ECSS-E-ST-50-XXX
Draft D: 29th February 2012

Space
Technology
Centre
University of Dundee

SpaceFibre Standard Draft D

LIKELY TO CHANGE!

Authors:	Steve Parkes
	Albert Ferrer
	Alberto Gonzalez
	Chris McClements
ESA Project Manager:	Martin Suess

This page is blank intentionally.

EUROPEAN COOPERATION
ECSS
FOR SPACE STANDARDIZATION

Space engineering

SpaceFibre

ECSS Secretariat
ESA-ESTEC
Requirements & Standards Division
Noordwijk, The Netherlands

Foreword

This Standard is one of the series of ECSS Standards intended to be applied together for the management, engineering and product assurance in space projects and applications. ECSS is a cooperative effort of the European Space Agency, national space agencies and European industry associations for the purpose of developing and maintaining common standards. Requirements in this Standard are defined in terms of what shall be accomplished, rather than in terms of how to organize and perform the necessary work. This allows existing organizational structures and methods to be applied where they are effective, and for the structures and methods to evolve as necessary without rewriting the standards.

This Standard has been prepared by the ECSS-E-ST-50-xxx Working Group, reviewed by the ECSS Executive Secretariat and approved by the ECSS Technical Authority.

Disclaimer

Published by: ESA Requirements and Standards Division
ESTEC, P.O. Box 299,
2200 AG Noordwijk
The Netherlands
Copyright: 2012 © by the European Space Agency for the members of ECSS

Change log

Draft	Authors	Details
Draft A 31st Oct 2007	S.M. Parkes C. McClements	Initial version.
Draft B 22nd Sept 2011	S.M. Parkes	Extensive revisions and re drafted in ECSS format.
Draft C 8th Dec 2011	S.M. Parkes Albert Ferrer	Technical corrections following prototyping and other minor revisions.
Draft D 29th Feb 2012	S.M. Parkes Albert Ferrer Alberto Gonzalez	Technical corrections and clarifications following review.

Table of contents

Figures

Tables

1 Scope

SpaceFibre is a very high-speed serial link designed specifically for use onboard spacecraft. It carries SpaceWire packets over virtual channels and provides a broadcast capability similar to SpaceWire time-codes but offering much more capability. SpaceFibre operates at 10 times the data-rate of SpaceWire and can run over fibre optic or copper media. When operating over fibre optic links SpaceFibre can run over distances of 100 m or more.

SpaceFibre is compatible with the packet level of the SpaceWire standard (ECSS-E-ST-50-12) and is therefore able to run the SpaceWire protocols defined in ECSS-E-ST-50-51C, 52C and 53C. This means that applications developed for SpaceWire can be readily transferred to SpaceFibre.

The aim of SpaceFibre is to provide point-to-point and networked interconnections for very high data-rate instruments, mass-memory units, processors and other equipment, on board a spacecraft.

This standard covers the protocols required to form a point-to-point link between two units. It does not cover the definition of SpaceWire packets and SpaceWire networks, which form the upper layers of SpaceFibre providing compatibility with SpaceWire at those levels.

The SpaceFibre standard specifies the interfaces to the user application and to the physical medium. Some other intermediate interfaces are also specified permitting interoperability at these intermediate levels. The functions that a SpaceFibre CODEC has to implement are specified. Preferred connector and cable characteristics for SpaceFibre optical and copper implementations are also specified.

This standard may be tailored for the specific characteristic and constraints of a space project in conformance with ECSS-S-ST-00.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this ECSS Standard. For dated references, subsequent amendments to, or revision of any of these publications do not apply. However, parties to agreements based on this ECSS Standard are encouraged to investigate the possibility of applying the more recent editions of the normative documents indicated below. For undated references, the latest edition of the publication referred to applies.

ECSS-S-ST-00-01	ECSS system - Glossary of terms
ECSS-E-ST-50-12C	Space engineering - SpaceWire - Links, nodes, routers and networks
ECSS-E-ST-50-51	Space engineering - SpaceWire protocol identification
ECSS-E-ST-50-52	Space engineering – Remote Memory Access Protocol

3
Terms, definitions and abbreviated terms

3.1 Terms defined in other standards

For the purpose of this Standard, the terms and definitions from ECSS-S-ST-00-01 and ECSS-E-50-50 apply.

3.2 Terms specific to the present standard

3.2..1 active lanes

the lanes that are ready to send data and control words, i.e. whose lane initialisation state machine is in the active state

3.2..2 bandwidth allowance

the amount of data that a virtual channel is allowed or expected to send in the bandwidth interval

3.2..3 bandwidth credit

the relative amount of link bandwidth that a virtual channel has accumulated

3.2..4 bandwidth credit limit

the maximum amount of positive or negative bandwidth credit that a virtual channel is allowed to accumulate

3.2..5 bandwidth used

the actual portion of link bandwidth that a virtual channel has actually used computed over the previous bandwidth measurement interval

3.2..6 character

data character or control code

3.2..7 comma

K28.5 control code

3.2..8 control code

8B/10B K-code

3.2..9 control word

comma or K28.3 or K28.4 control code followed by three data characters or EOP, EEP, or Null K-codes

3.2..10 current running disparity

the accumulated disparity of a bit stream from when it started to the present moment in time

3.2..11 data character

8-bit data value

3.2..12 data word

word of data comprising four SpaceWire N-Chars or Nulls

3.2..13 disparity

the number of ones in a bit stream minus the number of zeros in that bit stream

3.2..14 even disparity

the same number of ones and zeros in a bit stream

3.2..15 expected bandwidth

the proportion of overall link bandwidth that a virtual channel is expected to use

3.2..16 invalid symbol

symbol that contains a disparity error, i.e. it results in a running disparity greater than one or less than minus one, or is a symbol that does not occur in the 8B/10B decoding table, i.e. is not a valid symbol for an 8-bit data character or control code,

3.2..17 K-code

8B/10B control code

3.2..18 lane

SpaceFibre physical connection between two units

3.2..19 last frame bandwidth

the amount of data sent in the last data frame

3.2..20 link

SpaceFibre connection between two units that incorporates one or more lanes

3.2..21 link bandwidth

the amount of data that can be sent in one second over the SpaceFibre link

3.2..22 N-Char

SpaceWire data character, EOP or EEP

3.2..23 negative disparity

more zeros than ones in a bit stream

3.2..24 neutral disparity

the same number of ones as zeros in a bit stream

3.2..25 Null

character that can occur in a data frame after an EOP or EEP to fill the data word containing the EOP or EEP

3.2..26 ordered set

control word

3.2..27 permanent error

error on a link that cannot be recovered

3.2..28 persistent error

error on a link that can be recovered only by re-initialising the faulty link and resending the data

3.2..29 positive disparity

more ones than zeros in a bit stream

3.2..30 ready virtual channel

virtual channel with data ready to send and space in the virtual channel buffer at the far end of the link

3.2..31 required lanes

the lanes that are required to be used to form a SpaceFibre link

3.2..32 reserved bandwidth

the portion of link bandwidth that is set aside for use by a specific virtual channel using the bandwidth reservation quality of service

3.2..33 symbol

10-bit code resulting from 8B/10B encoding of a character

3.2..34 symbol rate

rate at which symbols can be handled in the transmitter and receiver

3.2..35 symbol word

a group of four consecutive symbols that when decoded will form a data word or control word

3.2..36 transient error

error on a link that can be recovered by resending the data without re-initialising the link

3.2..37 unrecognised symbol

symbol that does not appear in the 8B/10B symbol table

3.2..38 used bandwidth

the amount of data sent by a particular virtual channel in the last data frame, which is zero for all virtual channels except the one that sent the last data frame

3.2..39 used lane

lane that is being used by the SpaceFibre link

3.2..40 valid symbol

symbol that does not contain a disparity error and is found in the 8B/10B decoding table

3.2..41 word

data word or control word

3.2..42 word rate

rate at which words can be handled in the transmitter and receiver

3.3 Abbreviated terms

The following abbreviations are defined and used within this standard:

Abbreviation	**Meaning**
8B/10B	8-bit/10-bit
AC	alternating current
ACK	acknowledgement
BC	broadcast channel
B-TYPE	type of data in a broadcast frame
CRC	cyclic redundancy code
CML	current mode logic
CODEC	coder/decoder
DMA	direct memory access
EBF	end broadcast frame
EDF	end data frame
EEP	error end of packet
EOP	end of packet
FCT	flow control token
FDIR	fault detection, isolation and recovery
FIFO	first in first out
FR_SEQ#	frame sequence number
IACK	initialisation acknowledge control word
ID	identifier
iIDLE	inverse idle control word
iIACK	inverse initialisation acknowledge control word
iINIT	inverse initialisation control word
iLLCW	inverse lane layer control word
Inc	increment
INIT	initialisation control word
iSKIP	inverse skip control word
Len	length
LLCW	lane layer control word
LOS	loss of signal
LS	least-significant
LSB	least-significant bit
LSYNC	lane synchronisation control word
MAC	medium access controller

MS	most-significant
MSB	most-significant bit
NACK	negative acknowledgement
PCB	printed circuit board
PLL	phase locked loop
PRBS	pseudo-random bit sequence
QoS	quality of service
RMAP	remote memory access protocol
RX	receive
SBF	start of broadcast frame
SDF	start of data frame
SIF	start of idle frame
SOIS	spacecraft onboard interface services
TBA	to be advised
TBC	to be confirmed
TX	transmit
VC	virtual channel
VCB	virtual channel buffer
VHDL	VHSIC hardware description language
VHSIC	very high speed integrated circuit
VML	voltage mode logic

3.4 Conventions

In this document hexadecimal numbers are written with the prefix 0x, for example 0x34 and 0xDF15.

Binary numbers are written with the prefix 0b, for example 0b01001100 and 0b01.

Decimal numbers have no prefix.

4 Principles

4.1 SpaceFibre purpose

The aim of SpaceFibre is to provide point-to-point and networked interconnections for very high data-rate instruments, mass-memory units, processors and other equipment, on board a spacecraft. SpaceFibre is designed to operate using both existing and future space qualified SerDes interface devices, enabling practical implementations of SpaceFibre data-links on board spacecraft.

4.2 Guide to clause 5

Clause 5 of this standard provides the normative requirements. The SpaceFibre specification is separated into several functional layers. Each of these is placed into a separate subsection and provided with a service interface specification and a set of functional requirements. Each requirement has been designed to be verifiable.

Section 5.1 is a short overview of the following sub-sections.

Section 5.2 provides the service interface specification for the SpaceFibre interface. There are three service interfaces: the SpaceWire Packet Service which is used to send and receive SpaceWire packets over SpaceFibre; the Broadcast Message Service which is used to broadcast and receive short messages with low latency; and the Link Management Service which is used to configure and control the SpaceFibre link and to read status and error information.

Section 5.3 describes the formats of control words, SpaceWire characters, and frames which are used in SpaceFibre to initialise a link, to send data, and to detect and recover from errors.

Section 5.4 covers the virtual channel layer which is responsible for sending and receiving SpaceWire packets over SpaceFibre and for providing quality of service. Several qualities of service are supported concurrently by SpaceFibre: best effort, priority, bandwidth reservation, and scheduled. A SpaceWire packet is sent by placing it into a virtual channel buffer and received by reading it out of the corresponding virtual channel buffer at the other end of the SpaceFibre link. Each virtual channel buffer is configured to provide a specific quality of service. The SpaceWire packet information is segmented to support interleaving of data from several virtual channels taking into account the quality of service of each virtual channel. The virtual channel layer provides flow control across the link to avoid sending data when there is no room for it in buffers at the far end of the link. A medium access control in the virtual channel layer is responsible for appropriate multiplexing of data segments over the link, taking into account flow control information and quality of service.

Section 5.5 covers the broadcast message layer which is responsible for broadcasting and receiving short messages with low latency. The broadcast message service is intended to support time broadcast, synchronisation, fault signalling, and event signalling applications. The method for registering to receive broadcast messages is described. The way in which broadcast messages are broadcast are then considered along with the way in which they are validated before being passed further along the network or up to the end user application. The possible uses of the broadcast messages are then introduced.

Section 5.6 covers the framing layer which is responsible for encapsulating data, broadcast messages and flow control information into frames which are sent and received over the SpaceFibre link. The information in data frames is scrambled to mitigate EMC emissions.

Section 5.7 covers the retry layer which is responsible for error detection, isolation and recovery, at the link level. It adds frame sequence numbers and CRC checksums to the frames and flow control tokens (FCTs) from the framing layer. A retry buffer is provided to hold information until its correct reception at the far end of the link has been acknowledged. If a frame or FCT goes missing or arrives containing an error, the contents of the retry buffer are resent to rapidly recover from the fault. SpaceWire packet and broadcast messages are delivered without error, which simplifies error handling and FDIR at the application level. Negative acknowledgements are used to support rapid recovery from detected errors. The retry layer also provides a mechanism for sending idle frames when there is no application information to be sent. Idle frames optionally contain a pseudo-random bit sequence which can be used for bit error rate (BER) testing of the link.

Section 5.8 covers the lane control layer which is responsible multi-lane operation of a SpaceFibre link allowing information to be sent over several individual physical lanes to enhance throughput. The way in which multiple lanes are controlled and synchronised is specified, along with the mechanism for distributing information over several lanes on the transmit side and concentrating it back into a single information stream at the receive side of the link. A SpaceFibre link is the logical data link, which can comprise one or more physical lanes. The use of multiple lanes is optional.

Section 5.9 covers the lane layer which is responsible for sending information in the form of a stream of data and control words over a single lane. It provides mechanisms for initialising a lane, re-initialising the link in the event of a persistent error, and for adjusting for clock differences between the local clock and clock at the far end of the lane. The lane layer also provides an optional parallel loopback facility for test purposes.

Section 5.10 covers the encoding layer which is responsible for encoding data and control words for sending over a lane and for decoding those received from the other end of the lane. SpaceFibre uses 8B/10B encoding. In the receiver the encoding layer provides 8B/10B symbol synchronisation and data and control word synchronisation. Each data or control word is constructed from four 8B/10B symbols. 8B/10B encoding provides a DC balanced signal which can be AC coupled, supporting galvanic isolation.

Section 5.11 covers the serialisation layer which is responsible for transmitting the 8B/10B symbols as a serial bit stream and for recovering 8B/10B symbols from the received serial bit stream. The receiver provides bit synchronisation to recover the bit stream from the signals received by the physical layer. A mechanism for receive signal inversion is provided to permit freedom in routing the high-speed differential SpaceFibre signals on a PCB. A serial loopback facility is also provided for test purposes.

Section 5.12 covers the electrical and optical physical layers. The electrical characteristics of SpaceFibre drivers, receivers, PCB tracks, connectors and electrical cable are specified. The optical characteristics of the fibre optic version are provided. Where appropriate, connector mechanical information is also provided. SpaceFibre uses current mode logic (CML) for its electrical signalling.

Section 5.13 covers the management layer, which is responsible for configuring and controlling the SpaceFibre interface and for reporting error and status information. The values of the configuration parameters following reset are provided.

Section 5.14 covers conformance of implementation to the SpaceFibre specification and describes permitted partial implementations of the SpaceFibre specification.

4.3 SpaceFibre architecture

An overview of the SpaceFibre CODEC architecture is provided in Figure 4-1.

There are nine conceptual layers to the SpaceFibre CODEC:

- Virtual Channel and Flow Control: responsible for quality of service and flow control over the SpaceFibre link.
- Broadcast: responsible for broadcasting short messages across a SpaceFibre network and for receiving and checking those messages.
- Framing: responsible for framing SpaceWire packet data, broadcast messages and FCTs to be sent over the SpaceFibre link. It is also responsible for scrambling SpaceWire packet data for EMC mitigation purposes.
- Retry: responsible for recovering from transient and persistent errors on the SpaceFibre link, and for reporting errors and link failure. Detects missing and out of sequence frames.
- Lane Control: responsible for operating several SpaceFibre lanes in parallel to provide a higher data throughput and to provide redundancy with graceful degradation.
- Lane: responsible for initialising the lane, detecting lane errors and re-initialising the lane after an error has been detected.
- Encoding/Decoding: responsible for encoding data into symbols for transmission and decoding symbols into data for reception.
- Serialisation: responsible for serialising and de-serialising SpaceFibre symbols so that they may be transferred over the physical medium.

- Physical: responsible for transferring the electrical signals across a fibre optic or copper medium.

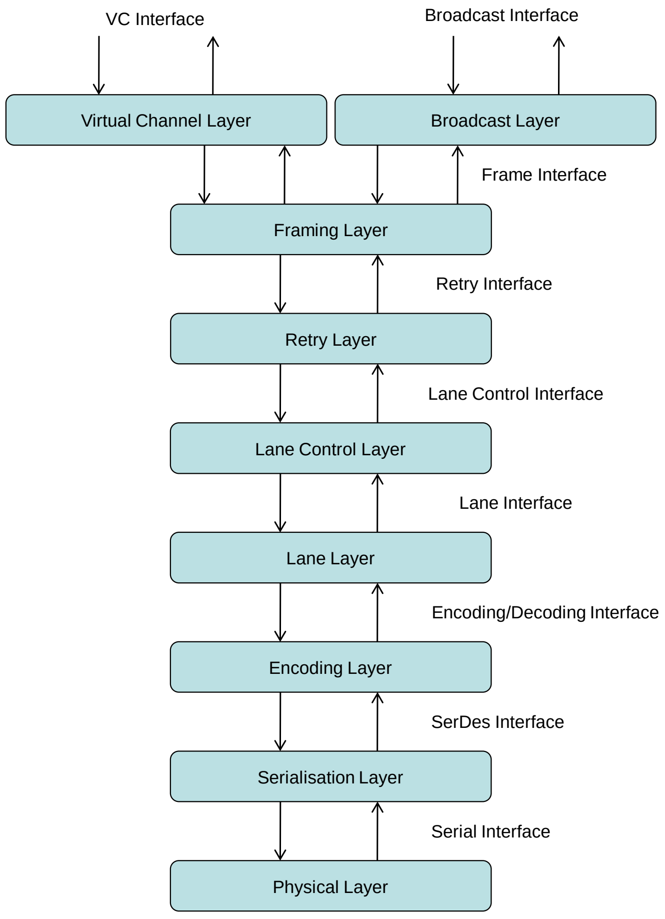


Figure 4-1 Overview of SpaceFibre CODEC

The detailed architecture of the SpaceFibre CODEC is illustrated in Figure 4-2.

The SpaceFibre CODEC performs several functions:

- Virtual channel buffering
- Segmentation
- Flow control
- Quality of service control
- Broadcast sequence number generation
- Broadcast validation
- EMC mitigation
- Framing
- PRBS testing
- Frame sequencing
- CRC check
- Frame retry buffering

- Idle frame insertion
- ACK/NACK
- Lane control
- Lane distribution/concentration
- Lane synchronisation
- Lane selection
- Lane initialisation and standby management
- Data rate adjustment
- Parallel loop-back
- Word synchronisation
- 8B/10B encoding and decoding
- Symbol synchronisation
- Receiver inversion
- Serialisation and de-serialisation
- Serial loop-back
- Line driver and receiver

Each of these functions is labelled in Figure 4-2 and described in the following sub-sections.

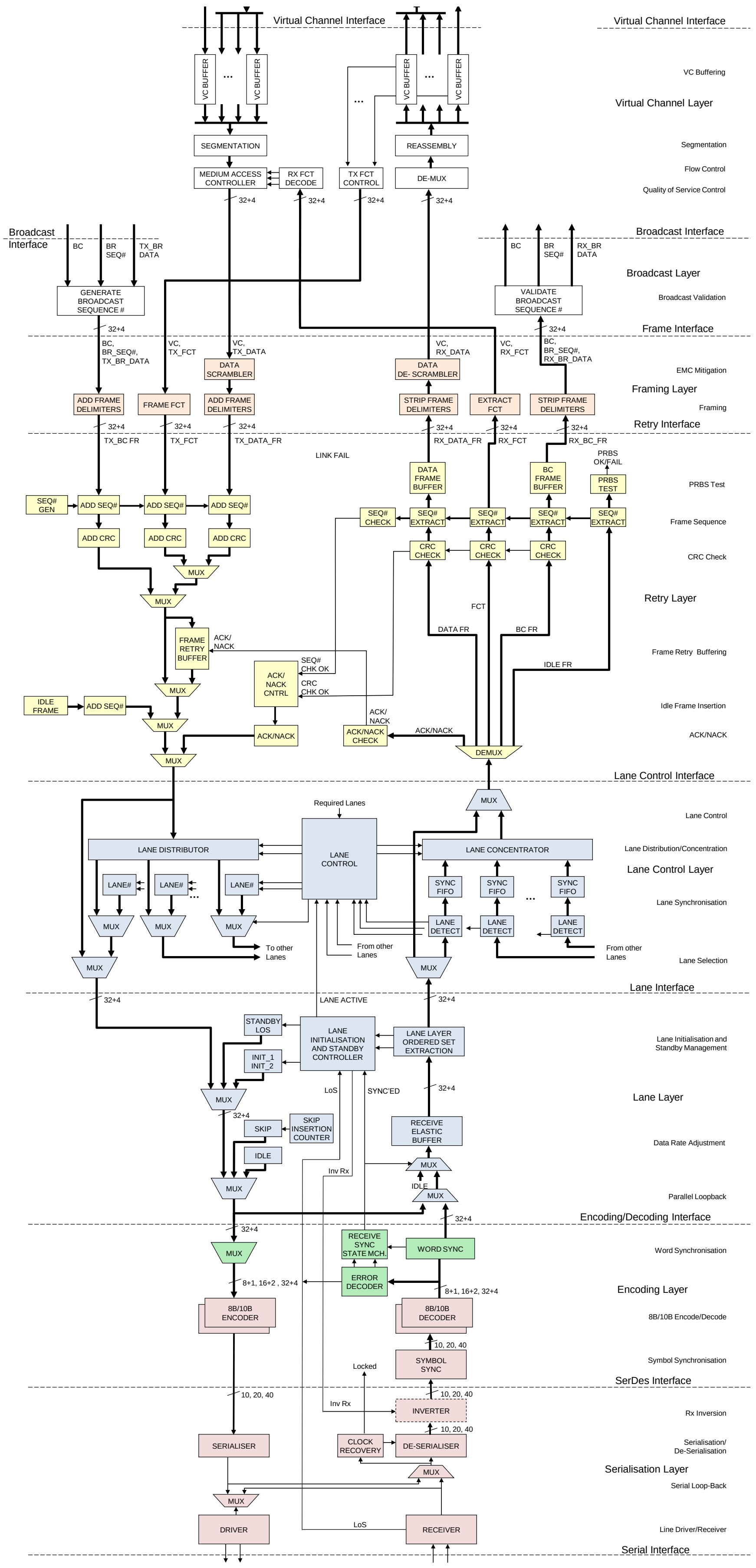


Figure 4-2 SpaceFibre CODEC Conceptual Architecture

4.3.1 SpaceFibre CODEC user interface

There are two different types of interface to the SpaceFibre CODEC: the virtual channel interface used to send and receive SpaceWire packets, and the broadcast channel interface used to broadcast short messages across a SpaceFibre network and to receive those broadcast messages.

The virtual channel interface of the SpaceFibre CODEC comprises a number of virtual channel buffers for sending SpaceWire packets (output VC buffers) and the same number for receiving SpaceWire packets (input VC buffers). There is also an interface for sending broadcast messages and an interface for receiving broadcast messages. The SpaceFibre CODEC is configured and controlled via registers the interface to which is application dependent.

The output VC buffer interface is used to send SpaceWire packets. Conceptually, each output VC buffer has a FIFO type interface that can accept SpaceWire data characters and EOP markers. To send a SpaceWire packet over a SpaceFibre virtual channel, the SpaceWire packet destination address and cargo are loaded sequentially into the appropriate output VC buffer, followed by an EOP. The specific interface to the VC buffer is application dependent.

Interfaces to the input VC buffers are used to read SpaceWire packets that have been received over the corresponding SpaceFibre virtual channel. Each input VC buffer has a FIFO type interface, from which SpaceWire data characters and EOP markers can be read.

The broadcast channel interface to the SpaceFibre CODEC comprises a set of registers for writing the parameters of a broadcast message (broadcast channel, broadcast sequence number, and the message) and a similar set of registers for reading received broadcast messages. The user registers to be notified on the reception of specific classes of broadcast message.

A service interface specification for the SpaceFibre CODEC is provided in section 5.2.

4.3.2 Virtual channel layer

The virtual channel layer is responsible for quality of service and flow control over the SpaceFibre link. It controls the quality of service related to delivery of SpaceWire packets.

4.3.2.1 Virtual channel buffering

The output virtual channel buffers (VCBs) are used to buffer SpaceWire packet data before that data is sent over the SpaceFibre link. Data is sent in frames containing up to 256 SpaceWire N-Chars or Nulls. The output VCBs permit this amount of data to be buffered before it is offered for transfer over the SpaceFibre link. Sending the data in frames and buffering data prior to framing, permits efficient interleaving of many SpaceWire packets travelling over different virtual channels over the SpaceFibre link.

The input VCBs provide a similar function for the reception of data arriving over the SpaceFibre interface. An input VCB provides storage for at least two maximum size data frame to ensure that when it arrives there is room for all the data it contains and that data can be transferred continuously if the end user can keep up with the data rate. The application using the SpaceFibre CODEC can then read data from the input VCB at its leisure, without causing loss of data on the SpaceFibre link.

4.3.2.2 Segmentation

Data is sent over the SpaceFibre link in a series of data frames which each contain up to 256 N-Chars or Nulls. The SpaceWire packet data in an output virtual channel buffer has to be segmented into chunks of up to 256 N-Chars or Nulls for placement into the data frames.

When a series of data frames are received for a particular virtual channel the data has to be extracted from them and reassembled into the SpaceWire packet data which is placed in the appropriate input virtual channel buffer.

4.3.2.3 Flow control

To manage the flow of data from all of the virtual channels across the SpaceFibre link it is necessary to know which output VCBs have data to send at one end of the link, and which input VCBs have space for more data at the other end of the link. Exchange of this information is performed with credit based flow control: by exchanging flow control tokens (FCTs) for data frames. The input VCBs are monitored to determine when they have space for another maximum-sized data frame (up to 256 N-Chars or Nulls). An FCT is sent to the other end of the link when a particular input VCB has space for another 256 N-Chars or Nulls. The TX FCT Controller manages the sending of the FCT. When the FCT is received at the other end of the link, it is passed to the RX FCT controller, which informs the MAC which VCBs are able to accept another data frame.

The potential loss of FCTs is handled by the retry layer, which ensures that FCTs cannot be lost unless the link suffers a permanent failure, in which case it is not possible to use that link any more.

4.3.2.4 Quality of service control

A medium access controller determines which VCB is allowed to send data over the SpaceFibre link. This depends on several things:

- Which output VCBs have data to send;
- Which input VCBs at the other end of the SpaceFibre link have space available to receive data;
- The arbitration or quality of service (QoS) policy in force for each Virtual Channel.

For SpaceFibre several quality of service policies are possible including:

- Best effort, where the only Virtual Channels with data to send are those with best effort quality of service, i.e. no other Virtual Channel with a different quality of service has data to send;
- Priority, where the Virtual Channel with the highest priority goes first;
- Bandwidth reserved, where the Virtual Channel with allocated bandwidth and recent low utilisation of the link will go first;

- Scheduled, where time-slots are defined by a local time register or broadcast messages, and only the Virtual Channel allocated to the current time-slot is permitted to send data. If this Virtual Channel has no data to send then another Virtual Channel may use this unused bandwidth opportunistically.

When a virtual channel has data to send in its output VCB and has room for data in the input VCB at the other end of the SpaceFibre link, it competes with other virtual channels in a similar state. The virtual channel permitted to send a frame of data will be the one with the most urgent need to send data according to the QoS policies of all the competing virtual channels. The virtual channel layer then passes a frame containing up to 256 N-Chars or Nulls from the selected output VCB to the framing layer for sending over the SpaceFibre link.

It is possible to have several virtual channels in a SpaceFibre interface, each operating with a different quality of service. Each virtual channel computes a precedence value based on its quality of service. The medium access controller uses this to determine which virtual channel is permitted to send a frame of data next.

Precedence for the priority quality of service is related to the various priority levels. Extremely urgent priority setting results in a very high precedence value, which will mean that an extremely urgent priority virtual channel will be able to send data just about as soon as it is ready to go. Other priority settings will have lower precedence values and so will compete with other qualities of service when requesting to send data. Each priority level has a fixed precedence.

Precedence for the bandwidth reserved quality of service is determined by the reserved bandwidth for the virtual channel and its recent used of link bandwidth. As a link sends data and uses up its reserved portion of the link bandwidth, so its precedence will drop, making it harder for it to compete with other virtual channels. When it is not able to send data because other virtual channels are sending data, its precedence will increase, making it easier for the link to compete against other virtual channels.

Precedence for the scheduled quality of service depends on a schedule table which specifies when a virtual channel is permitted to send data. To support the scheduled quality of service the link bandwidth is split into time-slots, either using a local time register set by a time broadcast message or using a synchronising broadcast message. The schedule table specifies when a virtual channel set to scheduled quality of service is allowed to send data. Such a virtual channel is not allowed to send data at any other time, only in its allocated time-slots. When a time-slot starts during which a scheduled virtual channel is allowed to send data, its precedence setting will be set to the highest possible precedence value, except for that f the highest priority level (extremely urgent priority). This ensures that during a time-slot the virtual channel scheduled to send data in that time-slot will be able to do so. If this virtual channel does not have any data to send, or there is no space in the virtual channel buffer at the other end of the link, another non-scheduled virtual channel is permitted to use the otherwise wasted bandwidth.

Precedence for the best effort quality of service is set to a very low value so that a best effort virtual channel will only be able to send data when there is no virtual channel with a different quality of service able to send data. If several best effort channels have data to send they will compete based on the bandwidth they expect to used (expected bandwidth).

Each virtual channel has its quality of service configured via the link management interface and is able to take on any of the possible SpaceFibre qualities of service.

4.3.3 Broadcast layer

The broadcast layer is responsible for broadcasting short messages across a SpaceFibre network and for receiving and checking those messages.

It receives broadcast messages to be sent from the user application and passes the information necessary to form a broadcast frame to the framing layer.

When a broadcast frame is received from the framing layer, the information it contains is taken out of the frame and passed to the user application.

4.3.3.1 Broadcast messages

A broadcast message is a short message that is sent by a node to all the other nodes on the SpaceFibre network. Broadcast messages propagate in a similar manner to SpaceWire time-codes. Each broadcast message contains a broadcast sequence number which is incremented each time a new broadcast message is sent. When a broadcast message arrives at a SpaceFibre receiver it is checked for errors and its broadcast sequence number is validated by comparing it to the broadcast sequence number of the last broadcast message received. The broadcast message is valid if its broadcast sequence number is one more than that of the previous broadcast message received. Only valid broadcast messages are passed out of the SpaceFibre CODEC. A SpaceFibre router will forward the broadcast message out of all of its SpaceFibre links except the one that the broadcast message was received on.

SpaceWire permits one set of time-codes to be broadcast, although by using the two flags in the time-code it is possible (but not legal according to the SpaceWire standard) to have four independent sequences of time-codes operating concurrently. SpaceFibre broadcast messages permit up to 256 independent sequences of broadcast messages each of which is referred to as a broadcast channel. Each broadcast channel has a broadcast channel identifier and its own broadcast sequence number.

The broadcast channels are split into three types:

- 0-31: Network management broadcast channels.
- 32-253: Node broadcast channels, with each broadcast channel associated with a node that has a logical address of the same value as the broadcast channel number.
- 254 & 255: Reserved broadcast channels.

The network management broadcast channels are split into three sub-types

- 0-3: Time distribution, which are used to provide fault tolerant distribution of system time over the SpaceFibre network.

- 4-7: Synchronisation, which are used to provide synchronisation services over the SpaceFibre network.
- 8-31: Network control, which are used to support configuration, control, and FDIR of a SpaceFibre network.

Broadcast messages also carry 8 bytes of data. A broadcast type field determines the meaning of the 8 bytes of data. For example, when type = TIME, the 8 bytes contain 8 bytes of time information. A broadcast message over one of the time synchronisation channels would typically be of type TIME and the eight data bytes would contain a system time value (un-segmented time).

Typically a particular broadcast channel will be used by a specific node to broadcast information to all other nodes on the SpaceFibre network. This can be used to signal events that occur in that node to other nodes on the network. Different nodes broadcast over different broadcast channels.

A user application of a SpaceFibre CODEC can subscribe to receive broadcast messages from specific broadcast channels and of specific broadcast type. In this way the application will only be notified and receive those broadcast messages that it is interested in.

4.3.3.2 Broadcast sequence number generation

The transmit side of the broadcast layer is responsible for generating the appropriate broadcast sequence number to be included in a broadcast message. Each broadcast channel has its own broadcast sequence number which is incremented each time a broadcast message is sent over that broadcast channel. The broadcast channel number, broadcast sequence number, and message to be sent in the broadcast message are passed to the framing layer for encapsulation into a broadcast frame and transmission over the SpaceFibre interface.

4.3.3.3 Broadcast validation

The receive side of the broadcast layer is responsible for validating the broadcast message by checking its broadcast sequence number. The broadcast layer receives the broadcast channel number, broadcast sequence number and message data from the framing layer for each correctly received broadcast message. It then checks that the broadcast message is valid i.e. its broadcast sequence number is one more than the broadcast sequence number of the last broadcast message received. If this is the case, the broadcast channel number, broadcast sequence number and message data are all passed to the user application. If the broadcast sequence number is not valid that broadcast message is discarded.

Note that the broadcast sequence number is different to the frame sequence number in the retry layer.

4.3.4 Framing layer

The framing layer is responsible for framing SpaceWire packet data, broadcast messages and FCTs to be sent over the SpaceFibre link. It is also responsible for scrambling SpaceWire packet data for EMC mitigation purposes.

The framing layer receives SpaceWire packet and FCT information from the virtual channel layer and broadcast message information from the broadcast message layer and puts this information into frames.

The framing layer passes the resulting data frames, FCTs and broadcast frames to the retry layer, each in the order in which they are to be sent over the SpaceFibre link. The retry layer controls the way these three streams of information are concentrated into the single stream of data/control words that flows over the SpaceFibre link.

The framing layer receives data frames, FCTs and broadcast frames that have been received and checked for errors by the retry layer. It removes the framing information and descrambles the data in data frames. Data frames and FCTs are passed up to the virtual channel layer and broadcast frames are passed to the broadcast layer.

4.3.4.1 EMC mitigation

Sending a constant bit pattern over a serial link can cause high levels of EM emission due to the energy being concentrated in a few frequency components. To avoid this, data may be scrambled before transmission by convolving the data sequence with a pseudo-random sequence. A pseudo-random sequence approximates white noise and thus has a wide bandwidth. Convolution of the data with a pseudo-random sequence broadens the frequency components, spreading the energy and reducing the peak emission levels.

SpaceFibre uses this technique to mitigate the EM emissions over copper SpaceFibre links. A scrambler is used to scramble the data in each data frame. The original data is then recovered by a de-scrambler at the other end of the link. To help with this process each frame is multiplied by the same pseudo-random sequence i.e. the pseudo-random generator is re-seeded with the same seed at the start of each frame. Broadcast frames and FCTs are not scrambled.

