Spaceborne Fiber Optic Data Bus (SFODB) Operational & Interface Description

Overview

This section introduces the SFODB functional elements, maps the relevant OSI protocol layers to the SFODB functional elements and highlights the key characteristics of the SFODB. No binding requirements are specified in this subsection. SFODB protocol layers and processes are described in their entirety for each functional element in the IEEE 1393-1999 SFODB protocol standard.

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.

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

The **FBIU**, or **F**iber-optic **B**us **I**nterface **U**nit, 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**, or **C**ontrol **F**iber-optic **B**us **I**nterface **U**nit, 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.

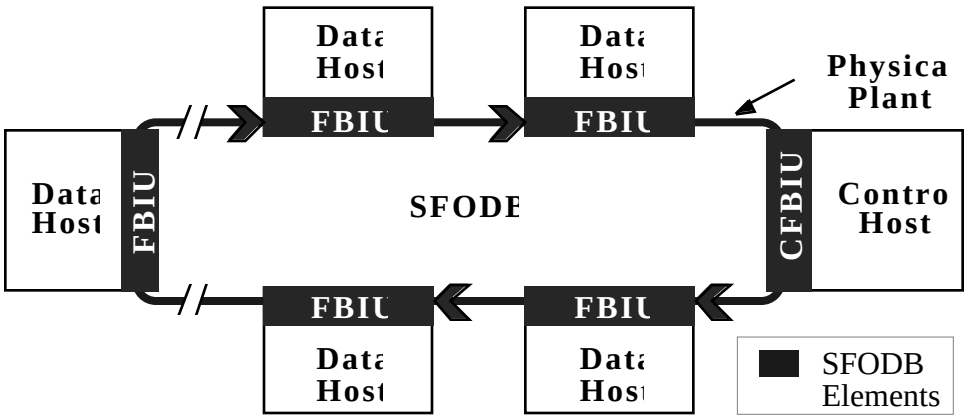

Figure 1 SFODB network topology

OSI protocol layer allocation

In an effort to standardize the protocol structure, the International Standards Organization (ISO) created the 7–layer Open System Interconnection (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 Management Layer. Figure 4-3 illustrates in a simplified form the allocation of these protocol layers to the SFODB functional element.

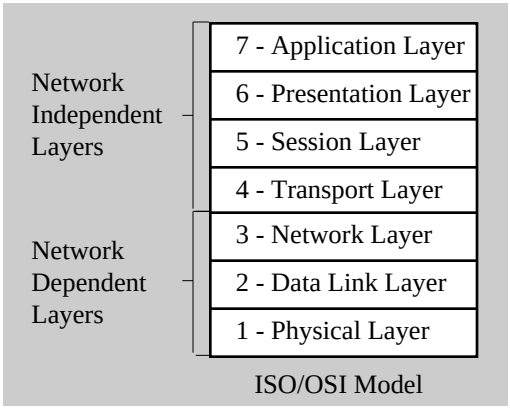

Figure 2 ISO/OSI model

a) Physical Layer - In the SFODB the Physical Layer is divided into two parts, the Media Dependent Physical Layer (MDPL) and the Non-Medial 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.

b) Data Link Layer - The Data Link Layer (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.

c) Management Layer - In the SFODB, the Network Layer functions associated with network bandwidth allocations and data routing, are considered part of the Management Layer. These non-real time Management Layer 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.

