU Fault indication.
- i) **ML_Control Host Interface Mgt_Control Host Interrupt** (optional)
The implementation of this discrete interface signal is optional and requires the incorporation of an additional CCIR register, **CCIR-XX0A**. The purpose of this signal is to provide the Control Host with a single interrupt signal. It can be used in place of **ML_Cmd Execution & Status Reporting_Control Host Interrupt**, **ML_Failure Reporting_CFBIU Fault**, and **ML_Control Host Interface Mgt_Init Failed**. If implemented, this signal will represent as a request to the Control Host to read the contents of **CCIR-XX0A**. **CCIR-XX0A** will define the source of the interrupt and determine any further required action by the Control Host.

4.2.1.2 Register interfaces

The CCIR allows the transfer of SFODB commands and status between the Control Host and the CFBIU. The CFBIU supports two types of command execution processes, Single Command execution and Block Load Command execution. The Single Command execution process allows the Control Host to issue a command by writing into CCIR-(0–5) and status the results of the command execution by reading CCIR-(6–9) (see Figure 4.6). The Block Load Command execution process allows the Control Host to format a series of commands into a predefined block of Control Host memory and direct the CFBIU to execute these commands sequentially using a DMA operation. The status of the command executions are written sequentially into another predefined block of Control Host memory by the CFBIU. The CCIR interfaces are listed in Table 4.2. The set of valid CFBIU Commands are defined in 4.2.1.4 and 4.2.1.5.

Table 4.2—CCIR interfaces

Control Host CCIR Interface	I/O	Bit width
Control Host_Ring Init	I	1
Control Host__Read Strobe	I	1
Control Host__Write Strobe	I	1
Control Host__Address	I/O	16
Control Host__Command/Status Word	I/O	16
ML_Control Host Interface Mgt_Command Busy Indicator	O	1
ML_Control Host Interface Mgt_Init Failed	O	1

- a) **Control Host__Ring Init**
Causes the CFBIU to execute an Initialize Command (see 4.2.1.7).
- b) **Control Host__Read Strobe**
Causes the CFBIU to place the contents of the CCIR corresponding to the **Control Host__Address** on the **Control Host__Command/Status Word**.

- c) **Control Host__Write Strobe**
Causes the CFBIU to write the contents of the **Control Host__Command/Status Word** into the CCIR corresponding to the **Control Host__Address**.
- d) **Control Host__Address**
 - The address (CCIR number 0 through 9) of the CCIR when the Control Host initiates the register access.
 - The address pointer to a Control Host memory location when the CFBIU is executing a Block Load Command in DMA mode.
- e) **Control Host__Command/Status Word**
 - The contents of the CCIR corresponding to the **Control Host__Address** when the Control Host is executing a CCIR Read operation.
 - A command word from the Control Host when the Control Host is executing a Single Command or the CFBIU is executing a Block Load Command or an Initialize Command.
 - A command response word from the CFBIU when the CFBIU is writing into a Control Host memory location during a Block Load Command execution.
- f) **ML_Control Host Interface Mgt__Command Busy Indicator**
Indicates to the Control Host that the CFBIU is in the process of executing a command.
- g) **ML_Control Host Interface Mgt__Init Failed**
Indicates the Initialize Command sequence has failed. The Control Host can either poll the contents of CCIR-(XX06–XX09) or read the last set of entries in the Control Host memory block allocated to Configuration Command Results to determine the cause of the Init Failed indication.

4.2.1.3 CCIR structure & definition

A set of ten (10), 16-bit CCIRs are used by the Control Host to issue configuration commands and status the command execution results (see Figure 4.6). Since there are only ten (10) CCIRs, the CFBIU uses the least significant byte of the CCIR Address to address the registers, the most significant byte is ignored. The “XX” in the CCIR Addresses represents a “Don’t Care” for the most significant byte of the address. The content and use of each CCIR is defined as follows:

CCIR-XX00—Command

Used to store the command word that selects the command or status operations to be performed by the CFBIU.

CCIR-XX01—CFBIU/FBIU Address & CCSR/FCSR Address

Used to store the CFBIU/FBIU Address and CCSR/FCSR Address, if the command is a Load or Poll Command. If the command is not a Load or Poll Command, the contents of the register is set to all zeros.

CCIR-XX02—Command Data Word 1

Stores the most significant 16 bits of the CCSR/FCSR configuration word, if the command is a Load Command.

CCIR-XX03—Command Data Word 2

Stores the least significant 16 bits of the CCSR/FCSR configuration word, if the command is a Load Command.

NOTE—A write operation to this register shall initiate the CFBIU command execution process.