When the link has no other data to send it will send IDLE control words to keep the link active. Sending repeated IDLEs will once again result in excessive EM emission spectral spikes. To avoid this problem, whenever there are no data frames to send an idle frame containing a pseudo-random bit sequence is sent, this is able to reduce the EM emission peaks by 7 to 10 dB. Idle frames are terminated as soon as another data frame becomes ready for transmission, so that the introduction of idle frames does not significantly affect the sending of data frames. Idle frames are generated by the retry layer.

This EMC mitigation technique depends on the spectrum of the data having relatively high peaks which are broadened by convolution with the pseudo-random bit sequence.

The data being scrambled can contain SpaceWire data characters, EOPs, EEPs and Nulls. Nulls are only used as fillers at the end of a frame. The data is scrambled but the EOPs, EEPs and Nulls are special K-codes and must not be scrambled. The scrambler includes the value of these K-codes in the scrambling of the data, but does not overwrite the K-code with the scrambled data.

4.3.4.2 Framing

Framing is the delimiting of data, broadcast and idle frames by control words that indicate the start and end of the frame. There are three types of start of frame: start of data frame (SDF), start of broadcast frame (SBF), and start of idle frame (SIF). There are two types of end of frame: end of data frame (EDF) and end of broadcast frame (EBF). The idle frame does not have a specific end of frame control word associated with it. The end of idle frame is indicated by the next start of frame (SDF, SBF or SIF) that is received.

4.3.4.2.1 Data frames

A data frame is illustrated in Figure 4-3.

0 7	8 15	16 23	24 31
COMMA	SDF	VC	Reserved
DATA 1 LS	DATA 1	DATA 1	DATA 1 MS
DATA 2 LS	DATA 2	DATA 2	DATA 2 MS
...	...	...	...
DATA N LS	DATA N	DATA N	DATA N MS
EDF	FR_SEQ#	CRC_LS	CRC_MS

Figure 4-3 Data Frame Format

A data frame is a series of 32+4 bit data words delimited by start of data frame (SDF) and end of frame (EDF) control words. Each data word holds four SpaceWire N-Chars or Nulls. The data frame is limited to a maximum frame size of 64 words (256 SpaceWire N-Chars or Nulls). The data field in a data frame is scrambled when it is transmitted and unscrambled when it is received.

The start of data frame includes the virtual channel number that identifies the virtual channel associated with this data frame.

The end of data frame includes space for information that will be supplied by the retry layer used for checking that there is no missing or duplicate frames (frame sequence number FR_SEQ#) and that there are no errors in the frame (CRC).

Each symbol in the data field of the data frame can contain a SpaceWire data character or a SpaceWire EOP or EEP or Null. A single data frame holds a chunk from a stream of SpaceWire packets, all of which are associated with the same virtual channel. For example, a data frame could contain any of the following:

- Start of a large SpaceWire packet,
- Middle of a large SpaceWire packet,
- End of a large SpaceWire packet,
- End of a large SpaceWire packet and start of following packet,
- Several small packets,
- End of a packet, followed by several small packets,
- Etc.

4.3.4.2.2 Broadcast frames

A broadcast frame is illustrated in Figure 4-4.

0 7	8 15	16 3	24 31
COMMA	SBF	BC	B_SEQ#/B_TYPE
DATA 1 LS	DATA 1	DATA 1	DATA 1 MS
DATA 2 LS	DATA 2	DATA 2	DATA 2 MS
EBF	Reserved	FR_SEQ#	CRC

Figure 4-4 Broadcast Frame Format

A broadcast frame contains the broadcast channel identifier, the broadcast sequence number, a broadcast type and a seven byte message.

The start of broadcast frame includes the broadcast channel number that identifies the broadcast channel (BC) associated with this broadcast frame, along with the broadcast sequence number (B_SEQ#) used to check that the broadcast message is valid.

The broadcast type specifies the type of broadcast message and hence the meaning of the following eight bytes of data.

The end of data frame includes information supplied by the retry layer for checking that there is no missing or duplicate frames (frame sequence number FR_SEQ#) and that there are no errors in the frame (CRC).

4.3.5 Retry layer

The retry layer is responsible for recovering from transient and persistent errors on the SpaceFibre link, and for reporting errors and link failure. It detects and corrects for missing, out of sequence and duplicate data frames, broadcast frames and FCTs, along with those that contain errors.

The retry layer receives data frames, FCTs and broadcast frames from the framing layer to be sent over the SpaceFibre link.

To support the retry operation it is necessary for the other end of the link to be able to detect missing, out-of-sequence, or erroneous frames and FCTs. This is achieved by adding a sequence number to each frame or FCT and then adding a CRC checksum to the data and broadcast frames. A single common series of sequence numbers is used for data frames, broadcast frames and FCTs, with the incrementing sequence number being applied in the order in which the frames and FCTs are sent over the link.

The CRC checksum covers the complete frame information including the start of frame and the end of frame containing the sequence number.

The complete data frames, broadcast frames and FCTs, are multiplexed together into a single stream of data ready to be sent over the SpaceFibre link. The broadcast frames and FCTs can be inserted at any point within a data frame. While data frames from different virtual channels are interleaved by the MAC in the virtual channel layer, broadcast frames and FCTs are not interleaved but inserted into the stream of data as soon as they are available to be sent. Broadcast frames and FCTs are interleaved, with broadcast frames having priority. FCTs cannot be inserted in a broadcast frame. The sequence numbers are only assigned and added to the frame as the end of the frame passes through the retry layer.

An example showing how data frames, broadcast frames and FCTs are combined to form the data stream is shown in Figure 4-6.

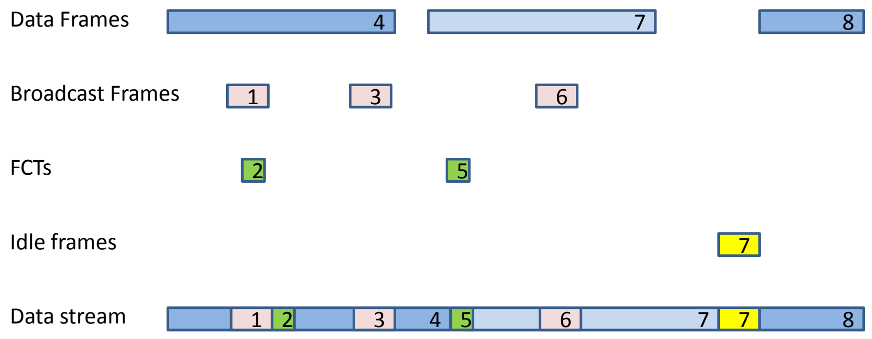


Figure 4-5 Frames and FCTs combined to form the data stream

As the data stream is formed and sent word by word over the SpaceFibre link, it is copied into the frame retry buffer.

On receipt of a data or broadcast frame or FCT over the SpaceFibre link the retry layer checks that it is in the correct sequence and does not contain errors and then passes it up to the framing layer. Each correctly received frame/FCT is acknowledged (ACK) so that the space that it is occupying in the frame retry buffer can be released to make room for other frames waiting to be sent. Each frame that is not correctly received results in a negative acknowledgement (NACK) being sent. The NACK contains the frame sequence number of the last correctly received frame/FCT. This results in the erroneous frame/FCT and all subsequent ones being resent. If a frame/FCT is lost completely, when the next correctly received frame or FCT is received, a NACK is sent resulting in that frame/FCT and all subsequent ones being retrieved from the frame retry buffer and resent. Each frame that is not correctly received is discarded.

If the SpaceFibre link is re-initialised, the current contents (i.e. not yet ACKed frames/FCTs) of the frame retry buffer are retransmitted.

When there are no frames/FCTs, ACKs or NACKs to send over the SpaceFibre link the retry layer will generate an idle frame to send. When idle frames are received they are checked to monitor error rates and then discarded. Idle frames are given the sequence number of the last sent data frame, broadcast frame or FCT. This enables the receiver to detect a previously missing or discarded frame/FCT and to send a NACK to request its retry. For example consider a data frame that does not arrive for whatever reason. Assume that there is no more data or FCTs to be sent, so an idle frame is sent. This arrives at the far end of the link carrying the sequence number of the data frame that has gone missing. The receiver realises that there is a missing frame and can then send a suitable NACK. Without this facility it could be a long time before there is more data to send prompting the retry. The retry would then rely on a time-out timer, which is much less responsive.

The retry layer sends and receives the data and control words making up the frames/FCTs and ACKs/NACKs to the lane control layer

4.3.5.1 Frame sequencing

The retry control function is responsible for checking that frames/FCTs arrive without error (CRC checksum correct), without duplicate, missing, and out-of-order frames (FR_SEQ# correct). It uses a CRC checksum and frame sequence number to determine if any of these errors have occurred.

There are two sequence number counters: one in the transmitter which produces the sequence numbers and one in the receiver that keeps track of the sequence number of the last correctly received frame/FCT and is used to check the subsequent frames/FCTs. On cold reset the sequence number in the transmitter and receiver are both set to zero.

The data frames, broadcast frames and FCTs all share the same frame sequence number. The frame sequence number is not to be confused with the broadcast sequence number used in broadcast frames. Broadcast frames have two sequence numbers: broadcast sequence number and frame sequence number.

When a data frame or broadcast frame or FCT is being sent the sequence number is incremented and its new value placed in the sequence number field.

When a data frame or broadcast frame or FCT is received without error, its sequence number is compared to the value of the receive sequence counter, which holds the sequence number of the last correctly received frame/FCT. If the frame sequence number is one more than the receive sequence counter, that frame is accepted, an ACK sent containing the frame sequence number, and the receive sequence counter incremented.

If the frame sequence number is not one more than the receive sequence counter, the frame/FCT is rejected and a NACK is sent containing the sequence number of the last correctly received frame/FCT, i.e. the current value of the receive sequence counter. The receive sequence counter is not changed.

Frames or FCTs that arrive in error are discarded.

When an ACK is received, the space in the frame retry buffer containing the frame/FCT being acknowledged is freed. If a NACK is received, the frame/FCTs in the retry buffer are resent. If the frame retry buffer becomes full, a time-out timer is started. If no ACK or NACK is received by the time this time-out timer expires all the frames/FCTs in the retry buffer are resent.

Persistent errors result in an error being signalled to the upper layers.

4.3.5.2 CRC check

Data frames and broadcast frames contain end user data which has to arrive without error.

A 16-bit CRC checksum is used to detect errors in data frames. This CRC covers the entire data frame including the SDF and EDF control words. The data values of any K-codes are included in the CRC, i.e. the CRC is applied to the 32-bits of the 32+4 bit data stream.

An 8-bit CRC checksum is used to detect errors in broadcast frames. The CRC covers the entire broadcast frame including the SBF and EBF control words. A shorter CRC can be used since a broadcast frame contains only 16 characters.

An 8-bit CRC checksum is used to detect errors in FCTs. The CRC covers the all the fields in the FCT control word.

4.3.5.3 Frame retry buffering

The frame buffer in the transmit side of the SpaceFibre CODEC is used to store data frames, broadcast frames and FCTs until their reception has been acknowledged by the far end of the SpaceFibre link. In the event of a negative acknowledgement (NACK) arriving indicating that a particular frame/FCT was missing or in error, the frame buffer resends all the data frames, broadcast frames and FCTs that have been sent after the one identified by the NACK as being received successfully.

It will also resend all the data frames, broadcast frames and FCTs it contains if the link is re-initialised or if there is an ACK/NACK time-out.

The frame buffers in the receive side of the SpaceFibre CODEC are used to store a data frame or a broadcast frame while it is being received and checked. Only frames/FCTs without error, without duplication, and in order are passed up to the framing layer for de-framing.

4.3.5.4 Idle/PRBS test

Idle frames are generated by the retry layer when there is no user information to send. On reception, idle frames are checked and then discarded. An idle frame is illustrated in Figure 4-6.

0 7	8 15	16 23	24 31
COMMA	SIF	FR_SEQ#	RESERVED
SEED LS	SEED	SEED	SEED MS
PRBS 1 LS	PRBS 1	PRBS 1	PRBS 1 MS
PRBS 2 LS	PRBS 2	PRBS 2	PRBS 2 MS
…	…	…	…
PRBS N LS	PRBS N	PRBS N	PRBS N MS

Figure 4-6 Idle Frame Format

An idle frame begins with a start of idle frame (SIF) control word. This begins with a comma, followed by a SIF symbol, a frame sequence number field (FR_SEQ#) that contains the sequence number of the last data frame, broadcast frame, or FCT, and a reserved field. The frame sequence numbering of an idle frame is a little different to that of a data frame, broadcast frame or FCT: the header of an idle frame contains a frame sequence number, which is the sequence number of the last frame or FCT sent, i.e. the current value of the frame transmit sequence counter is not incremented for an idle frame.

The SIF control word is followed by data words that contain a pseudo-random bit sequence (PRBS). There is no end of idle frame control word. The idle frame stops as soon as there is any useful data to send, indicated by a start of data frame or start of broadcast frame.

The seed used for the PRBS is contained in the first data word sent after the SIF control word. This enables the PRBS to be checked in the receiver. The value of the PRBS generator at the end of an idle frame is retained and used as the seed for the next idle word. The PRBS used for idle frames is different to that used for data frame scrambling.

The length of an idle frame is limited to a maximum of 64 PRBS data words. If there is no data frame or broadcast frame to send at that point a new idle frame is started with its seed taken from the end value of the PRBS of the previous idle frame.

The retry layer checks that the PRBS in each received idle frame is correct, by regenerating the PRBS from the seed contained at the start of each idle frame.

Once a link has been established and idle frames are being sent, the receiver will automatically and continuously check link integrity, supporting bit error rate (BER) testing. Sending idles can also be used to support measuring the eye pattern of the link to assess the performance of the physical layer.

4.3.5.5 ACK/NACK

ACKs are sent to indicate that a data frame, broadcast frame or FCT has been received without error and without missing, duplicate, or out of sequence frames or FCTs. An ACK is a control word that contains the frame sequence number of the frame or FCT being acknowledged i.e. of the last frame or FCT correctly received.

NACKs are sent to indicate that a missing data frame, broadcast frame or FCT has been detected. A NACK is a control word that contains the frame sequence number of the last frame or FCT correctly received.

When an ACK is received it causes the frame retry buffer to be updated by removing the frame or FCT that is being acknowledged from the retry buffer. Note that more than one frame of FCT can be acknowledged by an ACK.

When a NACK is received it causes the contents of the frame retry buffer to be resent, starting with the frame or FCT following that identified in the NACK as the last correctly received frame/FCT. The NACK also acknowledges any frames/FCTs prior to and including the one it identifies as the last correctly received frame/FCT.

ACKs and NACKs are injected into the transmit data stream with precedence over any data frame, broadcast frame or FCT.

4.3.6 Lane control layer

The lane control layer permits several lanes to be used in parallel to send data more quickly. For example, if a single lane is capable of sending data at 2Gbits/s, four lanes operating in parallel will permit a data rate of 8 Gbits/s to be achieved. This assumes that the implementation of the higher layers of the SpaceFibre CODEC can support the higher data rate provided by multiple lanes.

The lane control layer is responsible for operating several lanes in parallel to provide a higher data throughput and to provide redundancy with graceful degradation.

The lane control layer receives data and control words from the retry layer. Depending on the required number of lanes and the number of lanes currently operating, the lane control layer spreads the data and control words out over the lanes in use. This is done sequentially on a word by word basis. With the first word going to the first lane in use, the second word to the second and so on.

To ensure that the receiving end of the multilane SpaceFibre link is able to decode the received data and control words in the right order, the lane control layer sends lane synchronisation information across each lane whenever the number of available lanes changes. The receiver uses this information to synchronise the decoding of information from the lanes currently in use.

SpaceFibre permits any number of lanes to be operated in parallel up to a maximum of ten lanes (TBC). Since running a link over several lanes reduces the reliability of that link (any one of the N lanes failing will cause the link to fail) SpaceFibre allows additional lanes to be added to support both hot and cold redundancy and to support automatic graceful degradation in the event of a lane failure.

Implementation of lanes is not mandatory. A SpaceWire CODEC is permitted to have just one lane, in which case it does not need the lane control layer.

It is possible for lanes to be crossed over, e.g. lane connector number 1 to be connected to lane connector number 3 at the other end of the link.

4.3.6.1 Lane control

The lane control function is responsible for determining which lanes are to be used to send data/control words and which lanes are receiving data/control words.

On the sending side there are two parameters: the required number of lanes and the number of lanes currently operating (i.e. in the active state, see section 5.9.2). There are then three possibilities:

- Required number of lanes is equal to the number of lanes currently operating, in which case the information to be sent is distributed word by word to the currently operating lanes.
- Required number of lanes is greater than the number of lanes currently operating, in which case the information to be sent is distributed over the available operating lanes. The user application is informed that the requested number of lanes is not available. This provides support for graceful degradation. The link will continue to operate even when one of the requested lanes has failed. The data rate will be reduced but the link continues operating. A network management application could then activate another lane if available, restoring full data-rate operation. This provides cold standby.
- Required number of lanes (N) is less than the number of lanes currently operating, in which case the information to be sent is distributed over the first N lanes. The other operating lanes will only be used in the event of one of the used lanes failing. This provides hot standby.

The lane controller is informed by the lane layer which lanes are active and ready to send information.

4.3.6.2 Lane distribution/concentration

The lane distributor takes the stream of data/control words from the retry layer and spreads them out over the available lanes as determined by the lane controller. The first word goes to the lowest number used lane, the next word to the next lowest number used lane and so on until all the lanes have been provided with a word to send. The next word then goes to the lowest number used lane again. If three lanes are used, words are send in groups of three, one across each of the used lanes.

The lane concentrator does the opposite function to the lane distributor. It takes several data/control words arriving in parallel over the available lanes and passes them sequentially to the retry layer. The lane order in which data/control words arrive is determined by the lane controller.

4.3.6.3 Lane synchronisation

Lane synchronisation is performed following a cold reset of the SpaceFibre interface and whenever the number of required lanes, or number of operating lanes changes. In this event the lane controller causes each operational lane (up to the required number of lanes) to send its lane number over the lane using a lane synchronisation control word. For example, if there are four lanes and the third one is not operating, the other three lanes will be given the lane numbers 1, 2 and 3. Any operating lanes that are not to be used will transfer the lane number 0.

At the other end of the link the synchronisation control word will be detected in each operational lane and the lane controller informed of which lanes are to be used. The lane controller then instructs the lane concentrator which lanes to concentrate data from. Data/control words are then read from the operational lanes sequentially in lane number order.

It is possible that the words arrive in the lanes at different times. This is handled by the lane synchronisation FIFOs. As words arrive they are placed in these FIFOs. They are always read by the lane concentrator in lane number order as specified by the lane controller. A word can only be read from a specific FIFO when it contains one or more data/control words. If a word on a particular lane is late in arriving compared to the other lanes, the lane concentrator is then forced to wait for that word to arrive. Thereafter the lanes will be synchronised.

If a persistent error occurs, the SpaceFibre lane with the error will be re-initialised. While this is occurring, the lane controller will have detected an error and resynchronised the lanes using the lanes that are running properly. Any information that failed to be delivered successfully would then be resent by the retry layer. When the lane with the persistent error has re-initialised, the lane controller will resynchronise the lanes again permitting all lanes to be used once more. If a lane has a permanent error, re-initialisation of the lane will fail. In this case operation with the reduced number of lanes will continue.

It is possible that a lane synchronisation control word is corrupted or lost. If this happens the lane concentrator will not know the correct sequence for reading data from the lanes. There are several possible error cases:

Missing lane synchronisation symbol: detected as an active lane that does not receive a lane synchronisation symbol when the other active lanes receive one.

This is likely to be a temporary fault, so all lanes are reinitialised which will automatically cause resynchronisation of the lanes.

Duplicate lane synchronisation symbol: detected as two or more active lanes that receive lane synchronisation symbols with the same lane number. This is likely to be a serious fault in the transmitter and cannot normally occur. All lanes are reinitialised, causing resynchronisation of the lanes to attempt to recover from the fault.

If either type of error persists it must be flagged as a permanent fault to the application.

4.3.6.4 Lane selection

The lane selection function selects whether lanes are to be used or not. On the transmit side a selector determines if the data/control words are to come directly from the retry layer or via the lane circuitry.

On the receive side the data/control words are either passed directly to the retry layer, or via the lane circuitry.

4.3.7 Lane layer

The lane layer is responsible for initialising the lane, detecting link errors and re-initialising the lane after an error has been detected.

Once a lane has been initialised, the lane layer receives data/control words from the lane control layer for sending over the lane. It passes data/control words to the encoding/decoding layer for encoding and sending over the lane.

When data/control words are received over the lane, they are passed from the encoding/decoding layer to the lane layer. The lane layer strips out any control words related to the lane layer, using them for data rate adjustment, lane initialisation and standby management. The data words and remaining control words are passed up to the lane control layer.

4.3.7.1 Lane initialisation and standby management

Lane initialisation and standby management is handled by a lane state machine, which is responsible for initialising the lane prior to transfer of data frames, idle frames or broadcast frames. During lane initialisation bit-synchronisation, symbol-synchronisation and word-synchronisation are performed. A handshake protocol is used to ensure that both ends of the lane have achieved synchronisation. Capability negotiation is also performed during initialisation.

The lane initialisation state machine is also responsible for handling Loss of Signal (LoS) indication from the receiver. When Loss of Signal occurs the lane initialisation state machine sends LOST_SIGNAL control words to the far end of the lane (End A) to indicate that it has suffered LoS. End A of the lane can then record that a Loss of Signal occurred in End B. This is very useful for debugging the cause of a lane failure.

A lane can go into standby when there is no more data to send. In standby the transmitters are disabled to save power while the receivers are enabled ready to auto-start the lane when the other ends tries to initialise the lane again. If one end of the lane (End A) disables its transmitter, it will appear as a Loss of Signal to the other end (End B). To avoid this being detected as an error, End A sends Standby control words to End B to signal that it is about to put the lane into Standby. Both ends are then able to disable their transmitters without causing a Loss of Signal error.

4.3.7.2 Data rate adjustment

The two ends of the lane operate at the same bit rate within the limits of the crystal oscillators at either end. Frequency accuracy is normally ± 100 ppm for a typical crystal oscillator. An elastic-buffer is used in the receiver to compensate for any difference in the clocks at either end of the lane (see Annex A.6 for a description of an elastic buffer). If the transmitter is operating very slightly faster than the receiver then the receive elastic-buffer will start to fill up. If the transmitter is running more slowly than the receiver then the receive elastic-buffer will start to empty. Ideally the receive elastic-buffer should be kept half full. To do this the transmitter sends SKIP control words periodically, at least every 5000 control words or 32-bit data words. If the receive elastic-buffer is more than half full then the SKIP control word is skipped over when reading out of the buffer *i.e.* is ignored and the control word or data following is read out instead. This has then effect of reading out two control words or 32-bit data words rather than one so the buffer becomes emptier. If the receive elastic-buffer is less than half full then the SKIP control word is read out but the buffer output pointer is not updated so that the SKIP is read out twice. This slows down the emptying of the receive elastic-buffer and make the buffer fuller.

The receive elastic-buffer is an important element in the SpaceFibre interface: it allows data or control words to be read out of the 8B/10B encoder every cycle without having to do a hand-shake to see if a character is ready to be read. This significantly improves the speed of operation of the interface.

4.3.7.3 Parallel loop back

A parallel loop back is optionally provided at the bottom of the lane layer. This is used for testing of the upper layers of the SpaceFibre CODEC regardless of the operation of the encoding/decoding, serialisation and physical layers. It is also useful for testing the operation of one end of a SpaceFibre lane before the other end of the lane has been implemented.

4.3.8 Encoding/decoding layer

The encoding/decoding layer is responsible for encoding data/control words into symbols for transmission and decoding symbols into data/control words for reception. It receives data/control words from the lane layer encodes each of them into four symbols which are then passed to the serialisation layer for serialising and transmission over the physical layer.

Symbols are received from the serialisation layer, but they are not properly synchronised. The encoding/decoding layer synchronises the symbols and decodes them. Groups of four decoded symbols form a data/control word. The encoding/decoding layer is responsible for synchronising the symbols on a data/control word boundary and for forming the correctly synchronised data/control words.

4.3.8.1 Word synchronisation

A control word comprises a four symbol code: starting with a comma or other specific K-code (K28.3 or K28.4) and followed by three other symbols which are either data-symbols or K-symbols. The comma symbol in a control word is used to indicate the first symbol in a group of four that form a data/control word. The word synchroniser looks for the comma symbol, re-synchronising the word boundary when a comma is received. This synchronises subsequent data and control words.

4.3.8.2 8B/10B encoding and decoding

The 8B/10B encoding takes an 8-bit input together with a data/control (D/K) flag and provides a 10-bit code for serialisation. The 8B/10B decoder does the inverse operation to recover the 8-bit value and D/K flag. The 8B/10B encoding/decoding is described further in section A.2.

4.3.8.3 Symbol synchronisation

In the receiver once the bit synchronisation has been achieved and the bit stream recovered (see sections 4.3.9 and A.4) it is necessary to determine the boundary of the symbols in the bit stream (every 10-bits) and then to identify the data/control word boundary (every 40-bits). The comma symbol (see section A.2) is used to support both symbol and data/control word boundary detection. The comma symbol contains a unique 7-bit code which can be used to detect the boundary between the comma symbol and the next symbol. Control words normally begin with a comma symbol so the boundary between one data/control word and the next can also be detected using the comma symbol. Note that two control words, end of data frame (EDF) and end of broadcast frame (EBF), do not start with a comma.

4.3.8.4 Synchronisation on single disparity commas

Some radiation tolerant SerDes devices support symbol synchronisation on only positive disparity commas or only negative disparity only commas. This means that there is a possibility that symbol synchronisation might never occur if the symbol stream was such that the commas were always the wrong disparity for synchronisation.

To avoid this situation the control words used during initialisation have been specially designed to ensure symbol synchronisation on positive disparity commas, negative disparity commas, or both.

During initialisation INIT control words are sent continuously until the far end of the link has achieved bit, symbol and word synchronisation and sent back IACK control words. INITs have a special structure to ensure that synchronisation can take place on positive or negative only commas.

A comma (K28.5) always changes the running disparity e.g. if the disparity is positive immediately before a comma is received, it will be negative immediately afterwards. If the three symbols after the comma in an INIT all have even disparity, after they have been received the disparity will remain in the same state as after the comma. So, in the example, if the disparity is positive before an INIT is received, it will become negative immediately afterwards. Sending a pair of INITs one after the other is therefore enough to ensure proper synchronisation when the symbol synchroniser operates on only positive or only negative disparity commas.

It is possible that SKIP and IDLE control words are also sent during initialisation, so these two control codes are also designed to have the same disparity characteristics as an INIT control word.

4.3.9 Serialisation layer

The serialisation layer is responsible for serialising and de-serialising SpaceFibre symbols so that they may be transferred over the physical medium. It also provides an optional receiver inversion function which simplifies PCB layout and is recommended for all new SpaceFibre SerDes designs.

The serialisation layer receives symbols from the encoding/decoding layer. These are transferred either singly, in pairs or in groups of four, depending on the specific nature of the serialisation layer (e.g. external SerDes device).

Symbols received over the SpaceFibre physical layer are de-serialised by the serialisation layer and passed up to the encoding/decoding layer either singly, in pairs or in groups of four, depending on the specific nature of the serialisation layer.

4.3.9.1 Receiver inversion

Receiver inversion enables the PCB layout to be optimised for signal integrity. The receiver automatically detects the polarity of the signal being received and corrects the information being received accordingly. This permits the CML + and – signals to be swapped around on both the transmitter and receiver PCBs. The advantage is that the PCB layout can concentrate on signal integrity avoiding having to cross over CML + and – signals using multiple vias which would degrade signal integrity, especially at the high speeds at which SpaceFibre operates.

Some SerDes devices that can be used as part of a SpaceFibre CODEC design do not include receiver inversion and a PCB design using such a SerDes must not swap CML + and – lines. Furthermore if one of these devices is being used transmit signal inversion is not permitted at the far end of the link.

During initialisation the receiver has to detect whether the bit stream is inverted or not. To support this, the INIT and IACK control word symbols are selected to have valid inverse symbols, i.e. if the symbols is bit-wise inverted it will create a different, but valid symbol. The lane initialisation state machine can check for these inverted INIT or IACK symbols and change the polarity of the receive bit stream if they are received. SKIP and IDLE control words are also designed to have valid inverse symbols.

4.3.9.2 Serialisation and de-serialisation

Serialisation is the conversion of the 10-bit parallel symbols to a bit stream.

De-serialisation is the recovery of a 10-bit parallel stream from a serial bit stream. This requires clock recovery from the bit stream using a phase-locked loop (PLL) which is helped by the 8B/10B code giving a transition-rich set of transmitted codes.

4.3.9.3 Serial loop back

A serial loop-back function is provided to support testing of the complete SpaceFibre CODEC, including lanes, without the need for a physical loop-back. It also permits the testing of the physical layer before the upper layers of the CODEC at the far end of a link have been tested or integrated.

4.3.9.4 Line driver and receiver

The serialised data is driven onto a 100 ohm differential impedance line using current-mode logic (CML). This can then be used to drive a fibre optic transmitter. A fibre optic receiver converts the received optical signals back to CML which is then passed over a 100 ohm differential impedance line to the receiver. Both the CML driver and CML receiver are AC coupled.

CML is used on its own to provide a copper interface to SpaceFibre.

4.3.10 Physical layer

The physical layer is responsible for transferring the electrical signals across a fibre optic or copper medium.

4.4 Typical operation

The typical operation of the SpaceFibre interface will now be described with reference to Figure 4-2.

4.4.1 Sending data over virtual channels

The user application writes SpaceWire packet data characters and EOPs into one of the output virtual channel buffers (VCBs). The size of the output VCB might be small compared to the size of a packet so the packet information may have to be written in chunks into the output VCB. Each output VCB is large enough to contain at least two (TBC) complete data frames i.e. can contain at least 2 x 256 = 512 SpaceWire data characters or EOPs. The user application can write into several of the output VCBs at the same time since multiple SpaceWire packets can be transferred over the SpaceFibre link concurrently.

When there is enough data in an output VCB to fill a complete data frame, or when the VCB contains at least one EOP, the output VCB informs the medium access controller that it has some information ready to send.

The medium access controller receives information about the space available in each of the input VCBs at the other end of the SpaceFibre link from the RX FCT decoder. When a virtual channel has data to send (output VCB signals that it has data ready to send) and that virtual channel has room in its input VCB at the other end of the SpaceFibre link (RX FCT decoder signals that space is available), the medium access controller puts that virtual channel forward for arbitration.

Each virtual channel is assigned a quality of service (QoS), that determines which of several virtual channels put forward for arbitration will gain access to the SpaceFibre link and be able to send a data frame. Possible QoS include round robin or fair arbitration, priority arbitration, bandwidth reservation and scheduled arbitration. The QoS for a specific virtual channel is configurable.

From the virtual channels put forward for arbitration and the QoS regimes for each of these virtual channels, the medium access controller selects one of the virtual channels and passes data from it to the framing layer. Data frames from different output VCBs are interleaved by the medium access controller, before they are passed to the retry layer.

The RX FCT decoder receives FCTs from the far end of the link. Each FCT gives credit for sending a further 256 N-Chars or Nulls. A transmit credit counter contains 256 times the number of FCTs received minus the number of N-Chars or Nulls sent, i.e. credit received minus credit spent. Whenever this count is greater than or equal to 256 it indicates that the input VCB at the other end of the link has room to receive another full data frame (64 words).

The input virtual channel buffers at the local end of the link generate FCTs when they have space for another full data frame. These FCTs are produced by the TX_FCT controller which monitors the amount of space in each VCB to determine when it has enough unallocated space for another maximum size data frame. When there is enough space it sends an FCT to the framing layer for transmission to the RX FCT decoder at the other end of the link. The TX FCT controller at one end of the link and the RX FCT decoder at the other end of the link cooperate to control the flow of data frames across the link to avoid overflow of each of the input virtual channel buffers.

When the TX FCT controller needs to send an FCT to the other end of the link it is passed from the virtual channel layer to the framing layer.

4.4.2 Message broadcast

To send a broadcast message the user application simply writes the broadcast channel number, and the data to be broadcast to the broadcast message output registers. The user application is permitted to pass broadcast messages for broadcasting over the entire SpaceFibre network at the same time as it is writing SpaceWire packet data into the output VCBs.

Each broadcast channel has its own broadcast sequence counter. When a broadcast message is to be sent, the sequence counter for that broadcast channel is then incremented, and the broadcast channel number, broadcast data and the value of the sequence counter are passed to the framing layer.

4.4.3 Framing

The data sent from the selected virtual channel buffer is segmented into segments containing a maximum of 256 N-Chars or Nulls. This data is then scrambled by the data scrambler excluding any EOPs, EEPs or Nulls, i.e. only the data characters are scrambled. The scrambled data is then framed by adding a start of data frame control word at the front of the data and an end of data frame control word at the end.

The start of data frame control word contains the following information:

- Comma, which indicates that this is a control word;
- Start of Data Frame, which identifies this control word as a start of data frame;
- Virtual channel number, which identifies the virtual channel that this data frame is being sent over;

The end of data frame control word contains the following information:

- End of Data Frame control word, which identifies this control word as an end of data frame. This control word is K28.3 which is not a comma.
- Space for frame sequence number, which will be filled in by the retry layer;
- Space for a 16-bit CRC, which will be filled in by the retry layer;

FCTs from the virtual channel layer are encapsulated into a single control word containing the following information:

- FCT control symbol, which identifies this control word as an FCT. This control word is K23.7 which is not a comma.
- Virtual channel number, which identifies the virtual channel that this FCT relates to;
- Space for frame sequence number, which will be filled in by the retry layer.
- Space for an 8-bit CRC, which will be filled in by the retry layer;

Broadcast messages are framed by the framing layer but the data is not scrambled.

The start of broadcast frame control word contains the following information:

- Comma, which indicates that this is a control word;
- Start of Broadcast Frame (SBF), which identifies this control word as a start of broadcast frame;
- Broadcast channel number, which identifies the broadcast channel that this broadcast frame is being sent over;
- Broadcast sequence number, which is used for validating the broadcast message before it is propagated by a router or forwarded to an application in a node.

The end of broadcast frame control word contains the following information:

- End of Broadcast Frame (EBF) symbol, which identifies this control word as an end of broadcast frame. This control word is K28.4 which is not a comma.
- Space for frame sequence number, which will be filled in by the retry layer;

- Space for an 8-bit CRC, which will be filled in by the retry layer.

The data frames, FCTs and broadcast frames are passed to the retry layer.

4.4.4 Retry

To perform the error detection function for the retry function, it is necessary to add a frame sequence number and CRC checksum to the data frames, broadcast frames and FCT. This information is added as the frames enter the retry layer from the framing layer.

The various streams of information that are to be sent over the SpaceFibre link are then multiplexed into a single data stream. Any FCTs that the TX FCT controller requires to be transmitted are implanted amongst the data/control words from data frames. FCTs are thus sent very quickly without having to wait for the end of the current data frame.

Broadcast frames have to be sent as soon as possible since they might be carrying time-critical information. Broadcast frames are implanted amongst the data/control words from data frames, rather than interleaved between data frames. This means that the broadcast frames do not have to wait for the end of a data frame before it can be sent. While an FCT can be implanted inside a data frame it cannot be implanted inside a broadcast frame. Broadcast frames can be implanted at any point in a data frame.

The resulting stream of data and control words containing data frames embedded with FCTs and broadcast frames is passed to the lane control layer for sending over the SpaceFibre link. At the same time the data stream is copied into a frame retry buffer in case any frame or FCT is corrupted or goes missing.