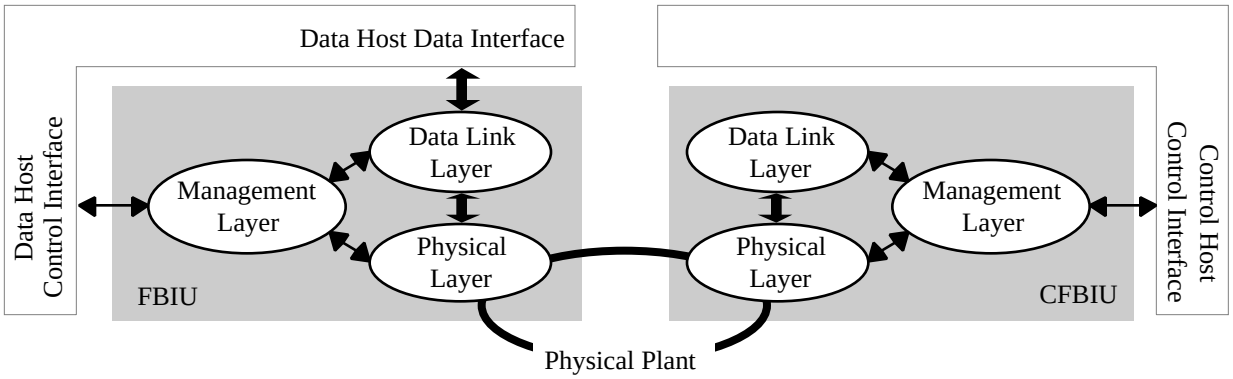

Figure 3 Protocol layer allocation

Network characteristics

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 meters.

The Media Dependent Physical Layer 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 multi-fiber cable, multi-pin connectors and specific termini are recommended for inter-operability, the user is free to select a different cable interconnect scheme.

The Non-Media Dependent Physical Layer is characterized by a continuous transmission, serial communications mode. Data is 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.

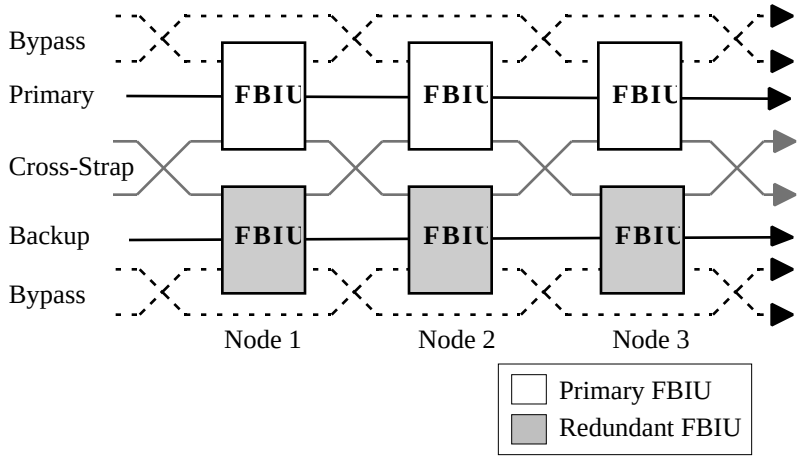

Figure 4 SFODB redundancy configuration

Data link layer characteristics

The Data Link Layer 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 services.

Transfer of data between Data Hosts on the SFODB ring is accomplished using a simple fixed length 32-Slot, 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 Latency Adjust Buffer (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.

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 subsection.

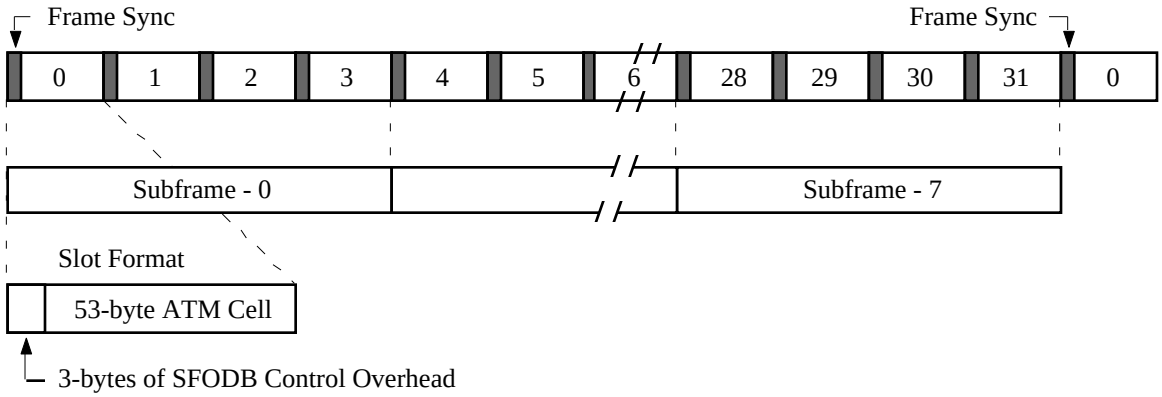

Figure 5 SFODB Frame format

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 the following items summarize the key characteristics of the SFODB Data Link Layer.

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 is 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 one byte. This is a product of the fixed-length frame format and the Latency Adjust Buffer (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 switch or packet switch 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.

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.

Receiving data The broadcast data is accepted or rejected using either a Dedicated Receive Slot process or a 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 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 is 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.

