School of Computing

Space Technology Centre
University of Dundee

SpaceNet - SpaceWire-T

Initial Protocol Definition

Revision:	Draft A Issue 3.1
Date:	24th August 2009
ESA Contract Number	220774-07-NL/LvH
Ref:	SpW-RT WP3-200.1

Space Technology Centre
School of Computing
University of Dundee
Dundee, DD1 4HN
Scotland, UK
spacetech.computing.dundee.ac.uk

Document Authors

Steve Parkes

Albert Ferrer Florit

Document Change Log

Date	Revision No	Comments
24th August 2009	Draft A Issue 3.1	Major revision removing all reliability and robustness related parts of the standard according to ESA's direction.
15th February 2009	Draft A Issue 2.2	General revision of document, correcting errors and adding clarifications
30th October 2008	Draft A Issue 2.1	Update following ESA and SciSys comments (track changes on). Description of complete SpaceWire protocol stack added with service interfaces.
8th September 2008	Draft A Issue 2.0	Update following early prototyping and inputs from SpaceWire Working Group
12th May 2008	Draft A Issue 1.1	Update following comments from ESA
25th March 2008	Draft A Issue 1.0	First issue

A comprehensive list of the changes made to this document in each major revision is provided in section 7.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 AIMS AND OBJECTIVES

The aim of this document is to present an initial set of protocols that meet the SpaceWire-T Requirements.

The various techniques that are being considered to meet the SpaceWire-T requirements are considered and described first. Then each function required to implement the protocols is defined in some detail.

WARNING

This current document is an early draft of the proposed standard and is for discussion purposes only. It will change after prototyping work has been completed. Applicable documents may also change.

DO NOT USE THIS DOCUMENT TO DESIGN DEVICES OR SYSTEMS!

1.2 BACKGROUND AND AUTHORS COMMENTS

The requirements for SpaceWire-RT were provided in WP2-100.1 SpaceWire-RT Requirements [AD8]. Note that RT stands for Real-Time (or alternatively Reliable and Timely). The requirements were subsequently agreed by the SpaceWire Working Group. A first draft of the initial protocols was provided in May 2008 (draft issue 1.1). These protocols were presented to the SpaceWire working group for comment and various parts of the protocols prototyped by University of Dundee. The draft standard was then updated according to the results of the prototyping and taking into account the comments from the SpaceWire working group. Detailed feedback was then provided by ESA and SciSys on the document resulting in over 90 RIDs. Action from these RIDs was agreed and the document updated accordingly resulting in version 2.2a of this document, which was presented to the SpaceWire Working Group.

The current version results from subsequent direction from ESA to remove the reliability parts of the protocol. Hence this document only covers provision of timeliness over SpaceWire. The assumption that ESA has made is that SpaceWire is sufficiently reliable for most applications without retry and redundancy mechanisms. In situations where SpaceWire is not sufficiently reliable for some applications then it is up to the application to provide reliability and redundancy mechanisms. SpaceWire-RT has now been temporarily renamed SpaceWire-T as the reliable part has been removed from the standard.

In addition ESA requested that the Basic QoS and the "allocated" and "opportunistic" mode of operation were also removed.

It should be noted that SpaceWire-T no longer provides a quality of service layer for SpaceWire since there is only one QoS offered for each type of network (asynchronous and scheduled). Furthermore it no longer implements the full set of QoS specified by CCSDS SOIS. ESA only requires the minimum SOIS QoS to be implemented, i.e. a Best Effort service.

Given the substantial change in requirements resulting from this redirection, it is recommended that the new requirements are documented and agreed by the SpaceWire Working Group prior to any further work on this standard. The specific requirements will drive the design decisions.

Please note the errors in the figure numbers are due to "track changes" being on. These errors disappear once changes are accepted and the document referencing updated.

1.3 Guide to Document

Section 2 lists the terms and definitions relevant to the SpaceWire-T set of protocols.

Section 3 provides an overview of the SpaceWire-T protocols and various techniques that are proposed for their implementation.

Section 4 provides the specification for an asynchronous network.

Section 5 provides the specification for a scheduled network.

Section 6 provides an initial look at the network configuration parameters required for SpaceWire-T. Substantially more work needs to be done on this section.

1.4 Acronyms and Abbreviations

AD	Applicable Document
BACK	BFCT Acknowledgement
BFCT	Buffer Flow Control Token
CCSDS	Consultative Committee for Space Data Systems
DP	Data PDU
ECSS	European Cooperation for Space Standardization
ESA	European Space Agency
ESTEC	ESA Space Technology and Research Centre
I/F	Interface

PC	Personal Computer
PDU	Protocol Data Unit
PUS	Packet Utilization Service
QoS	Quality of Service
RD	Reference Document
RMAP	Remote Memory Access Protocol
RMW	Read/Modify/Write
SDU	Service Data Unit
SOIS	Spacecraft Onboard Interface Services
SpW	SpaceWire
TCONS	Time-Critical Onboard Network Services
UDS	User Data Segment
UoD	University of Dundee

1.5 Reference Documents

The documents referenced in this document are listed in Table 1-1.

Table 1-1: Reference Documents

REF	Document Number	Document Title
RD1	UoD-SpaceNet v7, 23rd April 2007	Proposal for SpaceWire Network and Future Onboard Data-Handling, Technical, Management and Administrative Proposal
RD2	TEC-ED/WG/2005.15	SpaceWire Network "SpW-Net" SpaceWire and Future Onboard Data Handling SpaceNet Statement of Work Annex1

1.6 Applicable Documents

The documents applicable to this document are listed in Table 1-2.

Table 1-2: Applicable Documents		
REF	**Document Number**	**Document Title**
AD1	ECSS-E50-12A, January 2003	SpaceWire: Links, nodes, routers and networks
AD2	ECSS-E50-11A Draft 0.5	SpaceWire Protocols Feb 2008
AD3	CCSDS 850.0-G-R1.1 Draft Green Book	Spacecraft Onboard Interface Service Draft Informational Report Jan 2008
AD4	CCSDS 851.0-R-1.1 Draft Red Book	Spacecraft Onboard Interface Service Subnetwork Packet Service Draft Recommended Practice Jan 2008
AD5	CCSDS 852.0-R-1.1 Draft Red Book	Spacecraft Onboard Interface Service Subnetwork Memory Access Service Draft Recommended Practice Jan 2008
AD6	CCSDS 853.0-R-1.1 Draft Red Book	Spacecraft Onboard Interface Service Subnetwork Synchronisation Service Draft Recommended Practice Jan 2008
AD7	CCSDS 854.0-R-1.1 Draft Red Book	Spacecraft Onboard Interface Service Device Discovery Service Draft Recommended Practice Jan 2008
AD8	CCSDS 855.0-R-1.1 Draft Red Book	Spacecraft Onboard Interface Service Subnetwork Test Services Draft Recommended Practice Jan 2008
AD9	ISO 7498-1 1996	Information Technology – Open Systems Interconnect – Basic Reference Model: The Basic Model Available from http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html
AD10	SpW-RT WP3-100.1	SpaceWire-RT Requirements, February 2008
AD11	ECSS-E-ST-50-11C	SpaceWire Protocols Draft 1.3 July 2008

2 TERMS AND DEFINITIONS

In this section the terms and definitions proposed for the SpaceWire-T standard are provided in the form required by ECSS.

2.1 DEFINITIONS FROM THE OPEN SYSTEMS INTERCONNECTION (OSI) BASIC REFERENCE MODEL

layer subdivision of the architecture, constituted by subsystems of the same rank

protocol data unit (PDU) unit of data specified in a protocol and consisting of protocol-control-information and possibly user data.

service capability of a layer (service provider) together with the layers beneath it, which is provided to service-users.

service data unit (SDU) an amount of information whose identity is preserved when transferred between peer entities in a given layer and which is not interpreted by the supporting entities in that layer.

2.2 TERMS DEFINED IN THIS RECOMMENDATION

For the purposes of this Recommendation, the following definitions also apply. Many other terms that pertain to specific items are defined in the appropriate sections.

ACK acknowledgement

acknowledgement control PDU used to indicate reception of a data PDU without error and in the correct sequence

asynchronous network a network where there is no control of network bandwidth, packets are sent when generated, subject to prioritisation, provided that the required resources are not busy

asynchronous system system that uses a SpaceWire asynchronous network for communication between elements of the system

BACK buffer acknowledgement

buffer acknowledgement control PDU used to indicate reception of a buffer flow control token

BFCT buffer flow control token

buffer flow control token control PDU used to indicate availability of space in a destination buffer

byte 8-bits

channel identifier for network resources comprising a source buffer, the links on path through the SpaceWire network and a destination buffer

CRC cyclic redundancy code

cyclic redundancy code code used to check for errors in a packet

data PDU PDU containing a user data segment from an SDU

delimited having a known and finite length

destination destination node

destination channel buffer buffer in the destination node associated with a specific channel between a particular source and destination from which data, sent by the source and received at the destination, can be read

destination logical address logical address of the destination node

destination node node that a SpaceWire packet is being sent to

destination subnetwork service access point service access point that identifies the user entity to which a Packet Service SDU is required to be delivered

DLA destination logical address

DP data PDU

DSNSAP Destination Subnetwork Service Access Point

duplicate packet copy of a packet that has already been received once

epoch repeat cycle time for time-slot identifiers

input node node on a SpaceWire network that puts data into a channel for transfer across the network to another node (output node)

maximum user data segment maximum size of user data put into a segment by SpaceWire-T

MUDS maximum user data segment

output node node on a SpaceWire network that receives data from a channel

packet delimited byte aligned data unit

priority the relative precedence for sending of a data PDU relative to other data PDUs so that a higher priority data PDU normally gets sent before a lower priority one

QoS Quality of Service.

Quality of Service level of service that is requested and provided

Reliable data gets to the destination if it is at all possible even if there are transitory errors or even permanent errors provided that the network has some appropriate redundancy

scheduled network network that uses time-slots for dividing network bandwidth and schedules the traffic into those time-slots for transfer across the network

scheduled system system that uses a SpaceWire scheduled network for communication between elements of the system

segment piece of user data comprising one or more bytes

segmentation division of service data units by SpaceWire-T into shorter sections (segments) that are short enough to be sent over the SpaceWire-T network. SpaceWire-T is responsible for reassembling the segments back into service data units at the target.

sequence-number incrementing number given to each unique data PDU sent which is use to detect missing and duplicate data PDUs

service access point an interface to SpaceWire-T that is identified by a source or destination address and from which a user application can access the SpaceWire-T services

SLA source logical address

source source node

source channel buffer buffer in the source node associated with a specific channel between a particular source and destination into which data is written to be sent across the network from source to destination

source logical address logical address of the source node

source node node that is sending a SpaceWire packet

source subnetwork service access point service access point that identifies the user entity that wishes to access a SOIS service

SSNSAP Source Subnetwork Service Access Point

timely done at an appropriate time or within a specific time constraint

time-slot period of time between time-codes in a scheduled network which is the smallest unit of time used for time-division multiplexing in a SpaceWire-T network

UDS user data segment

user application a software or hardware system that is using the services of SpaceWire-T

user data data that a user of SpaceWire-T wishes to send across a SpaceWire network

user data segment a piece of a SDU that fits into a data PDU

2.3 JUSTIFICATION AND OTHER NOTES ON TERMINOLOGY

2.3.1 Maximum User Data Segment

The Maximum User Data Segment (MUDS) is required to ensure that different sources of data get fair access to the transmission medium, by multiplexing traffic on a data PDU by data PDU basis. When a large SDU is being sent, other sources can gain access to the transmission medium after each segment of the large SDU has been sent.

3 PROTOCOL OVERVIEW

In this section an informative description of the SpaceWire-RT protocols is provided as an introduction to the subsequent normative specifications in sections 4, 5, and 6.

3.1 LAYERED PROTOCOL STACK

SpaceWire-T is part of the layered protocol stack for SpaceWire which is illustrated in Figure 3-1.

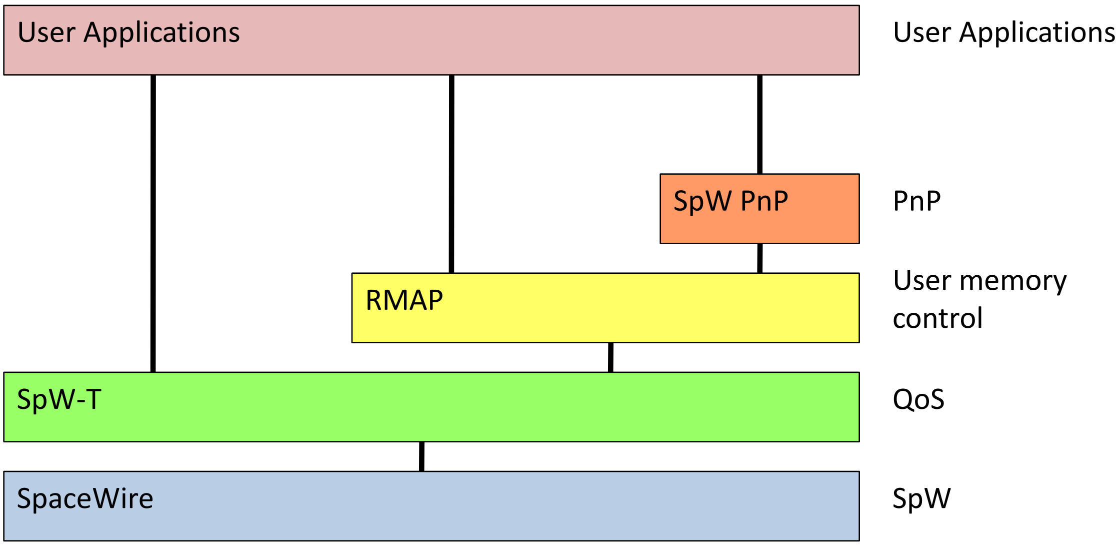

Figure 3-1 SpaceWire Layered Protocol Stack

SpaceWire is at the bottom of the protocol stack sending SpaceWire packets across the SpaceWire network from source to destination. Immediately on top of SpaceWire is SpaceWire-T providing quality of service. All traffic has to pass through SpaceWire-T otherwise the timeliness QoS cannot be ensured. User applications can talk directly to SpaceWire-T. RMAP provides a mechanism for reading from and writing to memory in a remote node. SpaceWire-PnP (plug and play) uses RMAP for configuration and discovery of nodes on the SpaceWire network. User applications can use the services provided by RMAP or SpaceWire-PnP as well as talking directly to SpaceWire-T. Note that if SpaceWire-T is not being used, RMAP, SpaceWire-PnP and user applications may run directly over SpaceWire but there is then limited QoS.