If there is no information to send over the link, an idle frame is automatically generated to fill the gap between useful pieces of information. This is necessary to keep the link synchronised.

In the case of it being necessary to retry sending one or more frames, the frames/FCTs to be resent are read out of the frame retry buffer.

Any ACKs or NACKs that need to be sent to indicate successful reception of a frame/FCT or to indicate that a frame/FCT is missing are implanted into the stream of words being passed to the lane control layer.

The precedence of information flowing into the lane control layer is therefore as follows:

- Broadcast frames highest precedence
- ACK/NACK
- FCTs
- Data frames
- Idle frames lowest precedence

4.4.5 Lane control - transmit

The transmit lane control function distributes the data and control words from the retry layer across the active lanes that are making up the SpaceFibre link.

Each lane has a separate physical connection. Running a SpaceFibre link over several lanes increases the data rate that can be achieved over the SpaceFibre link.

The data and control words from the retry layer are distributed word by word over the used lanes by the lane distributor, starting with the first word going to the lowest number used lane.

To provide a means of collecting the distributed words together again at the other end of the link, the lanes must be synchronised. At the transmit side lane numbers are inserted into each of the active lanes, whenever the lane controller detects that the number of required lanes has changed or one or more of the lanes has re-initialised. These lane numbers identify the order in which data and control words should be read out of the lanes and concentrated in the receiver at the far end of the link.

A final multiplexer in the lane control layer selects whether lanes are to be used or not.

The data and control words from the lane control layer are passed to the lane layer.

4.4.6 Lane transmission

The lane layer is responsible for establishing a connection over the physical SpaceFibre lane.

During lane initialisation special control words (INIT and IACK) are multiplexed into the transmitter to perform the initialisation handshake and configuration under direction of the lane initialisation and standby controller.

Once a connection has been established it indicates to the lane control level that the lane is ready (ACTIVE) and is then able to transfer data and control words.

To facilitate putting the lane into standby where the transmitters at both end of the lane are disabled the lane initialisation and standby controller injects standby control words into the transmit data stream. This causes the other end of the lane to disable its transmitter. The local transmitter is then disabled. In standby the receivers are still operational so that each end of the lane is able to respond when requested to re-initialise.

To support FDIR, if the local receiver detects that it has lost the receive signal, i.e. that the signal level is below a minimum acceptable level, the lane initialisation and standby controller will cause Loss of Signal (LoS) control words to be injected into the transmit data stream to inform the other end of the lane of the fault.

To support the receive elastic-buffer, skip control words are injected into the transmit data stream at a rate determined by the Skip Counter and at an interval no longer than 5000 control words or data words (see annex A.6).

Idle control words are sent when the lane is active and there is no other information to send.

The data and control words from the lane layer are passed to the encoder for 8B/10B encoding prior to transmission.

4.4.7 Encoding

The 8B/10B encoder encodes each data character or control code into a 10-bit code, which has special properties to support data transfer and recovery in the receiver (see annex A.2). The data words and control words are four symbols wide so conceptually four parallel 8B/10B encoders are required to perform the encoding.

4.4.8 Serialisation

The 10-bit code from the 8B/10B encoder is passed to the serialiser which converts it into a serial bit stream. This bit stream is driven on to the communications medium by the driver.

The input to the serialiser can be a 10, 20 or 40-bit wide interface from one, two or four 8B/10B encoders.

4.4.9 De-serialisation

The receiver takes the incoming bit stream and recovers the bit stream clock using a phase locked loop. The recovered bit clock is used by the de-serialiser to convert the serial bit stream into a slower speed, parallel data stream which can be handled a little more easily. This parallel data stream can be 10, 20, or 40 bits wide depending on the implementation of the SpaceFibre interface.

An inverter in the receiver is able to invert all the bits from the receiver. It does this on command from the lane initialisation and standby controller when an inverted data stream has been detected. The inversion by the inverter will then correct the received data. This capability is useful to simplify PCB layout for the very high-speed SpaceFibre signals. It is optional in SpaceFibre as some space qualified SerDes devices do not support receive bit inversion.

4.4.10 Decoding

The 10-bit symbols are distinguished in the parallel data stream by the symbol synchroniser. This unit determines the position of comma codes in the 10-bit parallel data stream and then outputs the data correctly aligned on the symbol boundaries.

The aligned 10-bit symbols are passed to an 8B/10B decoder which decodes the 10-bit symbols into 8-bit data characters each together with a D/K bit. The 8B/10B decoder is run from the received clock. The 8B/10B decoder detects various forms of error: invalid 10-bit code and incorrect running disparity. If an error occurs this is indicated to the receive synchronisation state machine. An error decoder may be necessary to decode the error indications from the SerDes and 8B/10B decoder into signals that are appropriate for the receive synchronisation state machine.

The stream of data+D/K characters are passed to the word synchroniser. Each data word starts with a comma symbol in the least significant symbol position. Each control word starts with a comma or other specific K-code depending on the particular control word. The word synchroniser synchronises on the position of the commas in the data stream to be able to extract subsequent data and control words from the symbol stream.

The receive synchronisation state machine monitors the operation of the symbol and word synchronisers and the 8B/10B decoder and determines when the receiver is synchronised or has lost sync. This state is indicated to the lane layer (SYNC'ED).

The received, synchronised data and control words are passed up to the lane layer.

4.4.11 Lane reception

The data or control words are buffered in a receive elastic-buffer, which copes with any differences between the receive clock and the local system clock (see annex A.6). Only words that are correctly synchronised are placed into the receive elastic-buffer. When the encoding/decoding layer indicates that the decoder has lost sync, the data and control words being written to the receive elastic-buffer are replaced by IDLE control words.

The output from the receive elastic-buffer is filtered for SKIP, IDLE and initialisation control words. SKIP and IDLE control words are discarded and initialisation control words (INIT, IACK, RCLR and RACK) are passed to the lane initialisation and standby controller along with the standby control words (STANDBY and LOS).

The lane initialisation and standby controller provides support for lane initialisation and recovery from errors. During initialisation it ensures that bit and character synchronisations are achieved and that the two ends of the lane are both ready to send and receive data.

The remaining data and control words are passed up to the lane control layer.

4.4.12 Lane control - receive

The lane control layer on the receive side concentrates the data and control words from the active lanes into a single data/control word stream that is passed up to the retry layer.

When a lane synchronisation control word is received it is detected by the lane detector in each lane and the lane number that it contains is passed to the lane controller. Lanes that are not to be used for transferring data/control words are indicated with a null lane number. The lane controller instructs the lane concentrator about the order in which data is to be read from the lanes as indicated by the lane number in the lane synchronisation control word received for each lane.

Data and control words are passed into a small synchronisation FIFO. The data in the FIFOs is read out by the lane concentrator in the order determined by the lane controller. The lane with lane number 1 has data read out of its synchronisation FIFO first, followed by the one with lane number 2, and so on until all the indicated lane numbers have been used. Data is not read out of any lane with a null lane number. Any delay on one or more physical lanes causing skew between lanes will be resolved by the ordering of lanes and the synchronisation FIFOs. If data is delayed on one lane the lane concentrator has to wait for that FIFO to have data ready, after going once round the lanes, data will be ready in each lane when the lane contractor wants to read from it. The maximum permitted skew between lanes determines the necessary depth of the synchronisation FIFO, which can be very small.

If lanes are not being used the lane concentrator can be bypassed using a multiplexer.

Data and control words from the lane concentrator are passed to the retry layer.

4.4.13 Error checking

The stream of data and control words are de-multiplexed in the retry layer so that each type of control word, data frame or broadcast frame can be dealt with appropriately.

ACKs and NACKs are stripped out of the data stream, checked and their contents (the frame/FCT number being ACKed or NACKed) is passed to the transmit frame retry buffer. ACKs result in frames/FCTs being deleted from the retry buffer, as they have arrived successfully. NACKs result in frames/FCTs being resent as they were not received correctly.

Data frames have their CRC checked. Any error results in the frame being discarded. If the CRC checksum is correct the frame sequence number is checked and should be one more than the last correctly received sequence number. If this is the case, the sequence checker informs the ACK/NACK controller to send an ACK and the data frame is passed to the framing layer. If the sequence number is not correct, the ACK/NACK controller is asked to send a NACK and the data frame is discarded. While the CRC and sequence number are being checked the data frame is temporarily stored in a data frame buffer. This permits it to be discarded if an error is detected, before the data frame is passed to the framing layer.

The FCT includes a frame sequence number and CRC checksum. The format of the FCT is checked and its frame sequence number and CRC validated. If the frame sequence number and CRC are correct an ACK is requested to be sent and the FCT is passed to the framing layer. If the frame sequence number or CRC are not correct a NACK is requested to be sent and the FCT is discarded.

Broadcast frames are processed in a similar way to data frames with their CRC and frame sequence number being checked.

Any idle frames have their sequence number checked. For an idle frame the sequence number should be the same value as the last received data frame, broadcast frame, or FCT. If this is not the case a NACK is requested to be sent, containing the frame sequence number of the last successfully received frame/FCT.

Following the start of idle frame (SIF) control word, the idle frame contains a PRBS which is checked for errors by the PRBS test function. If the link is idle it sends continuous idle frames which are checked continuously by the PRBS function giving an immediate indication of the status of the link and supporting eye pattern testing.

Correctly received data frames, FCTs and broadcast frames are passed up to the framing layer.

4.4.14 Frame stripping

The receive side of the framing layer, strips off the frame to expose the information contained in each frame.

Broadcast channel, broadcast sequence number and broadcast message data are extracted from each broadcast frame and passed up to the broadcast layer.

The FCT with its virtual channel number is passed to the virtual channel layer.

The data within a data frame is unscrambled and passed along with its virtual channel number to the virtual channel layer.

4.4.15 Broadcast message reception

The broadcast message needs to have its sequence number checked against the local broadcast counter for the relevant broadcast channel. It can then be signalled to a local application, or re-broadcast by a router.

4.4.16 Receiving data over virtual channels

The data from the data frame is placed in the appropriate virtual channel buffer as indicated by the virtual channel number in the data frame.

When 256 N-Chars or Nulls have been read out of an input virtual channel buffer, there is space for another frame to be received. This is indicated to the other end of the lane by sending an FCT containing the number of the virtual channel that has some more room. This is handled by the TX FCT controller.

Received FCTs are used by the RX FCT decoder to indicate to the medium access controller which virtual channels at the other end of the link have room for another frame.

Data from the input virtual channel buffers can be read out by the user application when required.

4.4.17 Loopback

Two loopback facilities are provided inside the CODEC for test purposes. The first provides loopback of the parallel data and control words and the second loops back the serial data before it is driven onto the physical medium. The serial loopback provides both local and remote loop back, *i.e.* serial data being sent is looped back into the receive de-serialiser and the bit stream data being received is fed back to the line driver.

4.4.18 Reset

There are two types of reset that can be applied to a SpaceFibre interface: cold reset and warm reset. Cold reset returns the device to its power on state. Warm reset is used after a persistent error has occurred to re-initialise a lane. Warm reset resets the lower layers of the SpaceFibre interface without resetting the upper layers. This permits rapid recovery of errors that may occur on a lane, in conjunction with the retry mechanism. The error recover is thus transparent to the higher layers of the SpaceFibre protocol and hence to the user applications.

5 Requirements

5.1 Overview

This section provides the normative requirements for SpaceFibre. It begins, in section 5.2 by specifying the services that the SpaceFibre CODEC provides. In section 5.3 the formats of data characters, symbols, words, control words, frames and packets are specified. The subsequent sections specify each of the functional layers of the SpaceFibre CODEC interface:

- Virtual channel layer (section 5.4)
- Broadcast message layer (section 5.5)
- Framing layer (section 5.6)
- Retry layer (section 5.7)
- Lane control layer (section 5.8)
- Lane layer (section 5.9)
- Encoding layer (section 5.10)
- Serialisation layer (section 5.11)
- Physical layer (section 5.12)
- Management layer (section 5.13

5.2 SpaceFibre CODEC service interface specification

5.2.1 SpaceWire packet service

The service primitives that shall be associated with the SpaceWire packet service are:

SEND_PACKET.request;
READ_PACKET.indication;

5.2.1.1 SEND_PACKET.request

5.2.1.1.1 Function

The SEND_PACKET.request primitive shall be used to send a SpaceWire packet through a virtual channel of a SpaceFibre link.

5.2.1.1.2 Semantics

The SEND_PACKET.request primitive shall provide the following parameters:

SEND_PACKET.request (Virtual Channel, SpaceWire Packet)

5.2.1.1.3 When Generated

When the user of the SpaceFibre CODEC has a packet to send over the SpaceFibre link, it shall generate a SEND_PACKET.request primitive to request to send the SpaceWire packet over a specific virtual channel of the SpaceFibre link.

5.2.1.1.4 Effect On Receipt

On receipt of the SEND_PACKET.request primitive the SpaceFibre CODEC shall send the SpaceWire packet over the specified virtual channel as soon as permitted by the SpaceFibre medium access controller.

5.2.1.2 READ_PACKET.indication

5.2.1.2.1 Function

The SpaceFibre CODEC shall pass a READ_PACKET.indication primitive to the Read SpaceWire Packet service user to indicate that a SpaceWire packet has arrived over a particular virtual channel and is waiting to be read.

5.2.1.2.2 Semantics

The READ_PACKET.indication primitive provides parameters as follows:

READ_PACKET.indication (Virtual Channel, Data).

5.2.1.2.3 When Generated

The READ_PACKET.indication primitive shall be passed to the Read Packet service user when a SpaceWire packet is received.

5.2.1.2.4 Effect On Receipt

The effect on receipt of the READ_PACKET.indication primitive by the Read Packet service user should result in that service user reading the received SpaceWire packet.

5.2.2 Broadcast message service

The service primitives that shall be associated with the broadcast message service are:

BROADCAST_MESSAGE.request;

BROADCAST_MESSAGE.indication;

BROADCAST_REGISTER.request;

BROADCAST_UNREGISTER.request.

5.2.2.1 BROADCAST_MESSAGE.request

5.2.2.1.1 Function

The BROADCAST_MESSAGE.request primitive shall be used to requests the SpaceFibre CODEC to send a broadcast message through a SpaceFibre broadcast channel.

5.2.2.1.2 Semantics

The BROADCAST_MESSAGE.request primitive shall provide the following parameters:

BROADCAST_MESSAGE.request (Broadcast Channel, Broadcast Type, Message).

5.2.2.1.3 When Generated

When the user of the broadcast message service has a broadcast message to send it shall generate a BROADCAST_MESSAGE.request primitive to request the SpaceFibre CODEC to send the broadcast message over a specific broadcast channel.

5.2.2.1.4 Effect On Receipt

On receipt of the BROADCAST_MESSAGE.request primitive the SpaceFibre CODEC shall send the broadcast message over the specified broadcast channel immediately, subject to link priority rules.

5.2.2.2 BROADCAST_MESSAGE.indication

5.2.2.2.1 Function

The function of the BROADCAST_MESSAGE.indication primitive shall be to indicate to a Broadcast Message service user application that a broadcast message has arrived over a particular broadcast channel and to pass that message to the service user. Only broadcast messages that have been registered will cause an indication (see section 5.2.2.3).

5.2.2.2.2 Semantics

The BROADCAST_MESSAGE.indication primitive provides parameters as follows:

BROADCAST_MESSAGE.indication (Application, Broadcast Channel, Broadcast Type, Message).

5.2.2.2.3 When Generated

The BROADCAST_MESSAGE.indication primitive shall be passed to the Broadcast Message service user application indentified by the Application parameter, when a valid broadcast message is received.

5.2.2.2.4 Effect On Receipt

The effect on receipt of the BROADCAST_MESSAGE.indication primitive by the Broadcast Message service user application shall be user defined.

5.2.2.3 BROADCAST_REGISTER.request

5.2.2.3.1 Function

The BROADCAST_REGISTER.request primitive shall be used to register to receive broadcast messages.

5.2.2.3.2 Semantics

The BROADCAST_REGISTER.request primitive shall provide the following parameters:

BROADCAST_REGISTER.request (Broadcast Channel, Broadcast Type, Application).

NOTE The Application parameter identifies the specific user application that is to be informed when a broadcast message with the particular broadcast channel or type is received.

NOTE The Broadcast Channel or Broadcast Type parameters are set to “All” to receive messages on all broadcast channels or all types of broadcast message.

5.2.2.3.3 When Generated

When an application wishes to receive broadcast messages from a specific broadcast channel, or of a specific type, or both, it shall first register to receive those messages by generating a BROADCAST_REGISTER.request primitive.

5.2.2.3.4 Effect On Receipt

On receipt of the BROADCAST_REGISTER.request primitive the SpaceFibre CODEC shall arrange for the application to be notified when a broadcast message arrives over the specified broadcast channel or with the appropriate type. This notification shall be performed using the BROADCAST_MESSAGE.indication.

5.2.2.4 BROADCAST_UNREGISTER.request

5.2.2.4.1 Function

The BROADCAST_UNREGISTER.request primitive shall be used to unregister an application from receive broadcast messages.

5.2.2.4.2 Semantics

The BROADCAST_UNREGISTER.request primitive shall provide the following parameters:

BROADCAST_UNREGISTER.request (Broadcast Channel, Broadcast Type, Application).

5.2.2.4.3 When Generated

When an application wishes to stop receiving broadcast messages from a specific broadcast channel, or of a specific type, or both, it shall generate a BROADCAST_UNREGISTER.request primitive.

5.2.2.4.4 Effect On Receipt

On receipt of the BROADCAST_UNREGISTER.request primitive the SpaceFibre CODEC shall remove the corresponding entry in the broadcast notification register to stop the application being notified of specific broadcast messages.

5.2.3 Link management service

The service primitives that shall be associated with the link status service are:

LANE_STATUS.indication.

5.2.3.1 LANE_STATUS.indication

5.2.3.1.1 Function

The function of the LANE_STATUS.indication primitive shall be to indicate that the status of the SpaceFibre link has changed.

5.2.3.1.2 Semantics

a. The LANE_STATUS.indication primitive shall provide parameters as follows:

   LANE_STATUS.indication(Status).

b. The Status parameter shall contain one of the following status types:

   1. TBD

5.2.3.1.3 When Generated

The LANE_STATUS.indication primitive shall be generated when the status of the SpaceFibre link changes.

5.2.3.1.4 Effect On Receipt

The effect on receipt of the LANE_STATUS.indication primitive by the SpaceFibre CODEC user shall be user defined.

5.3 Formats

In this section the formats of control words and frames are specified.

5.3.1 Control word format

Several types of control word shall be used by SpaceFibre:

- Lane control words
- Lane synchronisation control words
- Retry control words
- Framing control words
- Flow control words
- Receive error indication control words

5.3.1.1 Lane control words

a. The lane control words shall be used to initialise a SpaceFibre lane, to indicate loss of signal, and to indicate that a lane is about to go into standby.

NOTE The lane control words are constructed as shown in Table 5-1.

b. The comma shall be in the least significant symbol position and shall be sent first.

Table 5-1: Lane control words

Name	Control word	Function
SKIP	Comma, LLCW, SKIP, SKIP K28.5, D3.1, D6.1, D6.1	Sent every N control words or data words to support the receiver elastic buffer operation and skip-tick indication. N must be less than or equal to 5000. D6.1 has even disparity.
inverse SKIP	Comma, iLLCW, iSKIP, iSKIP K28.5, D28.6, D25.6, D25.6	The inverse SKIP is received when the receive polarity is inverted. Inverse SKIPs should be detected and treated as normal SKIPs. This allows the receive elastic buffer to operate even when the receive polarity is incorrect. D25.6 has even disparity.
IDLE	Comma, LLCW, IDLE, IDLE K28.5, D3.1, D6.2, D6.2	Sent during initialisation and whenever there is no data frame, idle frame or other control word to send. It keeps the lane active. D6.2 has even disparity.
inverse IDLE	Comma, iLLCW, iIDLE, iIDLE K28.5, D28.6, D25.5, D25.5	The inverse IDLE is received when the receive polarity is incorrect. If any IDLEs are inserted during lane initialisation with incorrect receive polarity, this will prevent the lane from detecting an error. Inverse IDLEs should be treated the same as normal IDLEs. D25.5 has even disparity.
INIT	Comma, LLCW, INIT, INIT K28.5, D3.1, D6.5, D6.5	Send as part of the initialisation handshake. D6.5 has even disparity.
inverse INIT	Comma, iLLCW, iINIT, iINIT K28.5, D28.6, D25.2, D25.2	Sent as part of the initialisation handshake if the signals are inverted.
IACK	Comma, LLCW, IACK, Capability	Send as part of the

	K28.5, D3.1, D6.6, D0.0-D31.7	initialisation handshake. The capability field describes the capability of the end of the lane sending the IACK. This can be used to exchange information about the capability of the SpaceFibre interface at the other end of the lane, so that the two ends of the lane can operate in the most efficient way possible.
RCLR	Comma, LLCW, RCLR, RCLR K28.5, D3.1, D9.1, D9.1	Recovery Clear. Sent by one end of the link to signal that it wants to recover from an error.
RACK	Comma, LLCW, RACK, RACK K28.5, D3.1, D9.2, D9.2	Recovery Acknowledgement. Sent to acknowledge reception of an RCLR control word.
STAND BY	Comma, LLCW, STBY, STBY K28.5, D3.1, D9.5, D9.5	Indicates that transmitter is moving to the Standby state and will tri-state its driver.
LOS	Comma, LLCW, LoS, LoS K28.5, D3.1, D9.6, D9.6	Indicates that the end of the link sending the LOST_SIGNAL control word has lost signal on its receiver.

NOTE The values and meanings of the Capability field in IACK is TBD.

5.3.1.1.2 Skip control word

a. The Skip control word (SKIP) shall be used to support operation of the receive elastic buffer in the SpaceFibre receiver.

b. The Skip control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Skip control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Skip control word shall identify the lane layer control word as being a Skip control word, and has the value D6.1.

e. The fourth and final symbol in the Skip control word shall be a copy of the third symbol.

5.3.1.1.3 Inverse skip control word

a. The Inverse Skip control word (iSKIP) shall contain the bit-wise inverse symbols of the Skip control word.

NOTE The symbols making up a Skip control word when inverted all form valid symbols.

NOTE The symbols making up a Skip control word form an inverse Skip control word when inverted

b. The Inverse Skip control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

NOTE The inverse of a comma (K28.5) is also a comma (K28.5), with the opposite disparity.

c. The second symbol in the Inverse Skip control word shall be the Inverse Lane Layer Control Word (iLLCW) identifier, which has the value D28.6 and identifies the control word as being the inverse of a control word generated and used by the lane layer.

d. The third symbol in the Inverse Skip control word shall identify the Lane Layer Control Word as being an Inverse Skip control word, and has the value D25.6.

e. The fourth and final symbol in the Inverse Skip control word shall be a copy of the third symbol.

NOTE The symbols making up an Inverse Skip control word form a Skip control word when inverted

f. The Inverse SKIP shall not be generated by the SpaceFibre CODEC.

NOTE The Inverse Skip is formed when the PCB layout in a SpaceFibre transmitter or receiver crosses over the two signals (CML+ and CML-) making up the differential signal.

5.3.1.1.4 Idle control word

a. The Idle control word (IDLE) shall be sent during initialisation and subsequently to keep the lane running when there is no other information to send.

b. The Idle control word (IDLE) shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Idle control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Idle control word shall identify the Lane Layer Control Word as being a Idle control word, and has the value D6.2.

e. The fourth and final symbol in the Idle control word shall be a copy of the third symbol.

5.3.1.1.5 Inverse idle control word

a. The Inverse Idle control word (iIDLE) shall contain the bit-wise inverse symbols of the Idle control word.

NOTE The symbols making up an Idle control word when inverted all form valid symbols.

NOTE The symbols making up a Idle control word form an inverse Idle control word when inverted

b. The Inverse Idle control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Inverse Idle control word shall be the Inverse Lane Layer Control Word (iLLCW) identifier, which has the value D28.6 and identifies the control word as being the inverse of a control word generated and used by the lane layer.

d. The third symbol in the Inverse Idle control word shall identify the Lane Layer Control Word as being an Inverse Idle control word, and has the value D25.5.

e. The fourth and final symbol in the Inverse Idle control word shall be a copy of the third symbol.

NOTE The symbols making up an inverse Idle control word form an Idle control word when inverted

f. The Inverse Idle shall not be generated by the SpaceFibre CODEC.

NOTE The Inverse Idle is formed when the PCB layout in a SpaceFibre transmitter or receiver crosses over the two signals (CML+ and CML-) making up the differential signal.

5.3.1.1.6 INIT control word

a. The INIT control word, used during lane initialisation, shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

b. The second symbol in the INIT control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

c. The third symbol in the INIT control word shall identify the Lane Layer Control Word as being an INIT control word, and has the value D6.5.

d. The fourth and final symbol in the INIT control word shall be a copy of the third symbol.

5.3.1.1.7 Inverse INIT control word

a. The Inverse INIT control word (iINIT) shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

b. The second symbol in the Inverse INIT control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D28.6 and identifies the control word as being a control word generated and used by the lane layer.

c. The third symbol in the Inverse INIT control word shall identify the Lane Layer Control Word as being an Inverse INIT control word, and has the value D25.2.

d. The fourth and final symbol in the Inverse INIT control word shall be a copy of the third symbol.

e. The Inverse INIT control word shall not be generated by the SpaceFibre CODEC.

NOTE The Inverse INIT is formed when the PCB layout in a SpaceFibre transmitter or receiver crosses over the two signals (CML+ and CML-) making up the differential signal.

5.3.1.1.8 IACK control word

a. The IACK control word, used during lane initialisation, shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

b. The second symbol in the IACK control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

c. The third symbol in the IACK control word shall identify the Lane Layer Control Word as being an IACK control word, and has the value D6.6.

d. The fourth and final symbol in the IACK control word (Capability) shall contain information about the capability of the lane, and be a data symbol with any value from D0.0 to D31.7.

NOTE It is not necessary for the Capability field to have valid inverse symbols, since by the time IACKs are being sent any necessary receiver inversion will have been completed.

e. The encoding of the Capability field is TBA.

5.3.1.1.9 Recovery Clear control word

a. The Recovery Clear control word (RCLR) shall be used to inform the far end of a lane that an error has been detected at the end of the link sending the RCLR control word .

NOTE The RLCR, RACK handshake clears the line of any data or control words except RCLR, RACK and IDLE, while keeping bit synchronisation of the receivers at both ends of the lane.

b. The Recovery Clear control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Recovery Clear control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Recovery Clear control word shall identify the Lane Layer Control Word as being a Recovery Clear control word, and has the value D9.1.

e. The fourth and final symbol in the Recovery Clear control word shall be a copy of the third symbol.

5.3.1.1.10 Recovery Acknowledge control word

a. The Recovery Acknowledge control word (RACK) shall be used to inform the far end of a lane that an RCLR control word has been received.

NOTE The RLCR, RACK handshake clears the line of any data or control words except RCLR, RACK and IDLE, while keeping bit synchronisation of the receivers at both ends of the lane.

b. The Recovery Acknowledge control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Recovery Acknowledge control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Recovery Acknowledge control word shall identify the Lane Layer Control Word as being a Recovery Acknowledge control word, and has the value D9.2.

e. The fourth and final symbol in the Recovery Acknowledge control word shall be a copy of the third symbol.

5.3.1.1.11 Standby control word

a. The Standby control word (STANDBY) shall be used to inform the far end of a lane that the SpaceFibre interface is about to go into standby mode with its line driver turned off.

b. The Standby control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Standby control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Standby control word shall identify the Lane Layer Control Word as being a Standby control word, and has the value D9.5.

e. The fourth and final symbol in the Standby control word shall be a copy of the third symbol.

5.3.1.1.12 Loss of signal control word

a. The Loss of Signal control word (LOS) shall be used to inform the far end of a lane that the local receiver is not receiving a signal.

b. The Loss of Signal control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the Loss of Signal control word shall be the Lane Layer Control Word (LLCW) identifier, which has the value D3.1 and identifies the control word as being a control word generated and used by the lane layer.

d. The third symbol in the Loss of Signal control word shall identify the Lane Layer Control Word as being a Loss of Signal control word, and has the value D9.6.

e. The fourth and final symbol in the Loss of Signal control word shall be a copy of the third symbol.

f. The Loss of Signal control word shall not be sent when the receiver has received a Standby control word signalling that the other end of the link is about to enter standby mode and turn off its line driver.

5.3.1.2 Lane synchronisation control words

a. Lane synchronisation control words shall be used to synchronise words flowing over multiple lanes.

NOTE Lane synchronisation control words are constructed as illustrated in Table 5-2.

Table 5-2: Lane Synchronisation Control words

Name	Control word	Function
LSYNC	Comma, LSYNC, LANE#, Reserved K28.5, D3.2, D0.0-D10.0, D0.0	Lane Synchronisation. Contains lane number, which indicates the order in which words are to be read from each lane, starting with lane number 1. A lane number of zero (null) indicates that although the lane is running it is not to be used as an active lane. Data will not be sent over a null lane and the lane concentrator will not read data from a null lane.

a. The Lane Sync control word (LSYNC) shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

b. The second symbol in the Lane Sync control word shall identify the control word as being a Lane Sync control word, and has the value D3.2.

c. The third symbol in the Lane Sync control word shall contain lane number, which indicates the order in which words are to be read from each lane, and shall be a data symbol in the range D0.0 to D10.0 (TBC).

d. The lane number D0.0 is the null lane number and indicates that this lane is not being used.

e. The lane numbers D1.0 to D10.0 (TBC) shall indicate the order in which data is to be extracted from each lane, starting with the lowest number lane first.

f. The fourth and final symbol in the Lane Sync control word shall be reserved and set to D0.0.

5.3.1.3 Retry control words

a. Retry control words shall be used to acknowledge data frames, broadcast frames and FCTs that are received correctly and to negatively acknowledge those that are received incorrectly.

NOTE The retry control words are constructed as illustrated in Table 5-3Table 5-5.

Table 5-3: Retry Control word

Name	Control word	Function
ACK	Comma, ACK, FR_SEQ#, CRC	Frame Acknowledge.

	K28.5, D3.5, D0.0-D31.7, D0.0-D31.7	Indicates that a data frame, broadcast frame or FCT has been received without error and in order. Sequence number is the frame sequence number (FR_SEQ#) from the data frame, broadcast frame or FCT that is being acknowledged. CRC is an 8-bit CRC that is used to confirm the integrity of the ACK.
NACK	Comma, NACK, FR_SEQ#, CRC K28.5, D3.6, D0.0-D31.7, D0.0-D31.7	Frame Negative Acknowledge. Indicates that a data frame, broadcast frame or FCT has not been received correctly. Sequence number is the frame sequence number (FR_SEQ#) of the last correctly received data frame, FCT, or broadcast frame. CRC is an 8-bit CRC that is used to confirm the integrity of the ACK.
RETRY	Comma, RETRY, FR_SEQ#, CRC K28.5, D5.1, D0.0-D31.7, D0.0-D31.7	Retry indication. Indicates to the far end of a link that a NACK has been received and the contents of the retry buffer is about to be transmitted. Sequence number is a copy of the frame sequence number (FR_SEQ#) in the NACK i.e. the last data frame, FCT, or broadcast frame correctly received at the far end of the link. CRC is an 8-bit CRC that is used to confirm the integrity of the RETRY.

5.3.1.3.2 ACK control word

a. The Acknowledgement control word (ACK), shall be used to indicate that a data frame, FCT, or broadcast frame has been received without error and in the correct order.

b. The ACK control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the ACK control word shall identify the control words as being an ACK control word, and has the value D3.5.

d. The third symbol in the ACK control word shall contain a frame sequence number (FR_SEQ#), which is the frame sequence number of the data frame, FCT, or broadcast frame that is being acknowledged, i.e. that has been correctly received at the far end of the link.

e. The fourth and final symbol in the ACK control word shall contain an 8-bit CRC covering all the symbols in the ACK control word, which is used to confirm the integrity of the ACK when received before its contents are acted upon.

NOTE The CRC includes the data part of the comma K-code.

5.3.1.3.3 NACK control word

a. The Negative Acknowledgement control word (NACK), shall be used to indicate that a data frame, FCT, or broadcast frame has not been received correctly.

b. The NACK control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the NACK control word shall identify the control words as being a NACK control word, and has the value D3.6.

d. The third symbol in the NACK control word shall contain a frame sequence number (FR_SEQ#), which is the frame sequence number of the last correctly received data frame, FCT, or broadcast frame.

NOTE All data frames, FCTs, and broadcast frames that have already been sent following that indicated in the NACK will be resent.

e. The fourth and final symbol in the NACK control word shall contain an 8-bit CRC covering all the symbols in the NACK control word, which is used to confirm the integrity of the NACK when received before its contents are acted upon.

NOTE The CRC includes the data part of the comma K-code.

5.3.1.3.4 RETRY control word

a. The Retry control word (RETRY), shall be used to indicate that the contents of the retry buffer is about to be sent.

b. The RETRY control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the RETRY control word shall identify the control words as being a RETRY control word, and has the value D5.1.

d. The third symbol in the RETRY control word shall contain a frame sequence number (FR_SEQ#), which is the frame sequence number contained in the NACK that caused the RETRY control word and contents of the retry buffer to be sent.

NOTE All data frames, FCTs, and broadcast frames that have already been sent following that indicated in the NACK will be resent.

e. The fourth and final symbol in the RETRY control word shall contain an 8-bit CRC covering all the symbols in the RETRY control word, which is used to confirm the integrity of the RETRY when received before its contents are acted upon.

NOTE The CRC includes the data part of the comma K-code.

5.3.1.4 Framing control words

a. Framing control words shall be used to encapsulate the data frames, broadcast frames, and idle frames being set across the link.

NOTE Framing control words are illustrated in Table 5-4.

Table 5-4: Data Framing Control words

Name	Control word	Function
SDF	Comma, SDF, VC, Reserved K28.5, D5.2, D0.0-D31.7, D0.0	Start of Data Frame. Contains type of frame and virtual channel, number.
SBF	Comma, SBF, BC, BC_SEQ# K28.5, D5.5, D0.0-D31.7, D0.0-D31.7	Start of Broadcast Frame.
SIF	Comma, SIF, FR_SEQ#, Reserved K28.5, D5.6, D0.0-D31.7, D0.0	Start of Idle Frame. Contains type of frame, and the FR_SEQ# of the last data frame, broadcast frame, or FCT sent. Note there is no end of idle frame control word.
EDF	K28.3, FR_SEQ#, CRC_LS, CRC_MS K28.3, D0.0-D31.7, D0.0-D31.7, D0.0-D31.7	End of Data Frame. Contains the frame sequence number and 16-bit CRC for the frame. Note that the EDF starts with K28.3 which is not a comma. This code differentiates all other control words from the EDF control word. Note the sequence number is over the link NOT per VC.
EBF	K28.1, Reserved, FR_SEQ#, CRC K28.1, D0.0, D0.0-D31.7, D0.0-D31.7	End of Broadcast Frame. Contains the frame sequence number and the 8-bit CRC for the frame. Note that the EBF starts with K28.1 which is not a comma. This code differentiates all other control words from the EBF control word. Note the sequence number is over the link NOT per VC.

5.3.1.4.2 Start of data frame control word

a. The Start of Data Frame control word (SDF), shall be used to indicate the start of a data frame.

b. The SDF control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the SDF control word shall identify the control words as being an SDF control word, and has the value D5.2.

d. The third symbol in the SDF control word shall contain the virtual channel number that this data frame is travelling over.

e. The fourth and final symbol in the SDF control word is reserved and shall be set to D0.0.