Management layer characteristics

The SFODB Management Layer 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 Management Layer 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 subsections 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 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 & 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 a 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 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 is 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 & 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.

Description of Operation

Control Host control interface

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

Discrete command & 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.

CFBIU Discrete Command 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
Control Host_Reset	I	1
ML_Rx Enable & Monitor_Rx Select Status [2:0]	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

Table 1 Discrete command & status interface

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

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 6-1.

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 6-2.

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 section4.2.1.8.
- 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 subsections 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 replace 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.

Register interfaces

The CFBIU Command Interface Registers (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 section4.2.3.

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

Table 2 CCIR interfaces

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 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 through CCIR-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.

CCIR structure & definition

A set of ten (11), 16-bit, CFBIU Command Interface Registers (CCIRs) are used by the Control Host to issue configuration commands and status the command execution results. Since there are only ten (11) 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:

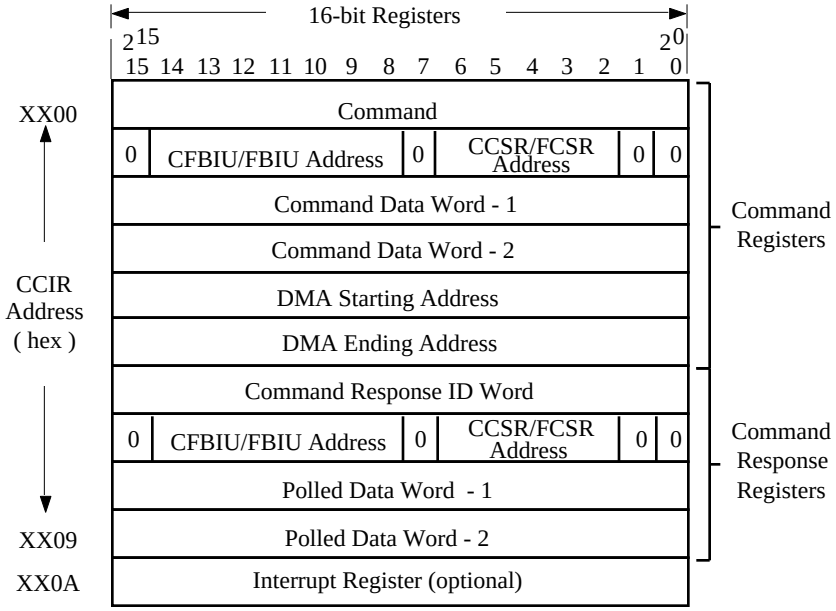

Figure 6 CCIR structure

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 - Configuration Data Word 1
Stores the most significant 16-bits of the CCSR/FCSR configuration word, if the command is a Load Command.

CCIR-XX03 - Configuration 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.

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 Verification Status
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 the CCIR-(XX06-XX09) are valid by setting the ML_Control Host Interface Process_Command Busy Indicator to False.

CCIR-XX0A - Interrupt Register (option)
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.

Bit 5	CCSR Parity Error
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 through CCIR-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.

Bit 5	Init Failed
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.

Bit 4	CFBIU Sync Loss
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.

Bit 3	Error Counter Max
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).

Bit 2	FBIU Poll Requested
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.

Bit 1	Frame Type = Synchronization
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.

Bit 0	Frame Type =	Error
1	TRUE	
0	FALSE	

CFBIU Single Command Definitions

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

- Load Command - Used to Load a CFBIU CCSR or a FBIU FCSR (See Figure 4-7).
- Poll Command - Used to Poll the contents of a CFBIU CCSR or a 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.

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.