The CCSDS Spacecraft Onboard Interface Services (SOIS) working group has defined a set of common communication services for use onboard a spacecraft. The SOIS subnetwork layer and three of the services provided are illustrated in Figure 3-2. The SOIS Packet Service provides for delivery of SOIS Packets across a subnetwork, the Memory Access Service for the access of memory devices on the subnetwork, and the Device Discovery Service supports plug-and-play capability with notification services.

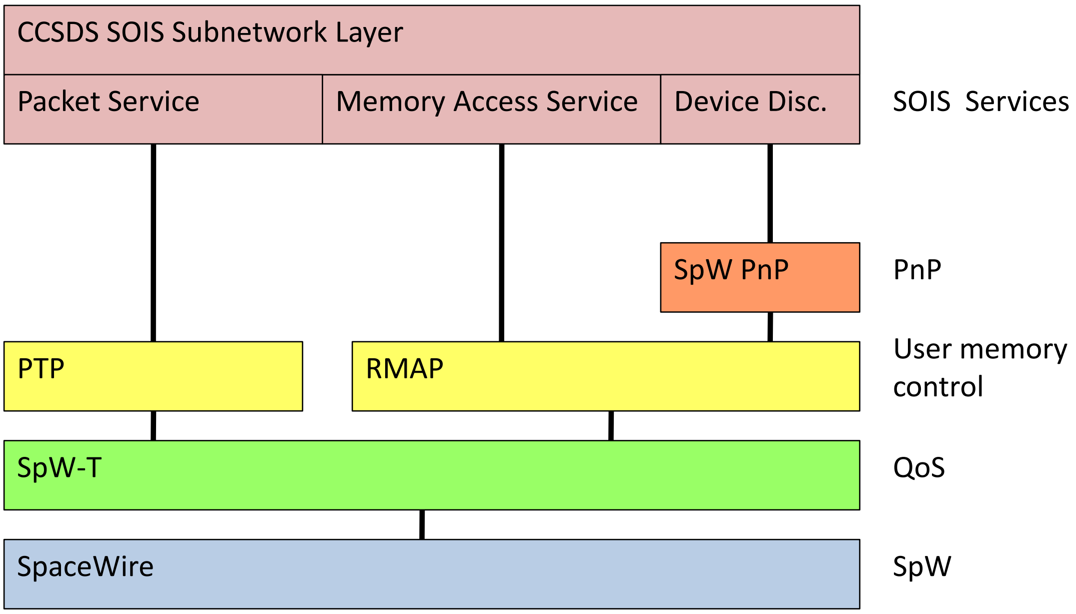

Figure 3-2 SOIS to SpaceWire Layered Protocol Stack

RMAP and SpaceWire-PnP provide the Memory Access Service and Device Discovery Services. The SOIS Packet Service is provided by a SpaceWire Packet Transfer Protocol that sends packets across the SpaceWire network, providing buffer management and flow control.

To send a SOIS Packet across a SpaceWire network using the CCSDS SOIS Packet Service the SOIS Packet is presented to the SOIS Packet Service and passed to the SpaceWire Packet Transfer Protocol (SpaceWire-PTP) which manages the destination buffer to avoid sending the SOIS Packet when there is no room for that complete SOIS Packet in the destination. SpaceWire-PTP passes the SOIS Packet as a stream of characters to SpaceWire-T which chops the stream into segments, sends them across the network, and reassembles the SOIS Packet at the destination. The SOIS Packet emerges from SpaceWire-T as a stream of characters which are read and put into appropriate SOIS Packet buffer space by SpaceWire-PTP. The complete buffered SOIS Packet is then made available to the user application via the SOIS Packet Service interface.

SpaceWire protocols are designed for efficient implementation, high performance and both hardware and software implementation. The SOIS services are designed for generic implementation over various buses or networks, providing a standard software interface.

3.2 SpaceWire-T Service Interfaces

There are five primitives currently defined for the SpaceWire-T service:

- Send_Data.request (channel, source address, destination address, cargo) which requests to send a Service Data Unit (SDU) from the source node where the request is being made to a destination node on a SpaceWire network;
- Receive_Data.indication (channel, source address, destination address, cargo) which indicates that a SpaceWire-RT data PDU has been received and which passes the SDU it carried to the SpaceWire user;
- Notify_Delivered.indication (channel, source address, destination address, SDU_ID) which indicates to the user that issued a Send_Data.request over a channel that provided assured or guaranteed services that the SDU was safely delivered to the destination.
- Notify_Error.indication (channel, source address, destination address, SDU_ID, error metadata) which indicates to the user that issued a Send_Data.request that there was a problem delivering the SDU over a channel that provided assured or guaranteed services.
- Configure.request (channel, configuration information) which configures the channel parameters.

Note that the network management including channel configuration is not included in this document.

3.3 Asynchronous and Scheduled Systems

There are two types of system that are supported by SpaceWire-T:

Asynchronous – where the sending of information over the SpaceWire network is asynchronous and priority is used to provide timeliness of delivery. Information in lower number source channels will be sent before information in higher number channels.

Scheduled – where information is sent over the SpaceWire network synchronously with each source channel being assigned one or more time-slots when it is allowed to transmit information. One or more source channels may be assigned to a single time-slot in which case the lower channel numbers allocated to a time-slot have priority over higher channel numbers assigned to that time-slot. Timeliness of delivery is controlled by a schedule table used to specify which source channels can send information in which time-slot. This provides deterministic delivery.

A user application writes into one of the source channel buffers available. This information is then transferred across the SpaceWire network and becomes available in the corresponding destination channel buffer. SpaceWire-T provides a stream service so that user data written into a source channel buffer appears in and can be read from the corresponding destination channel buffer. This is conceptually the same as a SpaceWire point-to-point link where data written into the source transmit FIFO appears in the destination receive FIFO.

3.4 QUALITY OF SERVICE

SpaceWire-T provides a limited quality of service layer for SpaceWire concentrating on timeliness only. It does NOT provide the full set of QoS defined by SOIS, only the minimum subset.

3.4.1 QoS in an Asynchronous Network

For an asynchronous network SpaceWire-T provides a Best Effort QoS with acknowledgment.

The **Best Effort QoS** provides a service which does not ensure delivery (i.e. does not provide any redundancy and does not retry in the event of a failure to deliver) and is not timely (i.e. does not deliver information within specified time constraints).This service does not deliver duplicate or out of sequence data PDUs. It provides an acknowledgement that packets have been received. The acknowledgement is not reliable (i.e. there are no retries on acknowledgements).

- Makes a single attempt to deliver data to its destination but cannot ensure that it will be delivered successfully.
- Data is provided without errors, i.e. a data PDU that arrives with an error is discarded.
- The order of data PDUs is preserved (within a priority value). Any out-of-order data PDUs are discarded.
- Data is provided without duplication, i.e. a data PDU that is a duplicate of a previous packet is discarded.
- Provides an acknowledgement to the source once data has been delivered successfully.
- Priority indicates the relative precedence with which data PDUs are handled by the sub-network. Higher priority data PDUs get sent before lower priority ones, provided that there is room in the higher priority destination buffers.

3.4.2 QoS in a Scheduled Network

For a scheduled network SpaceWire-T provides the reserved QoS.

The **Reserved QoS** provides a service which does not ensure delivery (i.e. does not provide any redundancy and does not retry in the event of a failure to deliver), but is timely (i.e. when a packet is delivered it is delivered on time).

- Makes a single attempt to deliver data to its destination but cannot ensure that it will be delivered successfully.
- Data is provided without errors, i.e. a data PDU that arrives with an error is discarded.

- The order of data PDUs is preserved.
- Any out-of-order DPs are discarded.
- Data is provided without duplication, i.e. a data PDU that is a duplicate of a previous packet is discarded.
- Provides an acknowledgement when requested to the source once data has been delivered successfully.
- Resources are reserved for transferring the data across the network.
- Priority indicates the relative precedence with which data PDUs are handled by the sub-network. Higher priority data PDUs get sent before lower priority ones, provided that there is room in the higher priority destination buffers

3.5 CHANNELS

A channel is a set of network resources that connects a source user application in a source node to a destination user application in a destination node. It includes the following:

- Buffer in the source into which data is written to send data across the network (source channel buffer)
- SpaceWire links over which the data PDUs travel
- Buffer in the destination from which data that has been received over the network can be read (destination channel buffer)

A channel connects a source channel buffer (the channel input) to a destination channel buffer (the channel output) via one or more SpaceWire links. Thus a channel is identified by the source logical address (SLA), the destination logical address (DLA) and a channel number. The channel number is used so that there can be more than one channel between a particular source and destination pair.

A channel is unidirectional sending data in one direction only from source to destination, although some control information for the channel does flow in the opposite direction. To provide bi-directional communication two unidirectional channels are required; one in each direction. A node can have one channel going to just one destination node, many channels each going to a different node, many channels all going to the same node, etc. Channels provide virtual unidirectional, point-to-point communications across a SpaceWire network. The entry to a channel in a source node is the source buffer where data is fed in by a user application. That data will appear at the exit of the channel (the destination buffer) in the destination node.

A source user application that wants to send information over a channel to a destination user application writes data into the appropriate source channel buffer when there is space in that buffer.

Data from this buffer is taken out in chunks with each chunk (referred to as a segment) being put in a separate SpaceWire packet. The SpaceWire packets are sent across the SpaceWire network using the links specified by the channel. When they arrive at the destination node the user information in the SpaceWire packets is extracted and put in the destination channel buffer ready for the destination user application to read. When there is no room in the destination channel buffer the source node is prevented from sending any further packets to that destination channel buffer using a flow-control mechanism.

Up to 32 channels can be implemented between each source and destination pair on an asynchronous system and up to 32 on a scheduled system. It is not necessary for a node to support all of these channels. A node only has to support the number of channels that it needs. Similarly a destination need only accept data from the sources that it expects to receive data from.

Priority is implemented using more than one channel between a source and destination. Information placed in the lower number channel will be transferred before information in a higher number channel.

An example of several channels between three nodes is illustrated in Figure 3-3.

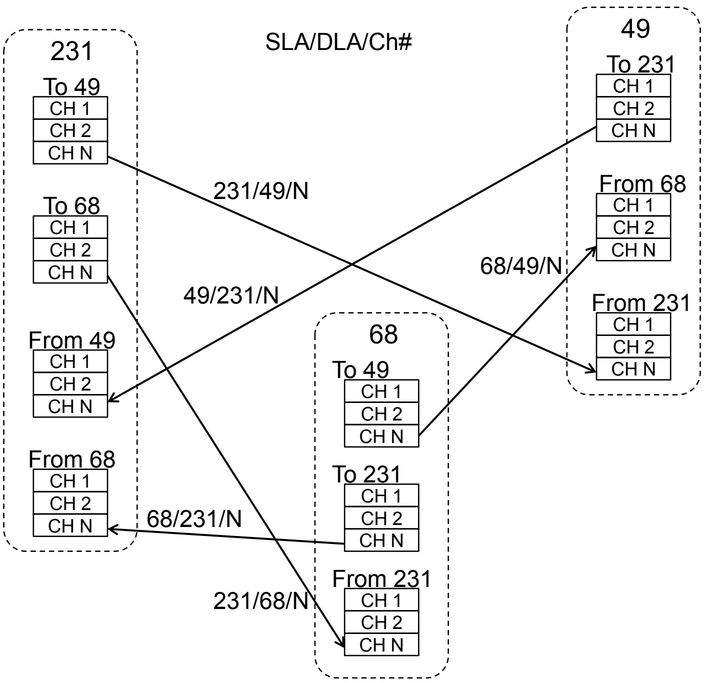

Figure 3-3 SpaceWire-T Channels

There are three nodes with SpaceWire logical addresses 49, 68 and 231, represented by the dashed rectangles. The source channel buffers in each node are grouped according to the destination for each channel. For example, node 68 has its source channel buffers to node 49 grouped under the heading “To 49”. Similarly, the destination channel buffers in each node are grouped according to the source for each channel. For example, node 49 has its destination channel buffers from node 68 grouped under the heading “From 68”. There are N unidirectional channels available from node 49 to node 68. Only the last one of these is indicated with an arrow from node 68 to node 49 for clarity. Source channel buffer 1 is connected to destination channel buffer 1, 2 to 2 and so on. The arrow between node 68 and 49 is labelled with the source logical address, destination logical address and channel number (SLA/DLA/CH#) which provides a network wide unique reference for that channel.

A pair of channels has been set up between node 49 and 231 using two unidirectional channels (231/49/N and 49/231/N). Similarly another pair of channels has been configured between nodes 68 and 231 (231/68/N and 68/231/N).

3.6 Scheduled System

3.6.1.1 Time-Slots

Reservation of network resources implies some means of controlling access to those resources. In SpaceWire-T this is done by scheduling traffic to use specific network resources at specific times. To achieve this, network bandwidth is split using equal divisions of time, known as time-slots. During each time-slot a discrete, constrained set of network communications can take place. Time-slots are distributed in SpaceWire using SpaceWire time-codes. There are 64 unique time-codes so a natural division is to have 64 time slots in a schedule or bandwidth allocation cycle. The 64 time-slots are referred to as an epoch. Time slots are used in the scheduled system for allocating network bandwidth to the network traffic.

The frequency of the time-codes depends upon the system requirements. This is discussed later in 3.7.3.8

3.6.1.2 Scheduling

Once the channels have been defined and the resources needed by each channel determined it is a relatively straightforward exercise to allocate the channels to a time-slot in a scheduled system, taking into account the required maximum data-rates that have to be supported for each unit. An example schedule is illustrated in Figure 3-4.