5.3.1.4.3 Start of broadcast frame control word

a. The Start of Broadcast Frame control word (SBF), shall be used to indicate the start of a broadcast frame.

b. The SBF control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the SBF control word shall identify the control words as being an SBF control word, and has the value D5.5.

d. The third symbol in the SBF control word shall contain the broadcast channel number that this broadcast frame is travelling over.

e. The fourth and final symbol in the SBF control word shall contain the broadcast sequence number (BC_SEQ#) for the broadcast channel.

NOTE The broadcast sequence number is used to support the broadcasting of the broadcast frame by SpaceFibre routers, in a similar way to the time-code value in SpaceWire supports the broadcast of time-codes.

5.3.1.4.4 Start of idle frame control word

a. The Start of Idle Frame control word (SIF), shall be used to indicate the start of a idle frame.

b. The SIF control word shall begin with a comma (K28.5), which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the SIF control word shall identify the control words as being an SIF control word, and has the value D5.6.

d. The third symbol in the SIF control word shall contain the frame sequence number of the last data frame, FCT or broadcast frame sent over the SpaceFibre link.

e. The fourth and final symbol in the SIF control word shall be reserved and contain the value D0.0.

NOTE There is no end of idle frame control word. Idle frames are ended by a SDF, SBF or SIF.

5.3.1.4.5 End of data frame control word

a. The End of Data Frame control word (EDF), shall be used to indicate the end of a data frame.

b. The EDF control word shall begin with the control code K28.3, which is in the least significant symbol position of the control word and is sent first.

NOTE K28.3 is not a comma.

c. The second symbol in the EDF control word shall contain the frame sequence number of the current data frame.

d. The third symbol of the EDF control word shall contain the least significant byte of a 16-bit CRC which covers the entire data frame including the SDF and EDF.

e. The fourth symbol of the EDF control word shall contain the most significant byte of a 16-bit CRC which covers the entire data frame including the SDF and EDF.

5.3.1.4.6 End of broadcast frame control word

a. The End of Broadcast Frame control word (EBF), shall be used to indicate the end of a broadcast frame.

b. The EBF control word shall begin with the control code K28.1, which is in the least significant symbol position of the control word and is sent first.

NOTE K28.1 is not a comma.

c. The second symbol in the EBF control word shall be reserved and contain the value D0.0.

d. The third symbol in the EBF control word shall contain the frame sequence number of the current broadcast frame.

e. The fourth symbol of the EBF control word shall contain an 8-bit CRC covering the entire broadcast frame including the SBF and EBF.

5.3.1.5 Flow control word

a. The Flow control word supports flow control across virtual channels.

NOTE The flow control word is illustrated in Table 5-5.

Table 5-5: Flow control word

Name	Control word	Function
FCT	FCT, Channel#., FR_SEQ#, CRC K28.6, D0.0-D31.7, D0.0-D31.7, D0.0-D31.7	Flow Control Token Indicates that the receive buffer for a specific virtual channel has room for another complete data frame. FCT is a K-code (K28.6) indicating that this control word is an FCT. Channel number identifies the virtual channel which this FCT is for. The FR_SEQ# is a frame sequence number added to the FCT by the retry layer to check for missing, duplicate or out of sequence data frame, broadcast frames and FCTs. CRC is an 8-bit CRC used to ensure that the FCT does not contain any errors.

a. The Flow Control Token control word (FCT), shall be used to indicate that the receive buffer for the specified virtual channel has room for another complete data frame.

b. The FCT control word shall begin with the K28.6 K-code which is in the least significant symbol position of the control word and is sent first.

c. The second symbol in the FCT control word shall contain the virtual channel number that this FCT is for.

d. The third symbol in the FCT control word shall contain the frame sequence number of the current FCT.

NOTE The FCT shares the same frame sequence numbers as the data frames and broadcast frames.

e. The fourth and final symbol in the FCT control word shall contain an 8–bit CRC covering the entire FCT, which is used to check the integrity of the FCT when received, before it is acted upon.

5.3.1.6 Receive error indication control word

a. The receive error indication control word is used by the encoding layer to indicate to a higher layer that a disparity error or invalid code error or other form of error was detected in the received data stream.

NOTE The receive error indication control words are illustrated in Table 5-5.

Table 5-6: Receive error indication control word

Name	Control word	Function
RXERR	Error, Error, Error, Error K0.0, K0.0, K0.0, K0.0	Receive error indication Indicates that an error has been detected in the received data stream by the decoder. Error is an invalid K-code (K0.0) which is used to indicate a symbol in a word is in error. The RXERR control word is made up of four Error symbols. Any word containing one or more symbols in error will be replaced by the RXERR control word. The received data stream is replace by RXERR control words whenever the receive synchronisation state machine is not in the ready state.

b. The receive error indication control word (RXERR), shall be used to indicate that the received data or control word contained an error or is likely to have contained an error.

c. The receive error indication control word shall comprise four error symbols (K0.0).

d. Since the receive error indication control word contains invalid symbols it shall not be transmitted, and is only used in the receiver to indicate receive errors to higher layers.

5.3.2 SpaceWire Characters

5.3.2.1 SpaceWire N-Chars

a. SpaceWire data characters shall be directly represented by data symbols D0.0 to D31.7.

NOTE For example, SpaceWire data character 0x39 is represented by data symbol D25.1 (see annex A.2).

NOTE The SpaceWire N-Chars are constructed as illustrated in Table 5-7.

Table 5-7: SpaceWire N-Char Symbols

Name	Symbol	Function
Data	D0.0 to D31.7	Each SpaceWire character contains one byte of data. Data byte 0x00 is represented by symbol D0.0, data byte 0x01 by D01.0, and so on up to data byte 0xFF which is represented by D31.7.
EOP	K28.2	Represents a SpaceWire EOP. This can occur at any point in the data field of a data frame, indicating the end of a SpaceWire packet. The data byte following the EOP is the first byte of the next packet.
EEP	K28.0	Represents a SpaceWire EEP. This can occur at any point in the data field of a data frame, indicating that the SpaceWire packet has terminated with an error. The data byte following the EEP is the first byte of the next packet.

b. The SpaceWire EOP shall be represented by K28.2.

NOTE K28.2 is not a comma.

c. The SpaceWire EEP shall be represented by K28.0.

NOTE K28.0 is not a comma.

d. The EOP and EEP symbols shall be allowed to appear anywhere within the data field of a data frame.

e. The EOP and EEP symbols shall indicate the end of a SpaceWire packet.

f. The data symbol following an EOP or EEP symbol shall be the first data symbol of the subsequent SpaceWire packet.

5.3.2.2 SpaceWire Null

a. The SpaceWire Null shall be represented by K23.7

NOTE K23.7 is not a comma.

NOTE The SpaceWire Null is constructed as illustrated in Table 5-8.

NOTE This character is called a SpaceWire Null because it appears in the SpaceWire N-Char stream. It is actually used to fill out space at the end of a SpaceWire packet.

Table 5-8: SpaceWire Null Symbol

Name	Symbol	Function
Null	K23.7	Used when there is not a multiple of 4 N-Chars in an output VCB to be sent. For example, if there are three N-Chars in an output VCB and the last one is an EOP, it is important to send this tail end of the SpaceWire packet, without waiting for data from another packet to be added to the buffer. Since SpaceFibre sends data in words of four N-Chars each it is necessary to have a character that will fill out the space at the end of the words. A sequence of nulls will contain one, two or three nulls but never any more.

b. The Null symbol shall be allowed to appear anywhere within the data field of a data frame.

NOTE This can be used to support 32-bit alignment of the VCBs. An example showing several small packets in part of a frame, some 32-bit aligned and others packed into 32-bits, is provided in Figure 5-1, where D represents a data character, E an EOP, and N a Null.

D	D	D	D
D	E	N	N
N	N	D	D
D	D	D	D
E	N	N	N
D	D	E	D
D	D	D	D
E	N	N	N

Figure 5-1 Nulls in a virtual channel buffer

c. When the data available from the output virtual channel buffer to be put in the data frame is not a multiple of four N-Chars, Null symbols shall be added to the end of the data field to pad the last word out to a complete data word.

NOTE This will happen when there is one or more EOPs or EEPs in the output VCB. If following the last EOP or EEP added to the output VCB, no more data has been added, there might not be a multiple of 4 N-chars in the buffer. When the last data incorporating the EOP or EEP is read out of the output VCB the space following the EOP or EEP must be filled with Null characters to form a complete 4 N-Char data word.

d. Nulls shall be transported across the SpaceFibre link and placed in the input virtual channel buffer at the receiving end of the link.

5.3.3 Frame Format

a. A frame shall contain user data, from a virtual channel or broadcast channel, or idle data.

b. Three types of frame shall be supported: data frame, broadcast frame, idle frame.

5.3.3.1 Data frame

a. A data frame shall start with a start of data frame (SDF) control word.

NOTE The data frame is illustrated in Figure 5-2.

0 7	8 15	16 23	24 31
COMMA	SDF	VC	RESERVED
DATA 1 LS	DATA 1	DATA 1	DATA 1 MS
DATA 2 LS	DATA 2	DATA 2	DATA 2 MS
…	…	…	…
DATA N LS	DATA N	DATA N	DATA N MS
EDF	FR_SEQ#	CRC_LS	CRC_MS

Figure 5-2 Data Frame Format

b. A data frame shall end with an end of data frame (EDF) control word.

c. A data frame shall contain between one and 64 data words each (TBC).

d. Each data word shall contain four SpaceWire N-Chars.

e. The virtual channel (VC) field in the data frame shall identify the VC sending the data frame and into which it is to be received.

f. The reserved field in the start of data frame (SDF) is reserved and shall be set to D0.0.

g. The end of data frame shall contain a 16-bit CRC covering the SDF control word, the data in the data frame, and the EDF control word.

h. The frame sequence number (FR_SEQ#) in the end of frame shall contain the sequence number of the frame.

5.3.3.2 Idle frames

a. An idle frame shall start with an idle frame control word.

NOTE An idle frame is illustrated in Figure 5-3.

0 7	8 15	16 23	24 31
COMMA	SIF	FR_SEQ#	RESERVED
SEED LS	SEED	SEED	SEED MS
PRBS 1 LS	PRBS 1	PRBS 1	PRBS 1 MS
PRBS 2 LS	PRBS 2	PRBS 2	PRBS 2 MS
…	…	…	…
PRBS N LS	PRBS N	PRBS N	PRBS N MS

Figure 5-3 Idle Frame Format

b. An idle frame shall contain between zero and 64 data words.

c. An idle frame shall end with the start of a data frame, broadcast frame, or next idle frame.

NOTE There is no end of idle frame control word.

d. The PRBS in an idle frame shall contain a pseudo-random bit sequence (PRBS).

NOTE Sending a PRBS avoids an EMC emission peak when idles frames are being transmitted and enables a PRBS test to be made at the receiver.

e. The Seed field in the start of idle frame (SIF) shall be set to a pseudo-random number which is used to seed the PRBS words in the idle frame.

NOTE This permits a SpaceFibre receiver to decode and check the PRBS in the idle frame. This can be used to check the health of the physical layer, after lane initialisation.

f. The PRBS shall be generated using the following algorithm

1. TBD

g. An idle frame shall be terminated as soon as there is a broadcast frame, or data frame to send.

h. An idle frame shall be terminated in any case when 64 PRBS words have been sent.

NOTE This means that the length of the idle frame can contain zero to 64 idle words depending on when there are more broadcast or data frames to send.

i. An idle frame shall contain at least the start of idle frame (SIF).

5.3.3.3 Broadcast frame

a. A broadcast frame shall start with a start of broadcast frame (SBF) control word.

NOTE The broadcast frame is illustrated in Figure 5-4.

0 7	8 15	16 23	24 31
COMMA	SBF	BC	B_SEQ#/B_TYPE
DATA 1 MS	DATA 1	DATA 1	DATA 1 MS
DATA 2 LS	DATA 2	DATA 2	DATA 2 MS
EBF	Reserved	FR_SEQ#	CRC

Figure 5-4 Broadcast Frame Format

b. A broadcast frame shall end with an end of broadcast frame (EBF) control word.

c. A broadcast frame shall contain two data words each containing four data bytes.

d. The broadcast channel (BC) field in the SBF control word shall identify the broadcast channel transmitting and receiving the broadcast frame.

e. The B_SEQ#/B_TYPE field shall contain two sub-fields: a 3-bit Broadcast Sequence Number (B_SEQ#) field and a 5-bit Broadcast Type (B_TYPE) field.

f. The Broadcast Sequence Number (B_SEQ#) field in the SBF shall contain an incrementing sequence number specific to the broadcast channel.

NOTE Each BC has its own broadcast sequence number (B_SEQ#) which is used to support the broadcast of the broadcast frame across a SpaceFibre network.

g. The Broadcast Type (B_TYPE) field shall contain a broadcast type, which indicates the type of broadcast message and the meaning of the subsequent eight data bytes.

h. The end of broadcast frame shall contain a reserved field set to D0.0.

i. The frame sequence number (FR_SEQ#) in the end of broadcast frame shall contain the sequence number of the frame.

NOTE This sequence number is used to support retry.

j. The end of broadcast frame shall contain an 8-bit CRC covering the SBF control word, the data in the data frame, and the EBF control word.

5.3.4 Control word and frame precedence

a. During lane initialisation, prior to a link being established, only the following Lane Layered Control Words shall be sent: SKIP, INIT, IACK, RCLR, RACK, IDLE.

b. During lane initialisation the Lane Layered Control Words shall have the following precedence:

(1) SKIP, highest precedence

(2) INIT, iINIT, IACK, iIACK, RCLR or RACK

(3) IDLE, lowest precedence.

c. Once one or more lanes have been established to form a link, the control words and frames shall have the following precedence:

(1) SKIP

(2) RCLR, RACK

(3) LOST_SIGNAL

(4) Standby

(5) LSYNC

(6) Broadcast frame

(7) ACK/NACK

(8) FCT

(9) Data frame

(10) Idle frame

(11) IDLE, lowest precedence.

d. A SKIP control word shall be inserted within a data frame, broadcast frame or idle frame.

e. A LOST_SIGNAL control word shall be inserted within a broadcast, data or idle frame.

f. A Standby control word shall be inserted within a broadcast, data or idle frame.

g. An LSYNC control word shall be inserted within a broadcast, data or idle frame.

h. A broadcast frame shall be inserted within a data.

i. A broadcast frame shall end an idle frame.

j. An ACK control word shall be inserted within a data or idle frame.

k. A NACK control word shall be inserted within a data or idle frame.

l. An FCT control word shall be inserted within a data frame.

m. An FCT control word shall end an idle fame.

n. A data frame shall end an idle frame.

o. An idle frame shall be sent only when there are no data frames, broadcast frames or FCTs to send.

p. An IDLE control word shall be sent only when there are no other data or control words to send.

q. While the contents of the retry buffer are being resent (see section 5.7), new data frames, FCTs, or broadcast frames cannot be sent and have to wait for the retry buffer to finish resending.

5.3.5 K-code summary

The meaning of K-codes within SpaceFibre shall be as summarised in Table 5-9.

Table 5-9: Meaning of K-codes

K-code	Meaning	Disparity
K0.0	RXERR	-
K28.0	SpaceWire EEP	even
K28.1	EBF	+2 or -2
K28.2	SpaceWire EOP	+2 or -2
K28.3	EDF	+2 or -2
K28.4	Not used	even
K28.5	Comma	+2 or -2
K28.6	FCT	+2 or -2
K28.7	Not used	even
K23.7	SpaceWire Null	even
K27.7	Not used	even
K29.7	Not used	even
K30.7	Not used	even

5.3.1 Control word symbol summary

The data values assigned to particular control word symbols within SpaceFibre shall be as summarised in Table 5-9.

Table 5-10: Meaning of control word symbols

D-code	Meaning	Disparity
D3.1 / D28.6	LLCW/iLLCW	even
D3.2	LSYNC	even
D3.5	ACK	even
D3.6	NACK	even
D5.1	RETRY	even
D5.2	SDF	even
D5.5	SDF	even
D5.6	SIF	even
D6.1 / D25.6	SKIP	even
D6.2 / D25.5	IDLE	even
D6.5 / D25.2	INIT	even
D6.6	IACK	even
D9.1	RCLR	even
D9.2	RACK	even
D9.5	STANDBY	even
D9.6	LoS	even

5.4 Virtual channel layer

5.4.1 Virtual channel buffering

a. The application interface to the SpaceFibre CODEC shall comprise one or more output virtual channel buffers and the same number of input virtual channel buffers.

b. There shall be a maximum of 256 virtual channels.

c. There shall be an output virtual channel buffer and an input virtual channel buffer for each virtual channel.

d. A particular virtual channel shall be identified by its virtual channel number.

e. An output virtual channel buffer shall be large enough to hold at least 512 SpaceWire N-Chars.

NOTE This corresponds to two full SpaceFibre data frames.

f. To send a SpaceWire packet over a particular virtual channel the application shall write the N-Chars forming the SpaceWire packet into the virtual channel buffer, starting with the leading N-Chars of the SpaceWire packet.

g. If the output virtual channel buffer becomes full the application shall wait until there is more room in the output virtual channel buffer.

NOTE SpaceWire packets can be any length. Packets greater than the size of the output virtual channel buffer will not fit into the buffer and will have to be written in as space is made available in the buffer when data is read out and sent over the SpaceFibre link.

h. As soon as one SpaceWire packet has been written into the output virtual channel buffer, it shall be possible to start writing the next SpaceWire packet to be sent over that virtual channel, provided that there is room in the output virtual channel buffer for at least one more N-Char.

NOTE This means that it is possible to have many small packets waiting for transmission in the output virtual channel buffer.

i. SpaceWire packets shall be sent over a virtual channel in the order in which they are written into the output virtual channel buffer for that virtual channel.

j. An output virtual channel at one end of a SpaceFibre link shall be paired with the input virtual channel at the other end of the SpaceFibre link which has the same virtual channel number.

k. When SpaceWire packets are received they shall be placed in the input virtual channel buffer for the virtual channel that they were sent over.

l. An input virtual channel buffer shall be large enough to hold at least 512 SpaceWire N-Chars (TBC).

m. SpaceWire N-Chars shall appear in an input virtual channel buffer in the same order in which they were written into the corresponding output virtual channel buffer at the other end of the link.

n. When there are N-Chars available in an input virtual channel buffer, the host application shall be informed that there is data available to read from that input virtual channel buffer.

o. The host application can read data from the input virtual channel buffer whenever it wants to, provided that there is data available in the input channel buffer.

5.4.2 Segmentation

a. Each output virtual channel buffer shall keep track of the number of N-Chars written into it and the number read out.

b. When there are at least 256 N-Chars in the output virtual channel buffer, or it contains at least one EOP or EEP, that virtual channel buffer shall indicate that is has data ready to form a data frame to a medium access controller, which controls which output channel is to be permitted to next send a data frame over the SpaceFibre link.

c. When informed that it is now permitted to send a data frame over the SpaceFibre link, an output virtual channel buffer shall pass 256 N-Chars, to the medium access controller for sending over the SpaceFibre link.

d. If the output virtual channel buffer contains one or more EOPs or EEPs and less than 256 N-Chars, it shall send all the N-chars it contains to the medium access controller.

e. If the number of N-Chars being sent in a data frame is not a multiple of four N-Chars, one, two or three Nulls shall be added to pad out the last data word of the data frame.

f. When a data frame is received, the N-Chars it contains shall be placed in the appropriate input virtual channel buffer.

5.4.3 Flow control

5.4.3.1 TX FCT Control

a. Each input virtual channel buffer shall keep track of the number of N-Chars and Nulls written into it from the SpaceFibre receiver, and read out using an input space counter, for that input virtual channel buffer.

NOTE If a frame contains an EOP or EEP it is possible that as well as N-Chars it contains some Nulls which are used to pad out the number of N-Chars to a multiple of four. At the link level these Nulls are treated in the same way as N-Chars and are transferred across the link and placed in the input virtual channel buffer at the far end of the link.

b. On cold-reset the input space counter shall be set to the amount of space in the buffer, counted as the number of N-Chars and Nulls the buffer can contain.

c. On warm-reset, the input space counter shall not be modified.

d. If the value of the input space counter is greater than or equal to 256 the input VC buffer shall request the transmitter to send an FCT.

NOTE After cold reset this can result in several FCTs being sent for a virtual channel if its input buffer can hold more than 256 N-Chars or Nulls.

e. For each FCT sent by a specific input virtual channel, 256 shall be subtracted from the corresponding input space counter to reserve that much space for the N-Chars and Nulls that the FCT will permit to be sent from the far end of the link.

NOTE An FCT is effectively exchanged for 256 N-Chars and Nulls.

f. Each time the end user application reads out an N-Char or Null from the input VC buffer, the input space counter shall be incremented by one.

g. The input space counter shall indicate the amount of space available in the input virtual channel buffer.

h. When FCTs are being requested to be sent for several different virtual channel buffers, they shall be arbitrated fairly.

5.4.3.2 RX FCT Control

a. Each output virtual channel buffer shall contain a FCT credit counter which shall indicate how much more data it is permitted to send.

b. On cold-reset, the FCT credit counter shall be set to zero.

c. On warm-reset, the FCT credit counter shall not be modified.

d. When an FCT is received for a particular virtual channel, 256 shall be added to the FCT credit counter for that virtual channel.

e. When a data frame is sent by a particular virtual channel, the number of N-Chars and Nulls sent in the data frame shall be subtracted from the FCT credit counter.

f. A virtual channel shall not be permitted to send data frames when its FCT credit counter is less than 256, unless its output VCB contains an EOP or EEP, in which case it can send N-Chars and Nulls up to the limit indicated by the FCT transmit credit counter for that virtual channel.

g. When a data frame is ready to send, the amount of data in the frame shall be compared to the value of the FCT credit counter and if the FCT credit counter is greater than or equal to the amount of data in the frame waiting to be sent, that data frame shall be sent.

5.4.4 Medium access control

a. A medium access controller shall determine which virtual channel shall be allowed to next send a data frame.

b. Only output virtual channel buffers indicating that they have data ready to form a data frame, shall be permitted to compete for sending the next data frame.

c. Only virtual channels whose FCT credit counter indicates that its input virtual channel buffer at the far end of the SpaceFibre link has room to accept the amount of data to be sent in the next data frame shall be permitted to compete for sending the next data frame.

d. Each output virtual channel buffer, with data ready to send in its output virtual channel buffer and space available in its input virtual channel buffer at the far end of the SpaceFibre link (a ready virtual channel), shall compete for sending the next data frame.

5.4.4.1 Precedence

a. The medium access controller shall use the precedence of each virtual channel to determine which ready virtual channel is permitted to send a data frame.

b. The precedence of each virtual channel shall be compared when the last word of the previous frame is being passed to the framing layer for transmission.

c. The ready virtual channel buffer that has highest precedence shall be permitted to send the next data frame.

d. The precedence of a virtual channel shall be determined by its quality of service parameters.

e. Precedence shall be calculated for the different qualities of service as outlined in Table 5-11 and detailed in the following clauses.

Table 5-11 Precedence for Different Qualities of Service

Quality of Service	**Precedence**
Best Effort	Not permitted to send data when a virtual channel supporting any other quality of service has data to send. If other best effort virtual channels have data to send, selection of which one is allowed to send data is made based on their expected use of link bandwidth and their recent use of link bandwidth. The virtual channel with high expected link bandwidth and low recent bandwidth utilisation will be permitted to send its data first. The combination of these two factors is referred to as bandwidth credit.
Priority	Fixed precedence levels corresponding to the various priority levels. A high priority virtual channel has a high precedence.
Bandwidth Reserved	Precedence depends up the reserved bandwidth for the virtual channel and its recent bandwidth utilisation. A virtual channel with large reserved bandwidth and low recent bandwidth utilisation will have high precedence. When a data frame is send by this virtual channel its precedence will drop. Its precedence will increase again over a period of time.
Scheduled	Time is separated into a series of time-slots during which a virtual channel can be scheduled to send data. When a time-slot arrives in which a virtual channel is scheduled to send data its precedence is set to the highest possible value, so that the virtual channel will be able to send one or more data frames straightaway. During all the other time-slots when the virtual channel is not scheduled to send data, it is not permitted to send any data even when no other virtual channel has data to send. If a scheduled virtual channel does not have data to send during its time-slot or does not have space in its VCBs at the far end of the link, other virtual channels (not scheduled ones) are permitted to use the otherwise

	wasted bandwidth.

5.4.4.2 Bandwidth credit

a. A virtual channel shall specify a portion of overall Link Bandwidth that it wishes or expects to use i.e. its Expected Bandwidth.

NOTE This information is to be provided for all virtual channels regardless of the quality of service they are providing.

b. For a virtual channel with Bandwidth Reserved quality of service the Expected Bandwidth shall be the Reserved Bandwidth.

c. Last Frame Bandwidth shall be the amount of data sent in the last data frame.

d. Used Bandwidth shall be the amount of data sent by a particular virtual channel in the last data frame.

NOTE This is zero except for all virtual channels except for the one that sent the last frame.

e. Bandwidth Allowance shall be calculated as follows:

$$BandwidthAllowance = ExpectedBandwidth \times LastFrameBandwidth$$

f. Bandwidth Credit is the accumulated Bandwidth Allowance less the Bandwidth Used.

g. Bandwidth Credit shall be calculated for each virtual channel as follows:

$$BandwidthCredit = \sum_{Frames} \frac{BandwidthAllowance - UsedBandwidth}{ExpectedBandwidth}$$

NOTE A value close to zero indicates nominal use of bandwidth by the virtual channel.

h. Bandwidth Credit shall be updated every time a data frame for any virtual channel has been sent.

i. Bandwidth Credit shall NOT be updated when no data has been send by any virtual channel.

NOTE This prevents all virtual channels reaching saturation of BandWidth Credit when there is no data being sent.

j. Bandwidth Credit shall be permitted to go negative.

NOTE A negative value indicates that the virtual channel is using more than its expected amount of link bandwidth.

NOTE A positive value indicates that the virtual channel is using less than its expected amount of link bandwidth.

k. Bandwidth Credit shall saturate at plus or minus the Bandwidth Credit Limit, i.e. if the Bandwidth Credit reaches a Bandwidth Credit Limit it is set to the value of the Bandwidth Credit Limit.

l. The Bandwidth Credit Limit shall be set to:

$$BandwidthCreditLimit = LinkBandwidth$$

NOTE This leads to a virtual channel with the full link bandwidth allocated to it reaching positive saturation saturating after 1 second when no data is being sent by that virtual channel. The virtual channel effectively starts to forget what it has send or not sent previously after this period of time.

m. When the Bandwidth Credit for a virtual channel reaches the positive Bandwidth Credit Limit it shall indicate in a status register that the virtual channel is using less bandwidth than expected.

NOTE A network management application is able to use this information to check correct utilisation of link bandwidth by its various virtual channels.

n. When the Bandwidth Credit for a virtual channel reaches the negative Bandwidth Credit Limit it shall indicate in a status register that the virtual channel is using more bandwidth than expected.

o. Bandwidth Credit shall be set to zero on cold reset.

p. Bandwidth Credit shall not be altered on warm reset.

5.4.4.3 Quality of service

a. Each virtual channel shall support the following qualities of service: best effort, priority, bandwidth reservation, and scheduled.

b. It shall be possible to set the quality of service of each virtual channel individually so that different qualities of service can be applied to different virtual channels.

c. A virtual channel shall compete with other virtual channels for sending frames over the link based on the current precedence of the virtual channel.

d. Upon cold reset the quality of service for each virtual channel shall be set to best effort.

e. Upon cold reset the Expected Bandwidth for each virtual channel shall be set to 1/N, where N is the number of virtual channels.

5.4.4.4 Best effort

a. Best effort quality of service shall permit a frame of data to be sent when there is no data frame with a different quality of service ready to be sent.

b. The precedence of all virtual channels providing best effort quality of service shall be set to just less than the minus Bandwidth Credit Limit, whenever any other virtual channel is ready to send data.

c. When the only virtual channels ready to send data are those providing best effort quality of service, their precedence shall be set to their Bandwidth Credit value.

d. When the only virtual channels ready to send data are those providing best effort quality of service, the one with the highest bandwidth credit shall be permitted to send a data frame.

e. When the only virtual channels ready to send data are those providing best effort quality of service and their bandwidth credits are all the same, any one of them shall be selected to send a data frame.

5.4.4.5 Priority

a. The precedence of a virtual channel with priority quality of service shall be set directly by the priority parameter for that virtual channel.

b. More than one virtual channel can be set to the same priority level in which case the one with highest bandwidth credit will be allowed to go first.

c. When implementing priority quality of service the bandwidth credit of the virtual channel shall be used to detect and indicate excessive use of link bandwidth by that virtual channel.

d. A virtual channel with priority quality of service shall be allocated a percentage of link bandwidth, which will be used to determine excessive or under utilisation of the link by that virtual channel.

e. There shall be fifteen priority levels whose precedence shall be equally spaced from the minus bandwidth credit limit to the plus bandwidth credit limit.

f. Priority 1 shall be the priority level with precedence just greater than the plus bandwidth credit limit.

g. Priority 7 shall be the priority level with precedence to zero.

h. Priority 15 shall be the priority level with precedence just less than the minus bandwidth credit limit.

5.4.4.6 Bandwidth reservation

a. The precedence of a virtual channel with bandwidth reserved quality of service shall be determined directly by the bandwidth credit of that virtual channel.

b. A virtual channel using bandwidth reservation shall compete with other virtual channels using bandwidth reservation or any other quality of service, based on its precedence.

5.4.4.7 Scheduled

a. The precedence of a virtual channel with scheduled quality of service shall be determined by the current time-slot and the schedule.

b. Time shall be split into a number of time-slots of equal duration specified by a time-slot duration

c. Time for use in the scheduled quality of service shall be taken from the local time register, which is regularly updated by the time-distribution broadcast channels.

d. The local time register shall provide an indication of when one time-slot finishes and the next one starts, together with the number of that time-slot.

e. The number of time-slots in a schedule shall be specified by a management parameter, see section 5.13

f. The schedule shall indicate in which time-slots a particular virtual channel is permitted to send data frames.

g. When a time-slot starts, for which a specific virtual channel is scheduled to send data, has data ready to send in its output virtual channel buffer and has space in its input virtual channel buffer at the far end of the SpaceFibre link, the virtual channel shall be given a precedence which is greater than that possible for any other quality of service, except for that of priority level 1 (extremely urgent priority), so that it sends its data immediately.

h. At the end of the time-slot, any virtual channel sending data frames shall cease sending them after the current data frame has been sent, unless that particular virtual channel buffer is also scheduled to send data in the next time-slot.

NOTE Alternatively the data frame being transmitted when the end of time-slot arrives could have an EDF inserted automatically.

i. The scheduling of traffic from virtual channels over the SpaceFibre link, shall take into account the fact that at the start of a time-slot, there might still be a complete data frame to send over the link from the previous time-slot.

j. If in a time-slot there is no data transfer specified in the schedule, or all the data that can be sent for the scheduled virtual channel has been sent, other non-scheduled virtual channels shall be permitted to send data frames.

k. If two virtual channels are both scheduled to send data in the same time-slot, the medium access controller shall send data from the virtual channel with the highest bandwidth credit first.

5.5 Broadcast message layer

5.5.1 Registering for broadcast services

a. An application shall be able to register to receive specific broadcast messages, specified by their broadcast channel, broadcast type or both.

NOTE For example, an application could register to receive all broadcast messages arriving over broadcast channel 56, or all broadcast messages of type "time", or all broadcast messages arriving over broadcast channel 56 which are of type "time".

b. When a broadcast message is received, an application registered to receive that specific broadcast message shall receive an indication, which contains the information received in the broadcast message.

c. An application shall be able to unregister from receiving specific broadcast messages.

d. On cold-reset no application shall be registered to receive broadcast messages.

e. On warm-reset the broadcast messages that an application is registered to receive shall remain unchanged.

5.5.2 Broadcast sequence number

a. The SpaceFibre interface shall be responsible for generating the broadcast sequence numbers for each broadcast channel.

b. A 3-bit broadcast counter shall be provided for each broadcast channel supported by the SpaceFibre interface.

c. In a SpaceFibre router one set of broadcast counters shall serve all the SpaceFibre interfaces in the router, rather than there being one set for each interface.

NOTE This is to support the broadcast message distribution mechanism.

d. On cold reset each broadcast counter shall be set to zero.

e. After a cold-reset the first broadcast message to be sent over each channel shall contain the broadcast sequence number one.

f. On warm-reset each broadcast counter shall remain unchanged.

g. When a broadcast message is to be sent over a specific broadcast channel the broadcast counter for that broadcast channel shall be incremented and the new value placed in the broadcast sequence number field of the broadcast message.

5.5.3 Broadcast validation

a. A broadcast message shall be validated before being indicated to an application.

b. If the broadcast sequence number in the broadcast message is one more than the value contained in the broadcast counter for that broadcast channel, the broadcast message shall be considered valid.

c. If the broadcast sequence number in the broadcast message is two more than the value contained in the broadcast counter for that broadcast channel, the broadcast message shall be considered valid, and a flag set to indicate that a broadcast message was missing.

NOTE This can happen if a broadcast message goes missing for whatever reason.

d. After a cold reset the first broadcast message for each broadcast channel shall be considered valid regardless of its broadcast sequence number.

e. If a broadcast message is valid, its reception shall be indicated to any application registered to receive broadcast messages from that broadcast channel or of that broadcast type.

f. If a broadcast message is not valid, its receipt shall not be indicated to any application registered to receive broadcast messages from that broadcast channel or of that broadcast type.

g. After a broadcast message has been received for a particular broadcast channel whether valid or not, the broadcast counter for that broadcast channel shall be loaded with the broadcast sequence number from that broadcast message.

5.5.4 Time broadcast

a. Broadcast channels 0, 1, 2 and 3 (time broadcast channels) shall be reserved for broadcasting system time information.

NOTE Together the four time broadcast channels can be used to implement a robust time distribution mechanism.

b. When the type field of a broadcast message contains the value zero (time type), the eight data characters shall contain system time information.

c. The time broadcast message format shall be as defined in

0 7	8 15	16 23	24 31
COMMA	SBF	BC	B_SEQ#/Time
Reserved	Fine Time LS	Fine Time	Fine Time MS
Coarse Time LS	Coarse Time	Coarse Time	Coarse Time MS
EBF	Reserved	FR_SEQ#	CRC

Figure 5-5 Time Broadcast Message Format

d. The Broadcast Type (B_TYPE) field in the start of broadcast frame control word shall be set to 0b00000, to indicate it is a time broadcast message.

e. The first character of the data field shall be reserved and set to D0.0.

f. The second, third and fourth characters of the data field shall contain the three fine time bytes of a CCSDS Unsegmented time Code (CUC), which is the sub-second time.

g. The second data word in the data field shall contain the four coarse time bytes of a CCSDS Unsegmented time Code (CUC), which is the time in seconds.

h. The CCSDS Unsegmented time Code (CUC) format shall be as defined in CCSDS 301.0-B-3, Time Code Formats, Jan 2002.

i. The Preamble field (P-field) of the CUC shall be carried implicitly by the time broadcast message with a value of:

   1. Bit 0, extension flag = 0, meaning not extended.
   2. Bits 1-3, time code identification = 001, meaning time epoch started on January 1st 1958.
   3. Bits 4-5, number of coarse time bytes = 11, meaning the coarse time is contained in 4 bytes.
   4. Bits 6-7, number of fine time bytes = 11, meaning the fine time is contained in 3 bytes.