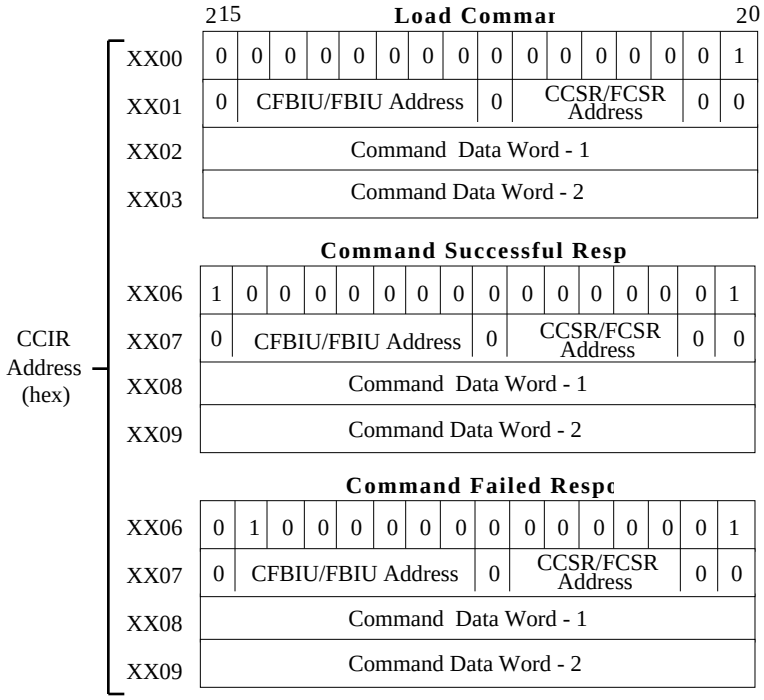

Figure 7 Load command and valid responses

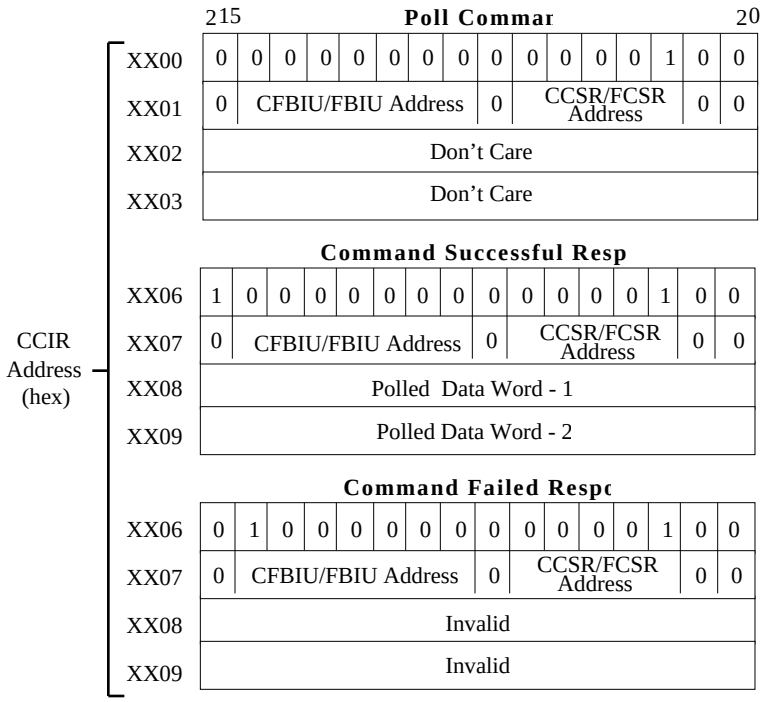

Figure 8 Poll command and valid responses

CFBIU Block Command Definitions

Block Commands allow the Control Host to load a set of sequential Configuration Commands into a predefined 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).

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.address.
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 lsb's 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.

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 the 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.

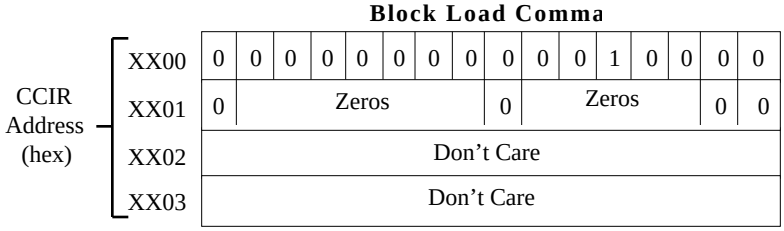

Figure 9 Block load command

Initialize Command & Ring Init execution

Upon receipt of a 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 the 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.