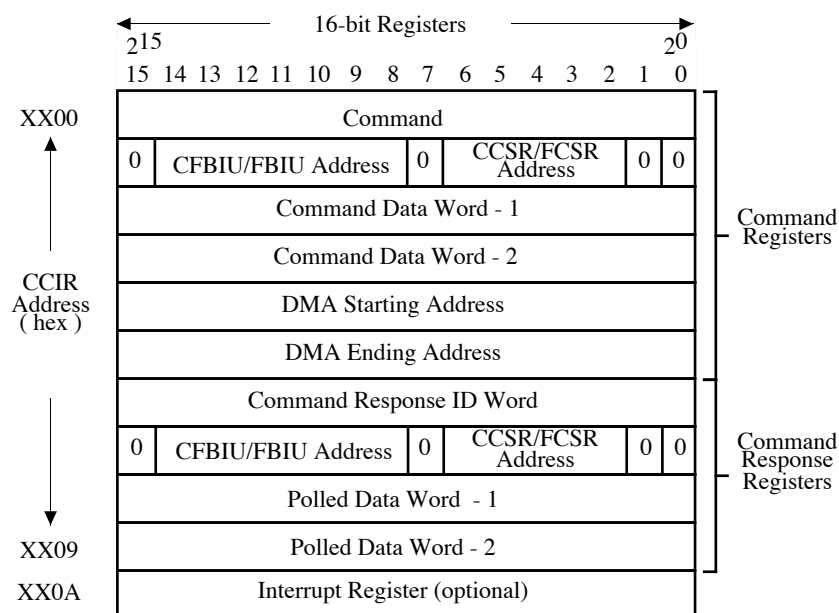


Figure 4.6—CCIR structure

CCIR-XX04—DMA Starting Address

Contains the starting address of a command block in Control Host memory. The most significant bit of this 16-bit data field shall be set to “0” and the 2-least significant bits of the 16-bit data field shall be set to “00.”

The CFBIU command execution process uses the most significant bit to select between the two Control Host memory blocks allocated to Configuration Commands and Configuration Command Responses. The 2-least significant bits are used internally by the CFBIU to sequence through the four command words required for each CCSR/FCSR load and poll operation.

CCIR-XX05—DMA Ending Address

Contains the last address of the command block in Control Host memory. The most significant bit of this 16-bit data field shall be set to “0” and the 2-least significant bits of the 16-bit data field shall be set to “11” for the same reasons stated above for the DMA Starting Address.

CCIR-XX06—Command Response ID Word

The Command Verification Status register is used by the CFBIU to report the status of the last executed command to the Control Host.

CCIR-XX07—CFBIU/FBIU Address & CCSR/FCSR Address

The CFBIU writes the polled CFBIU/FBIU Address and CCSR/FCSR Address into this register.

CCIR-XX08—Polled Data Word 1

The CFBIU writes the most significant 16 bits of data from the polled CCSR/FCSR into this register.

CCIR-XX09—Polled Data Word 2

The CFBIU writes the least significant 16 bits of data from the polled CCSR/FCSR into this register.

NOTE—The CFBIU notifies the Control Host when the contents of CCIR-(XX06–XX09) are valid by setting the ML_Control Host Interface Process_Command Busy Indicator to False.

CCIR-XX0A—Interrupt Register (optional)

The implementation of this register is optional. If this register is implemented, it requires the implementation of the discrete ML_Control Host Interface Mgt_Control Host Interrupt, which will be set to True each time the CFBIU writes to the register and will be set to False each time the Control Host reads this register.

Register Field Definitions:

Bits 15–7 are reserved.

Bit 6 CCSR Parity Error indicator. Indicates that a parity error has been detected in one of the CCSRs. Bits 31–27 of the CFBIU Status Register, CCSR-11, will contain the address of the CCSR register that caused the parity error.

<u>Bit 6</u>	<u>CCSR Parity Error</u>
1	TRUE
0	FALSE

Bit 5 Init Failed indicator. Indicates that a Command Execution has failed during the Initialization Command sequence. The Control Host should either poll the contents of CCIR-(XX06–XX09) or read the last set of entries in the Control Host memory block allocated to Configuration Command Results to determine which command failed to execute.

<u>Bit 5</u>	<u>Init Failed</u>
1	TRUE
0	FALSE

Bit 4 CFBIU Sync Loss indicator. Indicates that the CFBIU has lost Frame Sync and has set the Frame Type to SYNCHRONIZATION in an attempt to reestablish ring synchronization.

<u>Bit 4</u>	<u>CFBIU Sync Loss</u>
1	TRUE
0	FALSE

Bit 3 Error Counter Max indicator. Indicates that one of the CFBIU Error Counters has reached its maximum value. The Control Host should issue a Poll Command to CCSR-11 to determine which of the Error Counters is at its maximum value.

<u>Bit 3</u>	<u>Error Counter Max</u>
1	TRUE
0	FALSE

Bit 2 FBIU Poll Requested indicator. Indicates that an FBIU has set the **Rx Subframe_Poll for Status Indicator** field to True requesting the CFBIU poll the FBIU's Status Register. The CFBIU will automatically poll the FBIU Status Register and place results in CCIR-(XX06–XX09).

<u>Bit 2</u>	<u>FBIU Poll Requested</u>
1	TRUE
0	FALSE

Bit 1 Frame Type = Synchronization indicator. Indicates that the CFBIU is attempting to establish Frame Synchronization and is therefore incapable of executing configuration commands.

<u>Bit 1</u>	<u>Frame Type = Synchronization</u>
1	TRUE
0	FALSE

Bit 0 Frame Type = Error indicator. Indicates that the CFBIU has received an Error Frame Response from an FBIU indicating that the FBIU has lost Frame Synchronization. The Control Host should read CCIR- (XX06–XX09) to determine which FBIU was the source of the Error Frame Response.

<u>Bit 0</u>	<u>Frame Type = Error</u>
1	TRUE
0	FALSE

4.2.1.4 CFBIU Single Command Definitions

The CFBIU will, at a minimum, execute the following Single Commands:

- Load Command: Used to Load a CFBIU CCSR or an FBIU FCSR (see Figure 4.7).
- Poll Command: Used to Poll the contents of a CFBIU CCSR or an FBIU FCSR (see Figure 4.8).