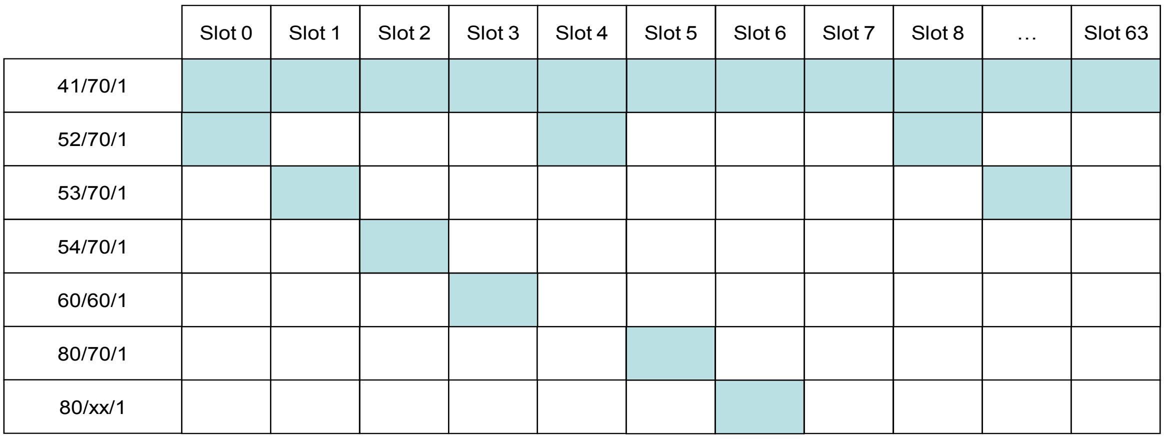

Figure 3-4 Slot allocation in scheduled system

The time-slots are listed along the top. In this example eight slots have been filled in which are repeated eight times for the full 64 time-slots. The nine channels are listed along the left hand side. The shaded boxes show when a particular channel is allowed to send data. For example Channel 52/70/1 is permitted to send data in time-slots 0, 4, 8, 12, etc.

In any one time-slot several communications may be happening concurrently provided that they do not conflict i.e. two or more channels communicating in any one time-slot do not require the same network resources. When GAR is being used this constraint alters as the bandwidth available increases with the number of links in a group. Since the links are full-duplex, bi-directional data flowing in one direction does not impede data flowing in the opposite direction.

As may be seen from the schedule table channel 41/70/1 may send data at any time: its traffic does not conflict with the other traffic flowing on the network. Channel 52/70/1 has 25% of the link bandwidth (one slot every four) and channel 53/70/1 has 12.5% bandwidth (one slot every eight).

When a particular time-slot comes around then the channels that are scheduled to send data in that time-slot are allowed to send one or more data PDUs provided that they fit within the duration of the time-slot.

3.6.1.3 Resources for Flow Control and Acknowledgements

As well as the main flow of DPs containing data, there will be traffic in the other direction containing acknowledgments and flow control information related to the transfer of the DPs.

In an asynchronous network where there is no control over resource utilisation, acknowledgments and flow control information can be sent without regard to possible delays over the SpaceWire network as timely delivery is not covered by the Best Effort QoS supported by an asynchronous network.

For a scheduled network where the resource utilisation is managed using time-slots, it is important that resource is allocated for the acknowledgements and also for any flow control information. To provide this additional resource the time-slots are made longer than needed to transfer the DPs so that there is time for acknowledgement and flow control information to be sent at the start of a time-slot. Time-slots are thus split into three parts:

- Acknowledgement phase
- Flow control phase
- DP transfer phase

During the DP transfer phase DPs are sent according to the schedule. During the acknowledgment phase acknowledgments are sent for the DPs transferred in the transfer phase, along with any flow control information.

To ease implementation the information is actually transferred in time-slots as follows:

1. Time-code received
2. Send acknowledgement for DPs received in previous time-slot
3. Wait a bit for acknowledgements to propagate across the network
4. Send flow control information
5. Wait a bit for flow control information to propagate across the network
6. Send DPs
7. Complete sending DPs before end of time-slot
8. Wait for next time-code

This results in the time-slot arrangement illustrated in Figure 3-5.

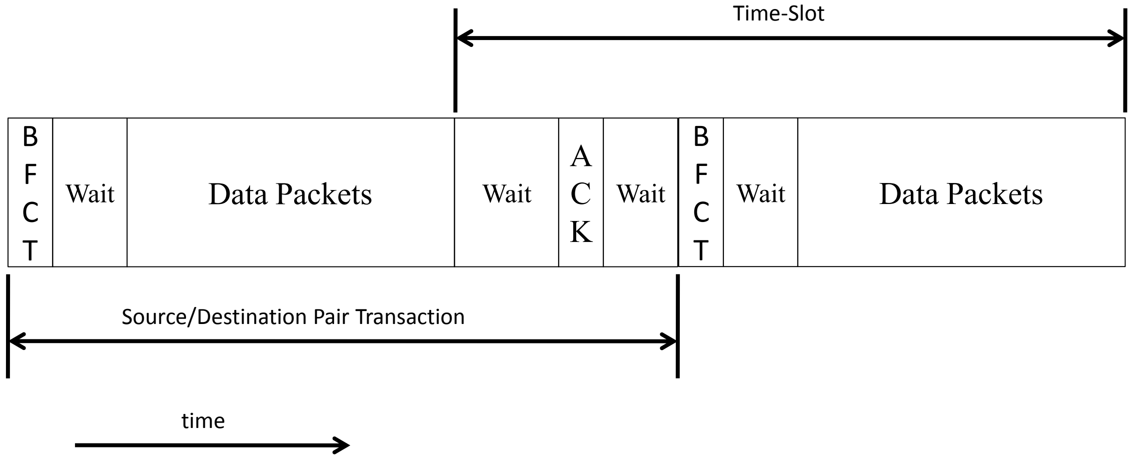

Figure 3-5 Time-Slot Arrangement

3.7 Architecture

The SpaceWire-T protocol includes the following functions:

- User application interface
- Segmentation
- End to end flow control
- Error detection
- Acknowledgement
- Address translation
- PDU encapsulation
- Priority
- Resource reservation
- Network management

In the following subsections these functions are described for an asynchronous system and then for a scheduled system. First the user application interface is described which is the same for both asynchronous and scheduled systems.

3.7.1 User Application Interface

SpaceWire-T provides a quality of service layer for a SpaceWire network. Any SpaceWire application is able to run over SpaceWire-T and benefit from the QoS provided. To achieve this SpaceWire-T uses the same conceptual model for the interfaces as SpaceWire. Data to be transferred over a SpaceWire network is put into a transmit FIFO at the source and appears in a receive FIFO at the destination. The destination is reached by following address information provided in the SpaceWire packet header. Packets are terminated by an EOP. The interface is a FIFO or stream type interface. SpaceWire-T uses the same model: it provides stream interfaces using source and destination buffers which effectively act as FIFOs.

The user application interface to SpaceWire-T is via the source and destination channel buffers. The source user application writes information to be transferred across the SpaceWire network into a source channel buffer. This is then readout by SpaceWire-T, and transferred across the SpaceWire network to the destination channel buffer associated with the source channel buffer. The destination user application can then read the information from the destination channel buffer.

There are five pieces of information that the source user application has to pass to the source channel buffer:

- **Channel number** – the number of the source and destination channel buffers to which the data to be transferred is to be written. The channel used will also determine the quality of service provided, since QoS parameters are associated with each channel.
- **Source** – the logical address of the source sending the information.
- **Destination** – the logical address of the destination to which the information is to be sent.
- **Cargo** – the data or other information that is to be transferred to the destination channel buffer.
- **Separator** – that separates out one complete piece of user application information from the next piece of user information being sent over a channel. For example a complete piece of information may be a CCSDS PUS packet or an RMAP command. SpaceWire-T will send information provided to a source channel buffer in segments when there is room in the destination channel buffer. Normally it sends segments of information that fill the maximum permitted DP. The last part of a complete piece of user information may not fill a DP, but should be transferred as soon as possible, without waiting for further user information to fill a maximum size DP. The separator is used to signal to SpaceWire-T that the information in the source channel buffer is to be sent straightaway without waiting for a full DP's worth of data. The separator is equivalent to the EOP in SpaceWire.

The destination channel buffer provides a similar interface to the destination user application with six pieces of information:

- **Channel number** – The channel number that the destination user application wants to read data from. Together the source and channel number identify the particular destination channel buffer from which data is to be read.
- **Indication** – An indication that a new piece of user information has started to arrive on a particular channel. The indication contains the source logical address and channel number which defines the destination channel buffer the information is being received in.
- **Source** – The logical address of the source that the destination user application wants to read data from.
- **Destination** – The logical address of the destination to which the information is being sent.
- **Cargo** – The data or other information that has been received from the related source channel buffer and placed in the destination channel buffer which is being read by the destination user application.

- **Separator** – An indication of when the last of the complete piece of user information from the source channel buffer has been read out of the destination channel buffer by the destination user application.

3.7.2 Asynchronous Network Functions

In this section the functions used in an asynchronous system to support the Best Effort service are introduced.

3.7.2.1 Segmentation

User information is passed to SpaceWire-T for sending across a SpaceWire network. The size of this user information is arbitrary and unknown a priori to SpaceWire-T. SpaceWire-T sends information across the SpaceWire network in protocol data units (DPs) each with a size up to a specific maximum DP size. To fit the user information into one or more DPs it has to be split into segments that fit into the available space in the DPs. The maximum size of the segment is less that the maximum DP size because the DP will also contain other information concerned with delivering the DP to its intended destination.

The segmentation function is responsible for splitting up the user information into user data segments no larger than the maximum user data segment (UDS) size. The maximum UDS size is 256 bytes. This allows a single byte to be used to specify the data length in a DP and keeps buffer memory requirements low. DPs with data length less than the maximum UDS size are allowed. A service data unit (SDU) greater than 256 bytes in size will be segmented into one or more DPs each containing 256 bytes of data followed by the last DP containing 256 or fewer bytes of data. The minimum UDS size is 1 byte. A data length of 1 means 1 byte of user data, 2 means 2 bytes and so on. A data length of 0 means 256 bytes of user data in the segment.

3.7.2.2 Address Translation

SpaceWire can provide up to 223 logical addresses, permitting up to 223 separate nodes. This number is adequate for most foreseen space missions, so for SpaceWire-T node identification uses the SpaceWire logical address. Path addressing may be used to route a packet to its destination but the node identification is done using the logical address.

A SpaceWire logical address is used to uniquely identify a node attached to the SpaceWire network. A node may be identified by more than one SpaceWire logical address, but there is only one node that has a specific logical address. For example node A can be identified by SpaceWire logical addresses 124 and 125, but logical address 132 cannot be used to identify both node B and node C (at the same time). Note that SpaceWire logical addresses are assigned to nodes not to SpaceWire ports (links) attached to a node. For example, if a node has logical address 212 and if this node has three SpaceWire ports, then SpaceWire packets can be routed to any of these three ports using the logical address 212.

The address translation function translates from the SpaceWire logical address to the SpaceWire address that will be used to send the packet across the network. The SpaceWire address can be a path, logical, or regional logical address or an address constructed using any combination of these addressing modes. The type of address used will be dependent upon the redundancy approach being used. Address resolution is used to determine the SpaceWire address bytes that are included in the header of the SpaceWire packet to route it along the required path across the SpaceWire network to its intended destination.

An example address translation scheme using look-up tables is shown in Figure 3-6.

SpaceWire Logical Address	SpaceWire Address
120	120
124	1, 6, 5, 2, 124
150	1, 132

Figure 3-6 Example Address Translation

SpaceWire logical address 120 identifies a node which is to be sent information using a SpaceWire logical address. In this case the SpaceWire Logical Address is used as the address of the SpaceWire packets that will contain the SDU.

SpaceWire logical address 124 identifies a node which is to be sent information using SpaceWire path addressing.

SpaceWire logical address 150 identifies a node which is to be sent information using a combination of path and logical addressing.

For acknowledgements and buffer flow control tokens the flow of information is in the opposite direction to the flow of DPs for a specific channel. The return path for the acknowledgements and buffer flow control tokens is normally the mirror of the sending path i.e. they follow the same path but in the opposite direction.

3.7.2.3 Error detection

Error detection is needed to ensure that the data is delivered without error. There are six possible types of error:

- Packet received with header error i.e. the header CRC has detected an error in the header.

- Packet delivered to wrong destination i.e. the destination SpaceWire logical address does not correspond to the SpaceWire address (or addresses) of the node that it has been delivered to.
- Packet received with data error i.e. the data CRC has detected an error in the data field.
- Missing packet or out-of sequence packet detected using sequence-number s i.e. the sequence-number of the packet received from a specific source logical address is not one more than the previous packet received from that address. Note that there is a separate sequence count for each channel.
- Duplicated packet received i.e. two packets with the same sequence-number are received. Note that since the sequence-number is an eight-bit number sequence-number and will roll over every 256 packets. Duplicate packet numbers must therefore occur within a certain number of packets, specifically within two times the maximum number of outstanding packets. Sequence-number incrementing and checking is modulo 256.
- SpaceWire Error End of Packet Error (EEP) i.e. somewhere on the path from source to destination a SpaceWire link error (parity, disconnect, credit or escape error) has occurred resulting in the packet being terminated prematurely by an EEP.

Note that a SpaceWire link-level error will result in a packet terminated by an EEP or a packet containing a header or data CRC error.

When an error occurs it is simply logged and optionally reported at the receiving node.

3.7.2.4 Priority

A separate source channel buffer and destination channel buffer are used for each level of priority to be supported for each destination. A DP will be sent from a source channel buffer with a lower channel number before sending a DP from a source channel buffer with a high channel number. The source channel buffers are allocated to destination and priority levels bearing this is mind. For example if there are four destinations (logical addresses 49, 62, 75 and 112) that a source (with logical address 88) needs to send information to and it wants to have three priority levels (low, medium, high) for communication to each of these destinations then the example channel allocation scheme shown in Table 3-1 can be used.