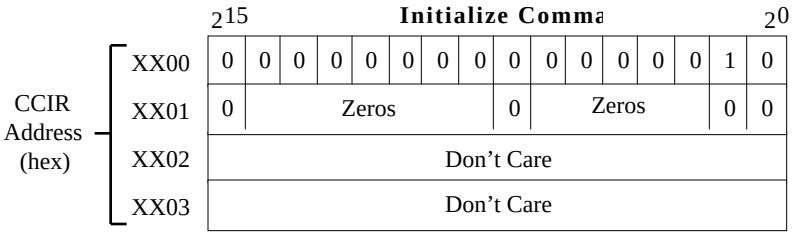

Figure 10 Initialize command

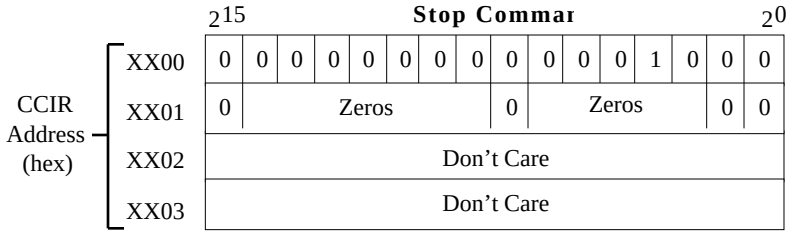

Figure 11 Stop command

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. An FBIU will generate an Error Frame when it is unable to acquire Frame Sync on the incoming serial Symbol stream.

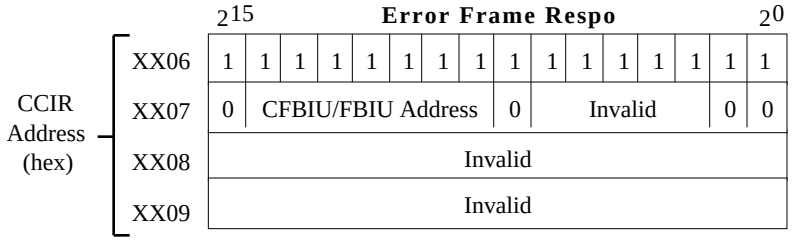

Figure 12 Error frame response

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-7 will contain the Address of the FBIU that generated the Poll for Status and CCIR-(8-9) will contain the contents of the FBIU Status Register(FCSR-19).

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.

Data Host interfaces

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

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.

Data Host Control 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

Table 3 Data Host control interfaces

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 6-1.

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 6-2.

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.

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

Table 0-4 Data Host data interfaces

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**.

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.

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.

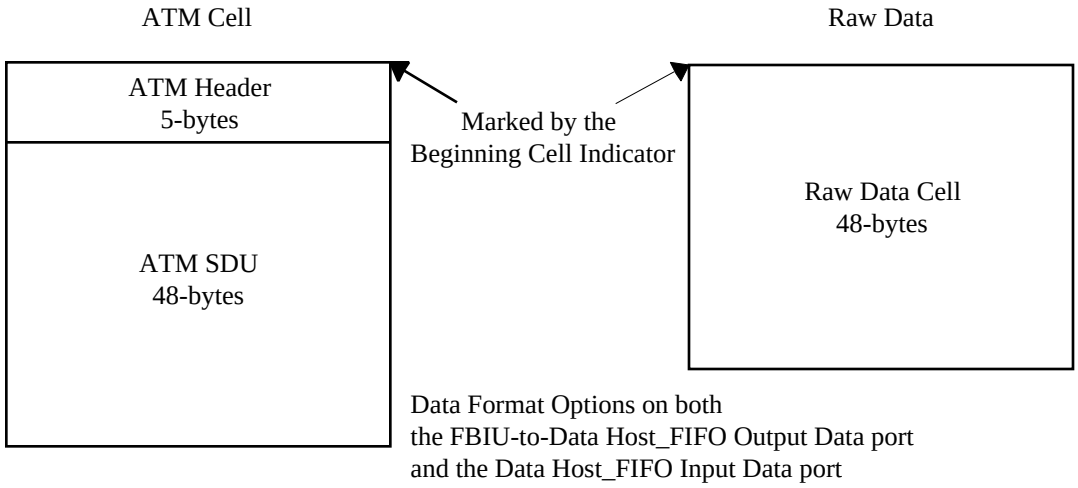

Figure 13 Data Host Data Format Options

SFODB Frame Structure

Frame, Subframe, Slot & Symbol Definition

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 Fame.

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 OH and a 53–Symbol encoded ATM Cell.

d. Symbol A Symbol is an 8B/10B encoded byte as defined in ANSI X3.230, Fibre Channel Physical and Signalling Interface (FC-PH), with Running Disparity implemented.

e. Frame Sync Symbol Two 8B/10B Control Codes (K Codes) are use as Frame Sync Symbols, K28.5 and K28.7. See the definition of Subframe Byte - 0 for Subframe “0” in section4.2.3.2.