The CFBIU will initiate a Single Command when the Control Host writes a valid Single Command into CCIR–XX00 through CCIR–XX03. A Command Busy Indicator set to True indicates to the Control Host that the CFBIU is executing the command.

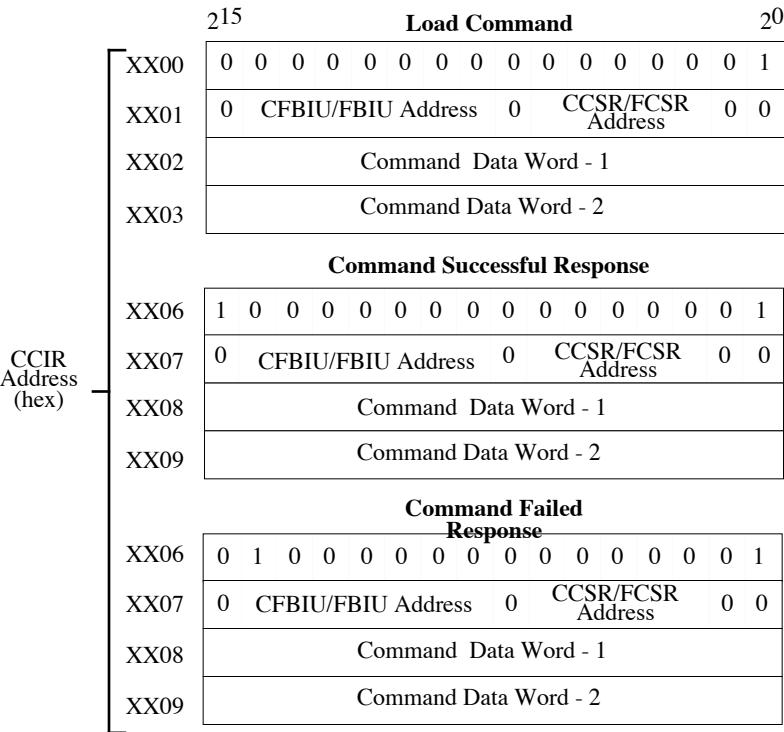


Figure 4.7—Load Command and valid responses

		215	Poll Command																20
CCIR Address (hex)	XX00		0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	XX01		0	CFBIU/FBIU Address								0	CCSR/FCSR Address				0	0	
	XX02		Don't Care																
	XX03		Don't Care																
	Command Successful Response																		
	XX06		1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	XX07		0	CFBIU/FBIU Address								0	CCSR/FCSR Address				0	0	
	XX08		Polled Data Word - 1																
	XX09		Polled Data Word - 2																
	Command Failed Response																		
	XX06		0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	XX07		0	CFBIU/FBIU Address								0	CCSR/FCSR Address				0	0	
	XX08		Invalid																
	XX09		Invalid																

Figure 4.8—Poll Command and valid responses

The CFBIU will write the results of the command execution into CCIR-XX06 through CCIR-XX09. A Command Busy Indicator set to False indicates to the Control Host that the CFBIU command execution is complete and the command execution results are valid.

Upon receipt of a Load or Poll Command from the Control Host,
 Set the Cmd Busy Indicator to True.
 Execute the Command.
 Load the Command Response into CCIR-(XX06–XX09).
 Set the Cmd Busy Indicator to False, notifying the Control Host
 that the command status information in CCIR-(XX06–XX09) is valid.

4.2.1.5 CFBIU Block Command definitions

Block Commands allow the Control Host to load a set of sequential Configuration Commands into a pre-defined block of Control Host memory and direct the CFBIU to execute these commands using a DMA process. The Configuration Command Response for each command executed during the Block Command process is written sequentially to another predefined block of Control Host memory by the CFBIU.

The CFBIU will, at a minimum, execute the following Block Commands:

- Block Load Command: Used to Load a group of CFBIU CCSRs and/or FBIU FCSRs defined by the CCIR-XX04 Starting Address and CCIR-XX05 Ending Address pointers (see Figure 4.9).
- Initialize Command: Used to Load all CFBIU CCSRs and FBIU FCSRs starting with a Control Host memory address pointer of “0000” and ending with the execution of the Stop Command (see Figure 4.10).
- Stop Command: Used to end the Initialize Command process (see Figure 4.11).

Block Load Command	
CCIR Address (hex)	
XX00	0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
XX01	0 Zeros 0 Zeros 0 0
XX02	Don't Care
XX03	Don't Care

Figure 4.9—Block Load Command

Initialize Command	
CCIR Address (hex)	
XX00	2^{15} 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 2^0
XX01	0 Zeros 0 Zeros 0 0
XX02	Don't Care
XX03	Don't Care

Figure 4.10—Initialize Command

Stop Command	
CCIR Address (hex)	
XX00	2^{15} 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 2^0
XX01	0 Zeros 0 Zeros 0 0
XX02	Don't Care
XX03	Don't Care

Figure 4.11—Stop Command

There are three methods that the Control Host can use to initiate a Block Command process.

Method - 1 Issue a Block Load Command

- Load a set of CCSR/FCSR commands into a contiguous block of Control Host memory.
- Write the Starting Address of this memory block into CCIR-XX04.
- Write the Ending Address of this memory block into CCIR-XX05.
- Write a Block Load Command into CCIR-XX00 through CCIR-XX03 (see Figure 4.9).