Table 3-1 Example Channel Numbering for Priority Arbitration

Destination	Channel Number SLA / DLA / Channel	Priority
49	88/49/1	High
49	88/49/2	Medium
49	88/49/3	Low
62	88/62/1	High
62	88/62/2	Medium
62	88/62/3	Low
75	88/75/1	High
75	88/75/2	Medium
75	88/75/3	Low
112	88/112/1	High
112	88/112/2	Medium
112	88/112/3	Low

Channel 88/49/1 is used for high priority traffic to the destination with logical address 49.

The source user application puts information into an appropriate source channel buffer depending on where it wants to send the data (destination) and the priority of the data. For example to send data with low priority from the source to the destination with logical address 75, the data is written into source channel buffer 88/75/3. SpaceWire-T will then transfer this information across the SpaceWire network when there is room in the destination channel buffer and after other higher priority traffic with space in its destination buffers has been transferred.

Note a priority scheme where the priority for each channel is explicitly declared rather than being the same as the channel number will also be considered. This is easier to manage but more difficult/expensive to implement in hardware.

Note that SpaceWire-T does not use the router priority feature implemented in the SpW-10X router.

3.7.2.5 Acknowledgement in an Asynchronous Network

When an acknowledgement is required the destination sends an acknowledgement to the source after each DP has been received successfully. When the source sends a DP, it starts a timer. When the DP arrives at the destination, an acknowledgement is returned to the source. If the source does not receive the acknowledgement before the timer times-out, the DP is assumed not to have arrived at the destination and the source cancels sending the rest of the SDU and informs the user application that the SDU was not delivered. If acknowledgements for all the DPs for an SDU are received successfully, the source informs the user application that the SDU has been delivered successfully.

The acknowledgement mechanism is illustrated in Figure 3-7 which shows an SDU being delivered successfully and in Figure 3-8 which shows a problem with SDU delivery.

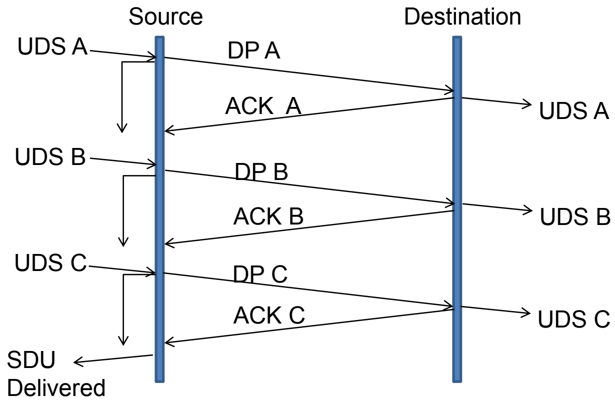

Figure 3-7 SDU Delivered Successfully

An SDU comprising three UDS is to be sent. UDS A is written into the source channel buffer to be sent to the destination. It is read out of this buffer and packaged into DP A which is sent across the SpaceWire network to the destination. A time-out timer is started when the DP is sent waiting for an ACK. When DP A arrives at the destination node the UDS is extracted from the DP and written into the destination channel buffer. An acknowledgement (ACK A) is sent back to the source to indicate that the DP arrived successfully. When ACK A arrives back at the source the time-out timer is cancelled and the next UDS sent. UDS B and UDS C are sent and acknowledged in the same way. When ACK C is received the User Application at the source is informed that the SDU has been delivered successfully.

Figure 3.8 shows what happens when an error occurs while trying to deliver a DP.

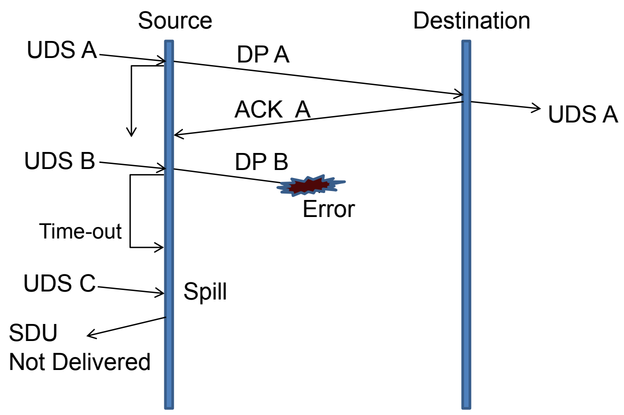

Figure 3-8 SDU Not Delivered

Again an SDU comprising three UDS is to be sent. UDS A is written into the source channel buffer, sent to the destination and acknowledged (ACK A). UDS B is then written into the source channel buffer and sent to the destination. When it is sent a time-out timer is started waiting for an ACK. On its way to the destination DP B is lost or corrupted. Since it does not arrive at the destination no ACK is returned to the source and the time-out timer expires. The User Application is then allowed to write the remainder of the SDU into the source channel buffer and it is spilt (discarded). Once the complete SDU has been received from the User Application, it is informed that the SDU could not be delivered.

3.7.2.6 Flow Control in an Asynchronous Network

End to end flow control is necessary to make sure that there is room in a buffer at the destination node before a DP is sent. This prevents the SpaceWire packet containing the DP being blocked by a destination that is not ready to receive it and thus being strung out across the SpaceWire network blocking other network traffic. Flow control is achieved by the destination channel buffer sending a buffer flow control token (BFCT) when it has enough room for another maximum length DP. To avoid a problem if a BFCT is lost BFCTs are acknowledged. If a BFCT acknowledgement (BACK) is not received within a certain time-out interval then the BFCT is resent. Each BFCT contains a sequence-number which increments each time another BFCT is sent for a specific channel. The sequence-number of the BFCT is also used in the BACK so that each BACK is related to a specific BFCT.

Note that a retry is required for the BFCT because if a BFCT is lost it could block the channel completely. This is much more serious than a loss of an SDU so a BFCT retry mechanism is provided.

The available buffer space is handled in units of the amount of space needed to hold a maximum sized DP because this is the unit of data that is sent. All DPs are the maximum DP size except the last one for a particular SDU.

Normal operation of the flow control mechanism is illustrated in Figure 3-9.

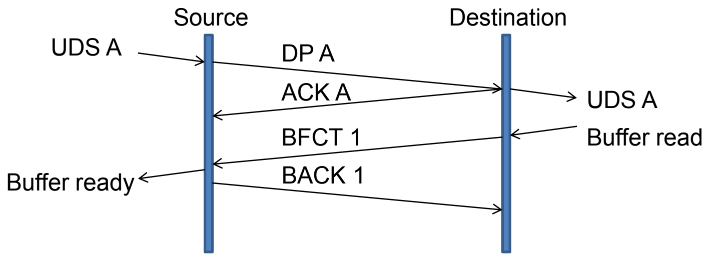

Figure 3-9 Flow Control Mechanism: Normal Operation

In Figure 3-9 a source node is sending data to a destination node across a channel. The source user application passes UDS A to SpaceWire-T to send. It is packaged into DP A and sent across the SpaceWire network. UDS A is extracted from the DP and put in the appropriate destination channel buffer. Sometime later UDS A is read by the destination user application freeing space in the destination channel buffer. This causes a BFCT to be sent back to the source. This BFCT has sequence-number 1 (BFCT 1). When it arrives at the source it signals to the source user application that there is space for another UDS in the destination channel buffer (buffer ready). The source user application can then submit another UDS for sending across the channel when it has more data to send. An acknowledgement to the BFCT (BACK 1) is returned to the destination.

The situation that occurs when a BFCT goes missing is illustrated in Figure 3-10, which follows on from the situation in Figure 3-9.

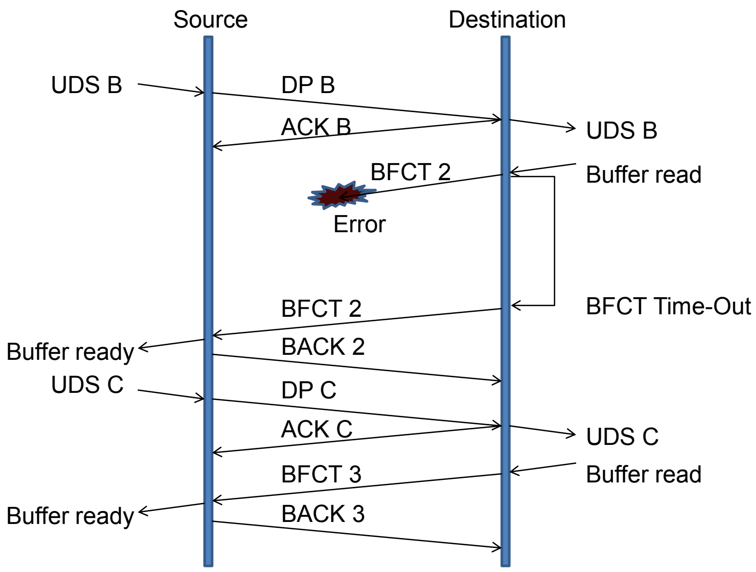

Figure 3-10 Flow Control Mechanism: BFCT Time-Out

The source user application passes UDS B to SpaceWire-T to send. It is packaged into DP B and sent across the SpaceWire network. When it arrives at the destination UDS B is extracted from the DP and put into the available space in the destination channel buffer. This UDS is read from the buffer and a BFCT with the next value sequence-number (BFCT 2) is returned to the source. Unfortunately this BFCT is lost or is corrupted on its way across the network. Since no BFCT arrives at the source the channel is blocked with the source being unable to send another DP because it does not know that there is space in the destination channel buffer.

To overcome the problem of a lost or corrupted BFCT, when the BFCT is sent the destination starts a time-out time waiting for the BACK. This timer is cancelled when a BACK arrives on the channel with a sequence-number equal to or greater than the sequence-number of the BFCT. If no BACK arrives before the time-out timer expires then the BFCT is resent. This ensures that the BFCTs are delivered and that the source can send data to the destination whenever there is room in the destination channel buffer.

So, returning to Figure 3-10, BFCT 2 has been lost or corrupted so no more DPs are received on that channel because the source does not know that there is space available in the destination channel buffer. The BFCT timer in the destination times out and BFCT 2 is resent. This time BFCT 2 arrives safely at the source and the fact that there is more space in the destination channel buffer is reported to the source user application (buffer ready). The source can then submit the next UDS for sending (UDS C).

Figure 3-11 shows what happens when a BFCT is lost but a subsequent BFCT is delivered successfully.

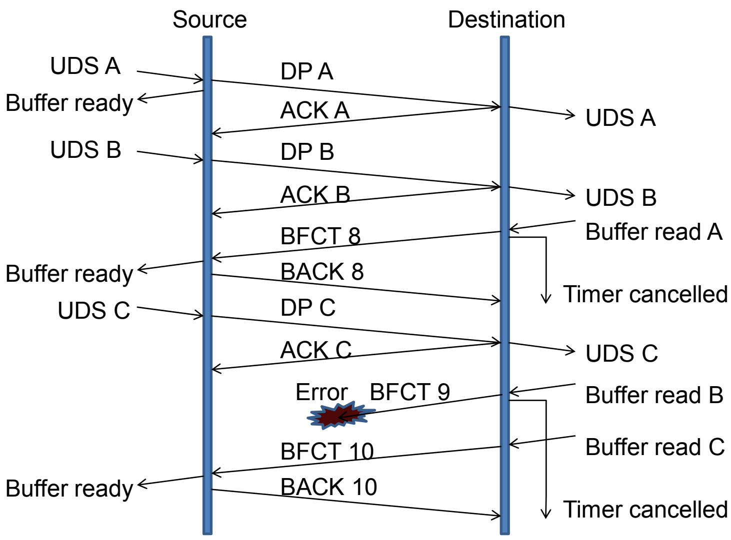

Figure 3-11 Flow Control Mechanism: Cancelling BFCT Time-Out Timers

The source has received two BFCTs already. The source user application passes UDS A to the channel to send. Since the destination channel buffer has room for two more UDSs, the UDS is packaged into DP A and sent across the SpaceWire network. When DP A reaches its destination it is placed in the destination channel buffer and an acknowledgement (ACK A) returned to the source. There is room for one more UDS in the destination channel buffer so the source user application passes UDS B to the channel to send. Packaged into DP B it travels across the SpaceWire network and UDS B is delivered to the destination channel buffer. The destination user application reads UDS A from the destination channel buffer sometime later and a BFCT (BFCT 8) is sent to the source to indicate there is room for one more DP in the destination channel buffer. A time-out timer is started when the BFCT is sent waiting for the corresponding FCAK. BFCT 8 arrives at the source and BACK 8 is returned to the destination cancelling the corresponding time-out timer when it arrives. UDS C is then transferred across the network and ACK C returned to the source. UDS B is read from the destination channel buffer by the destination user application, freeing space for another UDS in this buffer. BFCT 9 is sent to the source to inform it of the available space however it is corrupted or lost on its way across the SpaceWire network. The time-out timer will detect this eventually, but in the meantime UDS C is read from the destination channel buffer and another BFCT (BFCT 10) sent to the source. BFCT 10 arrives at the source safely and the source returns BACK 10 to the destination. Since the previous BFCT that the source received was BFCT 8 when BFCT 10 arrives, it knows that BFCT 9 must have also been sent and been lost, so it can safely assume that there is now space for two UDSs in the destination channel buffer. When BACK 10 arrives at the destination timers for any outstanding BFCT up to BFCT 10 are cancelled.