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

See Figure 4-15 and the following field definitions for a detail definition of each Subframe OH field.

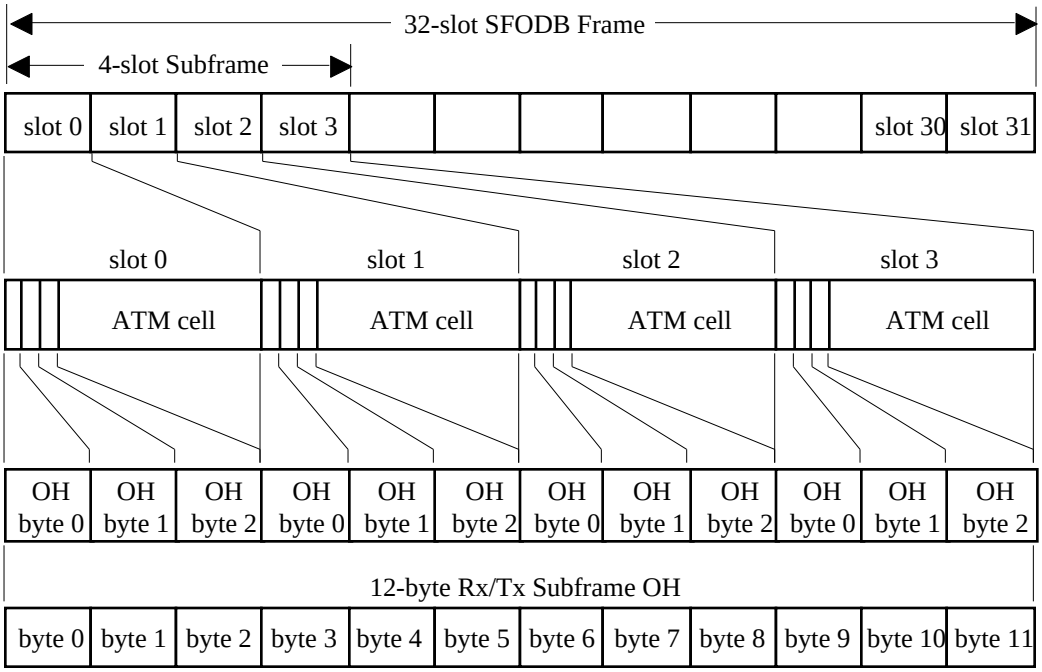

Figure 14 Subframe structure

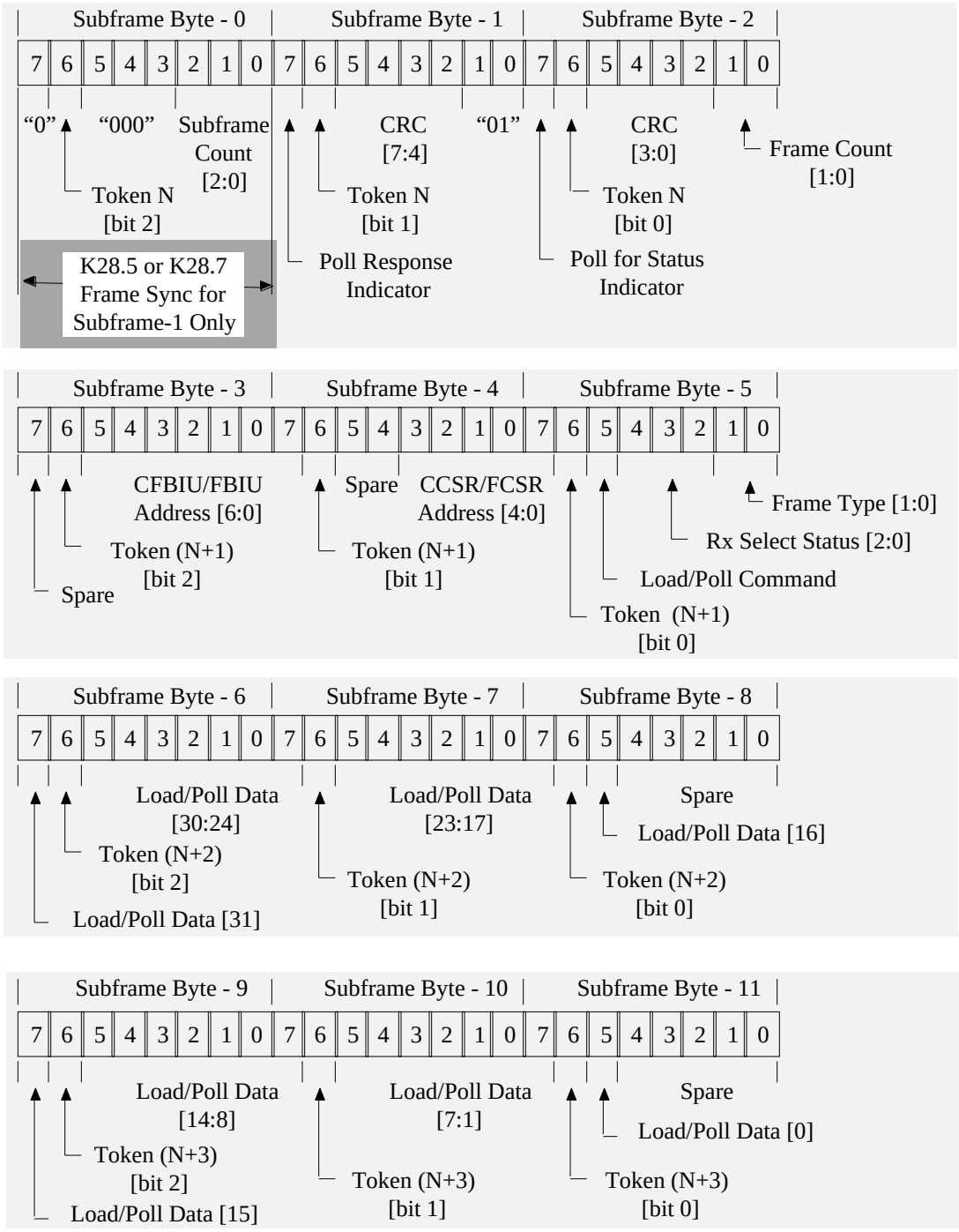

Figure 15 Subframe structure

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 #:	7	6	5	4	3	2	1	0	
	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.

Bits 7 is a static bit and is set to "0".

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

Bit 6	
0	FREE
1	CLAIM

Bits 5 - 3 are static bits and are set to "000".

Bits 2 - 0 contain the Subframe Count.

Bit #:	2	1	0	
	2^2	2^1	2^0	Subframe Count

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 msb's of the CRC for the previous subframe and the two static bits.

Bit 7 is the Poll Response Indicator.

Bit 7	
0	FALSE
1	TRUE

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

Bit 6	
0	FREE
1	CLAIM

Bits 5 - 2 contain the 4 msb's of the CRC for Subframe (N-1).

Bit #:	5	4	3	2	
	2^7	2^6	2^5	2^4	4 msb's of CRC for Subframe (N-1)

Bits 1 - 0 are static and set to "10".

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 lsb's of the CRC for the previous subframe and the Frame Count.