5.5.5 Synchronisation broadcast

a. Broadcast channels 4, 5, 6 and 7 (synchronisation broadcast channels) are reserved for broadcasting system synchronisation information.

5.5.6 Network management broadcast

a. Broadcast channels 7 to 31 shall be reserved for broadcasting network management information.

5.5.7 Node broadcast (TBC)

a. Broadcast channels 32 to 253 shall each be allocated to the node with the corresponding logical address.

b. The node allocated to a broadcast channel shall be the only node permitted to use that broadcast channel.

c. Broadcast channels 254 and 255 shall be reserved, and not used.

5.6 Framing layer

5.6.1 Framing layer service interface

The service primitives that shall be associated with this service are:

TX_DATA_FRAME.request;

RX_DATA_FRAME.indication;

TX_FCT.request;

RX_FCT.indication.

TX_BROADCAST_FRAME.request;

RX_BROADCAST_FRAME.indication;

5.6.1.1 TX_DATA_FRAME.request

5.6.1.1.1 Function

The virtual channel layer shall pass a TX_DATA_FRAME.request primitive to the framing layer to request to send a data frame containing some SpaceWire packet data over the SpaceFibre link.

5.6.1.1.2 Semantics

The TX_DATA_FRAME.request primitive shall provide the following parameters:

TX_DATA_FRAME.request (Virtual Channel, Data)

5.6.1.1.3 When Generated

The TX_DATA_FRAME.request primitive shall be generated when the virtual channel layer has a data to send over the SpaceFibre link.

5.6.1.1.4 Effect On Receipt

On receipt of the TX_DATA_FRAME.request primitive the framing layer shall frame the data from the virtual channel layer forming a data frame and send it over the SpaceFibre link.

5.6.1.2 RX_DATA_FRAME.indication

5.6.1.2.1 Function

The framing layer shall pass a RX_DATA_FRAME.indication primitive to the virtual channel layer to indicate that a data frame has been received.

5.6.1.2.2 Semantics

The RX_DATA_FRAME.indication primitive provides parameters as follows:

RX_DATA_FRAME.indication (Virtual Channel, Data).

5.6.1.2.3 When Generated

The RX_DATA_FRAME.indication primitive shall be passed to the virtual channel layer when a data frame has been received, de-capsulated and unscrambled.

5.6.1.2.4 Effect On Receipt

The effect on receipt of the RX_DATA_FRAME.indication primitive by the virtual channel layer shall be that the SpaceWire packet data is placed in the virtual channel buffer identified by the virtual channel parameter.

5.6.1.3 TX_FCT.request

5.6.1.3.1 Function

The virtual channel layer shall pass a TX_FCT.request primitive to the framing layer to request to send an FCT over the SpaceFibre link.

5.6.1.3.2 Semantics

The TX_FCT.request primitive shall provide the following parameters:

TX_FCT.request (Virtual Channel).

5.6.1.3.3 When Generated

The TX_FCT.request primitive shall be generated when one of the virtual channel input buffers has room for another data frame.

5.6.1.3.4 Effect On Receipt

On receipt of the TX_FCT.request primitive the framing layer shall immediately send an FCT for the specified virtual channel.

5.6.1.4 RX_FCT.indication

5.6.1.4.1 Function

The framing layer shall pass an RX_FCT.indication primitive to the virtual channel layer to indicate that an FCT has been received.

5.6.1.4.2 Semantics

The RX_FCT.indication primitive provides parameters as follows:

RX_FCT.indication (Virtual Channel).

5.6.1.4.3 When Generated

The RX_FCT.indication primitive shall be passed to the virtual channel layer when an FCT is received.

5.6.1.4.4 Effect On Receipt

The effect on receipt of the RX_FCT.indication primitive by the virtual channel layer shall be to enable another data frame from the specified virtual channel to be sent.

5.6.1.5 TX_BROADCAST_FRAME.request

5.6.1.5.1 Function

The virtual channel layer shall pass a TX_BROADCAST_FRAME.request primitive to the framing layer to request it to frame the broadcast message and to send it over the SpaceFibre link.

5.6.1.5.2 Semantics

The TX_BROADCAST_FRAME.request primitive shall provide the following parameters:

TX_BROADCAST_FRAME.request (Broadcast Channel, Broadcast Sequence Number, Message).

5.6.1.5.3 When Generated

The TX_BROADCAST_FRAME.request primitive shall be generated when a broadcast message is ready to send.

5.6.1.5.4 Effect On Receipt

On receipt of the TX_BROADCAST_FRAME.request primitive the framing layer shall encapsulate the broadcast message and immediately send it over the SpaceFibre link.

5.6.1.6 RX_BROADCAST_FRAME.indication

5.6.1.6.1 Function

The framing layer shall pass an RX_BROADCAST_FRAME.indication primitive to the broadcast layer to indicate that a broadcast frame has arrived and to pass its contents to the broadcast layer.

5.6.1.6.2 Semantics

The RX_BROADCAST_FRAME.indication primitive provides parameters as follows:

RX_BROADCAST_FRAME.indication (Broadcast Channel, Broadcast Sequence Number, Message).

5.6.1.6.3 When Generated

The RX_BROADCAST_FRAME.indication primitive shall be passed to the broadcast layer when a broadcast frame is received.

5.6.1.6.4 Effect On Receipt

The effect on receipt of the RX_BROADCAST_FRAME.indication primitive by the broadcast layer shall be for the broadcast layer to validate the broadcast frame and to pass valid broadcast messages up to the user application.

5.6.2 Framing

a. The framing layer shall encapsulate data from the virtual channel layer into data frames and pass them to the retry layer.

b. The framing layer shall receive flow control information from the virtual channel layer and pass it to the retry layer.

c. The framing layer shall encapsulate data from the broadcast message layer into broadcast frames and pass them to the retry layer.

d. Data frames from the retry layer shall be de-capsulated, unscrambled and passed up to the virtual channel layer.

e. FCTs received from the retry layer shall be passed up to the virtual channel layer.

f. Broadcast frames from the retry layer shall be de-capsulated and passed up to the broadcast channel layer.

5.6.3 EM emission mitigation

5.6.3.1 Data scrambling

a. The data field of data frames shall be scrambled prior to transmission of the frame by bit-wise multiplication of the data with a sequence of random numbers produced from a scrambling polynomial.

NOTE This scrambler is illustrated in Figure 5-6, which is included for clarity. The scrambler can be implemented in other ways.

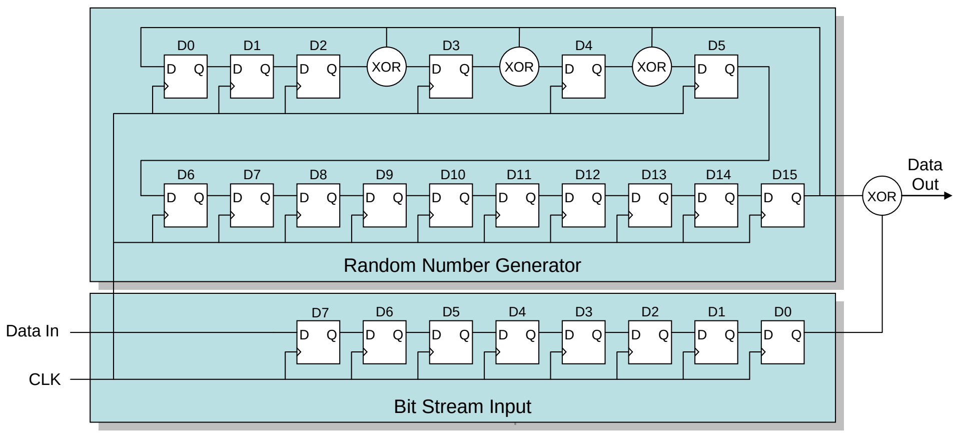


Figure 5-6 Scrambler / De-Scrambler

a. The scrambling polynomial to use shall be $G(x) = X^{16} + X^5 + X^4 + X^3 + 1$

b. The seed for the scrambler shall be 0xffff i.e. all flip-flops in the random number generator are set to one.

c. The scrambler shall be re-seeded at the start of every new data frame.

d. The single bit output sequence from the random number generator shall be XORed with the bit sequence from the data word.

e. The least-significant bit of the most-significant character in each word shall be scrambled first.

f. When an EOP or EEP or Null occurs in the data it shall not be replaced by scrambled data.

NOTE If the EOP or EEP or Null control code was replaced by scrambled data it would not be possible to distinguish it from a data value.

5.6.3.2 Unscrambling

a. When data frames are received they shall be unscrambled by multiplying (XORing) the received data with the same scrambling sequence as was used for scrambling.

b. The control codes representing EOP or EEP shall not be unscrambled.

5.6.4 Frame reception

a. Data frames that are received without error and in the correct order shall be passed from the retry layer to the framing layer.

b. FCTs received without error and in the correct order shall be passed from the retry layer to the framing layer.

c. Broadcast frames that are received without error and in the correct order shall be passed from the retry layer to the framing layer.

5.7 Retry layer

5.7.1 Retry layer service interface

The service primitives that shall be associated with the retry layer service are:

TX_DATA_FRAME.request;

RX_DATA_FRAME.indication;

TX_FCT.request;

RX_FCT.indication;

TX_BC_FRAME.request;

RX_BC_FRAME.indication;

5.7.1.1 TX_DATA_FRAME.request

5.7.1.1.1 Function

The framing layer shall pass a TX_DATA_FRAME.request primitive to the retry layer to send a data frame over the SpaceFibre link.

5.7.1.1.2 Semantics

The TX_DATA_FRAME.request primitive shall provide the following parameters:

TX_DATA_FRAME.request (Data Frame)

5.7.1.1.3 When Generated

The TX_DATA_FRAME.request primitive shall be passed to the retry layer when the framing layer has a data frame to send over the SpaceFibre link.

5.7.1.1.4 Effect On Receipt

On receipt of the TX_DATA_FRAME.request primitive the retry layer shall send the data frame over the SpaceFibre interface, resending the data frame in the event of it failing to be received correctly at the far end of the SpaceFibre link.

5.7.1.2 RX_DATA_FRAME.indication

5.7.1.2.1 Function

The retry layer shall pass a RX_DATA_FRAME.indication primitive to the framing layer to indicate that a data frame has been received.

5.7.1.2.2 Semantics

The RX_DATA_FRAME.indication primitive provides parameters as follows:

RX_DATA_FRAME.indication (Data Frame).

5.7.1.2.3 When Generated

The RX_DATA_FRAME.indication primitive shall be passed to the framing layer when a data frame is received without error and in the correct order.

5.7.1.2.4 Effect On Receipt

The effect on receipt of the RX_DATA_FRAME.indication primitive by the framing layer shall be that the contents of the received data frame is extracted and passed to the virtual channel layer.

5.7.1.3 TX_FCT.request

5.7.1.3.1 Function

The framing layer shall pass a TX_FCT.request primitive to the retry layer to send an FCT over the SpaceFibre link.

5.7.1.3.2 Semantics

The TX_FCT.request primitive shall provide the following parameters:

TX_FCT.request (FCT)

5.7.1.3.3 When Generated

The TX_FCT.request primitive shall be passed to the retry layer when the framing layer has a FCT to send over the SpaceFibre link.

5.7.1.3.4 Effect On Receipt

On receipt of the TX_FCT.request primitive the retry layer shall send the FCT over the SpaceFibre interface, resending the FCT in the event of it failing to be received correctly at the far end of the SpaceFibre link.

5.7.1.4 RX_FCT.indication

5.7.1.4.1 Function

The retry layer shall pass an RX_FCT.indication primitive to the framing layer to indicate that an FCT has been received.

5.7.1.4.2 Semantics

The RX_FCT.indication primitive provides parameters as follows:

RX_FCT.indication (FCT).

5.7.1.4.3 When Generated

The RX_FCT.indication primitive shall be passed to the framing layer when an FCT is received without error and in the correct order.

5.7.1.4.4 Effect On Receipt

The effect on receipt of the RX_FCT.indication primitive by the framing layer shall be for the information contained in the FCT to be extracted and passed on up to the virtual channel layer.

5.7.1.5 TX_BC_FRAME.request

5.7.1.5.1 Function

The framing layer shall pass a TX_BC_FRAME.request primitive to the retry layer to send a broadcast frame over the SpaceFibre link.

5.7.1.5.2 Semantics

The TX_BC_FRAME.request primitive shall provide the following parameters:

TX_BC_FRAME.request (Broadcast Frame)

5.7.1.5.3 When Generated

The TX_BC_FRAME.request primitive shall be passed to the retry layer when the framing layer has a broadcast frame or FCT to send over the SpaceFibre link.

5.7.1.5.4 Effect On Receipt

On receipt of the TX_BC_FRAME.request primitive the retry layer shall immediately send the broadcast frame over the SpaceFibre link, resending the broadcast frame in the event of it failing to be received correctly at the far end of the SpaceFibre link.

5.7.1.6 RX_BC_FRAME.indication

5.7.1.6.1 Function

The retry layer shall pass a RX_BC_FRAME.indication primitive to the framing layer to indicate that a broadcast frame has been received.

5.7.1.6.2 Semantics

The RX_BC_FRAME.indication primitive provides parameters as follows:

RX_BC_FRAME.indication (Broadcast Frame).

5.7.1.6.3 When Generated

The RX_BC_FRAME.indication primitive shall be passed to the framing layer when a broadcast frame is received without error and in the correct order.

5.7.1.6.4 Effect On Receipt

The effect on receipt of the RX_BC_FRAME.indication primitive by the framing layer shall be that the contents of the received broadcast frame is extracted and passed to the broadcast layer.

5.7.2 Data words identification state machine

a. The Data Word Identification state machine shall be used to identify which type of frame a data word belongs to, in order to support de-multiplexing of different frame types.

NOTE The state diagram for the Data Word Identification state machine is illustrated in Figure 5-7.

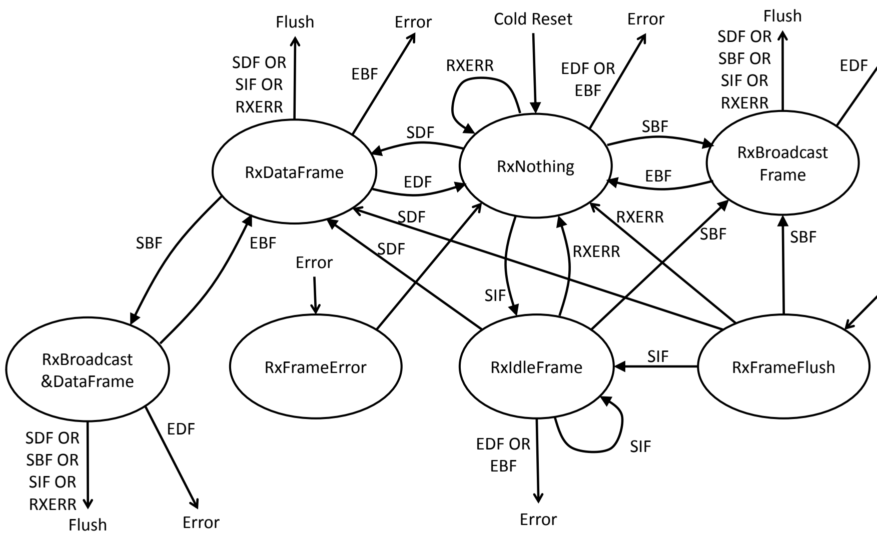


Figure 5-7 Data Word Identification State Machine

5.7.2.1 RxNothing state

a. The RxNothing state shall be entered on one of the following conditions:

1. Following a cold reset or warm reset.
2. From the RxDataFrame state, when an EDF is received.
3. From the RxBroadcastFrame state, when an EBF is received.
4. From the RxIdleFrame state, when an RXERR is received.
5. From the RxFrameFlush state, when the flush was caused by an RXERR,
6. Unconditionally, from the RxFrameError state.

b. When in the RxNothing state the Data Word Identification state machine shall initiate the following actions:

1. Indicate that no frame data is being received.
2. Discard any RXERR control words.

c. The Data Word Identification state machine shall leave the RxNothing State on one of the following conditions, which shall be evaluated in the order given:

1. When an EDF or EBF is received, move to the RxFrameError state.
2. When an SBF is received, move to the RxBroadcastFrame state.
3. When an SDF is received, move to the RxDataFrame state.
4. When an SIF is received, move to the RxIdleFrame state.

d. The RxNothing state is summarised in Table 5-12.

Table 5-12 RxNothing State

State	RxNothing
Entry	Following a cold reset or warm reset.

	From the RxDataFrame state, when an EDF is received. From the RxBroadcastFrame state, when an EBF is received. From the RxIdleFrame state, when an RXERR is received. From the RxFrameFlush state, when the flush was caused by an RXERR, Unconditionally, from the RxFrameError state.
Action	Indicate that no frame data is being received. Discard any RXERR control words.
Exit	When an EDF or EBF is received, move to the RxFrameError state. When an SBF is received, move to the RxBroadcastFrame state. When an SDF is received, move to the RxDataFrame state. When an SIF is received, move to the RxIdleFrame state.

5.7.2.2 RxDataFrame state

a. The RxDataFrame state shall be entered on one of the following conditions:

1. From the RxNothing state, when an SDF is received.
2. From the RxIdleFrame state, when an SDF is received.
3. From the RxBroadcast&DataFrame state, when an EBF is received.
4. From the RxFrameFlush state, when the flush was caused by an SDF.

b. When in the RxDataFrame state the Data Word Identification state machine shall initiate the following actions:

1. Indicate that data words being received belong to a data frame.

c. The Data Word Identification state machine shall leave the RxDataFrame state on one of the following conditions, which shall be evaluated in the order given:

1. When an EBF is received, move to the RxFrameError state.
2. When an SDF, SIF or RXERR is received, move to the RxFrameFlush state.
3. When an SBF is received move, to the RxBroadcast&DataFrame state.
4. When an EDF is received, move to the RxNothing state.

d. The RxDataFrame state is summarised in Table 5-13.

Table 5-13 RxDataFrame State

State	RxDataFrame
Entry	From the RxNothing state, when an SDF is received. From the RxIdleFrame state, when an SDF is received. From the RxBroadcast&DataFrame state, when an EBF is

	received. From the RxFrameFlush state, when the flush was caused by an SDF.
Action	Indicate that data words being received belong to a data frame.
Exit	When an EBF is received, move to the RxFrameError state. When an SDF, SIF or RXERR is received, move to the RxFrameFlush state. When an SBF is received move, to the RxBroadcast&DataFrame state. When an EDF is received, move to the RxNothing state.

5.7.2.3 RxBroadcastFrame state

a. The RxBroadcastFrame state shall be entered on one of the following conditions:

1. From the RxNothing state, when an SBF is received.
2. From the RxIdleFrame state, when an SBF is received.
3. From the RxFrameFlush state, when the flush was caused by an SBF.

b. When in the RxBroadcastFrame state the Data Word Identification state machine shall initiate the following actions:

1. Indicate that data words being received belong to a broadcast frame.

c. The Data Word Identification state machine shall leave the RxBroadcastFrame state on one of the following conditions, which shall be evaluated in the order given:

1. When an EDF is received, move to the RxFrameError state.
2. When an SDF, SBF, SIF or RXERR is received, move to the RxFrameFlush state.
3. When an EBF is received, move to the RxNothing state.

d. The RxBroadcastFrame state is summarised in Table 5-14.

Table 5-14 RxBroadcastFrame State

State	RxBroadcastFrame
Entry	From the RxNothing state, when an SBF is received. From the RxIdleFrame state, when an SBF is received. From the RxFrameFlush state, when the flush was caused by an SBF.
Action	Indicate that data words being received belong to a broadcast frame.
Exit	When an EDF is received, move to the RxFrameError state. When an SDF, SBF, SIF or RXERR is received, move to the

	RxFrameFlush state. When an EBF is received, move to the RxNothing state.

5.7.2.4 RxBroadcast&DataFrame state

a. The RxBroadcast&DataFrame state shall be entered on the following condition:

1. From the RxDataFrame state, when an SBF is received.

b. When in the RxNothing state the Data Word Identification state machine shall initiate the following actions:

1. Indicate that data words being received belong to a broadcast frame.

c. The Data Word Identification state machine shall leave the RxBroadcast&DataFrame state on one of the following conditions, which shall be evaluated in the order give:

1. When an EDF is received, move to the RxFrameError state.
2. When an SDF, SBF, SIF or RXERR is received, move to the RxFrameFlush state.
3. When an EBF is received, move to the RxDataFrame state.

d. The RxBroadcast&DataFrame state is summarised in Table 5-15.

Table 5-15 RxBroadcast&DataFrame State

State	RxBroadcast&DataFrame
Entry	From the RxDataFrame state, when an SBF is received.
Action	Indicate that data words being received belong to a broadcast frame.
Exit	When an EDF is received, move to the RxFrameError state. When an SDF, SBF, SIF or RXERR is received, move to the RxFrameFlush state. When an EBF is received, move to the RxDataFrame state.

5.7.2.5 RxIdleFrame state

a. The RxIdleFrame state shall be entered on one of the following conditions:

1. From the RxNothing state, when an SIF is received.
2. From the RxFrameFlush state, when the flush was caused by an SIF.

b. When in the RxIdleFrame state the Data Word Identification state machine shall initiate the following actions:

1. Indicate that data words being received belong to an idle frame.

c. The Data Word Identification state machine shall leave the RxDataFrame state on one of the following conditions, which shall be evaluated in the order give:

1. When an EDF or EBF is received, move to the RxFrameError state.
2. When an RXERR is received, move to the RxNothing state.
3. When an SBF is received, move to the RxBroadcastFrame state.
4. When an SDF is received, move to the RxDataFrame state.
5. When an SIF is received, remain in the RxIdleFrame state.

d. The RxIdleFrame state is summarised in Table 5-16.

Table 5-16 RxIdleFrame State

State	RxIdleFrame
Entry	From the RxNothing state, when an SIF is received. From the RxFrameFlush state, when the flush was caused by an SIF.
Action	Indicate that data words being received belong to an idle frame.
Exit	When an EDF or EBF is received, move to the RxFrameError state. When an RXERR is received, move to the RxNothing state. When an SBF is received, move to the RxBroadcastFrame state. When an SDF is received, move to the RxDataFrame state. When an SIF is received, remain in the RxIdleFrame state.

5.7.2.6 RxFrameFlush state

a. The RxFrameFlush state shall be entered on one of the following conditions:

1. From the RxDataFrame state, when an SDF, SIF or RXERR is received.
2. From the RxBroadcastFrame state, when an SDF, SBF, SIF or RXERR is received.
3. From the RxBroadcast&DataFrame state, when an SDF, SBF, SIF or RXERR is received.
4. From the RxIdleFrame state, when an RXERR is received.

b. When in the RxFrameFlush state the Data Word Identification state machine shall initiate the following actions:

1. Flush the frame receive buffers.

c. The Data Word Identification state machine shall leave the RxFrameFlush state on one of the following conditions, which shall be evaluated in the order give:

1. When an RXERR caused the flush, move to the RxNothing state.

2. When an SBF caused the flush, move to the RxBroadcastFrame state.
3. When an SDF caused the flush, move to the RxDataFrame state
4. When an SIF caused the flush, move to the RxIdleFrame state.

d. The RxFrameFlush state is summarised in Table 5-17.

Table 5-17 RxFrameFlush State

State	RxFrameFlush
Entry	From the RxDataFrame state, when an SDF, SIF or RXERR is received. From the RxBroadcastFrame state, when an SDF, SBF, SIF or RXERR is received. From the RxBroadcast&DataFrame state, when an SDF, SBF, SIF or RXERR is received. From the RxIdleFrame state, when an RXERR is received.
Action	Flush the frame receive buffers.
Exit	When an RXERR caused the flush, move to the RxNothing state. When an SBF caused the flush, move to the RxBroadcastFrame state. When an SDF caused the flush, move to the RxDataFrame state When an SIF caused the flush, move to the RxIdleFrame state.

5.7.2.7 RxFrameError state

a. The RxFrameError state shall be entered on one of the following conditions:
1. From the RxNothing state, when an EDF or EBF is received.
2. From the RxDataFrame state, when an EBF is received.
3. From the RxBroadcastFrame state, when an EDF is received.
4. From the RxBroadcast&DataFrame state, when an EDF is received.
5. From the RxIdleFrame state, when an EDF or EBF is received.

b. When in the RxFrameError state the Data Word Identification state machine shall initiate the following actions:
1. Flush the frame receive buffers.
2. Indicate and record the Receive Frame Error.

c. The Data Word Identification state machine shall leave the RxFrameError State on one of the following conditions, which shall be evaluated in the order given:
1. Unconditionally, move to the RxNothing state.

d. The RxFrameError state is summarised in Table 5-17.

Table 5-18 RxFrameError State

State	RxFrameError
Entry	From the RxNothing state, when an EDF or EBF is received. From the RxDataFrame state, when an EBF is received. From the RxBroadcastFrame state, when an EDF is received. From the RxBroadcast&DataFrame state, when an EDF is received. From the RxIdleFrame state, when an EDF or EBF is received.
Action	Flush the frame receive buffers. Indicate and record the Receive Frame Error.
Exit	Unconditionally, move to the RxNothing state.

5.7.2.8 FCTs

a. It shall be possible to receive an FCT in any state of the Data Word Identification state machine.

5.7.3 Frame sequence numbering

5.7.3.1 Frame sequence numbers on transmit

a. An 8-bit transmit frame sequence counter shall be maintained in the transmit side of the SpaceFibre CODEC, which holds the frame sequence number of the last data frame, broadcast frame or FCT control word that was sent.

b. The transmit frame sequence counter shall be cleared to zero on cold reset of the SpaceFibre CODEC.

c. The transmit frame sequence counter shall not be modified on warm reset of the SpaceFibre CODEC.

d. The single transmit frame sequence counter shall provide frame sequence numbers for data frames, FCTs and broadcast frames.

e. When an end of data frame, end of broadcast frame, or FCT control word passes into the retry layer from the framing layer, the current value of the transmit frame sequence counter shall be incremented and the new value placed in the frame sequence number (FR_SEQ#) field of the end of data frame, end of broadcast frame, or FCT control word.

f. No data frame, FCT or broadcast frame shall contain the same sequence number, ignoring rollover of the sequence counter.

NOTE Every 256 sequence numbers the counter will roll over repeating the series of sequence numbers.

g. On warm reset the transmit frame sequence counter shall remain unchanged.

h. On cold reset the transmit frame sequence counter shall be set to zero.

5.7.3.2 Frame sequence numbers on reception

a. An 8-bit receive frame sequence counter shall be maintained in the receive side of the SpaceFibre CODEC, which holds the frame sequence number of the last data frame, broadcast frame or FCT control word that was received correctly.

b. The receive frame sequence counter shall be cleared to zero on cold reset of the SpaceFibre CODEC.

c. The receive frame sequence counter shall not be modified on warm reset of the SpaceFibre CODEC.

d. The single receive frame sequence counter shall be used to check the sequence numbers of data frames, FCTs and broadcast frames.

e. When an end of data frame, end of broadcast frame, or FCT control word passes into the retry layer from the lane control layer, its frame sequence number (FR_SEQ#) shall be checked against the current value of the receive frame sequence counter.

f. If the frame sequence number has the same value as the receive frame sequence counter plus one, the data frame, FCT or broadcast frame shall be accepted.

g. When a data frame, FCT or broadcast frame is accepted, an acknowledgement (ACK) containing the value of the received frame sequence number shall be sent.

h. When a data frame, FCT or broadcast frame is accepted, the receive frame sequence counter shall be incremented.

i. If the frame sequence number does not have the same value as the receive frame sequence counter plus one, the data frame, FCT or broadcast frame shall be rejected.

j. When a data frame, FCT or broadcast frame is rejected, a negative acknowledgement (NACK) containing the value of the receive frame sequence counter shall be sent.

k. When a data frame, FCT or broadcast frame is rejected, the receive frame sequence counter shall not be incremented.

5.7.4 CRC for data frame

a. A 16-bit CRC checksum shall be applied to data frames.

b. The CRC shall cover an entire data frame, from and including the comma in the start of data frame control word, up to and including the frame sequence number in the end of data frame control word.

c. When a control code is to be included in the CRC calculation, its data value shall be used.

NOTE For example, K28.5 the comma code is replaced by D28.5 in the CRC calculation.

d. When an EOP or EEP or Null occurs in the data the value of the control code representing the EOP or EEP or Null shall be included in the CRC generation.

NOTE For example, an EOP is represented by K28.0 so the value D28.0 would be included in the CRC calculation. Including an EOP or EEP or Null in the CRC calculation means that they are protected by the CRC checksum.

e. The 16-bit CRC calculation is TBD.

5.7.5 CRC for broadcast frame, FCT, ACK and NACK

a. An 8-bit CRC checksum shall be applied to broadcast frames, FCTs, ACKs and NACKs.

b. The 8-bit CRC shall cover an entire broadcast frame, from and including the comma in the start of broadcast frame control word, up to and including the frame sequence number in the end of broadcast frame control word.

c. When a control code is to be included in the CRC calculation, its data value shall be used.

NOTE For example, K28.5 the comma code is replaced by D28.5 in the CRC calculation.

d. The 8-bit CRC calculation procedures shall:

1. use modulo 2 arithmetic for polynomial coefficients;

2. use a systematic binary *(n+16,n)* block code, where *(n+8)* is the number of bits of the codeword $c(x)$ and *n* is divisible by 8; *n* is the number of bits covered by the CRC;

3. use the following generating polynomial:

$$g(x) = x^8 + x^2 + x + 1$$

4. use byte format as input and output, for which the bits are represented as:

$$b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0$$

where b_7 is the most significant bit and b_0 is the least significant bit;

e. The 8-bit CRC generation procedure shall behave as follows:

1. The procedure accepts an *n*-bit input which is used to construct $m(x)$, where:

(a) the *n*-bit input is defined to be the set of bits $B_{i,j}$ grouped into $n/8$ bytes where $i = \{0,1,\ldots,n/8-1\}$ is the byte index and $j = \{7,6,\ldots,0\}$ is the bit index;

(b) the $n/8$ input bytes correspond to the fields covered by the CRC excluding the CRC byte; the first byte transmitted has index $i=0$; the last byte transmitted has index $i=n/8-1$;

(c) $m(x)$ is a polynomial $m_{n-1}x^{n-1}+m_{n-1}x^{n-2}+...+m_0x^0$ having binary coefficients m_i;

(d) $m(x)$ can be represented as an *n*-bit vector where coefficient m_{n-1} of the highest power of x is the most significant bit and coefficient m_0 of the lowest power of x is the least significant bit;

(e) the bit vector representation of $m(x)$ is formed by concatenating the $n/8$ bytes of the input in transmission order, where the least significant bit b_0 of each byte is taken first and the most significant bit b_7 of each byte is taken last:

$$m_{n-1}=B_{0,0},m_{n-2}=B_{0,1},m_{n-3}=B_{0,2},\ldots,m_{n-7}=B_{0,6},m_{n-8}=B_{0,7},$$
$$m_{n-9}=B_{1,0},m_{n-10}=B_{1,1},m_{n-11}=B_{1,2},\ldots,m_{n-15}=B_{1,6},m_{n-16}=B_{1,7},\ldots$$
$$m_7=B_{n/8-1,0},m_6=B_{n/8-1,1},m_5=B_{n/8-1,2},\ldots,m_1=B_{n/8-1,6},m_0=B_{n/8-1,7}$$

2. The procedure generates the remainder polynomial $r(x)$ given by the equation:

$$r(x)=\left[m(x)\cdot x^8\right]\text{modulo } g(x)$$

where $r(x)=r_7x^7+r_6x^6+\ldots r_0x^0$ and r_i are binary coefficients;

3. The Header and Data CRC are formed from the 8-bit vector representation of $r(x)$; the least significant bit b_0 of the CRC byte is coefficient r_7 of the highest power of x, while the most significant bit b_7 of the CRC byte is coefficient r_0 of the lowest power of x:

$$b_7=r_0,b_6=r_1,b_5=r_2,b_4=r_3,b_3=r_4,b_2=r_5,b_1=r_6,b_0=r_7$$

NOTE 1 The codeword $c(x)=m(x)\cdot x^8+r(x)$ is formed by concatenating the bit vector representations of $m(x)$ and $r(x)$.

NOTE 2 When a Galois version of a Linear Feedback Shift Register is used for CRC generation, its initial value is zero.

NOTE 3 Example VHDL and C-code implementations of this CRC algorithm are included in clause Annex B.

f. If the CRC generation procedure is applied to the bytes covered by the CRC *excluding* the CRC byte then the generated CRC shall be compared directly with the expected CRC byte. If the generated and expected CRC bytes are equal then no errors have been detected; if they are different then an error has been detected.

g. If the CRC generation procedure is applied to the bytes covered by the CRC *including* the CRC byte then the output of the CRC generation procedure shall be zero if no errors have been detected and non-zero if an error has been detected.

NOTE 1 When the codeword $c^*(x)$ is input to the CRC generator then the remainder is the syndrome: $s(x) = \left[c^*(x) \cdot x^8\right] \text{modulo}\, g(x)$.

NOTE 2 The codeword $c^*(x)$ is the concatenation of the Header or Data bytes covered by the CRC, followed by the CRC byte.

5.7.6 Frame retry buffering

a. Data frames, FCTs and broadcast frames shall be placed in a frame retry buffer, as they are passed to the lane control layer for sending over the SpaceFibre link.

b. Data frames, FCTs and broadcast frames in the frame retry buffer shall be identified by their frame sequence number.

c. When an ACK is received, all data frames, FCTs and broadcast frames with a frame sequence number less than or equal to the frame sequence number in the ACK shall be deleted from the frame retry buffer, taking into account counter roll over effects (i.e. calculation is modulo 256).

d. When a NACK is received, all the data frames, FCTs, or broadcast frames with a sequence number less than or equal to that of the NACK shall be removed from the frame retry buffer, since the NACK has signalled that they have been received correctly.

e. When a NACK is received, the data frame, FCT or broadcast frame, which has a sequence number one more than that of the NACK shall be resent together with all data frames, FCTs and broadcast frames which have higher value sequence numbers, since the NACK has signalled that they have not been received correctly.

NOTE See section 5.7.9.2 for reasons why a NACK is transmitted.

f. Any data frames, FCTs or broadcast frames, being resent shall remain in the same frame sequence number order in which they were originally placed in the frame retry buffer.

g. Any frames being resent may disregard any implanting of FCTs or broadcast frames within data frames, provided that they are resent in frame sequence number order.

h. A NACK received when the contents of the retry buffer are being resent shall be ignored and not cause all the contents to be resent again.

i. The number of retries shall be recorded and made available in a status register.

j. On cold reset the frame retry buffer shall be emptied.

k. On a cold reset the number of retries cleared to zero.

l. On warm reset the frame retry buffer shall not be emptied

m. On a warm reset the number of retries shall not be cleared.

n. The frame retry buffer shall be 256 times the number of lanes, words long (TBC).

NOTE The frame retry buffer shall be long enough to hold all the data and control words that can fit on the line for as long as it takes to send a frame and receive an ACK in reply.