The situation that occurs when a BACK goes missing is illustrated in Figure 3-12

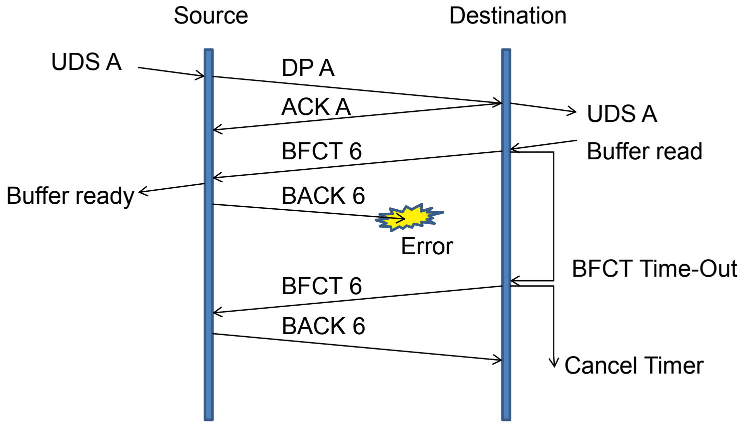

Figure 3-12 Flow Control Mechanism: Missing BACK

UDS A is transferred across the SpaceWire network to the destination channel buffer. A UDS is read out of the destination channel buffer freeing space for another UDS. BFCT 6 is sent to the source to let it know that more space is available and a time-out timer stared waiting for the corresponding

BACK. When BFCT 6 arrives at the source BACK 6 is returned to the destination so that it knows that BFCT 6 has been delivered successfully. Unfortunately BACK 6 is lost or corrupted on its way back to the destination. The BFCT time-out timer expires in the destination and BFCT 6 is resent. BFCT 6 arrives once more at the source. Since it is a duplicate it is ignored, but another BACK 6 is sent back to the destination. This time BACK 6 arrives at the destination successfully and the corresponding BFCT timer is cancelled.

If a SpaceWire packet arrives at a destination where there is no room in a buffer for DP it contains, that packet is spilt (discarded) immediately to prevent the SpaceWire network being blocked.

It is possible that when the BFCT 6 arrives and “buffer ready” is indicted to the source another UDS (UDS 6) is ready for sending straightaway. This situation is illustrated in Figure 3-13.

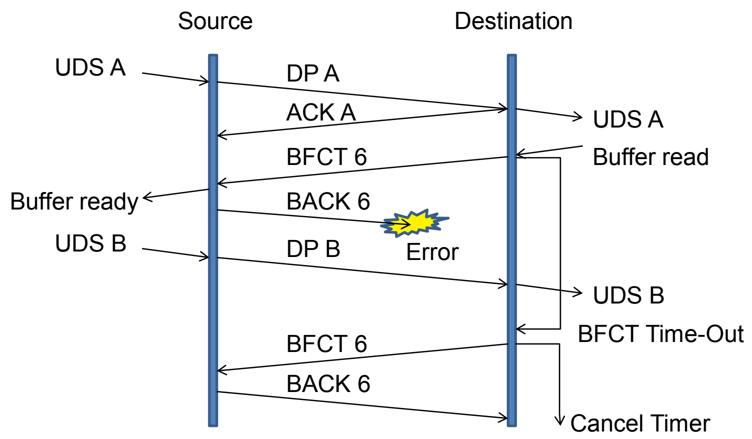

Figure 3-13 Flow Control Mechanism: Missing BACK, UDS Ready

When BFCT 6 arrives at the source BACK 6 is returned to the destination so that it knows that BFCT 6 has been delivered successfully. The fact that the destination buffer is ready to receive more data is indicated to the source. The source buffer has UDS B ready to send to this is transmitted in DP B. BACK 6 is lost or corrupted on its way back to the destination. DP B arrives at the destination, UDS B is extracted and is made available in the destination buffer.

The BFCT time-out timer expires in the destination since BACK 6 was not received and BFCT 6 is resent. BFCT 6 arrives once more at the source. Since it is a duplicate it is ignored, but another BACK 6 is sent back to the destination. This time BACK 6 arrives at the destination successfully and the corresponding BFCT timer is cancelled.

3.7.2.7 Encapsulation

The encapsulation function encapsulates DPs, ACKs, BFCTs and BACKs into SpaceWire packets.

3.7.2.7.1 DP Encapsulation

The DP encapsulation function encapsulates the UDS and associated parameters into a SpaceWire packet. The DP extraction function extracts the UDS from a SpaceWire packet.

The DP encapsulation is illustrated in Figure 3-14.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type / Redundancy
Channel	Sequence Number	Data Length	Header CRC
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data CRC	EOP		

Last octet sent

Figure 3-14 DP Encapsulation

The fields of the DP are described below:

The **destination SpaceWire address** is a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the required destination.

The **destination logical address** is a one byte field containing the logical address of the destination node.

The **protocol identifier** is a one byte field containing the SpaceWire-T protocol identifier value (0x03).

The **source logical address** is a one byte field that identifies the SpaceWire node sending the packet by its logical address.

The **channel** together with the destination and source logical address identifies the channel being used for communication and the associated source and destination channel buffers.

The **type** field is an eight bit field comprising the following sub-fields:

- The **packet type** is a 3-bit field containing the type of packet (DP, ACK, BFCT, BACK, Scheduled ACK (SACK) or Scheduled BFCT (SBFCT)).

- The **redundancy** field is a 2-bit field that is reserved.
- The **start/end** field contains 2-bits which indicate the position of the encapsulated UDS in the SDU as follows:
  - Start/end bits = 10: indicates the DP contains a UDS which is the start of a SDU
  - Start/end bits = 01: indicates the DP contains a UDS which is the end of a SDU.
  - Start/end bits = 00: indicates the DP contains a UDS which is the middle (neither start nor end) of a SDU.
  - Start/end bits = 11: indicates the DP contains a UDS which holds an entire SDU.
- The remaining 1-bit in the type field shall be reserved and set to zero.

The **sequence-number** is a one byte field containing an 8-bit sequence count used to detect missing DPs and BFCTs. There is a separate sequence count for each channel and for DPs and BFCTs within a channel.

The **data length** is a one byte field that specifies the number of data bytes in the data field. The value 0x00 means a data length of 256 bytes. All other values are directly the data length in bytes.

Note: Data length is not strictly necessary and may be removed/replaced in future revisions.

The **header CRC** is a one byte field containing an 8-bit CRC covering the header of the packet. This uses the same CRC format as the SpaceWire RMAP standard (ECSS-E50-51). The header CRC covers the header from the Destination Logical Address to the byte immediately prior to the header CRC. It does not include the Destination SpaceWire Address as this is deleted during passage through the SpaceWire network. The header CRC is used to check that the header is correct before the packet is processed. If there is an error in the header the entire packet is discarded.

The **data field** is a variable length field containing up to 256 data bytes.

The **data CRC** field contains an 8-bit CRC covering the data field only. This is used to confirm that the data has been delivered without error.

The **end of packet marker** is a SpaceWire control code that indicates the end of the SpaceWire packet and the start of the next one.

3.7.2.7.2 Control Code Encapsulation

The encapsulation of control codes (ACKs, BFCTs, and BACKs) is illustrated in Figure 3-15

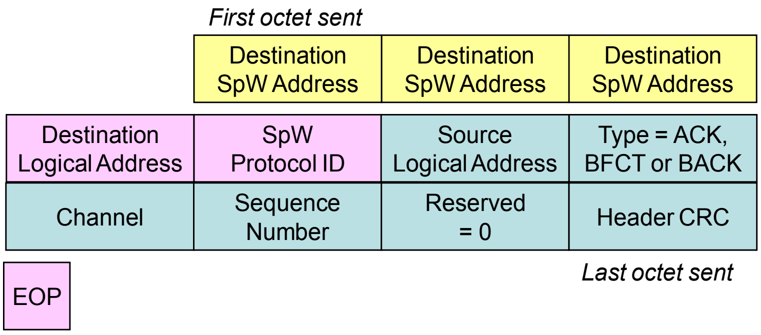

Figure 3-15 Control Code Encapsulation

The control code encapsulation is similar to the DP encapsulation. There is no data field and no data CRC. The data length field in the DP becomes a reserved field set to zero in the control code.

3.7.3 Scheduled Network Functions

In this section the functions used in a scheduled system to support the Resource Reserved service are introduced.

3.7.3.1 Segmentation

Segmentation is identical to that used for asynchronous networks, see section 3.7.2.1.

3.7.3.2 Address Translation

Address translation is identical to that used for asynchronous networks, see section 3.7.2.2.

3.7.3.3 Error detection

Error detection is identical to that used for asynchronous networks, see section 3.7.2.3.

3.7.3.4 Priority

When there are a number of channels between a source and destination pair assigned to a specific time-slot, the lowest number channel has priority. See section 3.7.3.5.

3.7.3.5 Scheduling

3.7.3.5.1 Scheduling and QoS

In a scheduled network the network bandwidth is separated using time-division multiplexing into a series of repeating time-slots. Each time-slot is allocated to one or more channels. A schedule table is used in each source to specify which source channel buffer(s) are allowed to send information during each time-slot. The schedule tables in every source are devised to avoid conflicts on the network. Only one source is allowed to send information at a time, or multiple sources can send information at the same time provided that they do not use any common network resource i.e. send information over the same SpaceWire link. Time-slots can be allocated to more than one channel between a specific source and destination pair. In this case the channel with the lowest channel number will be allowed to send information first.

An example schedule table supporting resource-reserved communication is shown in Table 3-2.

Table 3-2 Example Schedule Table

Source node 231

Time-slot	**Channel**
0	-
1	-
2	231/68/3
3	-
4	231/68/3 231/68/4 231/68/5
5	231/82/1
6	231/68/2
...	
62	-
63	-

During time-slots 0 and 1 the network is being used by other sources.

During time-slot 2, channel 3 from node 231 to 68 (channel 231/68/3) is allowed to send DPs. No other channel from node 231 to 68 is allowed to send DPs.

During time-slot 3, no channel is allowed to send a DP (presumably another source is scheduled to send a DP in this time-slot).

During time-slot 4, channels 3, 4 and 5 from node 231 to 68 (231/68/3) are allowed to send DPs. The lowest channel number has priority.

During time-slot 5, channel 1 from node 231 to node 82 is allowed to send DPs.

During time-slot 6, channel 2 from node 231 to 68 is allowed to send DPs.

In any time-slot if there is no DP ready to send then the time-slot will be unused.

3.7.3.6 Acknowledgement in an Scheduled Network

3.7.3.6.1 Nominal Operation of ACK

In a scheduled network an ACK is sent immediately after a series of DPs have been sent in a particular time-slot. The ACK is sent at the start of the time-slot following the one where the data was sent. The nominal operation of ACK in a scheduled network is illustrated in Figure 3-16.

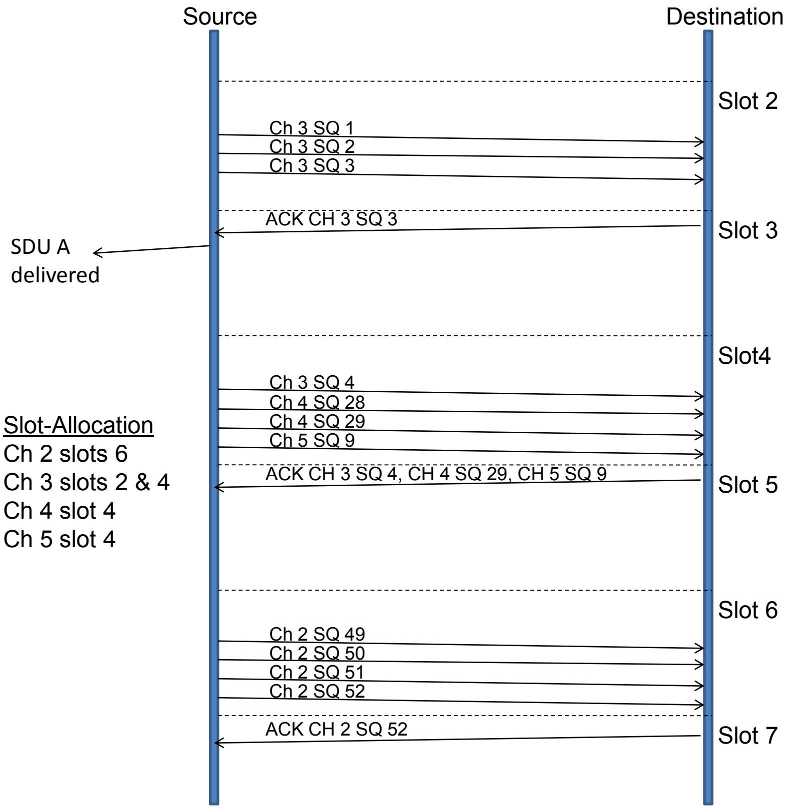

Figure 3-16 Nominal Operation of ACK in Scheduled Network

In this example there are several channels that communicate between a specific source and destination: Channel 2 is scheduled for time-slot 6, channel 3 for time-slots 2 and 4, and channels 4 and 5 are scheduled for time-slot 4 (along with channel 3).

In time-slot 2, after an appropriate wait for any acknowledgement traffic, channel 3 sends three DPs with sequence-number s 1, 2 and 3. These are all the DPs available to be sent over channel 3 so the remainder of the time-slot is unused. All of the DPs sent arrive at the destination with no errors. When time slot 3 starts an acknowledgement for channel 3 is sent which indicates that all DPs up to and including the one with sequence-number 3 have been successfully received. In this example the DP sent over channel 3 with sequence-number 3 is the last DP of an SDU, so the user application at the source is informed that it was delivered successfully.

The rest of time-slot 3 is used by some other source/destination pair to transfer information.

In time-slot 4 there are three channels that can send data. Assume that channel 3 has one DP to send and channels 4 and 5 have 2 and 1 respectively. The lowest priority channel with data to send (and flow control credit which will be considered later) will send its data first. Hence, channel 3 sends its one available DP, channel 4 sends its two, and channel 5 sends one. The DPs sent in time-slot 4 arrive without error so at the beginning of time-slot 5 an acknowledgement for channel 3 sequence–number 4, channel 4 sequence-number 19, and channel 5 sequence-number 9 is sent.

The next time-slot allocated to the specific source/destination pair being considered is time-slot 6. This time-slot is allocated to channel 2 which sends several DPs filling the time-slot capacity.

At the start of time-slot 7 an acknowledgement is sent for channel 2 sequence-number 52.