Bit 7 is the Poll for Status Indicator.

Bit 7	
0	FALSE
1	TRUE

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

Bit 6	
	FREE
1	CLAIM

Bits 5 - 2 contain the 4 lsb's of the CRC for Subframe (N-1).

Bit #:	5	4	3	2	
	2^3	2^2	2^1	2^0	4 lsb's of CRC for Subframe (N-1)

Bits 2 - 0 contain the Frame Count.

Bit #:	1	0	
	2^1	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 msb's of the FBIU Address.

Bits 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.

Bit 6	
0	FREE
1	CLAIM

Bits 5 - 0 contain the 6 msb's of the FBIU Address.

Bit #:	5	4	3	2	1	0	
	2^6	2^5	2^4	2^3	2^2	2^1	6 msb's 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 msb's of the FCSR Address.

Bit 7 contains the lsb of the FBIU Address.

Bit 7	
2^0	lsb of the FBIU Address

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

Bit 6	
0	FREE
1	CLAIM

Bits 5 - 4 are spare bits and preset to "00".

Bits 3 - 0 contain the 4 msb's of the FCSR Address.

Bit #:	3	2	1	0	
	2^4	2^3	2^2	2^1	msb's 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.

Bit 7	
2^0	lsb of the FCSR Address

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

Bit 6	
0	FREE
1	CLAIM

Bits 5 is the Load/Poll Command.