NOTE When multiple lanes are used proportionally more data can be held on the line.

o. If the frame retry buffer becomes full, no more data frames, FCTs or broadcast frames shall be sent until there is more room in the frame retry buffer.

p. If the frame retry buffer becomes full, a link failure timer shall be started.

q. The link failure timer shall expire TBA μs after it has been started.

r. The link failure timer shall be cancelled, when an ACK or NACK is received.

s. If the link failure timer expires, a link failure signal shall be asserted, and indicated to the user application.

t. When the frame retry buffer is full, only ACKs, NACKs, and idle frames shall be sent, by the retry layer.

5.7.7 Receive frame buffering

a. While a data frame or broadcast frame is being received, their words shall be buffered in order to be able to check the CRC and frame sequence number prior to the frame being accepted and passed up to the Framing layer.

b. If frame information is received in an incorrect order (see section 5.7.2) and the Data Word Identification state machine enters the RxFrameFlush state, the contents of the buffer shall be flushed, and the SDF or SBF at the start of the next frame placed in the buffer.

c. Only data frames, FCTs, and broadcast frames that are received without error and in the correct sequence shall be passed up to the framing layer.

d. Data frames, FCTs, and broadcast frames that are received containing errors or in an incorrect sequence shall be discarded.

5.7.8 Idle frames

a. Idle frames shall be generated when there are no data frames, FCTs or broadcast frames to send.

b. Idle frames shall also be generated when the retry buffer is full preventing any new data frames, FCTs, or broadcast frames from being sent.

c. The frame sequence number field in the start of idle frame shall be set to the current value of the transmit frame sequence counter.

NOTE The transmit frame sequence counter contains the sequence number of the last data frame, FCT or broadcast frame sent. After a cold reset this counter is set to zero so if an idle frame is the first frame sent after cold reset, its frame sequence number field will be zero. If a data frame, FCT, or broadcast frame is the first frame sent after a cold reset its frame sequence number field will be one.

d. The start of idle frame control word shall be followed by a series of pseudo-random data words which will form a pseudo random bit sequence when transmitted.

e. The first pseudo-random word shall contain the seed that is to be used for generating the subsequent series of pseudo-random data words.

f. The following algorithm shall be used to generate the series of pseudo-random data words. xxx TBA

g. At the end of an idle frame the next pseudo-random data word shall be generated and stored for use as the seed for the next idle frame.

h. When the next idle frame is to be sent the stored seed value shall be used as the first pseudo-random data word, i.e. the seed, in that idle frame.

i. After cold reset the stored seed shall be set to 0xffff ffff (xxx TBC).

j. After warm reset the stored seed shall not be modified.

5.7.9 ACK/NACK control

5.7.9.1 Sending ACKs

a. An ACK shall be sent as soon as possible whenever a data frame, FCT, or broadcast frame is received without error and in the correct frame sequence number order.

b. The ACK shall contain the frame sequence number of the data frame, FCT, or broadcast frame that caused the ACK to be sent, i.e. the last correctly received data frame, broadcast frame or FCT.

c. ACKs are not buffered and shall be sent as soon as possible, taking into account control word precedence, see section 5.3.4.

5.7.9.2 Sending NACKs

a. A NACK shall be sent as soon as possible whenever a data frame, FCT, or broadcast frame is received which either contains an error or is not in the correct frame sequence number order, i.e. its frame sequence number is not one more than the value in the receive frame sequence counter.

b. A NACK shall also be sent as soon as possible if the type of frame received cannot be identified or if an unrecognised word is received or if a word is received in an unexpected place in a frame.

c. A NACK shall also be sent as soon as possible when an RXERR control word is received by the Data Word Initialisation state machine while in the RxDataFrame, RxBroadcstFrame, or RxDataBroadcastFrame states.

d. The NACK shall contain the frame sequence number of the last correctly received data frame, FCT, or broadcast frame, which is the value contained in the receive frame sequence counter.

e. If a received idle frame contains a frame sequence number which is not the same as the receive frame sequence counter, a NACK shall be sent containing the sequence number of the last correctly received data frame, FCT, or broadcast frame.

NOTE The idle frame sequence number ought to have the same sequence number as the last correctly received data frame, FCT, or broadcast frame i.e. be the same as the current value of the receive frame sequence counter. If this is not the case it indicates that a frame that has been sent has not been received, so a NACK is sent to indicate the last correctly received data frame, FCT or broadcast frame.

NOTE When an idle frame is received that has the correct frame sequence number there is no need to send an ACK. The correct frame sequence number indicates that the last data frame, FCT or broadcast frame has been received correctly. If the ACK sent for this frame has gone missing, it does not matter unless the frame retry buffer fills up which will not happen unless more data frames, FCTs or broadcast frames are sent. In which case other ACKs will be sent.

5.7.9.3 Receiving ACKs

a. When an ACK is received it shall be checked for errors using its CRC.

b. When an ACK is received with a valid CRC, the data frames, FCTs, or broadcast frames that have a frame sequence number less than or equal to that contained in the ACK, taking into account the rollover of the frame sequence number at 256, so that the comparison of the frame sequence number is modulo 256, shall be removed from the frame retry buffer.

5.7.9.4 Receiving NACKs

a. When a NACK is received it shall be checked for errors using its CRC.

b. When a NACK is received with a valid CRC, the data frames, FCTs, or broadcast frames that have a frame sequence number less than or equal to that contained in the NACK, taking into account the rollover of the frame sequence number at 256, so that the comparison of the frame sequence number is modulo 256, shall be removed from the frame retry buffer.

c. When a NACK is received with a valid CRC, the data frames, FCTs, or broadcast frame in the frame retry buffer that have a frame sequence number greater than that contained in the NACK, taking into account the rollover of the frame sequence number at 256 so that the comparison of the frame sequence number is modulo 256, shall be taken from the frame retry buffer in frame sequence number order and resent.

d. If the frame retry buffer is already resending information when a NACK is received the NACK shall be ignored.

e. If a NACK is received and the frame retry buffer is empty, an idle frame shall be sent.

NOTE This is to recover from an error happening in an idle frame while the retry buffer is empty, i.e. with no data frame, broadcast frame or FCT sent previously.

5.7.10 PRBS test

a. When an idle frame is received the first data word after the start of idle frame shall be loaded as the seed into a PRBS generator.

b. The PRBS generator shall be used to generate a 32-bit pseudo-random data word for every data word that is received in the idle frame, using the same algorithm as was used to generate the pseudo-random data words in the idle frame.

c. The PRBS data words generated by the PRBS generator shall be used to check each subsequent data word received in the idle frame.

d. If any of the received data words do not match that produced by the PRBS generator, a PRBS error signal shall be asserted.

5.8 Lane control layer

5.8.1 Lane control layer service interface

The service primitives that shall be associated with the lane control layer service are:

TX_WORD.request;

RX_WORD.indication;

5.8.1.1 TX_WORD.request

5.8.1.1.1 Function

The retry layer shall pass a TX_WORD.request primitive to the lane control layer to send a data word or control word that forms part of a data frame, broadcast frame, idle frame, FCT, ACK or NACK over the SpaceFibre link.

5.8.1.1.2 Semantics

The TX_WORD.request primitive shall provide the following parameters:

TX_WORD.request (Word)

5.8.1.1.3 When Generated

The TX_WORD.request primitive shall be passed to the lane control layer when the retry layer has information to send over the SpaceFibre link.

5.8.1.1.4 Effect On Receipt

On receipt of the TX_WORD.request primitive the lane control layer shall send the data word or control word over the SpaceFibre interface.

5.8.1.2 RX_WORD.indication

5.8.1.2.1 Function

The lane control layer shall pass a RX_WORD.indication primitive to the retry layer to indicate that a data word or control word has been received.

5.8.1.2.2 Semantics

The RX_WORD.indication primitive provides parameters as follows:

RX_WORD.indication (Word).

5.8.1.2.3 When Generated

The RX_WORD.indication primitive shall be passed to the retry layer when a data word or control word is received.

5.8.1.2.4 Effect On Receipt

The effect on receipt of the RX_WORD.indication primitive by the retry layer shall result in the word being accepted by the retry layer.

5.8.2 Lane control

a. The Required Number of Lanes management parameter (required lanes) shall specify the number of lanes that are to be used to form the SpaceFibre link.

b. The number of lanes that are active and ready to send data and control words (active lanes) shall be indicated from the lane layer to the lane control layer.

c. If the number of required lanes is less than the number of active lanes, the number of used lanes shall be set to the number of required lanes.

d. If the number of required lanes is the same as the number of active lanes, the number of used lanes shall be set to the number of required lanes.

e. If the number of required lanes is more than the number of active lanes, the number of used lanes shall be set to the number of active lanes.

f. Each lane shall be given a lane number, starting at 1 and incrementing for each physical link.

NOTE For example, if there are four possible lanes they would be given lane numbers 1, 2, 3 and 4.

5.8.3 Lane synchronisation

a. When the number of required lanes or the number of active lanes changes, lane synchronisation shall occur.

b. During lane synchronisation the number of used lanes shall be determined.

c. Lanes shall be assigned lane umbers as follows: the lowest number lane that is active is assigned lane number 1, the next lowest number lane that is active is assigned lane number 2, and subsequent lanes are assigned lane numbers in this way until enough lanes have been assigned lane numbers to cover the number of used lanes.

d. All other physical links that are active shall be assigned a null lane number (zero).

e. An LSYNC control word shall be sent over each of the active lanes containing the lane number assigned to that lane.

f. The LSYNC control word shall be sent over each of the active lanes at approximately the same time, i.e. within the time it takes to transmit one control word (TBC).

g. Once the LSYNC control word has been sent, data and control words can be distributed over the used lanes for transmission.

h. When an LSYNC control word is received, the lane synchroniser shall wait for LSYNC control words to be received over each of the other active lanes.

i. When LSYNC control words have been received on all active lanes, the lane concentrator shall be updated with the lanes it is to concentrate data from.

j. The active lanes that receive an LSYNC control word containing a null (zero) shall not be included in the lane concentration.

k. The active lanes that receive an LSYNC control word containing a lane number shall be included in the lane concentration.

l. If an LSYNC control word is not received on a lane when LSYNC control words are received on the other active lanes, all the lanes are instructed to reinitialise.

NOTE Link re-initialisation will automatically invoke resynchronisation of the lanes.

m. If the received LSYNC control words do not form a proper integer series (i.e. 1, 2, 3, etc) of lane numbers with no duplicate lane numbers, other than possible duplicate null lanes, and no missing lane numbers, all the lanes are instructed to reinitialise.

NOTE This is likely to be because it was lost and replaced by an RXERR control word or the result of a serious error in the transmitter, which re-initialisation might not correct.

n. If after invoking re-initialisation of lanes, lane synchronisation still fails, the error shall be flagged as a permanent error and the user application informed.

5.8.4 Lane distribution

a. Data and control words to be sent over the SpaceFibre link shall be distributed over the used lanes.

b. The first data or control word shall be sent over the lowest number used lane, the next data or control word over the next lowest number used lane, and so on with data or control words being sent over each of the used lanes.

c. Data and control words shall not be sent over active lanes which are not used lanes.

NOTE Such lanes are providing hot standby and will be sending IDLE control words.

5.8.5 Lane concentration

a. When a data or control word arrives over a lane, it shall be placed in a small synchronisation FIFO.

b. There shall be one synchronisation FIFO for each lane.

c. The synchronisation FIFO shall be able to store at least three data or control words.

d. Data or control words shall be read out of the synchronisation FIFOs for each used lane in lane number order.

e. Following lane synchronisation, the first data or control word shall be read out of the lane with the lowest lane number, the next data or control word over the next lowest number used lane, and so on with data and control words being read out of the synchronisation FIFOs of each of the used lanes.

NOTE The lane numbers of the concentrator can be different to that of the distributor.

5.8.6 Lane selection

It shall be possible to switch off or bypass multi-lane operation so that the SpaceFibre link operates over one lane only.

5.9 Lane layer

5.9.1 Lane layer service interface

The service primitives that shall be associated with the lane layer service are:

TX_WORD.request;

RX_WORD.indication;

LINK_STATE.indication.

5.9.1.1 TX_WORD.request

5.9.1.1.1 Function

The lane control layer shall pass a TX_WORD.request primitive to the lane layer to send a data word or control word that forms part of a data frame, broadcast frame, idle frame, FCT, ACK or NACK over the SpaceFibre link.

5.9.1.1.2 Semantics

The TX_WORD.request primitive shall provide the following parameters:

TX_WORD.request (Word)

5.9.1.1.3 When Generated

The TX_WORD.request primitive shall be passed to the lane layer when the lane control layer has information to send over the SpaceFibre link.

5.9.1.1.4 Effect On Receipt

On receipt of the TX_WORD.request primitive the lane layer shall send the data word or control word over the SpaceFibre link.

5.9.1.2 RX_WORD.indication

5.9.1.2.1 Function

The lane layer shall pass a RX_WORD.indication primitive to the lane control layer to indicate that a data word or control word has been received.

5.9.1.2.2 Semantics

The RX_WORD.indication primitive provides parameters as follows:

RX_ WORD.indication (Word).

5.9.1.2.3 When Generated

The RX_ WORD.indication primitive shall be passed to the lane control layer when a data word or control word is received.

5.9.1.2.4 Effect On Receipt

The effect on receipt of the RX_ WORD.indication primitive by the lane control layer shall be that the received word is taken by the lane control layer and interleaved with words from other lanes.

5.9.1.3 LANE_STATE.indication

5.9.1.3.1 Function

The lane layer shall pass a LANE_STATUS.indication primitive to the lane control layer to indicate that the status of the lane has changed.

5.9.1.3.2 Semantics

The LANE_STATUS.indication primitive provides parameters as follows:

LANE_STATUS.indication (Lane Status).

NOTE The lane status values are TBD.

5.9.1.3.3 When Generated

The LANE_STATUS.indication primitive shall be passed to the lane control layer when the lane status has changed.

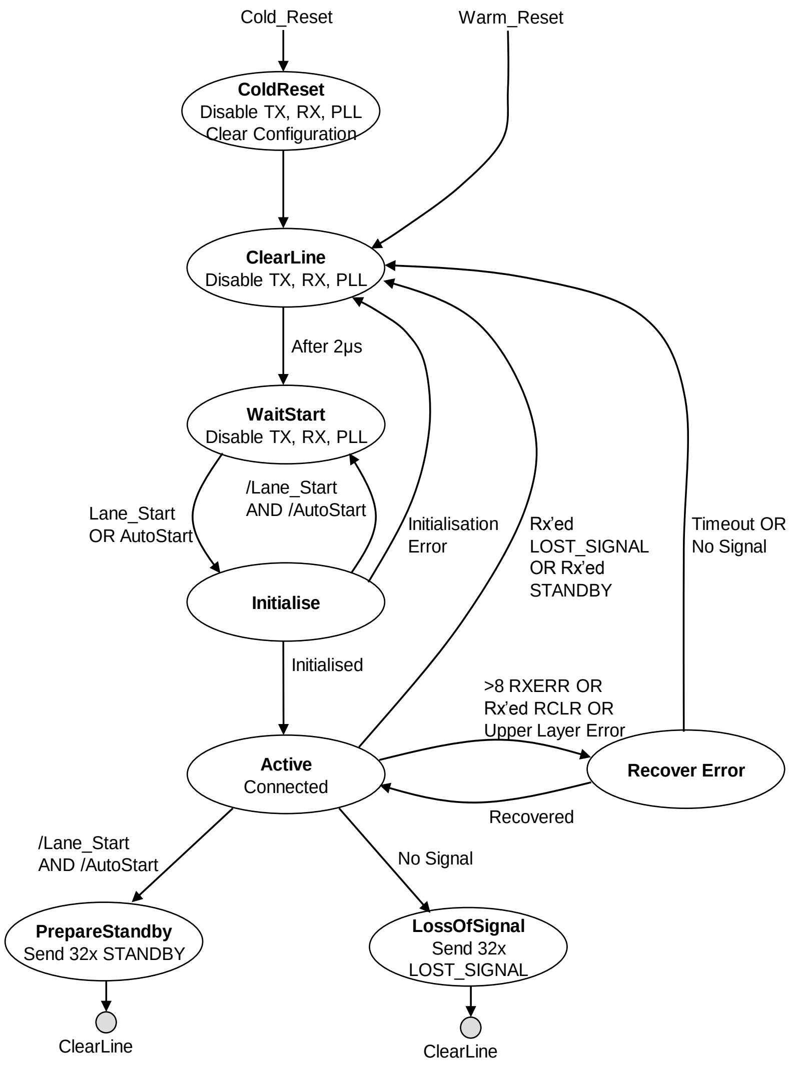
5.9.1.3.4 Effect On Receipt

On receipt of the LANE_STATUS.indication primitive the lane control layer shall react depending on the lane status. If the lane status is lane not ready, the lane control layer will stop sending words to the lane layer and cease reading words from that layer. The lane control layer will also resynchronise the remaining active lanes. If the lane status is lane ready, the lane control layer will resynchronise the active lanes and start sending words to the lane layer and reading words from that layer.

5.9.2 Lane initialisation and standby management

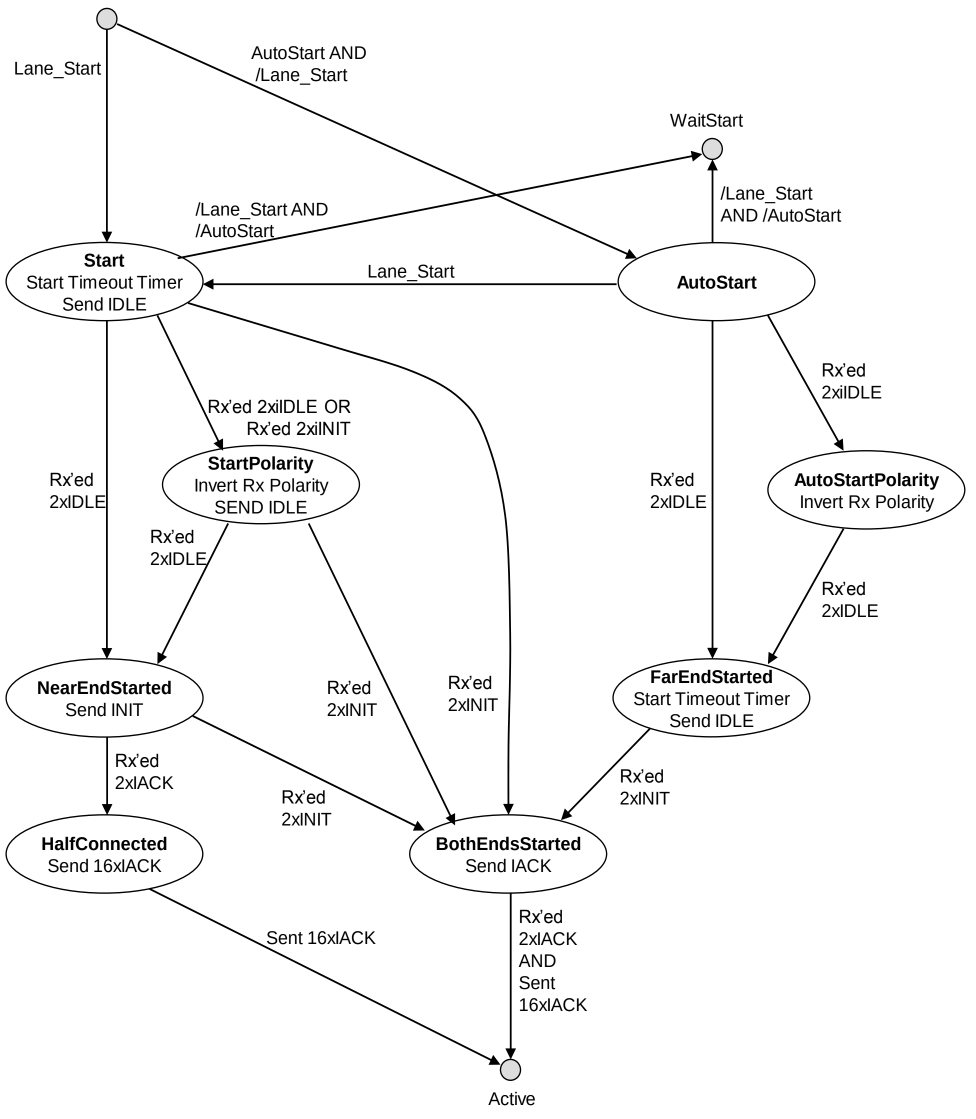
a. The Lane Initialisation state machine shall control the initialisation of the SpaceFibre link: establishing the connection and responding to standby management requests.

NOTE The state diagram for the lane initialisation state machine is illustrated in Figure 5-8.



No Signal means no signal detected at receiver inputs.
Initialisation errors are listed in Figure 5-9.

Figure 5-8 Overall Lane Initialisation State Machine



Errors in the Receive Synchronisation state machine, which result in an RXERR control word are ignored.
Eight consecutive RXERR control words in any state except Start and AutoStart cause a transition to the ClearLine state.
A Timeout error in any state except the AutoStart states causes a transition to the ClearLine state.
No Signal at receiver in any state except the Start and AutoStart states causes a transition to the ClearLine state.

Figure 5-9 Initialise State

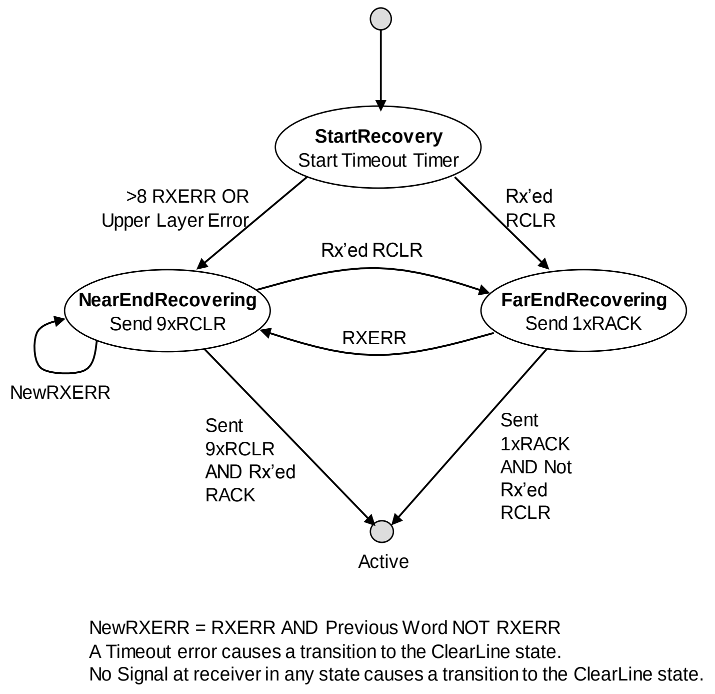

StartRecovery
Start Timeout Timer
>8 RXERR OR Upper Layer Error
Rx'ed RCLR
Rx'ed RCLR
NearEndRecovering
Send 9xRCLR
FarEndRecovering
Send 1xRACK
RXERR
NewRXERR
Sent 9xRCLR AND Rx'ed RACK
Sent 1xRACK AND Not Rx'ed RCLR
Active
NewRXERR = RXERR AND Previous Word NOT RXERR
A Timeout error causes a transition to the ClearLine state.
No Signal at receiver in any state causes a transition to the ClearLine state.

Figure 5-10 Recover Error State

5.9.2.1.2 ColdReset State

a. The Lane Initialisation state machine shall enter the ColdReset state on one of the following conditions:

   1. Power on reset.

   2. ColdReset command.

      NOTE Commands are issued via the Network Management interface which sets appropriate bits in control registers.

b. When in the ColdReset state the Lane Initialisation state machine shall initiate the following actions:

   1. Reset of the SpaceFibre Lane.

   2. Reset of all the Lane Configuration registers.

   3. Disable the transmitter driver, receiver and related transmit and receive PLLs.

c. The Lane Initialisation state machine shall leave the ColdReset state on one of the following conditions:

   1. Unconditionally, move to the ClearLine state.

d. The ColdReset state is summarised in Table 5-19.

Table 5-19 ColdReset State

State	ColdReset

Entry	Power on reset. ColdReset command.
Action	Reset of the SpaceFibre Lane. Reset of all the Lane Configuration registers. Disable the transmitter driver, receiver and related transmit and receive PLLs.
Exit	Unconditionally, move to the ClearLine state.

5.9.2.1.3 ClearLine state

a. The Lane Initialisation state machine shall enter the ClearLine state on one of the following conditions:

1. From the ColdReset state, unconditionally.
2. On WarmReset command.
3. From the Start state, when the initialisation timeout timer expires.

   NOTE The initialisation timeout timer is started in the Start state and stopped in the Active state. It will timeout if initialisation takes longer than 20 µs. This period allows plenty of time for the receive clock recovery at the far end of the lane to lock onto the transmitted signal, a response to be generated, returned to the near end and the local receiver to lock onto the returned signal.

4. From the StartPolarity state, when the initialisation timeout timer expires.
5. From the NearEndStarted state, when the initialisation timeout timer expires.
6. From the BothEndsStarted state, when the initialisation timeout timer expires.
7. From the FarEndStarted state, when the initialisation timeout timer expires.
8. From the HalfConnected state, when the initialisation timeout timer expires.
9. From the StartPolarity state, when a 1 µs polarity timeout timer expires.

   NOTE 1 µs is sufficient for the change in receiver polarity to propagate through the receive pipeline and receive elastic buffer.

10. From the AutoStartPolarity state, when a 1 µs polarity timeout timer expires.
11. From the NearEndRecovering state, when a 2 µs recovery timeout timer expires.

12. From the FarEndRecovering state, when a 2 µs recovery timeout timer expires.
13. From the StartPolarity state, when more than eight consecutive RXERR control words are received.
14. From the AutoStartPolarity state, when more than eight consecutive RXERR control words are received.
15. From the NearEndStarted state, when more than eight consecutive RXERR control words are received.
16. From the BothEndsStarted state, when more than eight consecutive RXERR control words are received.
17. From the FarEndStarted state, when more than eight consecutive RXERR control words are received.
18. From the HalfConnected state, when more than eight consecutive RXERR control words are received.
19. From the StartPolarity state, when No Signal is detected at the receiver.
20. From the AutoStartPolarity state, when No Signal is detected at the receiver.
21. From the NearEndStarted state, when No Signal is detected at the receiver.
22. From the BothEndsStarted state, when No Signal is detected at the receiver.
23. From the FarEndStarted state, when No Signal is detected at the receiver.
24. From the HalfConnected, when No Signal is detected at the receiver.
25. From the StartRecovery state, when No Signal is detected at the receiver.
26. From the NearEndRecovering state, when No Signal is detected at the receiver.
27. From the FarEndRecovering state, when No Signal is detected at the receiver.
28. From the PrepareStandby state, after sending 32 STANDBY control words.
29. From the LossOfSignal state, after sending 32 LOST_SIGNAL control words.

b. When in the ClearLine state the Lane Initialisation state machine shall initiate the following actions:

1. Start a 2 µs timeout timer.
2. Disable the transmitter driver, receiver and related transmit and receive PLLs.
3. Do NOT reset the Lane Configuration registers.

4. Switch off receiver bit inversion.

c. The Lane Initialisation state machine shall leave the ClearLine state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.

2. When the 2 μs timeout timer expires, move to the WaitStart state.

NOTE This is more than long enough for signals to propagate through the longest permitted fibre optic cable (100m) and back again, and to allow time for the signals to be processed at each end of the cable. The result is that the lanes forming the link will be completely reset at the near end of the link and at least warm reset at the far end.

d. The ClearLine state is summarised in Table 5-20.

Table 5-20 ClearLine State

State	ClearLine
Entry	From the ColdReset state, unconditionally. On WarmReset command. From the Start state, when the initialisation timeout timer expires. From the StartPolarity state, when the initialisation timeout timer expires. From the NearEndStarted state, when the initialisation timeout timer expires. From the BothEndsStarted state, when the initialisation timeout timer expires. From the FarEndStarted state, when the initialisation timeout timer expires. From the HalfConnected state, when the initialisation timeout timer expires. From the StartPolarity state, when a 1 μs polarity timeout timer expires. From the AutoStartPolarity state, when a 1 μs polarity timeout timer expires. From the NearEndRecovering state, when a 2 μs recovery timeout timer expires. From the FarEndRecovering state, when a 2 μs recovery timeout timer expires. From the StartPolarity state, when more than eight consecutive RXERR control words are received. From the AutoStartPolarity state, when more than eight consecutive RXERR control words are received. From the NearEndStarted state, when more than eight

	consecutive RXERR control words are received. From the BothEndsStarted state, when more than eight consecutive RXERR control words are received. From the FarEndStarted state, when more than eight consecutive RXERR control words are received. From the HalfConnected state, when more than eight consecutive RXERR control words are received. From the StartPolarity state, when No Signal is detected at the receiver. From the AutoStartPolarity state, when No Signal is detected at the receiver. From the NearEndStarted state, when No Signal is detected at the receiver. From the BothEndsStarted state, when No Signal is detected at the receiver. From the FarEndStarted state, when No Signal is detected at the receiver. From the HalfConnected, when No Signal is detected at the receiver. From the StartRecovery state, when No Signal is detected at the receiver. From the NearEndRecovering state, when No Signal is detected at the receiver. From the FarEndRecovering state, when No Signal is detected at the receiver. From the PrepareStandby state, after sending 32 STANDBY control words. From the LossOfSignal state, after sending 32 LOST_SIGNAL control words.
Action	Start a 2 μs timeout timer. Disable the transmitter driver, receiver and related transmit and receive PLLs. Do NOT reset the Lane Configuration registers. Switch off receiver bit inversion.
Exit	When ColdReset is asserted, move to the ColdReset state. When the 2 μs timeout timer expires, move to the WaitStart state.

5.9.2.1.4 WaitStart State

a. The Lane Initialisation state machine shall enter the WaitStart state on one of the following conditions:

1. From the ClearLine state, after waiting there for 2μs.

b. When in the WaitStart state the Lane Initialisation state machine shall initiate the following actions:

   1. Do NOT reset the Lane Configuration registers.

   2. Disable the transmitter driver, receiver and related transmit and receive PLLs.

   3. Switch off receiver bit inversion.

c. The Lane Initialisation state machine shall leave the WaitStart state on one of the following conditions, which are to be evaluated in the order given:

   1. When ColdReset is asserted, move to the ColdReset state.

   2. When WarmReset is asserted, move to the ClearLine state.

   3. When Lane_Start is asserted, move to the Start state.

   4. When Auto_Start is asserted and Lane_Start is NOT asserted, move to the AutoStart state.

      NOTE The Start state and AutoStart states are sub-states of the Initialise state as detailed in Figure 5-9.

d. The WaitStart state is summarised in Table 5-21.

Table 5-21 WaitStart State

State	WaitStart
Entry	From the ClearLine state, after waiting there for 2μs.
Action	Do NOT reset the Lane Configuration registers. Disable the transmitter driver, receiver and related transmit and receive PLLs. Switch off receiver bit inversion.
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When Lane_Start is asserted, move to the Start state. When Auto_Start is asserted and Lane_Start is NOT asserted, move to the AutoStart state.

5.9.2.1.5 Initialise State

a. The Initialise state is a super-state, which is responsible for initialising the lane to prepare it for sending and receiving data, and which shall operate as detailed in Figure 5-9.

5.9.2.1.6 Active State

a. The Lane Initialisation state machine shall enter the Active state when initialisation is complete or following successful error recovery.

b. The Lane Initialisation state machine shall enter the Active state on one of the following conditions:

1. From BothEndsStarted state, when at least sixteen IACK control words have been send AND two IACK control words have been received.
2. From the HalfConnected state, when sixteen IACK control words have been sent.

   NOTE The BothEndsStarted and Connected State states are sub-states of the Initialise state as detailed in Figure 5-9.

c. When in the Active state the Lane Initialisation state machine shall initiate the following actions:

1. Stop the initialisation timeout timer.
2. Stop the recovery timeout timer.
3. Enable the transmitter, transmitter PLL, receiver, and receiver PLL.
4. Enable transmission of data and control words from the lane control or retry layer.
5. Pass received data and control words up to the lane control or retry layer.
6. Filter out Lane control words (see Table 5-1) so that they are not passed up to the lane control or retry layer.

   NOTE They are used by the Lane Initialisation state machine and nothing is passed up to the lane control or retry layer in their place.

d. The Lane Initialisation state machine shall leave the Active state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When a STANDBY control word is received, move to the ClearLine state.
4. When a LOST_SIGNAL control word is received, move to the ClearLine state.
5. When No Signal At Receiver signal is asserted indicating that there is a loss of signal at the receiver, move to the LossOfSignal state.
6. When the Lane_Start and Auto_Start signals are both de-asserted, move to the PrepareStandby state.
7. When an RCLR control word is received while in the Active state, move to the StartRecovery state.

   NOTE The StartRecovery, NearEndRecovering state and FarEndRecovering states are sub-states of the Recover Error state as detailed in Figure 5-10

8. When there is a persistent problem with symbol synchronisation in the receiver resulting in more than more than eight consecutive RXERR control words, move to the StartRecovery state.

   NOTE If there are more than more than eight consecutive RXERR control words, there is a significant problem and it is necessary to recover from the error. If there are fewer than eight consecutive RXERR, it is possible that transient error will not affect the high level information being sent i.e. might be idles in an IDLE frame. Any significant error will then be picked up by the Data Word Identification state machine in the retry layer, causing error recovery.

9. When an error occurs in an upper layer, move to the StartRecovery state.

e. The Active state is summarised in Table 5-22.

Table 5-22 Active State

State	Active
Entry	From BothEndsStarted state, when at least sixteen IACK control words have been send AND two IACK control words have been received. From the HalfConnected state, when sixteen IACK control words have been sent.
Action	Stop the initialisation timeout timer. Stop the recovery timeout timer. Enable the transmitter, transmitter PLL, receiver, and receiver PLL. Enable transmission of data and control words from the lane control or retry layer. Pass received data and control words up to the lane control or retry layer. Filter out Lane control words (see Table 5-1) so that they are not passed up to the lane control or retry layer.
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When a STANDBY control word is received, move to the ClearLine state. When a LOST_SIGNAL control word is received, move to the ClearLine state. When No Signal At Receiver signal is asserted indicating that there is a loss of signal at the receiver, move to the LossOfSignal state. When the Lane_Start and Auto_Start signals are both de-asserted, move to the PrepareStandby state.

	When an RCLR control word is received while in the Active state, move to the StartRecovery state. When there is a persistent problem with symbol synchronisation in the receiver resulting in more than more than eight consecutive RXERR control words, move to the StartRecovery state. When an error occurs in an upper layer, move to the StartRecovery state.

5.9.2.1.7 PrepareStandby state

a. The Lane Initialisation state machine shall enter the PrepareStandby state on one of the following conditions:

1. From the Active state, when Lane_Start and Auto_Start signals are both de-asserted.

b. When in the PrepareStandby state the Lane Initialisation state machine shall initiate the following actions:

1. Send 32 STANDBY control words.

NOTE 32 STANBY control words are sufficient to make sure that the first STANDBY control word has passed through the receive pipeline and receive elastic buffer at the far end of the lane, before the subsequent No Signal at receiver, caused by the near end of the lane entering the ClearLine state, is detected by the far end receiver.

c. The Lane Initialisation state machine shall leave the PrepareStandby state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When 32 STANDBY control words have been sent, move to the ClearLine state.

d. The PrepareStandby state is summarised in Table 5-23.

Table 5-23 PrepareStandby State

State	PrepareStandby
Entry	From the Active state, when Lane_Start and Auto_Start signals are both de-asserted.
Action	Send 32 STANDBY control words.
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When 32 STANDBY control words have been sent, move to the ClearLine state.