3.7.3.6.2 Lost Data PDUs

If a DP is lost or corrupted during transmission, any more DPs that are ready to send it the time-slot will be sent. An acknowledgement will be sent at the end of the time-slot and this will indicate that one or more of the DPs sent were not received successfully. The remainder of the SDU being sent will be discarded and the user application informed that the SDU was not delivered successfully. This is illustrated in Figure 3-17.

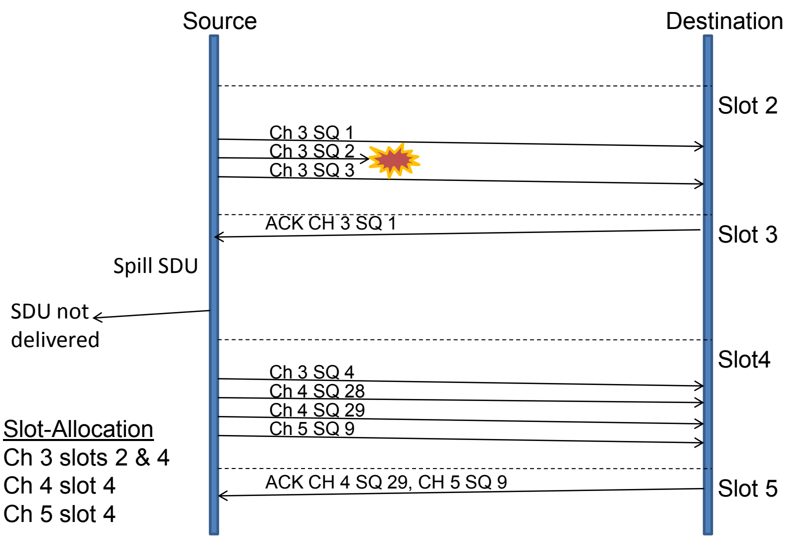

Figure 3-17 ACK in Scheduled Network

3.7.3.6.3 Lost Acknowledgements

A lost acknowledgement will have the same effect as a lost DP, resulting in the discarding of the SDU being sent and an indication to the user application that the SDU was not delivered successfully.

3.7.3.7 Flow Control in a Scheduled Network

Flow control in a scheduled system is handled in a completely different manner to that of an asynchronous system. The primary driver for this is that it is difficult to schedule the BFCTs and BACKs as they travel in opposite directions and a BFCT can occur at any time. Since communications between a particular source/destination pair takes place in one or more specific time-slots, flow control can be sent at the end of the time-slot (or the beginning of the next one) for all the channels between the particular source/destination pair. As flow control information is being sent regularly there is no need for BACKs and the related time-out timers. Furthermore 4-bits serve to indicate whether a particular destination channel buffer is able to accept up to fifteen DPs or not. The number of possible channels between a particular source and destination pair is limited to 32 to constrain the size of packet needed to convey the BFCT information (see section 3.7.3.7). Since 4-bits are used a maximum of fifteen DPs can normally be sent in one time-slot. This depends on the specific time-slot timing which will be considered later.

The flow control information for all the channels between a particular source/destination pair is sent at the beginning of a time-slot. This provides up to date information on the status of the destination channel buffers.

The flow control mechanism for a scheduled system is illustrated in Figure 3-18 .

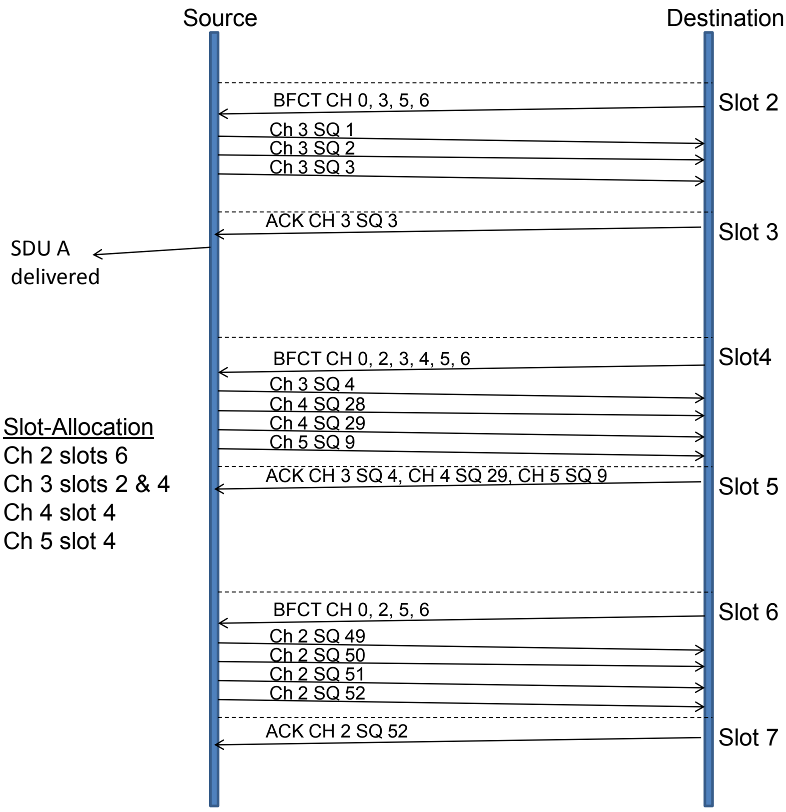

Figure 3-18 Scheduled Flow-Control

A BFCT is sent at the beginning of each time-slot from the destination to the source indicating the amount of space available in each channel. For example in Figure 3-18 at the start of slot 2 a BFCT is sent to indicate that there is room in the destination buffers for channels 0, 3, 5, and 6. For the sake of clarity on the diagram, the specific amount of space available in each of these destination buffers has not been included.

It is possible that the BFCT is lost or corrupted. This situation is illustrated in Figure 3-19. In time-slot 4 the BFCT is lost or corrupted. Since the source does not know what space is available in the destination no DPs are sent in this time-slot.

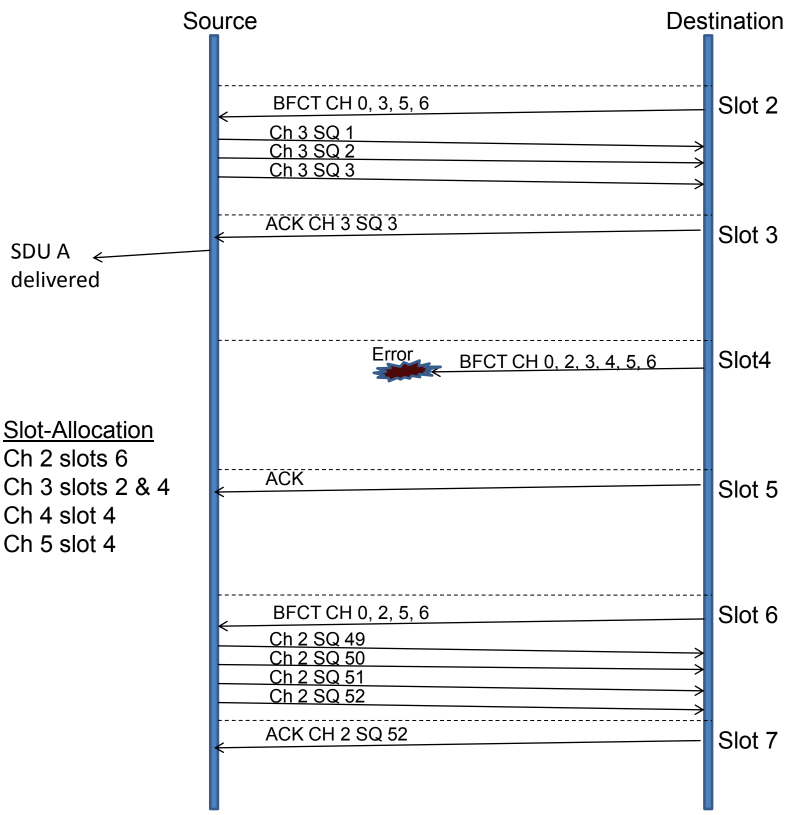

Figure 3-19 Error in Scheduled Flow-Control

3.7.3.8 Time-Slot Timing

The timing of a time-slot in a scheduled network is illustrated in Figure 3-20. Note that the specific timing values provided are an example only and correspond to a hardware implementation of SpaceWire-T.

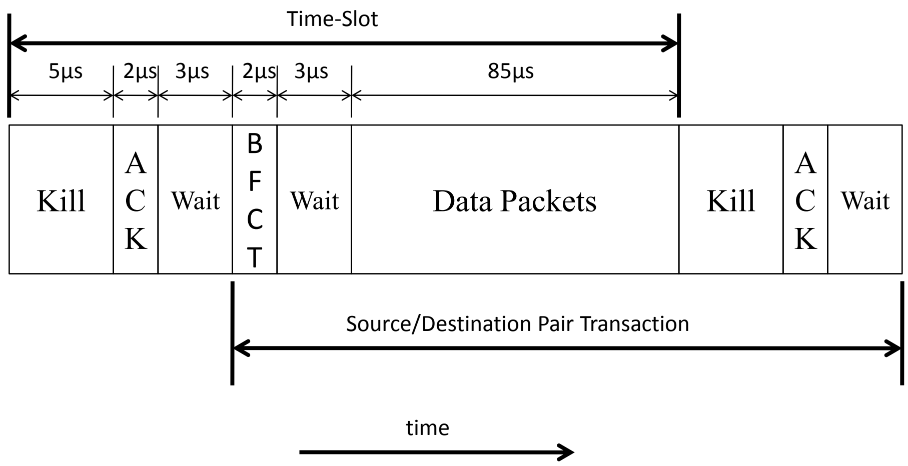

Figure 3-20 Time-Slot Timing

To understand the time-slot timing, consider first the source/destination pair transaction shown in Figure 3-20. This starts with the destination sending the BFCT to the source informing the source about the current status of the destination channel buffers related to the source/destination pair. To allow the BFCTs to propagate through the network unhindered by other packets a gap (wait) follows the BFCT where no transmissions take place. After the wait interval data PDUs are sent from the source to the destination over one or more channels. There is a limit to the number of DPs that can be sent so that they do not extend beyond the end of the Data PDUs interval. At the end of the Data PDUs interval the next time-code arrives. When this happens any packets still being sent are terminated abruptly and a "kill" interval allows time for all traffic on the network to die out.

The killing of packets only happens on the sending side. Packets being received are not killed. Note that if any packets are still being sent when the time-code is distributed, the scheduling has failed and an error is reported. The "killing" of packets still being sent when the time-code arrives is done to prevent fault propagation.

Following the "kill" interval the acknowledgements are sent in the ACK interval for the relevant DPs sent in the previous time-slot. ACK is followed by another wait period to allow the ACKs to propagate across the network. Following this wait interval the next BFCT is sent.

The actual time-slot covers the period from the "kill" interval to the end of the data PDU transfer. This is so that the arrival of the next time-code can terminate the sending of any data PDUs that are still being sent when the time-code arrives. The "kill", ACK, wait, BFCT and wait intervals are then deterministic in length, so that no other traffic is flowing when the Data PDU interval starts.

Typical timing when the SpaceWire link speeds are all set to 200 Mbps are shown in Figure 3-20. This timing results in a time-slot of 100 μs and an epoch of 6.4 ms. Six maximum size DPs can be sent in the 85 μs Data PDU interval. Note that the DP data length field has a maximum value of 256 and is not a user configurable parameter. There is a maximum of 6 DPs in a time-slot.

3.7.3.9 Scheduled Encapsulation

The encapsulation function encapsulates DPs, ACKs, BFCTs and BACKs into SpaceWire packets.

3.7.3.9.1 DP Encapsulation

The DP encapsulation function encapsulates the UDS and associated parameters into a SpaceWire packet. The DP extraction function extracts the UDS from a SpaceWire packet.

The DP encapsulation is illustrated in Figure 3-21 and is the same as that for an asynchronous network.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type / Redundancy
Channel	Sequence Number	Data Length	Header CRC
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data CRC	EOP		

Last octet sent

Figure 3-21 DP Encapsulation

The fields of the DP are described below:

The **destination SpaceWire address** is a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the required destination.

The **destination logical address** is a one byte field containing the logical address of the destination node.

The **protocol identifier** is a one byte field containing the SpaceWire-T protocol identifier value (0x03).

The **source logical address** is a one byte field that identifies the SpaceWire node sending the packet by its logical address.

The **type** field is an eight bit field comprising the following sub-fields:

- The **packet type** is a 3-bit field containing the type of packet (DP).
- The **redundancy** field is a 2-bit field which is reserved.
- The **start/end** field contains 2-bits which indicate the position of the encapsulated UDS in the SDU as follows:
  - Start/end bits = 10: indicates the DP contains a UDS which is the start of a SDU
  - Start/end bits = 01: indicates the DP contains a UDS which is the end of a SDU.

    - Start/end bits = 00: indicates the DP contains a UDS which is the middle (neither start nor end) of a SDU.
    - Start/end bits = 11: indicates the DP contains a UDS which holds an entire SDU.
- The remaining 1-bit in the type field shall be reserved and set to zero.

The **channel number** together with the destination and source logical addresses identifies the channel being used for communication and the associated source and destination channel buffers.

The **sequence-number** is a one byte field containing an 8-bit sequence count used to detect missing DPs and BFCTs. There is a separate sequence count for each channel.

The **data length** is a one byte field that specifies the number of data bytes in the data field. A data length of 0 means there are 256 bytes in the data field.

The **header CRC** is a one byte field containing an 8-bit CRC covering the header of the packet. This uses the same CRC format as the SpaceWire RMAP standard (ECSS-E50-11). The header CRC covers the header from the Destination Logical Address to the byte immediately prior to the header CRC. It does not include the Destination SpaceWire Address as this is deleted during passage through the SpaceWire network. The header CRC is used to check that the header is correct before the packet is processed. If there is an error in the header the entire packet is discarded.

The **data field** is a variable length field containing up to 256 data bytes.

The **data CRC** field contains a 8-bit CRC covering the data field only. This is used to confirm that the data has been delivered without error.

The **end of packet marker** is a SpaceWire control code that indicates the end of the SpaceWire packet and the start of the next one.