Method - 2 Issue an Initialize Command

- Load a set of CCSR/FCSR commands into a contiguous block of Control Host memory starting at 0000-hex.
- Write a Stop Command (see Figure 4.11) as the last command of this contiguous block.
- Write an Initialize Command into CCIR-XX00 through CCIR-XX03 (see Figure 4.10).

Method - 3 Issue a Ring Init Command

- Load a set of CCSR/FCSR commands into a contiguous block of Control Host memory starting at address 0000–hex.
- Write a Stop Command (see Figure 4.11) as the last command of this contiguous block.
- Assert Control Host_Ring Init.

Once the Block Command execution process has been initiated, the CFBIU will sequence through each group of four command words stored in the Control Host memory block allocated to Configuration Commands and execute each command. The CFBIU will write the Configuration Command Responses from each Configuration Command executed during this process into the Control Host memory block allocated to Configuration Command Responses.

The CFBIU uses the msb of the address pointer to select between the Control Host memory block allocated to Configuration Commands and the Control Host memory block allocated to Configuration Command Responses. That is, the CFBIU uses an address pointer msb of “0” when reading each Configuration Command and an address pointer msb of “1” when writing the Configuration Command Responses of each command. The CFBIU uses the same 15 lsbs to both read the Configuration Commands and write the Configuration Command Responses creating a one-to-one map between Configuration Commands and Configuration Command Responses within the allocated Control Host memory.

If a command fails to execute at any point during the Block Command execution process, indicated by the assertion of the ML_Cmd Execution & Status Reporting_Execute Failed, the Control Host Interface Mgt process will set the Init Failed to true and halt the process. When the Init Failed is set to true, CCIR-(XX06–XX09) contain the Command Failed Response of the command that failed to execute.

4.2.1.6 Block Load Command execution

- Upon receipt of a Block Load Command from the Control Host (see Figure 4.9),
- Set the Cmd Busy Indicator to True.
- Initiate a DMA Read from the Control Host memory block allocated to Configuration Commands starting with the DMA Starting Address in CCIR-XX04.
- Write the first four (4) entries from the Control Host memory into CCIR-(XX00–XX03).
- Execute the Command.

- When the command execution is complete,
- Transfer the contents of Status Registers CCIR-(XX06–XX09) into the Control Host memory block allocated to Configuration Command Responses using a DMA Write process.

- Continue the above four (4) word command execution and status reporting processes until the DMA Ending Address in CCIR-XX05 is reached
- Or a command fails to execute.

- If a command fails to execute,
- Then,
 - Stop the Block Load Command process.
 - Set the Init Failed indicator to True.
 - Set the Cmd Busy Indicator to False.
- Else,
 - When the DMA Ending Address in CCIR-XX05 is reached,
 - Then,
 - Execute the last command.
 - Set the Cmd Busy Indicator to False.

4.2.1.7 Initialize Command and Ring Init execution

Upon receipt of an Initialize Command from the Control Host (see Figure 4.10)
or the Assertion of the Control Host_Ring Init discrete,
Set the Cmd Busy Indicator to True.
Initiate a DMA Read from the Control Host memory block allocated to
Configuration Commands starting with the DMA Starting Address equal to “0000-hex.”
Write the first four (4) entries from the Control Host memory into CCIR-(XX00–XX03)
and Assert the Execute Command indicator.

When the command execution is complete,
Transfer the contents of Status Registers CCIR-(XX06–XX09) into the Control Host memory block
allocated to Configuration Command Responses using a DMA Write process.

Continue the above four (4) word command execution and status reporting processes
Until a Stop Command is read from the Control Host memory,
Or a command fails to execute.

If a command fails to execute,
Then,
Stop the Block Load Command process.
Set the Init Failed indicator to True.
Set the Cmd Busy Indicator to False.
Else,
When a Stop Command is read from the Control Host memory,
Then,
Stop the Block Load Command process.
Set the Cmd Busy Indicator to False.

4.2.1.8 Error frame report

The CFBIU will notify the Control Host when an Error Frame has been received from one of the FBIUs by asserting the ML_Cmd Execution & Status Reporting_Control Host Interrupt. The CFBIU/FBIU Address field of CCIR-7 will contain the Address of the FBIU that is generating the Error Frame (see Figure 4.12). An FBIU will generate an Error Frame when it is unable to acquire Frame Sync on the incoming serial Symbol stream.

		2 ¹⁵	Error Frame Response												2 ⁰	
CCIR Address (hex)	XX06	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	XX07	0	CFBIU/FBIU Address								0	Invalid		0	0	
	XX08	Invalid														
	XX09	Invalid														

Figure 4.12—Error Frame response

4.2.1.9 Poll for Status Request report

The CFBIU will notify the Control Host when an FBIU has requested a Poll for Status indicating a problem within the FBIU or a Host Service Request by asserting the ML_Cmd Execution & Status

Reporting_Control Host Interrupt. CCIR-(XX06–XX09) will contain a Poll Command Successful Response as defined in Figure 4.8. The CFBIU/FBIU Address field of CCIR-XX07 will contain the Address of the FBIU that generated the Poll for Status and CCIR-(XX08–XX09) will contain the contents of the FBIU Status Register (FCSR-19).