Bit 5	
0	POLL
1	LOAD

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

Bit #:	4	3	2	
	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 #: 1	0	
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.

Bit 7 and 5 - 0 contains the Load/Poll Data bits as defined below.

Bits #: 7	5	4	3	2	1	0	
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.

Bit 6	
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.

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

Bits #: 7	5	4	3	2	1	0	
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.

Bit 6	
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 #: 7	5	
2^{17}	2^{16}	Load/Poll Data

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

Bit 6	
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.

Bit 7 and 5 - 0 contains the Load/Poll Data bits as defined below.

Bits #: 7	5	4	3	2	1	0	
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.

Bit 6	
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.

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

Bits #: 7	5	4	3	2	1	0	
2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	Load/Poll Data

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

Bit 6	
0	FREE
1	CLAIM

Subframe Byte - 11

Byte-11 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 #: 7	5	
2^{1}	2^{0}	Load/Poll Data

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

Bit_6	
0	FREE
1	CLAIM

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

Sample startup scenario

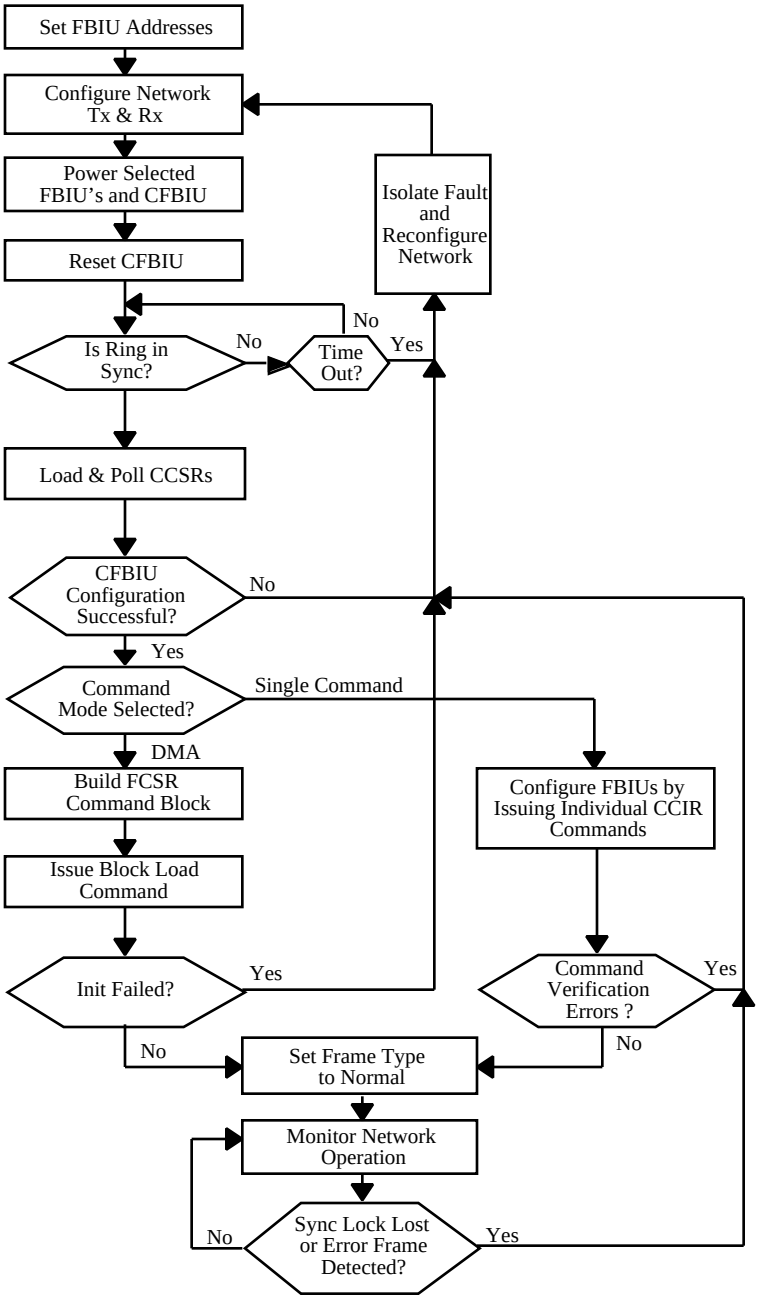

Figure 16 Sample startup scenario