3.7.3.9.2 Control Code Encapsulation

The encapsulation of scheduled acknowledgements (SACK) and scheduled BFCTs (SBFCT) are illustrated in Figure 3-22 and Figure 3-23 respectively.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type = SACK
Channel Number	Sequence Number	Channel Number	Sequence Number
Channel Number	Sequence Number	Channel Number	Sequence Number
Channel Number	Sequence Number	Channel Number	Sequence Number
Header CRC	EOP		

Figure 3-22 Scheduled ACK Encapsulation

The scheduled ACK is similar to other PDUs except that it contains a string of channel numbers and sequence-number s in pairs. Each channel number holds the number of the channel being acknowledged. The following sequence-number is the sequence-number of the last DP being acknowledged in the corresponding channel. There is one channel/sequence-number pair for each channel being acknowledged. Only guaranteed and assured PDs are acknowledged.

The fields of the schedule ACK are described below:

The **destination SpaceWire address** is a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the required destination. Since this is an acknowledgement the destination is the channel input node.

The **destination logical address** is a one byte field containing the logical address of the channel input node.

The **protocol identifier** is a one byte field containing the SpaceWire-T protocol identifier value (0x03).

The **type** field is an eight bit field comprising the following sub-fields:

- The **packet type** is a 3-bit field containing the type of packet (scheduled ACK).
- The **redundancy** field is 2-bit field which is reserved.
- The other 3-bits in the type field are reserved and are set to zero.

The **source logical address** is a one byte field that identifies the SpaceWire node sending the packet by its logical address. This is the logical address of the channel output node.

Each **channel number** is the number of a channel being acknowledged.

Each **sequence-number** is a one byte field containing an 8-bit sequence count used to detect missing DPs. There is a separate sequence count for each channel.

The **header CRC** is a one byte field containing an 8-bit CRC covering the header of the packet. This uses the same CRC format as the SpaceWire RMAP standard (ECSS-E50-11). The header CRC covers the header from the Destination Logical Address to the byte immediately prior to the header CRC. It does not include the Destination SpaceWire Address as this is deleted during passage through the SpaceWire network. The header CRC is used to check that the header is correct before the packet is processed. If there is an error in the header the entire packet is discarded.

The **end of packet marker** is a SpaceWire control code that indicates the end of the SpaceWire packet and the start of the next one.

The scheduled BFCT is similar to other PDUs except that it contains a string of 4-bit codes representing the status of all the destination channel buffers in the destination node (output node). These 4-bit codes represent space for zero, one, two, up to fifteen UDS in the corresponding destination channel buffer. The 4-bit codes are in channel number order starting with channel zero. All channels that are implemented in a destination are included in the scheduled BFCT information.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type = SBFCT
BFCT 0-1	BFCT 2-3	BFCT 4-5	BFCT 6-7
BFCT 8-9	BFCT 10-11	BFCT 12-13	BFCT 14-15
BFCT 16-17	BFCT 18-19	BFCT 20-21	BFCT 22-23
BFCT 24-25	BFCT 26-27	BFCT 28-29	BFCT 30-31
Header CRC	EOP		

Figure 3-23 Scheduled BFCT Encapsulation

The fields of the scheduled BFCT are described below:

The **destination SpaceWire address** is a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the required destination. Since this is a BFCT the destination is the channel input node.

The **destination logical address** is a one byte field containing the logical address of the channel input node.

The **protocol identifier** is a one byte field containing the SpaceWire-T protocol identifier value (0x03).

The **source logical address** is a one byte field that identifies the SpaceWire node sending the packet by its logical address. This is the logical address of the channel output node.

The **type** field is an eight bit field comprising the following sub-fields:

- The **packet type** is a 3-bit field containing the type of packet (scheduled BFCT).
- The **redundancy** field is a 2-bit field reserved field.
- The other 3-bits in the type field are reserved and are set to zero.

Each **BFCT** field contains two 4-bit BFCT codes indicating the destination channel buffer status of four destination channel buffers. The first BFCT field contains information about channels 0 and 1, the next BFCT field relates to channels 2 and 3, and so on.

The **4-bit BFCT code** is 0 when there is no room in the destination channel buffer. Values between 1 and 15 indicate room for 1 to 15 maximum sized DPs, respectively.

The **header CRC** is a one byte field containing an 8-bit CRC covering the header of the packet. This uses the same CRC format as the SpaceWire RMAP standard (ECSS-E50-11). The header CRC covers the header from the Destination Logical Address to the byte immediately prior to the header CRC. It does not include the Destination SpaceWire Address as this is deleted during passage through the SpaceWire network. The header CRC is used to check that the header is correct before the packet is processed. If there is an error in the header the entire packet is discarded.

The **end of packet marker** is a SpaceWire control code that indicates the end of the SpaceWire packet and the start of the next one.

4 ASYNCHRONOUS NETWORK SPECIFICATION

This section provides the normative specification for SpaceWire-T running over an asynchronous network.

4.1 USER APPLICATION INTERFACE

4.1.1 Source User Interface

a) The source user interface shall consist of one or more source channel buffers.

b) Each source channel buffer shall be associated with a destination channel buffer in a destination node.

c) There shall be a one to one association between a source channel buffer and a destination channel buffer.

d) Source channel buffers shall be grouped into destination groups i.e. according to the destination node that they are associated with.

e) A maximum of 32 source channel buffers shall be provided in a destination group.

f) Fewer than 32 source channel buffers can be implemented in a destination group.

g) Source channel buffers shall be numbered consecutively from zero.

h) Source channel buffer zero shall be reserved for configuration. It is mandatory to implement this channel.

i) All other source channel buffers except zero shall be available for user application.

j) The prime and zero or more alternative routes to the destination node shall be associated with the source channel buffer. Normally there is one prime route for each destination group and one route for the redundant and any other alternative route to the destination.

k) The QoS provide shall be dependent upon the type of network implemented: Best Effort for an asynchronous network and Resource-Reserved for a scheduled network.

l) The source channel buffer shall indicate to the source user application interface when it has room for more information to send to the destination channel buffer.

m) The source user application shall write data to be sent to the destination channel buffer into the source channel buffer associated with that destination channel buffer.

n) When a complete user application SDU has been written into a source channel buffer the source user application shall indicate to the source channel buffer that the end of an SDU has been reached. This is so that the contents can be transferred as soon as possible without waiting for more data to be put into the source channel buffer.

o) When a complete SDU has been successfully received at the destination the source user application shall be informed that it arrived safely.

p) When an error occurs while sending an SDU the remainder of that SDU shall be discarded and the source user application informed that delivery of the SDU failed.

4.1.2 Destination User Interface

a) The destination user interface shall consist of one or more destination channel buffers.

b) Each destination channel buffer shall be associated with a source channel buffer in a source node.

c) There shall be a one to one association between a destination channel buffer and a source channel buffer.

d) Destination channel buffers shall be grouped into source groups i.e. according to the source node that they are associated with.

e) A maximum of 32 destination channel buffers shall be provided in a source group.

f) Fewer than 32 destination channel buffers may be implemented in a source group.

g) Destination channel buffers shall be numbered consecutively from zero.

h) Destination channel buffer zero shall be reserved for configuration.

i) All other destination channel buffers except zero shall be available for user application.

q) The route back to the source node for acknowledgements and BFCTs shall be associated with the destination channel buffer. Normally there is one route for each source group.

r) The QoS provide shall be dependent upon the type of network implemented: Best Effort for an asynchronous network and Resource-Reserved for a scheduled network.

j) The destination channel buffer shall indicate to the destination user application when it has data available.

k) The destination user application shall read data from the destination channel buffer.

l) Data should be read out in segments of 256 bytes except for the last segment in a SDU.

m) The destination channel buffer shall indicate to the destination user application when a complete user application data unit has been read out of the destination channel buffer.

4.2 Segmentation

a) Data in a source channel buffer shall be split up into user data segments (UDS) no larger than the maximum UDS size of 256 bytes.

b) Each UDS shall be encapsulated in a DP for transferring across the SpaceWire network.

c) An UDS shall be put into a DP and sent across the network whenever there is room in the destination buffer for a complete maximum size DP, and when there is either

   i. 256 or more bytes of user data in the source channel buffer

   ii. Less than 256 bytes of user data followed by an end of user data character.

d) The DP shall be flagged to indicate whether the UDS it contains is the start of an SDU, middle of an SDU, end of an SDU, or both the beginning and end of an SDU (i.e. the SDU fits into a single UDS and hence single DP).

4.3 End to End Flow Control

a) A source channel buffer shall only request to send data to a destination channel buffer when there is room for the maximum size UDS (256 bytes) in the destination channel buffer.

b) The destination channel buffer shall signal that space is available for another maximum size UDS by sending a buffer flow control token to the source channel buffer associated with the destination channel buffer.

c) The buffer flow control token shall contain the logical address of the destination node that contains the destination channel buffer, the channel number of the destination channel buffer, and a BFCT sequence-number.

d) The BFCT sequence-number shall increment each time enough space for a further maximum size UDS becomes available in the destination channel buffer

e) When the BFCT is incremented a BFCT shall be sent to the source node containing the source channel buffer that is sending data to the destination channel buffer.

f) The BFCT sequence-number shall be an 8-bit count.

g) The BFCT sequence-number shall wrap round to 0 when the maximum value (255) is reached.

h) When a BFCT is sent a time-out timer shall be started waiting on the arrival of an acknowledgement to the BFCT (BACK).

i) When a BFCT arrives at the source node it shall be passed to the source channel buffer identified by the destination logical address and channel number contained in the BFCT.

j) The source channel buffer shall keep a note of the sequence-number of the last BFCT it received.

k) The source channel buffer shall keep a note of the amount of available space in the destination channel buffer.

l) If the sequence-number of the BFCT received is greater or equal (modulo 256) to the sequence-number of the last BFCT received then the source channel buffer

   i. Shall update its record of the amount of available space in the destination channel buffer accordingly,

   ii. Shall send an acknowledgement to the BFCT (BACK) containing the sequence-number of the BFCT received.

m) If the sequence-number of the BFCT received is less than the sequence-number of the last BFCT received then the source channel buffer shall ignore the BFCT as it is a duplicate of a previously sent BFCT.

n) When a BACK is received at a node it shall be passed to the appropriate destination channel buffer as determined by the channel number.

o) The destination channel shall keep a record of the sequence-number of the last BACK received.

p) If the sequence-number of the BACK is greater than the sequence-number of the last BACK received,

   i. The timers for all BFCTs with sequence-number s from one more than the last BACK received to the BACK just received shall be cancelled,

   ii. The record of the sequence-number of the last BACK received shall be updated to the sequence-number of the BACK just received.

q) If a BFCT time-out timer expires,

   i. A BFCT shall be sent with the sequence-number of the highest (modulo 256) sequence-number DP received in sequence,

   ii. The time-out timer shall be restarted.

r) If a timer-out timer for a BFCT expire more that a maximum allowed number of times the local host system shall be informed of the error.

4.4 SENDING AND RECEIVING DPS

4.4.1 Types of error:

a) The SpaceWire-T receiver shall detect the following types of error:

  i. Packet received with header error
  ii. Packet delivered to wrong destination or with wrong SpaceWire Protocol ID
  iii. Packet received with data error
  iv. Missing or out of sequence packet
  v. Duplicate packet
  vi. SpaceWire Error End of Packet Error (EEP)

b) If and when a source detects that there has been an error in transmitting an SDU, the remainder of that SDU shall be discarded and the source user application of the failure to deliver it successfully.

4.4.2 CRC generation and checking:

a) The header CRC shall be an 8-bit CRC which is the same as used for RMAP.

b) The header CRC shall cover all bytes in the header from the destination logical address to and including the byte immediately prior to the header CRC.

c) The data CRC shall be an 8-bit CRC which is the same as used for RMAP.

d) The data CRC shall cover all data bytes in the data field.

4.4.3 Sending a DP

a) Each source channel buffer shall have a last sent sequence-number variable which shall hold the sequence-number of the last DP sent from that source channel buffer.

b) Each source channel buffer shall have a last acknowledged sequence-number variable which shall hold the sequence-number of the last DP that has been acknowledged.

c) After power up or reset the last sent and last acknowledged sequence-number variables shall be set to zero so that the sequence-number of the first DP sent from each channel shall be one.

d) When the source channel buffer has a UDS ready to send and there is room for it in the destination channel buffer, it shall request for it to be sent it with the next sequence-number for that channel i.e. the sequence-number will be one more (modulo 256) than that of the last sequence-number variable for that channel.

e) When a DP is sent, a timeout timer shall be started waiting for an acknowledgement from that packet.

f) When a DP is sent, the last sent sequence-number shall be incremented so that it contains the sequence-number of the last DP sent.

4.4.4 Receiving a DP

a) When a DP is received without error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP:

   i. The UDS shall be extracted from the DP and placed in the destination channel buffer.

   ii. An acknowledgement shall be sent to the source of the DP containing:

      a. The logical address of the source that sent the DP being acknowledged,

      b. The logical address of the destination that received the DP and is sending the acknowledgement,

      c. The channel number from the DP, i.e. the channel number in the source,

      d. The sequence-number from the DP.

   iii. The last received sequence-number variable in the destination channel buffer shall be updated to the sequence-number of the DP just received.

4.4.5 Receiving a DP when the Destination Channel Buffer is Full

a) When a DP is received without error and there is no room in the destination channel buffer specified by the source logical address and channel number:

   i. The DP shall be discarded.

   ii. The error shall be logged as a flow control error.