4.2.1.10 CFBIU Sync Loss notification

The CFBIU will notify the Control Host when the CFBIU has lost Frame synchronization by asserting the ML_Cmd Execution & Status Reporting_Control Host Interrupt. The Rx Frame Type Status field of the CFBIU status register, CCSR-11_Rx Frame Type Status, will indicate that the SFODB is no longer in Normal mode (see 7.3.4.4.6). The Control Host should poll all CCSR and FCSR configuration registers to verify the SFODB configuration; reconfigure the SFODB ring, if required; and then, set the CFBIU Frame Type to NORMAL by executing a Load Command to CCSR-0.

4.2.2 Data Host interfaces

This subclause defines the control interface between the FBIU and the Data Host. The operation of each functional interface signal listed is defined in detail in Clause 6.

4.2.2.1 Control interfaces

The discrete command and status interfaces listed in Table 4.3 allow the Data Host to set the FBIU's Address, select and status the FBIU Tx and Rx configuration and request a predefined service from the Control Host.

Table 4.3—Discrete command and status interfaces

Discrete command and status interfaces	I/O	Bit width
Data Host_Host Tx/Rx Control	I	1
Data Host_Tx Select [2:0]	I	3
Data Host_Rx Select [2:0]	I	3
ML_Rx Enable & Monitor_Rx Select Status [2:0]	O	3
Data Host_Host Service Request	I	1
Data Host_FBIU Address	I	7

- a) **Data Host_Host Tx/Rx Control**
Defines whether the Data Host discrete commands or the contents of FCSR-8 will control the selection of the optical transmitters and receivers.
- b) **Data Host_Tx Select [2:0]**
Controls the selection of the optical transmitter, if Data Host_Host Tx/Rx Control is True (see Table 6.3).

- c) **Data Host_Rx Select [2:0]**
Controls the selection of the optical receiver, if Data Host_Host Tx/Rx Control is True (see Table 6.4).
- d) **ML_Rx Enable & Monitor_Rx Select Status [2:0]**
Indicates which of the optical receivers is selected.
- e) **Data Host_Host Service Request**
Used by the Data Host to request any predefined service from the Control Host.
- f) **Data Host_FBIU Address**
Used by the Data Host to set the FBIU's node Address (0–127). Each FBIU in the network must have a unique FBIU Address in order to communicate properly with the CFBIU.

4.2.2.2 Data interfaces

The Data Host data interfaces are shown in Table 4.4.

FBIU-to-Host FIFO Interfaces

- a) **Data Host_FIFO Clear**
Used by the Data Host to Clear the FBIU-to-Host FIFO.
- b) **FBIU-to-Host FIFO_Transfer Enable**
Used by the FBIU to control the FBIU-to-Host FIFO Read access.
- c) **FBIU-to-Host FIFO_FIFO Output Data**
The FBIU-to-Host FIFO output data port. The output data format options are illustrated in Figure 4.13. Output data format selection is controlled by FCSR-8_Host-to-FBIU ATM/Raw Data Select.

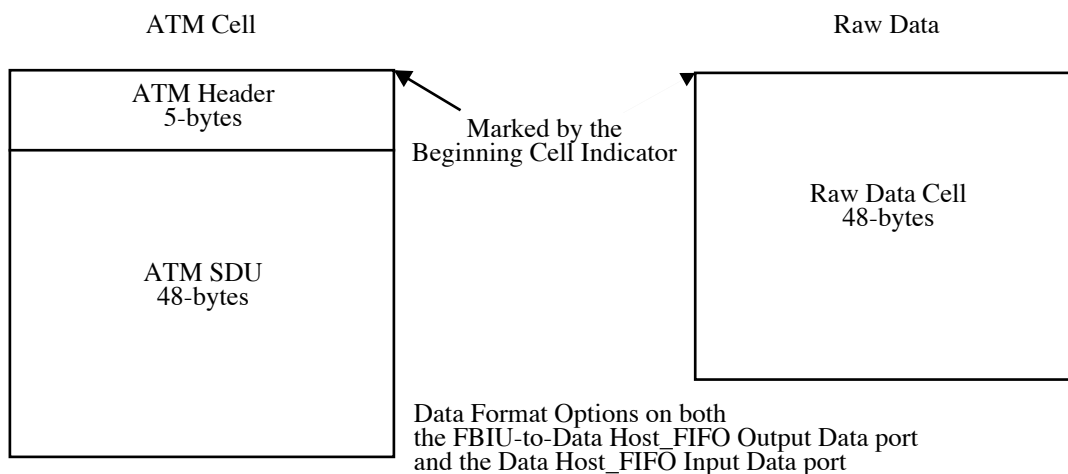


Figure 4.13—Data Host data format options

- d) **FBIU-to-Host FIFO_Beginning Cell Indicator**
Used by the FBIU to indicate the Beginning of a 53-byte ATM Cell or 48-byte Raw Data Cell (see Figure 4.13).
- e) **FBIU-to-Host FIFO_FIFO Full**
Used by the FBIU to indicate that the FBIU-to-Host FIFO is full.