5.9.2.1.8 LossOfSignal state

a. The Lane Initialisation state machine shall enter the LossOfSignal state on one of the following conditions:

   1. From the Active state when No Signal at Receiver signal is asserted, indicating that there is no signal present on the receiver inputs.

b. When in the LossOfSignal state the Lane Initialisation state machine shall initiate the following actions:

   1. Send 32 LOST_SIGNAL control words.

c. The Lane Initialisation state machine shall leave the LossOfSignal state on one of the following conditions, which are to be evaluated in the order given:

   1. When ColdReset is asserted, move to the ColdReset state.
   2. When WarmReset is asserted, move to the ClearLine state.
   3. When 32 LOST_SIGNAL control words have been sent, move to the ClearLine state.

d. The LossOfSignal state is summarised in Table 5-24.

Table 5-24 LossOfSignal State

State	LossOfSignal
Entry	From the Active state when No Signal at Receiver signal is asserted, indicating that there is no signal present on the receiver inputs.
Action	Send 32 LOST_SIGNAL control words.
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When 32 LOST_SIGNAL control words have been sent, move to the ClearLine state.

5.9.2.1.9 RecoverError State

a. The RecoverError state is a super-state, which is responsible for recovering from temporary errors and which shall operate as detailed in Figure 5-10.

5.9.2.1.10 Start state

a. The Lane Initialisation state machine shall enter the Start state on one of the following conditions:

   1. From the WaitStart state, when Lane_Start is asserted.

b. When in the Start state the Lane Initialisation state machine shall initiate the following actions:

1. Start a 20 μs initialisation timeout timer, on entry to the state.

   NOTE 20 μs allows plenty of time for the receive clock recovery at the far end of the lane to lock onto the transmitted signal, a response to be generated and returned to the near end.

2. Send IDLE control words.
3. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the Start state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When Lane_Start is not-asserted and AutoStart is not-asserted, move to the WaitStart state.
4. When two IDLE control words are received, move to the NearEndStarted state.
5. When two inverse IDLE control words are received, move to the StartPolarity state.
6. When two inverse INIT control words are received, move to the StartPolarity state.
7. When two INIT control words are received, move to the BothEndsStarted State.
8. When the initialisation timeout timer expires, move to the ClearLine state.

d. The Start state is summarised in Table 5-25.

Table 5-25 Start State

State	Start
Entry	From the WaitStart state, when Lane_Start is asserted.
Action	Start a 20 μs initialisation timeout timer, on entry to the state. Send IDLE control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When Lane_Start is not-asserted and AutoStart is not-asserted, move to the WaitStart state. When two IDLE control words are received, move to the NearEndStarted state. When two inverse IDLE control words are received, move to the StartPolarity state. When two inverse INIT control words are received, move to the StartPolarity state. When two INIT control words are received, move to the

	BothEndsStarted State. When the initialisation timeout timer expires, move to the ClearLine state.

5.9.2.1.11 AutoStart state

a. The Lane Initialisation state machine shall enter the AutoStart state on one of the following conditions:

   1. From the WaitStart state, when AutoStart is asserted and Lane_Start is NOT asserted.

b. When in the AutoStart state the Lane Initialisation state machine shall initiate the following actions:

   1. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the AutoStart state on one of the following conditions, which are to be evaluated in the order given:

   1. When ColdReset is asserted, move to the ColdReset state.
   2. When WarmReset is asserted, move to the ClearLine state.
   3. When Auto_Start is de-asserted and Lane_Start is NOT asserted, move to the WaitStart state.
   4. When Lane_Start is received, move to the Start state.
   5. When two IDLE control words are received, move to the FarEndStarted state.
   6. When two inverse IDLE control words are received, move to the AutoStartPolarity state.

d. The AutoStart state is summarised in Table 5-26.

Table 5-26 AutoStart State

State	AutoStart
Entry	From the WaitStart state, when AutoStart is asserted and Lane_Start is NOT asserted.
Action	Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When Auto_Start is de-asserted and Lane_Start is NOT asserted, move to the WaitStart state. When Lane_Start is received, move to the Start state. When two IDLE control words are received, move to the FarEndStarted state. When two inverse IDLE control words are received, move to the AutoStartPolarity state.

5.9.2.1.12 StartPolarity state

a. The Lane Initialisation state machine shall enter the StartPolarity state on one of the following conditions:

   1. From the Start state, when two inverse IDLE control words are received.

   2. From the Start state, when two inverse INIT control words are received.

b. When in the StartPolarity state the Lane Initialisation state machine shall initiate the following actions:

   1. Start a 1 μs polarity timeout timer.

      NOTE 1 μs is sufficient for the change in receiver polarity to propagate through the receive pipeline and receive elastic buffer.

   2. Switch on receiver bit inversion.

   3. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the StartPolarity state on one of the following conditions, which are to be evaluated in the order given:

   1. When ColdReset is asserted, move to the ColdReset state.

   2. When WarmReset is asserted, move to the ClearLine state.

   3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

   4. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control word, move to the ClearLine state.

   5. When two IDLE control words are received, move to the NearEndStarted state.

   6. When two INIT control words are received, move to the BothEndsStarted state.

   7. When the polarity timeout timer expires, move to the ClearLine state.

   8. When the initialisation timeout timer expires, move to the ClearLine state.

d. The StartPolarity state is summarised in Table 5-27.

Table 5-27 StartPolarity State

State	StartPolarity
Entry	From the Start state, when two inverse IDLE control words are received. From the Start state, when two inverse INIT control words are received.
Action	Start a 1 μs polarity timeout timer.

	Switch on receiver bit inversion. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control word, move to the ClearLine state. When two IDLE control words are received, move to the NearEndStarted state. When two INIT control words are received, move to the BothEndsStarted state. When the polarity timeout timer expires, move to the ClearLine state. When the initialisation timeout timer expires, move to the ClearLine state.

5.9.2.1.13 AutoStartPolarity state

a. The Lane Initialisation state machine shall enter the AutoStartPolarity state on one of the following conditions:

   1. From the AutoStart state, when two inverse IDLE control words are received.

b. When in the AutoStartPolarity state the Lane Initialisation state machine shall initiate the following actions:

   1. Start a 1 μs polarity timeout timer.

      NOTE 1 μs is sufficient for the change in receiver polarity to propagate through the receive pipeline and receive elastic buffer.

   2. Switch on receiver bit inversion.

   3. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the AutoStartPolarity state on one of the following conditions, which are to be evaluated in the order given:

   1. When ColdReset is asserted, move to the ColdReset state.

   2. When WarmReset is asserted, move to the ClearLine state.

   3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

   4. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state.

5. When two IDLE control words are received, move to the FarEndStarted state.

6. When the polarity timeout timer expires, move to the ClearLine state.

d. The AutoStartPolarity state is summarised in Table 5-28.

Table 5-28 AutoStartPolarity State

State	AutoStartPolarity
Entry	From the AutoStart state, when two inverse IDLE control words are received.
Action	Start a 1 μs polarity timeout timer. Switch on receiver bit inversion. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state. When two IDLE control words are received, move to the FarEndStarted state. When the polarity timeout timer expires, move to the ClearLine state.

5.9.2.1.1 NearEndStarted state

a. The Lane Initialisation state machine shall enter the NearEndStarted state on one of the following conditions:

1. From the Start state, when two IDLE control words are received.

2. From the StartPolarity state, when two IDLE control words are received.

b. When in the NearEndStarted state the Lane Initialisation state machine shall initiate the following actions:

1. Send INIT control words.

2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the NearEndStarted state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.

2. When WarmReset is asserted, move to the ClearLine state.

3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

4. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state.

5. When two INIT control words are received, move to the BothEndsStarted state.

6. When two IACK control words are received, move to the HalfConnected state.

7. When the initialisation timeout timer expires, move to the ClearLine state.

d. The NearEndStarted state is summarised in Table 5-30.

Table 5-29 NearEndStarted State

State	NearEndStarted
Entry	From the Start state, when two IDLE control words are received. From the StartPolarity state, when two IDLE control words are received.
Action	Send INIT control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state. When two INIT control words are received, move to the BothEndsStarted state. When two IACK control words are received, move to the HalfConnected state. When the initialisation timeout timer expires, move to the ClearLine state.

5.9.2.1.2 BothEndsStarted state

a. The Lane Initialisation state machine shall enter the BothEndsStarted state on one of the following conditions:

1. From the Start state, when two INIT control words are received.

2. From the StartPolarity state, when two INIT control words are received.

b. When in the BothEndsStarted state the Lane Initialisation state machine shall initiate the following actions:

1. Send IACK control words.

2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the BothEndsStarted state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When No Signal is detected at the receiver inputs, move to the ClearLine state.
4. When a receive error is detected by the receive synchronisation state machine which results in eight conecutive RXERR control words, move to the ClearLine state.
5. When two IACK control words are received and at least sixteen IACK control words have been sent, move to the Active state.
6. When the initialisation timeout timer expires, move to the ClearLine state.

d. The BothEndsStarted state is summarised in Table 5-30.

Table 5-30 BothEndsStarted State

State	BothEndsStarted
Entry	From the Start state, when two INIT control words are received. From the StartPolarity state, when two INIT control words are received.
Action	Send IACK control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in eight conecutive RXERR control words, move to the ClearLine state. When two IACK control words are received and at least sixteen IACK control words have been sent, move to the Active state. When the initialisation timeout timer expires, move to the ClearLine state.

5.9.2.1.3 FarEndStarted state

a. The Lane Initialisation state machine shall enter the FarEndStarted state on one of the following conditions:

1. From the AutoStart state, when two IDLE control words are received.

2. From the AutoStartPolarity state, when two IDLE control words are received.

b. When in the FarEndStarted state the Lane Initialisation state machine shall initiate the following actions:

1. Start a 20 μs initialisation timeout timer, on entry to the state.

2. Send IDLE control words.

3. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the FarEndStarted state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.

2. When WarmReset is asserted, move to the ClearLine state.

3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

4. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state.

5. When two INIT control words are received, move to the BothEndsStarted state.

6. When the initialisation timeout timer expires, move to the ClearLine state.

d. The FarEndStarted state is summarised in Table 5-31.

Table 5-31 FarEndStarted State

State	FarEndStarted
Entry	From the AutoStart state, when two IDLE control words are received. From the AutoStartPolarity state, when two IDLE control words are received.
Action	Start a 20 μs initialisation timeout timer, on entry to the state. Send IDLE control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state. When two INIT control words are received, move to the BothEndsStarted state. When the initialisation timeout timer expires, move to the ClearLine state.

5.9.2.1.4 HalfConnected State

a. The Lane Initialisation state machine shall enter the HalfConnected State on one of the following conditions:

 1. From the NearEndStarted state, when two IACK control words are received.

b. When in the HalfConnected State the Lane Initialisation state machine shall initiate the following actions:

 1. Send sixteen IACK control words.
 2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the HalfConnected State on one of the following conditions, which are to be evaluated in the order given:

 1. When ColdReset is asserted, move to the ColdReset state.
 2. When WarmReset is asserted, move to the ClearLine state.
 3. When No Signal is detected at the receiver inputs, move to the ClearLine state.
 4. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state.
 5. When sixteen IACK control words have been sent, move to the Active state.
 6. When the initialisation timeout timer expires, move to the ClearLine state.

d. The HalfConnected State is summarised in Table 5-30.

Table 5-32 HalfConnected State

State	HalfConnected
Entry	From the NearEndStarted state, when two IACK control words are received.
Action	Send sixteen IACK control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in more than eight consecutive RXERR control words, move to the ClearLine state. When sixteen IACK control words have been sent, move to the Active state. When the initialisation timeout timer expires, move to the

	ClearLine state.

5.9.2.1.5 StartRecovery state

a. The Lane Initialisation state machine shall enter the StartRecovery state on one of the following conditions:

  1. From the Active state, when a receive error is detected by the Receive Synchronisation state machine, which results in more than eight consecutive RXERR control words.

  2. From the Active state, when an RCLR control word is received.

  3. From the Active state, when an error is detected in an upper layer of the SpaceFibre protocol.

b. When in the StartRecovery state the Lane Initialisation state machine shall initiate the following actions, which are to be evaluated in the order given:

  1. Start a 2 μs recovery timeout timer.

  2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the StartRecovery state on one of the following conditions:

  1. When ColdReset is asserted, move to the ColdReset state.

  2. When WarmReset is asserted, move to the ClearLine state.

  3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

  4. When receiving eight or more RXERR control caused entry to the StartRecovery state, move to the NearEndRecovering State.

  5. When an error being detected in an upper layer protocol caused entry to the StartRecovery state, move to the NearEndRecovering State.

  6. When an RCLR control word caused entry to the StartRecovery state, move to the FarEndRecovering State.

a. The StartRecovery state is summarised in Table 5-30.

Table 5-33 StartRecovery State

State	StartRecovery
Entry	From the Active state, when a receive error is detected by the Receive Synchronisation state machine, which results in more than eight consecutive RXERR control words. From the Active state, when an RCLR control word is received. From the Active state, when an error is detected in an upper layer of the SpaceFibre protocol.
Action	Start a 2 μs recovery timeout timer. Receive lane layer control words (LLCW).

Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When receiving eight or more RXERR control caused entry to the StartRecovery state, move to the NearEndRecovering State. When an error being detected in an upper layer protocol caused entry to the StartRecovery state, move to the NearEndRecovering State. When an RCLR control word caused entry to the StartRecovery state, move to the FarEndRecovering State.

5.9.2.1.6 NearEndRecovering state

a. The Lane Initialisation state machine shall enter the NearEndRecovering state on one of the following conditions:

1. From the StartRecovery state, when more than eight consecutive RXERRs control words caused entry to the StartRecovery state.
2. From the StartRecovery state, when en error in an upper layer protocol caused entry to the StartRecovery state.
3. From the NearEndRecovering state when an RXERR is received after one or more other control or data words have been received.

NOTE I.e. when a new RXERR occurs sends a new RCLR.

b. When in the NearEndRecovering state the Lane Initialisation state machine shall initiate the following actions, which are to be evaluated in the order given:

1. Send nine RCLR control words.
2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the NearEndRecovering state on one of the following conditions:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When No Signal is detected at the receiver inputs, move to the ClearLine state.
4. When an RCLR control word has been received, move to the FarEndRecovering State.
5. When nine RCLR control words have been sent and a RACK control word has been received, move to the Active state.
6. When the recovery timeout timer expires, move to the ClearLine state.

d. The NearEndRecovering state is summarised in Table 5-30.

Table 5-34 NearEndRecovering State

State	NearEndRecovering
Entry	From the StartRecovery state, when more than eight consecutive RXERRs control words caused entry to the StartRecovery state. From the StartRecovery state, when en error in an upper layer protocol caused entry to the StartRecovery state. From the NearEndRecovering state when an RXERR is received after one or more other control or data words have been received.
Action	Send nine RCLR control words. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When an RCLR control word has been received, move to the FarEndRecovering State. When nine RCLR control words have been sent and a RACK control word has been received, move to the Active state. When the recovery timeout timer expires, move to the ClearLine state.

5.9.2.1.7 FarEndRecovering state

a. The Lane Initialisation state machine shall enter the FarEndRecovering state on one of the following conditions:

1. From the StartRecovery state, when receipt of an RCLR control word caused entry to the StartRecovery state.
2. From the NearEndRecovering state, when an RCLR control word is received.

b. When in the FarEndRecovering state the Lane Initialisation state machine shall initiate the following actions:

1. Send one RACK control word.
2. Receive lane layer control words (LLCW).

c. The Lane Initialisation state machine shall leave the FarEndRecovering state on one of the following conditions, which are to be evaluated in the order given:

1. When ColdReset is asserted, move to the ColdReset state.
2. When WarmReset is asserted, move to the ClearLine state.
3. When No Signal is detected at the receiver inputs, move to the ClearLine state.

4. When a receive error is detected by the receive synchronisation state machine which results in an RXERR control word, move to the NearEndRecovering state.

5. When one RACK control word has been sent, move to the Active state.

6. When the recovery timeout timer expires, move to the ClearLine state.

d. The FarEndRecovering state is summarised in Table 5-31.

Table 5-35 FarEndRecovering State

State	FarEndRecovering
Entry	From the StartRecovery state, when receipt of an RCLR control word caused entry to the StartRecovery state. From the NearEndRecovering state, when an RCLR control word is received.
Action	Send one RACK control word. Receive lane layer control words (LLCW).
Exit	When ColdReset is asserted, move to the ColdReset state. When WarmReset is asserted, move to the ClearLine state. When No Signal is detected at the receiver inputs, move to the ClearLine state. When a receive error is detected by the receive synchronisation state machine which results in an RXERR control word, move to the NearEndRecovering state. When one RACK control word has been sent, move to the Active state. When the recovery timeout timer expires, move to the ClearLine state.

5.9.3 Data rate adjustment

a. The transmitters at each end of a lane shall send data at the same nominal data signalling rate.

b. The maximum permitted difference in the data signalling rates, when each end of the lane is operating at nominally the same data signalling rate (e.g. 2 Gbits/s) shall be 100 parts per million.

NOTE The data signalling rate of a transmitter at one end of a lane and a receiver at the other end of a lane might be different due to differences in the local clocks being used at each end of the lane.

c. A receive elastic buffer shall be used to compensate for differences in the data signalling rate at each end of the lane.

d. Skip control words shall be inserted regularly in the transmit data stream.

NOTE When a SKIP arrives at the receiver at the other end of the lane it might have been inverted by PCB layout to form an inverse SKIP control word.

e. A skip control word shall be sent every 5,000 words.

f. When a skip or inverse skip control word is read from a receive elastic buffer that is more than half full, the skip or inverse skip control word shall be removed from the receive elastic buffer and the following character shall be read from the buffer.

g. When a skip or inverse skip control word is read from a receive elastic buffer that is less than half full, the skip or inverse skip control word shall be left in the buffer, so that it can be read again the next time the buffer is read.

h. A particular skip or inverse skip control word shall be read at most twice, i.e. the second time it is read it is removed from the receive elastic buffer.

5.9.4 Idle words

a. When there is no other data or control word to send an idle word shall be sent.

NOTE It is essential not to have gaps in the data being sent because the PLL in the receiver has to maintain lock on the incoming data stream.

b. When received, idle or inverse idle control words shall be discarded when they are read out of the receive elastic buffer.

NOTE Although only IDLEs are sent they might be inverted into inverse IDLEs by the time they reach the receiver.

5.9.5 Parallel loopback

a. A parallel loopback facility should be provided in the SpaceFibre CODEC for test purposes.

NOTE This facility is optional.

b. When enabled, the parallel loopback shall pass the stream of words to be transmitted from the lane layer to the receive input of the lane layer.

NOTE This connects the output of the transmit side of the lane layer to the input of the receive side of the lane layer as illustrated in Figure 4-2.

5.10 Encoding layer

5.10.1 Encoding layer service interface

The service primitives that shall be associated with the encoding layer service are:

TX_WORD.request;

RX_WORD.indication;

SYNC.indication.

5.10.1.1 TX_WORD.request

5.10.1.1.1 Function

The lane layer shall pass a TX_WORD.request primitive to the encoding layer to send a data word or control word over the lane.

5.10.1.1.2 Semantics

The TX_WORD.request primitive shall provide the following parameters:

TX_WORD.request (Word)

5.10.1.1.3 When Generated

The TX_WORD.request primitive shall be passed to the encoding layer when the lane layer has information to send over the lane.

5.10.1.1.4 Effect On Receipt

On receipt of the TX_WORD.request primitive, the encoding layer shall encode the data word or control and send it over the lane.

5.10.1.2 RX_WORD.indication

5.10.1.2.1 Function

The encoding layer shall pass a RX_WORD.indication primitive to the lane layer to indicate that a data word has been received.

5.10.1.2.2 Semantics

The RX_WORD.indication primitive provides parameters as follows:

RX_ WORD.indication (Word).

5.10.1.2.3 When Generated

The RX_ WORD.indication primitive shall be passed to the lane layer when a data or control word is received.

5.10.1.2.4 Effect On Receipt

The effect on receipt of the RX_ WORD.indication primitive by the lane layer shall be that the received word is placed in the receive elastic buffer.

5.10.1.3 SYNC.indication

5.10.1.3.1 Function

The encoding layer shall pass a SYNC.indication primitive to the lane layer to indicate that the receiver is synchronised to the incoming data stream and is receiving data and control words.

5.10.1.3.2 Semantics

The SYNC.indication primitive provides parameters as follows:

SYNC.indication (Sync Status).

5.10.1.3.3 When Generated

The SYNC.indication primitive shall be passed to the lane layer when the receiver synchronisation state changes, i.e. when the receiver either achieves synchronisation or loses synchronisation.

5.10.1.3.4 Effect On Receipt

On receipt of the SYNC.indication primitive, the lane layer shall react depending on the lane status. If the lane is synchronised, the lane layer will start sending data frames, broadcast frames, FCTs, ACKs and NACKs to the encoding layer. If the lane has lost synchronisation, the lane layer will try to re-initialise the link.

5.10.2 8B/10B encode/decode

a. The SpaceFibre CODEC shall use 8B/10B encoding to encode each 8-bit data character or control code into a 10-bit symbol that is transmitted.

b. To ensure DC balancing of the transmitted signal account shall be kept of the current running disparity in the transmitter.

c. If the current running disparity is positive when encoding an 8-bit character or control code, the symbol for that data character or control code which has negative disparity shall be used.

d. If the current running disparity is positive when encoding an 8-bit character or control code, and there is no symbol for that data character or control code which has negative disparity, the symbol with neutral shall be used.

e. If the current running disparity is negative when encoding an 8-bit character or control code, the symbol for that data character or control code which has positive disparity shall be used.

f. If the current running disparity is negative when encoding an 8-bit character or control code, and there is no symbol for that data character or control code which has positive disparity, the symbol with neutral shall be used.

g. To detect disparity errors account shall be kept of the current running disparity in the receiver.

h. If the current running disparity is more than plus one or less minus one, this shall indicate a disparity error.

i. If a disparity error occurs, it shall be indicated to the receive synchronisation state machine and the current symbol shall be set to K0.0.

j. When a symbol is received it shall be decoded into an 8-bit data character or control code using the 8B/10B symbol table.

k. If an unrecognised symbol is received then a symbol error shall be indicated to the receive synchronisation state machine and the current symbol shall be set to K0.0.

l. The 8B/10B encoding shall follow the encoding detailed in table TBC

NOTE The 8B/10B encoding table has yet to be added.

5.10.3 Symbol synchronisation

a. The boundary between symbols shall be determined by detecting the unique comma sequences.

b. The **Positive Comma** sequence is 0011111010.

c. The **Negative Comma** sequence is 1100000101.

NOTE The full 10 bit comma sequences are used for symbol synchronisation.

d. Both positive and negative commas shall be detected and used for synchronisation.

e. Synchronisation may be performed on only positive commas or only negative commas when using legacy SerDes devices.

f. The form of symbol synchronisation, positive or negative commas or both, shall be indicated to the far end of the lane in the capability field of IACK during lane initialisation.

g. The 10-bit wide input stream shall be aligned or realigned to form a stream of correctly aligned symbols so that each 10-bit group contains one complete symbol.

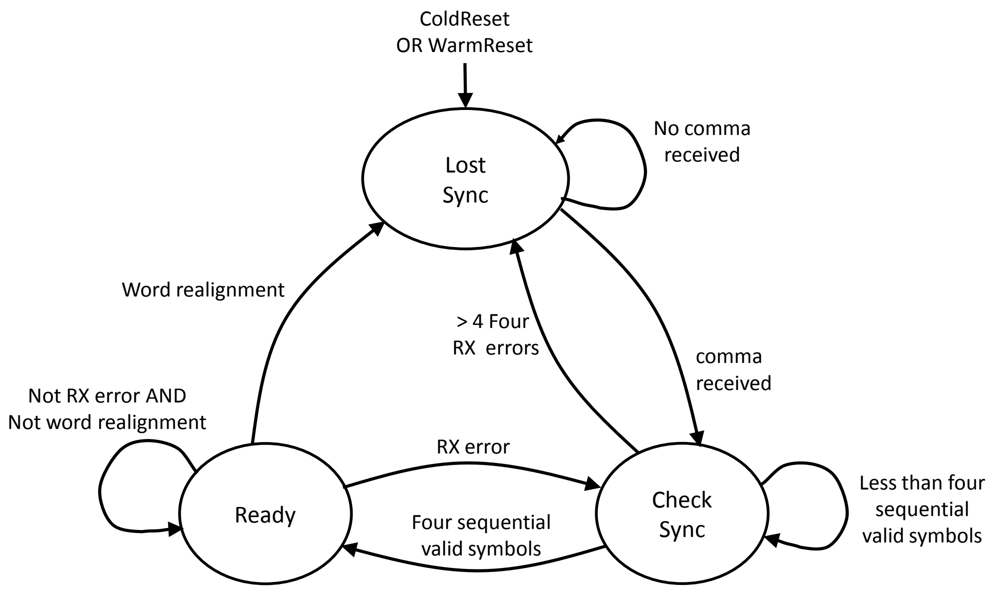
h. The 10-bit wide input stream shall be realigned every time a comma sequence is detected in a different position to the position of the last comma detected.

i. When a 20-bit or 40-bit interface is being used from the 8B/10B receiver, realignment may occur on the first comma in a word, i.e. when there are two or more commas in a 20-bit or 40-bit word it will be the comma in the lower significant bit position that is used for symbol realignment.

5.10.3.2 Receive Synchronisation State Machine

a. A receive synchronisation state machine shall be used to determine when incoming symbols are properly synchronised.

NOTE The state diagram for the receive synchronisation state machine is illustrated in Figure 5-11.



RX error = disparity or invalid symbol error

Word realignment = word synchronisation lost (i.e. comma in position other than LS byte)

Figure 5-11 Receive Synchronisation State Machine

5.10.3.2.1 LostSync State

a. The LostSync state shall be entered on one of the following conditions:

1. ColdReset command.
2. WarmReset command.
3. From the CheckSync state, when a total of more than four symbols are received that contain a disparity error or are invalid.
4. From the Ready state, when word realignment occurs.

b. When in the LostSync state the Receive Synchronisation state machine shall initiate the following actions:

1. Replace any received data and control words with RXERR control words.

c. The Receive Synchronisation state machine shall leave the LostSync State on one of the following conditions, which shall be evaluated in the order given:

1. When a comma is received, move to the CheckSync state.

d. The LostSync state is summarised in Table 5-36.

Table 5-36 LostSync State

State	LostSync
Entry	ColdReset command. WarmReset command. From the CheckSync state, when a total of more than four symbols are received that contain a disparity error or are

	invalid. From the Ready state, when word realignment occurs.
Action	Replace received data and control words with RXERR control words.
Exit	When a comma is received, move to the CheckSync state.

5.10.3.2.2 CheckSync State

a. The CheckSync state shall be entered on one of the following conditions:

  1. From the LostSync state, when a comma is received.
  2. From the Ready state, when a disparity error occurs or an invalid symbol is detected.

b. When in the CheckSync state the Receive Synchronisation state machine shall initiate the following actions:

  1. Replace any received words with RXERR control words.
  2. Count the number of symbols received that are invalid or contain a disparity error.

c. The Receive Synchronisation state machine shall leave the CheckSync State on one of the following conditions, which shall be evaluated in the order given:

  1. When ColdReset is asserted, move to the LostSync state.
  2. When WarmReset is asserted, move to the CheckSync state.
  3. When a total of more than four symbols are received that are invalid or contain a disparity error, move to the LostSync state.

     NOTE These four invalid or erroneous symbols need not be sequential i.e. there may be valid symbols interspersed amongst them

  4. When four consecutive valid symbols have been received, move to the Ready state.

d. The CheckSync state is summarised in Table 5-37.

Table 5-37 CheckSync State

State	CheckSync
Entry	From the LostSync state, when a comma is received. From the Ready state, when a disparity error occurs or an invalid symbol is detected.
Action	Replace any received words with RXERR control words. Count the number of symbols received that are invalid or contain a disparity error.
Exit	When ColdReset is asserted, move to the LostSync state. When WarmReset is asserted, move to the CheckSync state. When a total of more than four symbols are received that are

	invalid or contain a disparity error, move to the LostSync state. When four consecutive valid symbols have been received, move to the Ready state.

5.10.3.2.3 Ready State

a. The Ready state shall be entered on one of the following conditions:

   1. From the CheckSync state, when four consecutive valid symbols have been detected.

b. When in the Ready state the Receiver Synchronisation state machine shall initiate the following actions:

   1. Receive symbols.

c. The Receiver Synchronisation state machine shall leave the Ready state on one of the following conditions, which shall be evaluated in the order given:

   1. When ColdReset is asserted, move to the LostSync state.
   2. When WarmReset is asserted, move to the CheckSync state.
   3. When a disparity error occurs or an invalid symbol is received, move to the CheckSync state.
   4. When word realignment occurs, move to the LostSync state.

d. The Ready state is summarised in Table 5-38.

Table 5-38 Ready State

State	Ready
Entry	From the CheckSync state, when four consecutive valid symbols have been detected.
Action	Receive symbols.
Exit	When ColdReset is asserted, move to the LostSync state. When WarmReset is asserted, move to the CheckSync state. When a disparity error occurs or an invalid symbol is received, move to the CheckSync state. When word realignment occurs, move to the LostSync state.

5.10.4 Word Synchronisation

a. A symbol word shall comprise four symbols.

b. The first symbol to be transmitted in a symbol word shall be the least significant symbol.

c. A comma shall only occur in the least significant symbol position of a symbol word.

d. Word synchronisation shall be performed whenever a comma is received.

e. Word synchronisation shall be achieved by selecting symbols in consecutive groups of four so that the comma appears in the least significant symbol position.

f. On word synchronisation, the first word shall comprise the comma control symbol in the least significant symbol position, together with the following three symbols.

g. Each subsequent group of four symbols shall form each of the following symbol words.

h. Word realignment shall occur when a comma occurs in any position other than the least significant symbol position of a word.

i. If a word contains a K0.0 symbol indicating an error that word together with the previous word shall be set to the RXERR control word.

NOTE This means that if a word contains a K0.0 symbol, all the symbols in that word and the previous word will be set to K0.0. This is used to indicate a receiver error to the higher layers.

5.11 Serialisation layer

5.11.1 SerDes interface

a. The SerDes Interface shall pass coded, but unsynchronised, symbols between the Encoding and Serialisation parts of the SpaceFibre CODEC.

b. The SerDes interface shall comprise a Transmit SerDes interface and a Receive SerDes interface.

c. The SerDes interfaces shall be 10-bit, 20-bit or 40-bit wide.

5.11.1.2 Transmit SerDes Input

The Transmit SerDes Input shall contain the signals listed in Table 5-39, Table 5-40, or Table 5-41.

Table 5-39 Transmit SerDes Interface (10-bit)

Signal	Dir	Function
SerDes_Txdata(9:0)	In	10-bit wide data containing one symbol for transmission.

Table 5-40 Transmit SerDes Interface (20-bit)

Signal	Dir	Function
SerDes_Txdata(19:0)	In	20-bit wide data containing two symbols for transmission.

Table 5-41 Transmit SerDes Interface (40-bit)

Signal	Dir	Function
SerDes_Txdata(39:0)	In	40-bit wide data containing four symbols for transmission.

5.11.1.3 Receive SerDes Output

The Receive SerDes Output shall contain the signals listed in Table 5-42, Table 5-43, or Table 5-44.

Table 5-42 Receive SerDes Interface (10-bit)

Signal	Dir	Function
SerDes_Rxdata(9:0)	Out	10-bits of received data. This data is NOT symbol synchronised i.e. the 10-bits can contain some bits from one symbol and the rest of the bits from the next symbol.

Table 5-43 Receive SerDes Interface (20-bit)

Signal	Dir	Function
SerDes_Rxdata(19:0)	Out	20-bits of received data. This data is NOT symbol synchronised i.e. the 20-bits can contain some bits from one symbol, the following complete symbol, and the rest of the bits from the next symbol.

Table 5-44 Receive SerDes Interface (40-bit)

Signal	Dir	Function
SerDes_Rxdata(39:0)	Out	40-bits of received data. This data is NOT symbol synchronised i.e. the 40-bits can contain some bits from one symbol, the following complete three symbols, and the rest of the bits from the next symbol.

5.11.2 Bit Synchronisation

a. The receive clock used to sample the incoming bit stream, shall be generated by a clock recovery circuit that matches the phase of a local receive clock to the transitions of the incoming bit stream.

b. Sampling of the bit stream should be close (+/- 20% TBC) to the centre of the bit period.

c. The clock recovery circuit should indicate in a status register when bit synchronisation is achieved.

5.11.3 Serialiser/deserialiser

a. 10-bit symbols shall be transmitted serially over the physical medium.

b. At the transmitter a serialiser shall be used to convert each parallel 10-bit symbol into a serial bit stream with each bit of the symbol being send one after the other.

NOTE One, two or four symbols can be provided to the serialiser in parallel.

c. The least significant bit of the 10-bit symbol shall be transmitted first.

d. At the receiver the incoming bit stream shall be converted to a 10-bit word by sampling the bit stream with a receive clock in a deserialiser.

NOTE The deserialiser does not necessarily produce a stream of 10-bit symbols because the boundary of the symbols is not known by the deserialiser.

5.11.4 Inversion

The received symbols shall be bit-wise inverted if requested by the lane initialisation state machine.

5.11.5 Serial loopback

a. A serial loopback facility shall be provided in the SpaceFibre CODEC for test purposes.

b. The serial loopback shall connect the bit stream output from the serialiser in the transmitter directly into the serial input of the deserialiser in the receiver.

c. The serial loopback shall connect the bit stream input from the receiver directly to the transmitter driver output.

5.12 Physical layer

5.12.1 Serial Interface

a. The Serial Interface shall pass serial data out of and into the SpaceFibre CODEC.

b. The serial interface shall comprise a transmitter serial output and a receiver serial input.

5.12.1.2 Transmit Serial Output

The Transmit Serial Output shall contain the signals listed in Table 5-45.

Table 5-45 Transmit Serial Interface

Signal	Function
Txp	Positive side of the differential serial transmitter output.
Txn	Negative side of the differential serial transmitter output.

5.12.1.3 Receive Serial Input

The Receive Serial Input shall contain the signals listed in Table 5-46.

Table 5-46 Receiver Serial Interface

Signal	Function
Rxp	Positive side of the differential serial receiver input.
Rxn	Negative side of the differential serial receiver input.

5.12.2 Electrical SpaceFibre medium

5.12.2.1 Electrical SpaceFibre Driver and Receiver

The driver and receiver for SpaceFibre operation over copper shall use Current Mode Logic (CML), or compatible driver and receiver.

5.12.2.2 Electrical SpaceFibre PCB Tracks

a. The PCB tracks for electrical SpaceFibre shall be 100 ohms differential impedance.

b. Two pairs of differential PCB tracks shall be used for a bi-directional SpaceFibre link, one pair for each direction.

5.12.2.3 Single-ended Electrical Connectors

a. TBA

5.12.2.4 SpaceFibre Electrical Cables

a. TBA

5.12.2.5 Differential Electrical Connectors

a. TBA

5.12.2.6 Differential Electrical Cables

a. TBA

5.12.3 Fibre optic driver and receiver

5.12.3.1 Fibre Optic Driver and Receiver

a. TBA

5.12.3.2 Fibre Optic Connectors

a. TBA

5.12.3.3 Fibre Optic Cables

a. TBA

5.13 Management layer

NOTE This section is in the process of being written.