Table 4.4—Data Host data interfaces

Data Host data interfaces	I/O	Bit width
FBIU-to-Host FIFO interfaces		
Data Host_FIFO Clear	I	1
FBIU-to-Host FIFO_FIFO Output Data	O	32
FBIU-to-Host FIFO_Beginning Cell Indicator	O	1
FBIU-to-Host FIFO_Transfer Enable	O	1
FBIU-to-Host FIFO_FIFO Full	O	1
Host-to-FBIU FIFO interfaces		
Data Host_FIFO Clear	I	1
Data Host_FIFO Input Data	I	32
Data Host_Beginning Cell Indicator	I	1
Host-to-FBIU FIFO_Transfer Enable	O	1
Host-to-FBIU FIFO_FIFO Full	O	1

Host-to-FBIU FIFO Interfaces

- a) **Data Host_FIFO Clear**
Used by the Data Host to Clear the Host-to-FBIU FIFO.
- b) **Host-to-FBIU FIFO_Transfer Enable**
Used by the FBIU to control the Host-to-FBIU FIFO Write access.
- c) **Data Host_FIFO Input Data**
The Host-to-FBIU FIFO input data port. The input data format options are illustrated in Figure 4.13. Input data format selection is controlled by FCSR-8_Host-to-FBIU ATM/Raw Data Select.
- d) **Data Host_Beginning Cell Indicator**
Used by the Data Host to indicate the Beginning of a 53-byte ATM Cell or 48-byte Raw Data Cell (see Figure 4.13).
- e) **Host-to-FBIU FIFO_FIFO Full**
Used by the FBIU to indicate that the Host-to-FBIU FIFO is full.

4.2.3 SFODB Frame Structure

4.2.3.1 Frame, Subframe, Slot, and Symbol definitions

Figure 4.14 illustrates the SFODB Frame, Subframe, and Slot structure.

- a) **Frame:** A Frame is defined as 32 contiguous Slots. A Frame Sync Symbol defines the beginning of each SFODB Frame.
- b) **Subframe:** A Subframe is defined as 4 contiguous Slots. There are eight (8) Subframes in each Frame.
- c) **Slot:** A Slot is defined as 56-Symbols. Each Slot contains 3-Symbols of encoded Subframe overhead (OH) and a 53-Symbol encoded ATM Cell.
- d) **Symbol:** A Symbol is an 8B/10B encoded byte, as defined in ANSI X3.230-1994, with Running Disparity implemented.
- e) **Frame Sync Symbol:** Two 8B/10B Control Codes (K Codes) are used as Frame Sync Symbols, K28.5 and K28.7 (see the definition of Subframe Byte-0 for Subframe “0” in 4.2.3.2).

4.2.3.2 Subframe OH field definitions

The Subframe OH used to transfer Configuration and Status information between the CFBIU and the FBIUs is a 12-byte field composed of 3 bytes of overhead from four consecutive SFODB frame Slots (see Figure 4.14). The Rx Subframe OH and Tx Subframe OH are identical in format and content. The Rx and Tx distinguish between the Subframe OH received by the CFBIU/FBIU and the Subframe OH transmitted by the CFBIU/FBIU respectively. There are 8 Subframes within each 32-Slot SFODB Frame. The Overhead bytes that make up each Subframe are defined as follows:

Subframe-0	Overhead bytes 0 through 2	from Slots 0 through 3
Subframe-1	Overhead bytes 0 through 2	from Slots 4 through 7
Subframe-2	Overhead bytes 0 through 2	from Slots 8 through 11
Subframe-3	Overhead bytes 0 through 2	from Slots 12 through 15
Subframe-4	Overhead bytes 0 through 2	from Slots 16 through 19
Subframe-5	Overhead bytes 0 through 2	from Slots 20 through 23
Subframe-6	Overhead bytes 0 through 2	from Slots 24 through 27
Subframe-7	Overhead bytes 0 through 2	from Slots 28 through 31

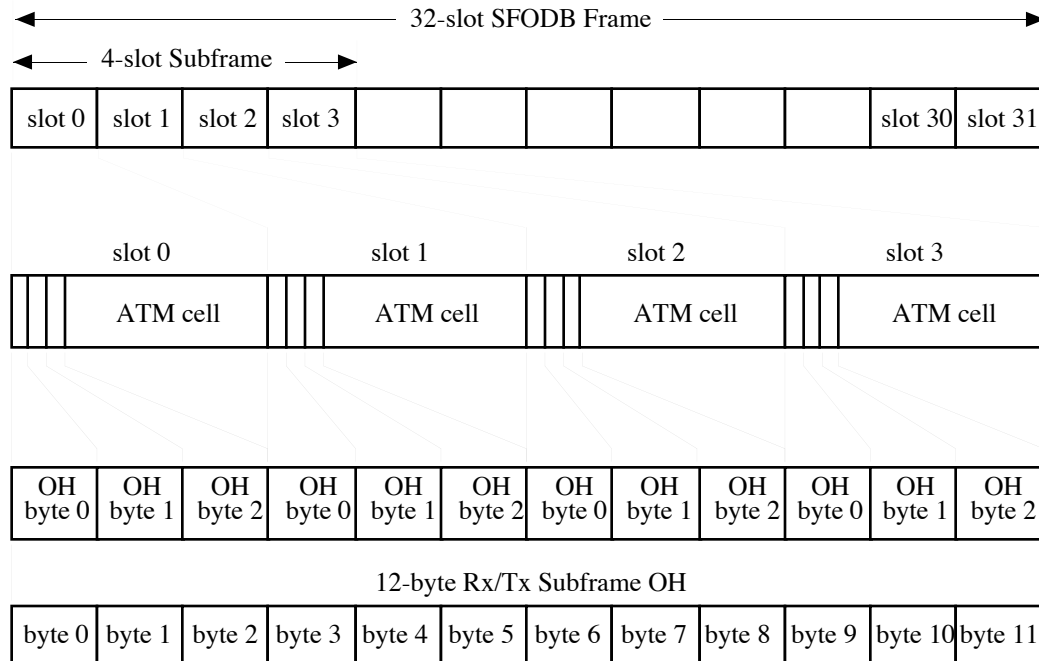


Figure 4.14—Subframe structure

See Figure 4.15 and the following field definitions for a detailed definition of each Subframe OH field.

Subframe Byte-0 for Subframe “0”

Byte-0 of Subframe “0” contains the K28.5 or K28.7 Frame Start Delimiter. This unique 8B/10B command code occurs once at the start of each frame and represents Subframe Count “000.” The decoded K28.5 and K28.7 command codes maintain the integrity of the Free and Claim Token bit respectively.

Bit #:	<u>7</u>	<u>6</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	1	0	1	1	1	1	0	0	K28.5 code
	1	1	1	1	1	1	0	0	K28.7 code

Subframe Byte-0 for Subframes “1–7”

Byte-0 of Subframes “1–7” contains four static bits, the first bit of the majority vote Token field associated with the frame Slot and the Subframe Count.

Bit 7 is a static bit and is set to “0.”

Bit 6 is the first bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 5–3 are static bits and are set to “000.”

Bits 2–0 contain the Subframe Count.

Bit #:	<u>2</u>	<u>1</u>	<u>0</u>	
	2^2	2^1	2^0	Subframe Count

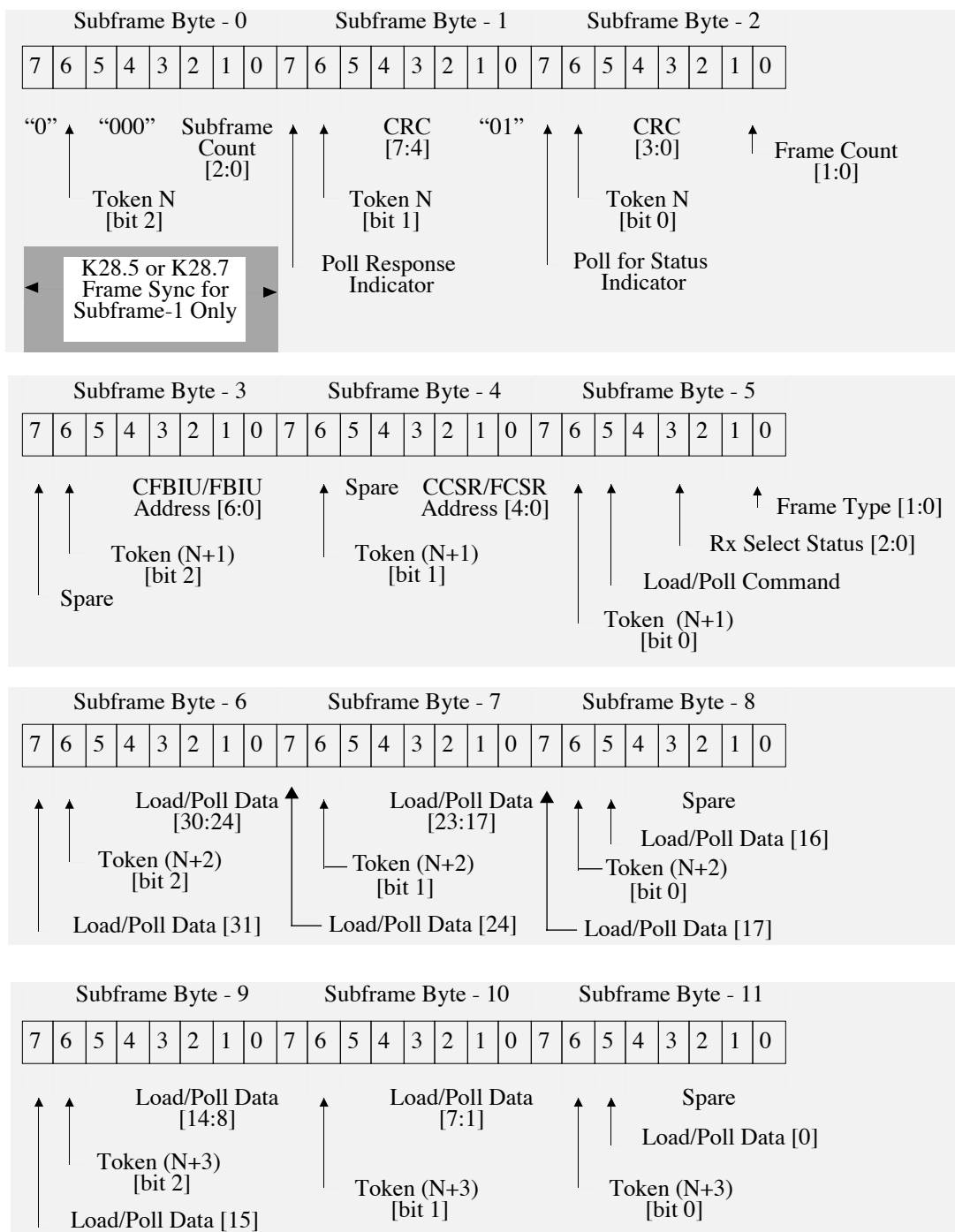


Figure 4.15—Subframe structure

Subframe Byte-1

Byte-1 of all Subframes contains the Poll Response Indicator, the second bit of the majority vote Token field associated with the frame Slot, the 4 msbs of the cyclic redundancy check (CRC) for the previous subframe and the two static bits.

Bit 7 is the Poll Response Indicator.