5.13.1 Reset

Layer	**Variable**	**Cold Reset**	**Warm Reset**
Virtual Channel	Output VCBs	Flushed	Unchanged
	Input VCBs	Flushed	Unchanged
	FCT counter	Cleared	Unchanged
	Bandwidth credit	Set to zero	Unchanged
	Input space counter	Set to buffer size	Unchanged
	FCT credit counter	Set to zero	Unchanged
Broadcast Message	BC Sequence counter	Set to zero	Unchanged
Framing			
Retry	Transmit frame sequence counter	Set to zero	Unchanged
	Receive frame sequence counter	Set to zero	Unchanged
	Number of retries	Set to zero	Unchanged
	PRBS seed	Set to 0xffff ffff	Unchanged
Lane Control	Required number of lanes	Set to one	Unchanged
	Distribution lane numbers	Clear	Clear

	Concentration lane numbers	Clear	Clear
Lane	Capability parameters	Reset	Unchanged
	Receiver bit inversion	Off	Off
Encoding			
Serialisation	PLL	Reset	Unchanged
Physical			

5.14 SpaceFibre conformance

5.14.1 Overview

5.14.2 Partial implementations

Single lane

Bit inversion

Only positive or only negative comma synchronisation

Parallel loopback

Annex A(informative) Serial Data Link Concepts

This section provides an overview of several key concepts for high-speed serial data links.

A.1 Data Scrambling

Data scrambling is a technique used to reduce the electro-magnetic (EM) emissions from a communications system. The data signal is convolved with a wideband signal which results in the spectrum of the data being broadened. Possible peaks in the EM spectrum of the original data signal are spread out reducing the energy at any single frequency. Note that data scrambling does not guarantee reduced peaks in the EM spectrum since it is possible that the scrambling produces a bit sequence with a higher spectral peaks than the original signal. However, for regular bit sequences it is likely to reduce the spectral peaks.

A random number generator is used to produce the wideband signal which is XORed with the data being transmitted. At the beginning of each frame being transmitted the random number generator is reseeded with a specific value. A similar random number generator in the receiver, seeded with the same seed as in the transmitter at the start of every new frame, is used to de-scramble the data. The incoming data is XORed with the random number sequence to reveal the original data stream.

The random number generator is implemented using a linear feedback shift register as shown in Figure A-1. An example scrambling/de-scrambling polynomial is that used in PCI-Express:

$$G(x) = X^{16} + X^5 + X^4 + X^3 + 1$$

The seed for the random number generator is $FFFF_h$ *i.e.* all flip-flops in the random number generator are set to 1.

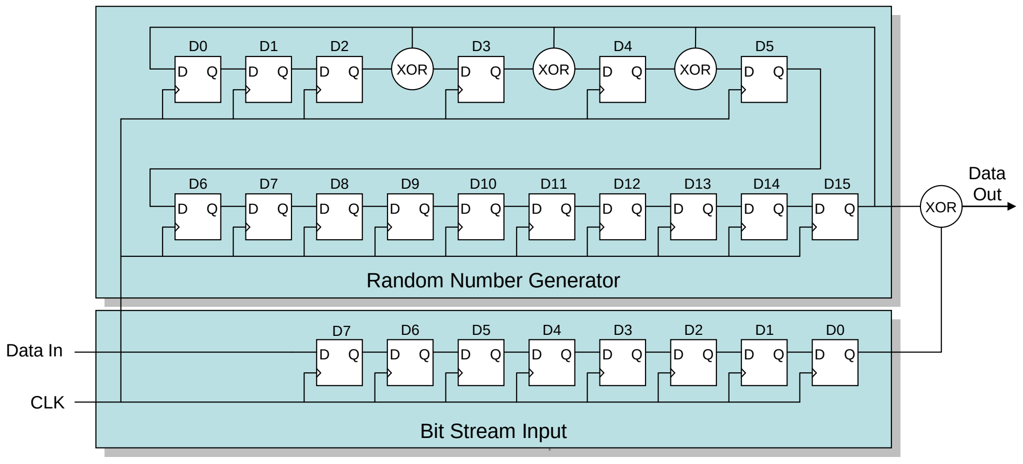


Figure A-1 Scrambler / De-Scrambler

A.2 8B/10B Encoding and Decoding

8B/10B encoding encodes 8-bit data bytes into 10-bit characters for transmission. The 8B/10B encoding has several advantages over direct 8-bit transmission.

1. It provides a transmitted data stream with roughly the same number of 1's as 0's giving the data a zero DC bias, improving the transmission characteristics and enabling AC coupling.
2. Since a 10-bit code has 1024 possible values and not all of these are needed to send an 8-bit value there are spare valid codes left over that can be used for control codes.
3. It guarantees that there will be sufficient number of bit transitions in the serial data stream to enable the recovery of the bit clock using a phase-locked loop. A maximum of five consecutive ones or zeros are ensured with 8B/10B encoding.
4. Since all characters, both data and control characters, are transmitted with 10-bits the bit and character transmission rates are constant simplifying the transmission and reception of characters.
5. Codes that are unused by the 8B/10B encoding can be used to detect link errors *i.e.* if an unused code occurs then there has been a transmission error.
6. The current running disparity following 8B/10B encoding is always +1 or -1, any other value indicates a disparity error.

To avoid significant DC components 8B/10B encoding uses only the 10-bit codes that contain either 5 ones and 5 zeros, 6 ones and 4 zeros, or 4 ones and 6 zeros. There are enough of these to encode the 8-bit data byte and several possible control codes. Characters encoded with 5 ones and 5 zeros have neutral disparity and will produce zero DC bias. However, if a sequence of bytes was transmitted that contained characters all with 6 ones and 4 zeros the DC component would slowly increase. A similar opposite effect would occur if the characters all contained 4 ones and 6 zeros. To prevent this increasing DC bias and to maintain an equal number of transmitted ones and zeros each character with an unequal number of ones and zeros has two possible codes one with 6 ones and 4 zeros and the other with 4 ones and 6 zeros.

Every time the transmitter sends a character with 6 ones and 4 zeros it will record the fact that it has sent more ones than zeros and the next time it has to send a character with an uneven number of bits it will choose the code that has 4 ones and 6 zeros. This keeps the average number of ones and zeros the same and eliminates any DC bias in the transmitted signal. The Current Running Disparity variable is set to one (positive) when more ones have been sent than zeros and to zero (negative) when more zeros have been sent than ones. Characters with 5 ones and 5 zeros have neutral disparity and do not affect the Current Running Disparity value. When a character with an unequal number of ones and zeros is to be sent, the value of the Current Running Disparity will determine which of the two possible 10-bit codes will be sent. If the Current Running Disparity is positive then the option with 4 ones and 6 zeros is sent, if it is negative then the other option with 6 ones and 4 zeros is transmitted.

Once all 256 possible values of an 8-bit data byte have been assigned a code with 5 ones and 5 zeros or a pair of codes with unequal numbers of ones and zeros, there are just 12 valid codes left out of the possible 1024 values of a 10-bit code. The others have more than six ones or more than six zeros and are invalid.

A.2.1 8B/10B Encoding

8B/10B encoding is normally done using a pair of look-up tables as shown in Figure A-2 rather than a single look-up table.

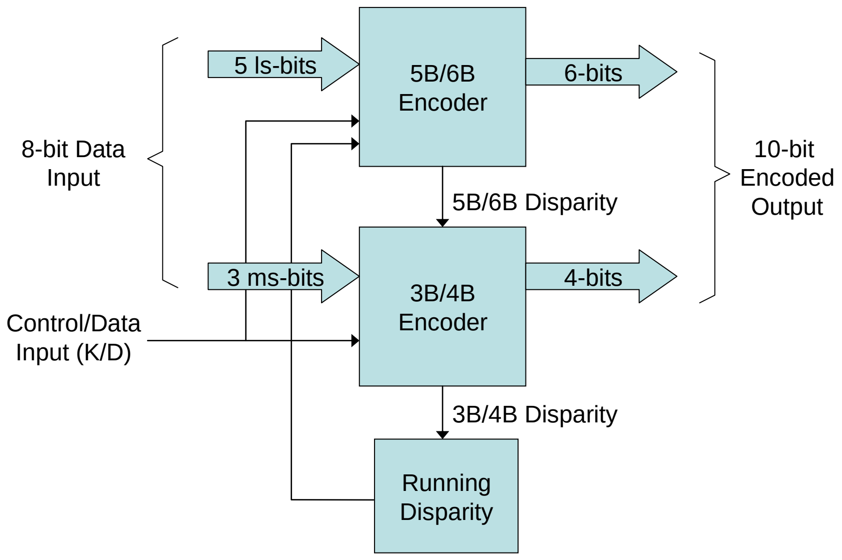


Figure A-2 8B/10B Encoder

The 5B/6B and 3B/4B approach to 8B/10B encoding has lead to a specific notation for representing codes resulting from this encoding. This is illustrated in Figure A-3.

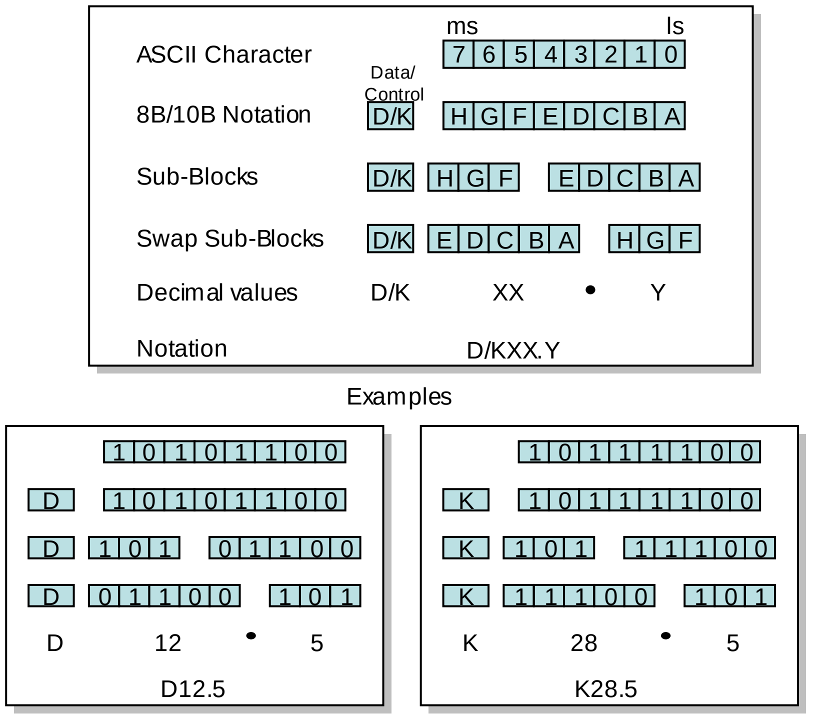


Figure A-3 8B/10B Notation

The five least-significant bits are encoded first using a 5B/6B encoder. This takes into account whether the 5 least significant bits are part of a control or data word as determined by the K/D input and also the current running disparity of the link. The 5B/6B encoding table is given in Table A-1. This table has the following properties:

- The six bit outputs consist of either three ones and three zeros, four ones and two zeros or two ones and four zeros.
- When the output code has neutral disparity (three ones and three zeros) there is one code independent of the running disparity (except for D07.y which has two codes based on the running disparity). The complement of a neutral disparity code also has neutral disparity and, except for D07.y, corresponds to a different input symbol. By definition, using a neutral disparity code will not affect the running disparity.
- When the output code has non-neutral disparity (four ones and two zeros or two ones and four zeros) there are two alternative codes provided which are the complement of each other. The code that is applied when the current running disparity of the link is negative (-ve) has four ones and two zeros which will then make the disparity out of the 5B/6B encoder positive. Similarly when the current running disparity is positive (+ve) the code with two ones and four zeros is applied making the 5B/6B disparity negative.
- The coding table is organised to minimise the amount of logic needed to implement it so that wherever possible there is a one to one mapping of bits from the 5-bit input to the 6-bit output. Note that K28.y must be treated as a special case.

The six-bit output of the 5B/6B encoder forms the six least-significant bits of the 8B/10B encoder output. The 5B/6B disparity is used in the encoding of the three most-significant bits of the 8-bit input data. The 5B/6B disparity, three most significant bits of the input data and the control/data flag (K/D) are fed into a separate 3B/4B encoder which produces the four most significant bits of the 8B/10B encoder output and a new value for the running disparity. The contents of the 3B/4B encoding table are given in Table A-2. This table has the following properties:

- Only 13 possible codes are valid, those shown in Table A-2.
- When the output code has non-neutral disparity there are two codes which are the complement of each other (with the exception of the codes for Dxx.7). One of these two codes will be used depending on the 5B/6B disparity. If the 5B/6B disparity is negative then the option with three ones and one zero will be used resulting in an overall positive disparity which will be the new value of the running disparity. The opposite is the case when the 5B/6B disparity is positive.
- The encoding for Dxx.7 has an alternative coding to prevent five consecutive ones being transmitted. The -ve current running disparity alternative (0111_b) is used for D17.7, D18.7 and D20.7. The +ve current running disparity alternative (1000_b) is used for D11.7, D13.7 and D14.7. This does complicate the encoding somewhat because these special cases have to be identified in the input data stream and the alternative code activated.

The complete 8B/10B encoding is performed by combining the results of the 5B/6B and 3B/4B encoding steps.

A.2.2 8B/10B Decoding

The task of decoding 8B/10B symbols is more complicated than the encoding process since a large number of input codes are mapped to a few valid output codes. Care must be taken to ensure that invalid 8B/10B codes are not accidentally considered to be valid simply because the 5B/6B and 3B/4B components are individually valid. An example is 1110101110_b which has a valid 5B/6B component 111010_b (-D23.y) and a valid 3B/4B component 1110_b (Dxx.7 normal encoding). This is invalid because the alternative Dxx.7 encoding 0111_b encoding ought to have been used.

Additional care must be taken with the 3B/4B decoder since the 4-bit input cannot distinguish between K and D codes. For example, 0110_b represents either -Kxx.1, or Dxx.6 or +Kxx.6.

Table A-1 5B/6B Encoding

Input		Output	
Data Input	Data bits 43210 (EDCBA)	Current Running Disparity -ve abcdei	Current Running Disparity +ve abcdei
D00.y	00000	100111	011000
D01.y	00001	011101	100010
D02.y	00010	101101	010010
D03.y	00011	110001	
D04.y	00100	110101	001010
D05.y	00101	101001	
D06.y	00110	011001	
D07.y	00111	111000	000111
D08.y	01000	111001	000110
D09.y	01001	100101	
D10.y	01010	010101	
D11.y	01011	110100	
D12.y	01100	001101	
D13.y	01101	101100	
D14.y	01110	011100	
D15.y	01111	010111	101000
D16.y	10000	011011	100100
D17.y	10001	100011	
D18.y	10010	010011	
D19.y	10011	110010	
D20.y	10100	001011	
D21.y	10101	101010	
D22.y	10110	011010	
D/K23.y	10111	111010	000101
D24.y	11000	110011	001100
D25.y	11001	100110	
D26.y	11010	010110	
D/K27.y	11011	110110	001001
D28.y	11100	001110	
K28.y	11100	001111	110000
D/K29.y	11101	101110	010001
D/K30.y	11110	011110	100001
D31.y	11111	101011	010100

Table A-2 3B/4B Encoding

Input		Output	
Data Input	**Data bits 765 (HGF)**	**5B/6B Disparity -ve fghj**	**5B/6B Disparity +ve fghj**
D/Kxx.0	000	1011	0100
Dxx.1	001	1001	
Kxx.1	001	0110	1001
Dxx.2	010	0101	
Kxx.2	010	1010	0101
D/Kxx.3	011	1100	0011
D/Kxx.4	100	1101	0010
Dxx.5	101	1010	
Kxx.5	101	0101	1010
Dxx.6	110	0110	
Kxx.6	110	1001	0110
Dxx.7	111	1110/0111	0001/1000
Kxx.7	111	0111	1000

The 12 control characters are listed in Table A-3. Three of these characters (K28.1, K28.5 and K28.7) contain a unique seven bit pattern (0011111 or 1100000) which does not occur in any of the data codes and which cannot be produced by concatenating any other two data or control codes. This pattern is known as the "comma" pattern and is widely used for performing receive code synchronisation (character alignment). The comma pattern is underlined in Table A-3.

Note that K28.7 followed by certain other data or control codes can produce a false comma, but the correct one comes first.

Table A-3 8B/10B Control (K) Codes

Input	Output	
Special Character Name	Current Running Disparity -ve	Current Running Disparity +ve
K28.0	001111 0100	110000 1011
K28.1	001111 1001	110000 0110
K28.2	001111 0101	110000 1010
K28.3	001111 0011	110000 1100
K28.4	001111 0010	110000 1101
K28.5	001111 1010	110000 0101
K28.6	001111 0110	110000 1001
K28.7	001111 1000	110000 0111
K23.7	111010 1000	000101 0111
K27.7	110110 1000	001001 0111
K29.7	101110 1000	010001 0111
K30.7	011110 1000	100001 0111

A.2.3 Disparity

The initial disparity can be either positive or negative, i.e. +1 or -1. A symbol can have a disparity of +2 (six ones and four zeros), 0 (five ones and five zeros) or -2 (four ones and six zeros). If the disparity of a new symbol is anything other than +2, 0 or -2 it is invalid. When a new symbol arrives its disparity is calculated based on the current running disparity plus the disparity of the new symbol. The possible results are (running disparity + new symbol disparity):

(+1) + (+2) = +3 which is invalid

(+1) + (0) = +1

(+1) + (-2) = -1

(-1) + (+2) = +1

(-1) + (0) = -1

(-1) + (-2) = -3 which is invalid

When an invalid disparity arises it is an indication that something has gone wrong with the link and the link needs to be re-initialised. The running disparity can be tracked as soon as link is initialised.

When the 8B/10B encoder/decoder is separated into 5B/6B and 3B/4B encoders/decoders the above rules apply to both encoders/decoders. The disparity of each sub-code must be +2, 0 or -2 and the running disparity at the end of each encoder/decoder must be +1 or -1.

Table A-4 and Table A-5 show how errors can be captured by monitoring for invalid codes and disparity errors. The transmitter 5B/6B and 3B/4B codes are shown followed by the running disparity (+1 or -1) after the code has been sent. The initial running disparity is -1 in both examples. In Table A-4 a single bit error converts the D00.0 character sent into a code whose 4B component does not appear in the 3B/4B coding table and has a disparity of -4. This is immediately detected as a coding error.

Table A-4 Detection of error by invalid code

Character	Transmitted				Received			
D00.0	100111	+1	0100	-1	100111	+1	0000	-3 ERROR

In Table A-5 an error occurs in the first line with the D08.1 character being changed to the D05.1 character. D05.1 is a valid character so goes undetected. The running disparity should however be positive but because of the error it is negative. The characters that follow have neutral disparity so the running disparity remains unchanged and no error is detected. Eventually a character, D15.1, is sent which does not have neutral disparity. At the transmitter the running disparity is negative prior to D15.1 so that character is encoded as 101000 1001 which has negative disparity. When this is received at the receiver the negative disparity causes an error because the running disparity there is already negative. The error has been caught by disparity but several characters were sent before the error became apparent.

The receiver should look out for both invalid characters and disparity errors. It is also important that a CRC code is added to each packet sent to ensure that any error in a packet is detected.

Table A-5 Detection of error by invalid disparity

Char	Transmitted				Received				Char
D08.1	111001	+1	1001	+1	101001	-1	1001	-1	D05.1
D09.1	100101	+1	1001	+1	100101	-1	1001	-1	D09.1
D10.1	010101	+1	1001	+1	010101	-1	1001	-1	D10.1
D11.1	110100	+1	1001	+1	110100	-1	1001	-1	D11.1
D12.1	001101	+1	1001	+1	001101	-1	1001	-1	D12.1
D13.1	101100	+1	1001	+1	101100	-1	1001	-1	D13.1
D14.1	011100	+1	1001	+1	011100	-1	1001	-1	D14.1
D15.1	101000	-1	1001	-1	101000	-3 ERROR			D15.1

A.3 Serialisation and De-Serialisation

Serialisation is the conversion of a parallel data stream into a serial one. The parallel 10-bit data word is loaded into a shift register and then shifted out using a transmit clock signal to drive the shift register. A new character has to be loaded into the parallel input of the shift register as soon as the previous 10-bit character has been shifted out to prevent a gap in the serial data.

De-serialisation is the opposite of serialisation. The serial data is shifted into a shift register using a receive clock (also called a bit clock). Recovery of the receive clock from the transmitted serial data stream is described in section A.4 Once a full 10-bit character has been shifted into the shift register it is read out in parallel. The 10-bit character must be read out at the correct point in the serial data stream *i.e.* when a complete new 10-bit character is in the shift register. Character synchronisation is described in section A.5.

A.4 Receive Clock Recovery

Recovery of the receive clock (bit clock) from the received serial data stream is done using a phase-locked loop (PLL). A typical phase-locked loop is shown in Figure A-4.

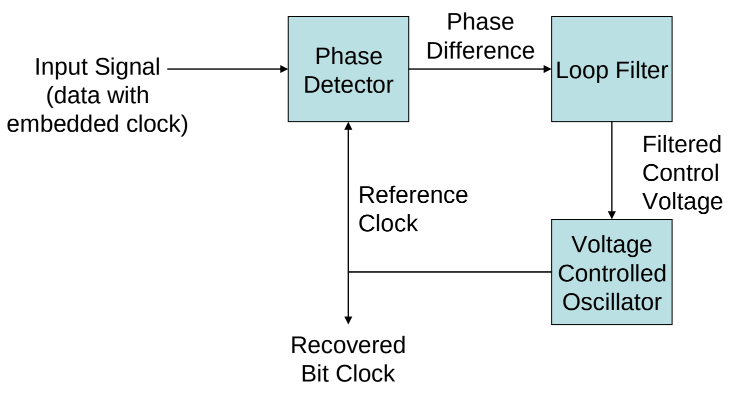


Figure A-4 Typical Phase-Locked Loop

The phase of the incoming data stream is compared to the phase of a reference clock signal. The detected phase difference is filtered, removing noise and providing an average phase difference. The filtered phase difference is used to control the frequency of the reference clock. If there is a positive phase difference with edges in the data stream occurring before edges in the reference clock then the reference clock frequency must be increased so that it catches up with the data stream edges. If there is a negative phase difference then the reference clock is occurring too early so must be slowed down, in which case the reference clock frequency is reduced.

When locked so that there is no phase difference, the reference clock can be used to recover the data-bits from the serial stream.

High frequency phase locked loops are normally implemented using a voltage controlled oscillator and an analogue loop filter.

The time taken for a PLL to lock onto a signal is dependent upon the design of the PLL and the difference between the input bit stream phase and the PLL reference clock phase. Typically it takes at least 5000 edges in the bit stream for a PLL to lock although it can be substantially longer for some PLL designs.

A.5 Symbol Synchronisation

Symbol synchronisation is necessary in the receiver to separate out each symbol from the received bit stream. To do this is it necessary to identify where a symbol starts, after that each individual symbol can be separated by simply counting 10-bits for each symbol. Identifying the start of a symbol is used using the 8B/10B Comma bit sequences. Comma sequences are unique seven bit sequences:

- Plus Comma 0011111
- Negative Comma 1100000

An illustration of symbol synchronisation using a plus comma is shown in Figure A-5. The start of the next character occurs on the fourth bit after the end of the detected plus comma.

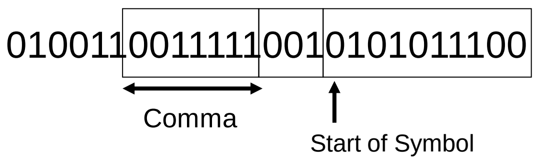


Figure A-5 Symbol Synchronisation Using a Plus Comma

Note that SpaceFibre uses all 10-bits of the commas for symbol synchronisation, rather than just 7-bits.

There are two principal means of performing symbol synchronisation. The first method, shown in Figure A-6, performs the symbol synchronisation after de-serialisation, while the second method, illustrated in Figure A-7, does it during de-serialisation.

Figure A-6 shows the received bit stream being fed into the de-serialising shift register. As soon as ten bits have been received the de-serialised data is loaded into a register. The exact position of the ten bits in the data stream is not important. After a further ten bits have been received the data in the register is loaded into a second register and the de-serialised data is loaded into the first register. The 20 bits in these two registers are examined for a possible comma, using the comma detect circuitry. The combinatorial logic in the comma detect circuitry outputs a Comma Detect signal when a comma is found and registers the position of the start of the next character. The Start of Symbol is used to drive a 20:10 multiplexer which select the ten-bits of a character from the 20-bits in the two registers.

With this approach comma detection is done at a clock rate of one tenth of that of the bit stream. The comma detect circuitry has to simultaneously look for a comma in ten possible bit positions, requiring 10 correlators.

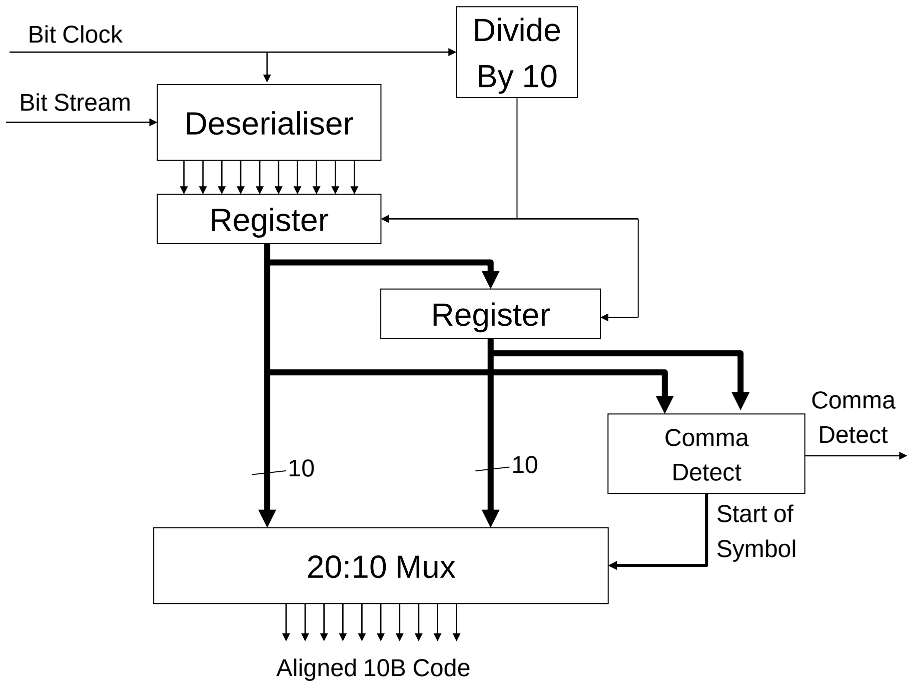


Figure A-6 Symbol Alignment After De-serialisation

The other approach, shown in Figure A-7, performs comma detection at the bit clock rate. The bit stream is fed into a 10-bit shift register (de-serialiser). The 10-bit parallel output from the shift register fed to a 10-bit character register and to a Comma Detect circuit. The Comma Detect circuit looks for a Comma in the last seven bits of the shift register (*i.e.* the first seven bits to enter the shift register). When a comma is detected the data in the shift register are loaded into the data register. A bit counter is used to count the 10-bits in each character, loading the data register from the shift register every 10 bits. The comma detect circuit resets the counter, re-synchronising the bit counter and forcing the data in the shift register to be loaded into the data register.

This approach requires the comma detect circuitry to operate at the rate of the bit clock and needs a high-speed bit counter. The amount of circuitry is significantly less that the other approach.

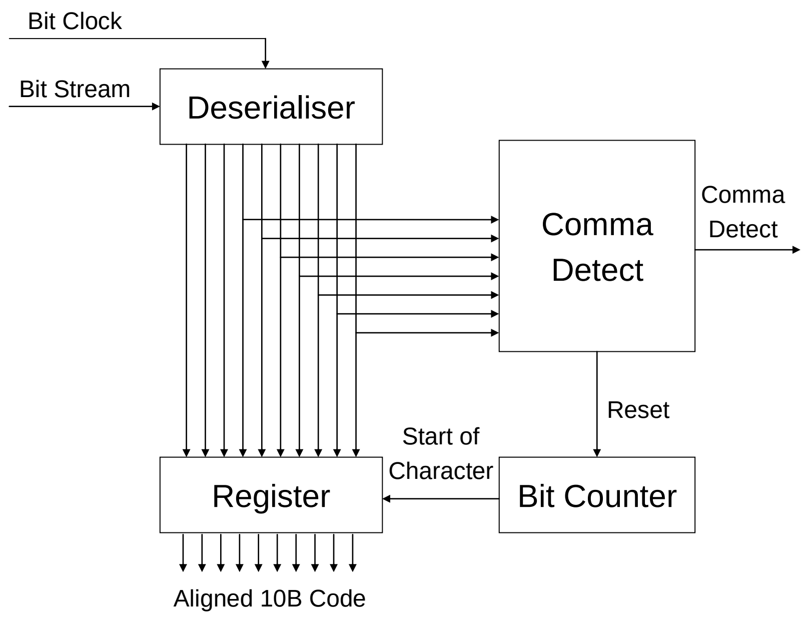


Figure A-7 Character Alignment During De-serialisation

A.6 Receive Elastic Buffer

The two ends of a link are both expected to operate at the same frequency. In practice, however, there will be slight differences in the clocks at the two ends of the link. This can cause receive buffer overflow or under-run problems unless the difference in the two clock speeds is compensated for. This is achieved using a Receive Elastic Buffer and associated SKIP characters.

The receive clock (bit clock) is recovered from the incoming bit stream, so is at the same frequency as the transmit clock at the other end of the link. After de-serialisation and character synchronisation the incoming data must be transferred from the receive clock domain to the local system clock domain. In passing between these two clock domains, slight differences in the clock frequencies must be accommodated. This is achieved using the Receive Elastic Buffer.

The normal situation with a Receive Elastic Buffer is illustrated in Figure A-8. Data is written into the buffer using a write pointer which operates at the receive character rate (RXRECCLK). It is read out by read pointer which operates at the user system character rate (RXUSRCLK).

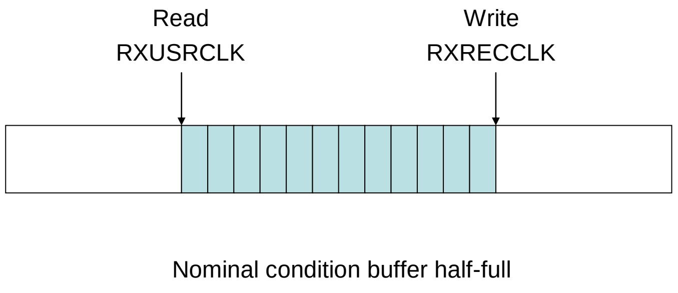


Figure A-8 Receive Elastic Buffer - Nominal Condition

Any difference between the two clock frequencies is compensated for using a special character, SKIP, inserted every so often into the data stream

If the RXUSRCLK is faster than RXRECCLK the buffer will slowly empty. When the buffer is less than half full, implying that RXUSRCLK is faster than RXRECCLK, extra SKIP characters are added to the Receive Elastic Buffer. This may be done when a SKIP character is read out of the buffer, by simply not incrementing the read pointer, so that the SKIP character will be read a second time. The effect is to add an extra SKIP character to the data stream, temporarily slowing down the RXUSRCLK to compensate for it being faster than RXRECCLK. This is illustrated in Figure A-9.

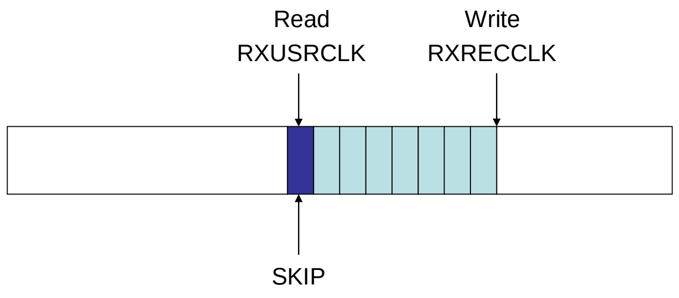


Figure A-9 Receive Elastic Buffer Emptying

If RXUSRCLK is slower than RXRECCLK then the Receive Elastic Buffer will slowly fill up. When the buffer is more than half full, SKIP characters are skipped. This is done by incrementing the read pointer past a SKIP character, *i.e.* if after reading a character, the next character to be read is a SKIP character, it is ignored and the read pointer is moved to point to the following character instead. The effect is to remove SKIP characters from the buffer, temporarily speeding up the RXUSRCLK to make up for the fact that it is slower than RXRECCLK. This is shown in Figure A-10. Note that the SKIP operation requires the elastic buffer to know in advance that a SKIP is present in the buffer without reading it otherwise the SKIP operation will have no effect.

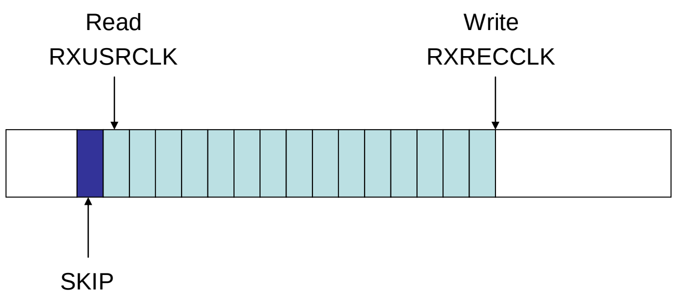


Figure A-10 Receive Elastic Buffer Filling Up

For the Receive Elastic Buffer to work properly there must be sufficient SKIPs in the data stream, so that they can be remove if necessary. The frequency of SKIPs depends on the size of the elastic buffer and the maximum frequency difference between RXUSRCLK and RXRECCLK.

Assume that the nominal operational frequency is F symbols per second and that the maximum clock difference, is D Hz, then the time, T, taken for the elastic buffer to have one symbol too many or one symbol too few is given by:

$T = 1 / (\text{Receive Clock Frequency} – \text{User Clock Frequency})$

$T = 1/ ((F+D) – (F-D))= 1/ (2D)$

In this time the number of symbols sent is

$N = (F+D).T$

which is approximately

$N \approx F/(2D)$

since F is much greater than D.

Now D/F is the maximum clock drift, so

$N \approx 1/(2P)$

Where P is the maximum drift in the clock.

For a ±100 ppm maximum clock drift, which is readily achievable using crystal oscillators, D, is 10^{-4}, and the number of symbols sent before the elastic buffer is one symbol out is 5000. A SKIP symbol must thus be sent every 5000 symbols to prevent the Elastic buffer from ever being more than one symbol out. This is the case independent of the size of the symbols.

In SpaceFibre the receive elastic buffer stores control and data words rather than individual symbols. The SKIP control word is therefore four symbols lone, i.e. one word long.

Annex B (informative) Example of SpaceFibre CRC implementation

B.1 Overview

In this example implementations of the CRC used by SpaceFibre are provide in VHDL and C-code.

B.2 VHDL implementation of SpaceFibre CRC

Bibliography

ECSS-ST-S-00	ECSS system - Description, implementation and general requirements
ECSS-E-ST-50	Space engineering - Communications
ECSS-E-HB-50	Space engineering - Communications guidelines
ECSS-E-ST-40	Space engineering - Software
http://www.spacewire.esa.int	SpaceWire website