<u>Bit 7</u>	
0	FALSE
1	TRUE

Bit 6 is the second bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 5–2 contain the 4 msbs of the CRC for Subframe (N-1).

Bit #:	$\frac{5}{2^7}$	$\frac{4}{2^6}$	$\frac{3}{2^5}$	$\frac{2}{2^4}$	4 msbs of CRC for Subframe (N-1)
--------	-----------------	-----------------	-----------------	-----------------	----------------------------------

Bits 1–0 are static and set to “01.”

Subframe Byte-2

Byte-2 of all Subframes contains the Poll for Status Indicator, the third bit of the majority vote Token field associated with the frame Slot, the 4 lsbs of the CRC for the previous subframe and the Frame Count.

Bit 7 is the Poll for Status Indicator.

<u>Bit 7</u>	
0	FALSE
1	TRUE

Bit 6 is the third bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 5–2 contain the 4 lsbs of the CRC for Subframe (N-1).

Bit #:	$\frac{5}{2^3}$	$\frac{4}{2^2}$	$\frac{3}{2^1}$	$\frac{2}{2^0}$	4 lsbs of CRC for Subframe (N-1)
--------	-----------------	-----------------	-----------------	-----------------	----------------------------------

Bits 1–0 contain the Frame Count.

Bit #:	$\frac{1}{2^1}$	$\frac{0}{2^0}$	Frame Count
--------	-----------------	-----------------	-------------

Subframe Byte-3

Byte-3 of all Subframes contains one spare bit, the first bit of the majority vote Token field associated with the frame Slot and the 6 msbs of the FBIU Address.

Bit 7 is a spare bit and is preset to a Logic “0.”

Bit 6 is the first bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 5–0 contain the 6 msbs of the FBIU Address.

Bit #:	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^6	2^5	2^4	2^3	2^2	2^1	6 msbs of the FBIU Address

Subframe Byte-4

Byte-4 of all Subframes contains the lsb of the FBIU Address, the second bit of the majority vote Token field associated with the frame Slot and the 4 msbs of the FCSR Address.

Bit 7 contains the lsb of the FBIU Address.

<u>Bit 7</u>	
2^0	lsb of the FBIU Address

Bit 6 is the second bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 5–4 are spare bits and preset to “00.”

Bits 3–0 contain the 4 msbs of the FCSR Address.

Bit #:	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^4	2^3	2^2	2^1	msbs of the FCSR Address

Subframe Byte-5

Byte-5 of all Subframes contains the lsb of the FCSR Address, the third bit of the majority vote Token field associated with the frame Slot, the Load/Poll Command field, the Rx Select Status field and the Frame Type field.

Bit 7 contains the lsb of the FCSR Address.

<u>Bit 7</u>	
2^0	lsb of the FCSR Address

Bit 6 is the third bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bit 5 is the Load/Poll Command.

<u>Bit 5</u>	
0	POLL
1	LOAD

Bits 4–2 contain the 3-bit Rx Select Status field. This field is only valid if the Frame Type indicates ERROR.

Bit #:	<u>4</u>	<u>3</u>	<u>2</u>	
	1	0	0	Rx Selected [2] - Primary
	0	1	0	Rx Selected [1] - Cross-Strap
	0	0	1	Rx Selected [0] - Bypass

Bits 1–0 contain the Frame Type.

Bit #:	<u>1</u>	<u>0</u>	
	0	0	SYNCHRONIZATION
	0	1	CONFIGURATION
	1	1	NORMAL
	1	0	ERROR

Subframe Byte-6

Byte-6 of all Subframes contains Load/Poll Data and the first bit of the majority vote Token field associated with the frame Slot.

Bits 7 and 5–0 contains the Load/Poll Data bits as defined below.

Bits #:	<u>7</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	Load/Poll Data

Bit 6 is the third bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Subframe Byte-7

Byte-7 of all Subframes contains Load/Poll Data and the second bit of the majority vote Token field associated with the frame Slot.

Bits 7 and 5–0 contains the lsb of the Load/Poll Data bits as defined below.

Bits #:	<u>7</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^{24}	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	Load/Poll Data

Bit 6 is the second bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Subframe Byte-8

Byte-8 of all Subframes contains Load/Poll Data and the third bit of the majority vote Token field associated with the frame Slot.

Bits 7 and 5 contains the lsb of the Load/Poll Data bits as defined below.

Bits #:	<u>7</u>	<u>5</u>	
	2^{17}	2^{16}	Load/Poll Data

Bit 6 is the third bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Bits 4–0 are spare bits and are preset to “00000.”

Subframe Byte-9

Byte-9 of all Subframes contains Load/Poll Data and the first bit of the majority vote Token field associated with the frame Slot.

Bits 7 and 5–0 contain the Load/Poll Data bits as defined below.

Bits #:	<u>7</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	Load/Poll Data

Bit 6 is the third bit of the majority vote Token for Slot-N.

<u>Bit 6</u>	
0	FREE
1	CLAIM

Subframe Byte-10

Byte-10 of all Subframes contains Load/Poll Data and the second bit of the majority vote Token field associated with the frame Slot.

Bits 7 and 5–0 contain the lsb of the Load/Poll Data bits as defined below.

Bits #:	<u>7</u>	<u>5</u>	<u>4</u>	<u>3</u>	<u>2</u>	<u>1</u>	<u>0</u>	
	2^8	2^7	2^6	2^5	2^4	2^3	2^2	Load/Poll Data