4.4.6 Receiving a Packet Containing Errors

a) When a packet is received with a header error indicated by the header CRC, or by an EOP or EEP being received before the header CRC is received:

   i. The packet shall be discarded,

   ii. The error shall be logged as a header error.

b) When a packet is received with a correct header CRC but the destination logical address does not match a logical address accepted by the destination node:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid destination error.

c) When a packet is received with a correct header CRC, but the source logical address is not one that the destination node accepts data from:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid source error.

d) When a packet is received with a correct header CRC, but the combination of the source logical address and the channel number does not identify a destination channel buffer:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid channel error.

e) When a packet is received with a correct header CRC, but the sequence-number is not equal to or one more than the value of the last received sequence-number :

   i. The packet shall be discarded,

   ii. The error shall be logged as an out of sequence error.

f) When a packet is received with a correct header CRC, but the sequence-number is equal to the value of the last received sequence-number :

   i. The packet shall be discarded,

   ii. The occurrence of a duplicate packet shall be logged.

g) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but a data CRC is detected after the UDS has been written to the destination channel buffer:

   i. The DP shall be discarded,

   ii. The UDS shall be removed from the destination channel buffer,

   iii. The error shall be logged as a data CRC error.

h) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but an EOP occurs before the data CRC:

   i. The DP shall be discarded,

   ii. Any data written to the destination channel buffer from this DP shall be removed,

   iii. The error shall be logged as an early EOP error.

i) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but an EEP occurs before or immediately after the data CRC:

   i. The DP shall be discarded,

   ii. Any data written to the destination channel buffer from this DP shall be removed,

   iii. The error shall be logged as an EEP error.

j) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but the data field is larger than specified in the data length field:

   i. The DP shall be discarded,

   ii. Any data written to the destination channel buffer from this DP shall be removed,

   iii. The error shall be logged as a data length error.

4.4.7 Receiving an Acknowledgement:

a) When an acknowledgement (ACK) is received it shall be passed to the appropriate source channel buffer as determined by the channel number.

b) If the destination logical address and channel number in the acknowledgement do not correspond to a valid channel;

   i. The acknowledgement shall be discarded

   ii. The error shall be logged as an ACK destination error.

c) If the sequence-number in the acknowledgement is greater than the last acknowledged sequence-number and less than or equal to the sequence-number of the last DP sent,

i. Each UDS with a sequence-number greater than the last acknowledged sequence-number and less than or equal to the sequence-number of the acknowledgement shall have its time-out timer cancelled

ii. The last acknowledged sequence-number variable shall be set to the sequence-number of the acknowledgement.

d) If the sequence-number in the acknowledgement is less than or equal to the last acknowledged sequence-number and less than or equal to the sequence-number of the last DP sent,

i. The acknowledgement shall be discarded,

ii. The occurrence of a duplicate acknowledgement shall be logged.

e) If the sequence-number in the acknowledgement is greater than the sequence-number of the last DP sent,

i. The acknowledgement shall be discarded,

ii. An invalid acknowledgement sequence-number shall be recorded.

4.4.8 Acknowledgement Time-Out:

a) When an acknowledgement time-out timer expires,

i. All acknowledgement time-out timers that are still running waiting for acknowledgements for DPs sent from the source channel buffer shall be cancelled.

ii. The current SDU being sent shall be discarded.

iii. The user application shall be informed that the SDU was not delivered successfully.

4.5 Address Translation

a) SpaceWire logical addresses shall be used to identify nodes on a SpaceWire-T network.

b) The maximum number of nodes permitted shall be 223 which is the maximum number of unreserved SpaceWire logical addresses.

c) A node may be identified by more than one SpaceWire logical address.

d) The address translation function shall translate from the SpaceWire logical address of a node to a SpaceWire address that is used to send a SpaceWire packet across the network.

e) The SpaceWire address may be a SpaceWire path address, logical address, regional logical address or an address constructed using any combination of these addressing modes.

f) An address translation table shall translate from the identity of the node that a SpaceWire packet is to be sent (i.e. its logical address) to one or more SpaceWire addresses.

4.6 ENCAPSULATION

4.6.1 Data PDU

a) The DP encapsulation function shall encapsulate a UDS and associated parameters into a SpaceWire packet.

Note: The DP encapsulation is illustrated in Figure 4-1.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type / Redundancy
Channel	Sequence Number	Data Length	Header CRC
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
CRC MS	CRC LS	EOP	

Last octet sent

Figure 4-1 DP Encapsulation

b) The DP extraction function shall extract a UDS from a SpaceWire packet.

c) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the required destination (output node).

d) The destination logical address shall be a one byte field containing the logical address of the destination node (output node).

e) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

f) The source logical address shall be a one byte field that identifies the SpaceWire node sending the packet (input node) by its logical address.

g) The source logical address shall also identify the group of destination channel buffers associated with the source node (input node).

h) The type field shall be an eight bit field comprising the following sub-fields:

   i. The packet type shall be a 3-bit field containing the type of packet (DP, ACK, BFCT, BACK, Scheduled ACK (SACK) or Scheduled BFCT (SBFCT)).

   ii. The redundancy field shall be a two bit field that is reserved.

   iii. An ACK or BACK should use the same redundancy index as the corresponding DP or BFCT.

   iv. The start/end field contains 2-bits which indicate the position of the encapsulated UDS in the SDU as follows:

      i. Start/end bits = 10: indicates the DP contains a UDS which is the start of a SDU

      ii. Start/end bits = 01: indicates the DP contains a UDS which is the end of a SDU.

      iii. Start/end bits = 00: indicates the DP contains a UDS which is the middle (neither start nor end) of a SDU.

      iv. Start/end bits = 11: indicates the DP contains a UDS which holds an entire SDU.

   v. The remaining 1-bit in the type field shall be reserved and set to zero.

i) The channel field shall (together with the destination and source logical addresses) identify the destination channel buffer.

j) The sequence-number shall be a one byte field containing an 8-bit sequence count used to detect missing DPs and BFCTs. There is a separate sequence count for each channel and for DPs and BFCTs within a channel.

k) The data length shall be a one byte field that specifies the number of data bytes in the data field.

l) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

m) The header CRC shall use the same CRC format as the SpaceWire RMAP standard (ECSS-E50-11).

n) The header CRC shall cover the header from the Destination Logical Address to the byte immediately prior to the header CRC.

o) The header CRC shall not cover the Destination SpaceWire Address as this is deleted during passage through the SpaceWire network.

p) The header CRC shall be used to check that the header is correct before the packet is processed.

q) The data field shall be a variable length field containing up to 256 data bytes.

r) The data field shall contain the UDS.

s) The data CRC field shall contain an 8-bit CRC covering the data field only.

t) The data CRC shall use the same CRC format as the SpaceWire RMAP standard (ECSS-E50-11).

u) The data CRC shall be used to confirm that the data has been delivered without error.

v) The end of packet marker shall be a SpaceWire control code that indicates the end of the SpaceWire packet and the start of the next one.

4.6.2 ACK

The encapsulation of an acknowledgement (ACK) control code is illustrated in Figure 4-2.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type = ACK
Channel	Sequence Number	Reserved = 0	Header CRC
EOP			

Last octet sent

Figure 4-2 ACK Encapsulation

a) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the acknowledgement across the SpaceWire network back to the source (input node) of the DP that is being acknowledged.

b) The destination logical address shall be a one byte field containing the logical address of the source of the DP being acknowledged i.e. the logical address of the input node.

c) The destination logical address shall be copied from the source logical address field of the DP being acknowledged.

d) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

e) The source logical address shall be a one byte field that identifies the SpaceWire node sending the acknowledgement by its logical address.

f) The source logical address shall be copied from the destination field of the DP that is being acknowledged.

g) The type field shall be an eight bit field comprising the following sub-fields:

  i. The packet type shall be a 3-bit field containing the type of packet i.e. an ACK.

  ii. The redundancy field shall be a 2-bit field that is reserved.

  iii. The other 3-bits in the type field shall be reserved and set to zero.

h) The channel field shall (together with the destination and source logical addresses) identify the source channel buffer in the input node that sent the DP being acknowledged.

i) The sequence-number shall be a one byte field containing the sequence-number copied from the DP being acknowledged.

j) The reserved field shall be a one byte field that is set to zero.

k) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

4.6.3 BFCT

The encapsulation of BFCT control codes is illustrated in Figure 4-3.

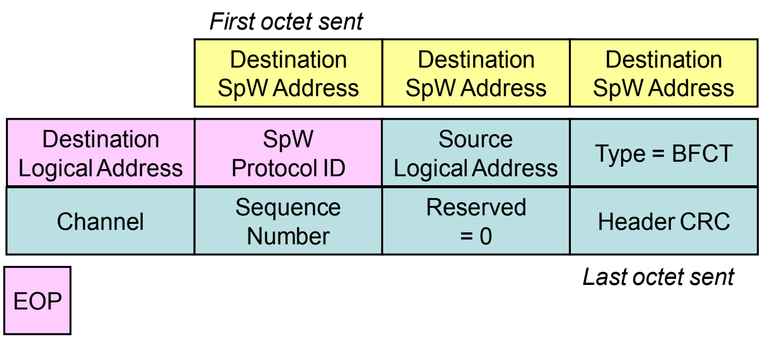

Figure 4-3 BFCT Encapsulation

a) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the BFCT across the SpaceWire network to the source (input node) of DPs for the destination channel buffer that is sending the BFCT.

b) The destination logical address shall be a one byte field containing the logical address of the source (input node) of DPs for the destination channel buffer that is sending the BFCT.

c) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

d) The source logical address shall be a one byte field that identifies the SpaceWire node sending the BFCT by its logical address.

e) The type field shall be an eight bit field comprising the following sub-fields:

   i. The packet type shall be a 3-bit field containing the type of packet i.e. a BFCT.

   ii. The redundancy field shall be a 2-bit field that is reserved.

   iii. The other 3-bits in the type field shall be reserved and set to zero.

f) The channel field shall (together with the destination and source logical addresses) identify the source channel buffer in the input node to which the BFCT is being sent.

g) The BFCT sequence-number shall be a one byte field containing an 8-bit sequence count used to detect missing BFCTs.

h) The reserved field shall be a one byte field that is set to zero.

i) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

4.6.4 BACK

The encapsulation of BACK control codes is illustrated in Figure 4-4.

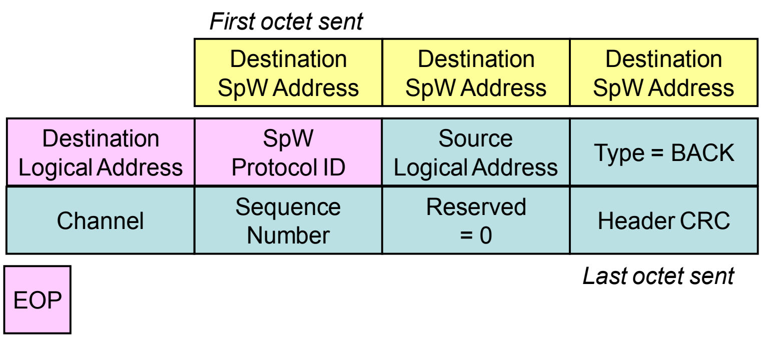

Figure 4-4 BACK Encapsulation

a) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the packet across the SpaceWire network to the destination node (output node) containing the destination channel buffer that sent the BFCT being acknowledged.

b) The destination logical address shall be a one byte field containing the logical address of the destination node (output node) containing the destination channel buffer that sent the BFCT being acknowledged.

c) The destination logical address shall be a copy of the destination logical address in the BFCT that is being acknowledged.

d) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

e) The source logical address shall be a one byte field that identifies the SpaceWire node (input node) sending the BACK by its logical address.

f) The source logical address shall be a copy of the source logical address in the BFCT that is being acknowledged.

g) The type field shall be an eight bit field comprising the following sub-fields:

   i. The packet type shall be a 3-bit field containing the type of packet i.e. a BACK.

   ii. The redundancy field shall be a 2-bit field that is reserved.

   iii. The other 3-bits in the type field shall be reserved and set to zero.

h) The channel field shall (together with the destination and source logical addresses) identify the destination channel buffer in the output node that sent the BFCT being acknowledged.

i) The BFCT sequence-number shall be a one byte field containing the BFCT sequence-number copied from the BFCT that is being acknowledged.

j) The reserved field shall be a one byte field that is set to zero.

k) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

4.7 PRIORITY

a) A separate channel shall be used for each level of priority to be supported for each type of QoS and for each source/destination pair.

b) Priority shall be associated with the channel number.

c) A DP shall be sent from a source channel buffer with a lower channel number before sending a DP from a source channel buffer with a high channel number.

d) The user application can prioritise the order in which it puts SDUs into a specific channel.

5 SCHEDULED NETWORK SPECIFICATION

This section provides the normative specification for SpaceWire-T running over a scheduled network.

5.1 USER APPLICATION INTERFACE

5.1.1 Source User Interface

a) The source user interface for a scheduled network shall be identical to that for an asynchronous network (see sub-clause 4.1.1).

5.1.2 Destination User Interface

a) The source user interface for a scheduled network shall be identical to that for an asynchronous network (see sub-clause 4.1.2).

5.2 SEGMENTATION

a) Segmentation for a scheduled network shall be identical to that for an asynchronous network (see sub-clause 4.2)

5.3 END TO END FLOW CONTROL

a) A source channel buffer shall only request to send data to a destination channel buffer when there is room for the maximum size UDS (256 bytes) in the destination channel buffer.

b) Each destination channel buffer shall signal the space available in the buffer by sending a scheduled buffer flow control token (SBFCT) near the start of each time-slot assigned to the source/destination pair for communication.

c) The scheduled buffer flow control token shall contain buffer status information on every destination channel buffer that connects to the present source node.

d) The buffer status information shall be encoded in 4-bits as follows:

   i. 0000 = no room in the destination channel buffer

   ii. 0001 to 1111 = room for one to fifteen maximum sized UDS in the destination channel buffer, respectively. If more space is available the value fifteen shall be used.

e) The buffer status information for the channel 0 shall be first followed by that for channel 1 and then sequentially all the other channels between the source and destination pair.

f) There shall be a maximum of 32 channels between any source and destination pair

g) The scheduled buffer flow control token shall contain the logical address of the destination node sending the token.

h) When a SBFCT arrives at the source node the destination channel buffer information it contains shall be passed to the set of source channel buffers identified by the destination logical address within the SBFCT.

i) Each source channel buffer shall keep a note of the space available in the corresponding destination channel buffer.

5.4 Sending and Receiving DPs

5.4.1 Types of error:

a) The SpaceWire-T receiver in a scheduled network shall detect the following types of error:

i. Packet received with header error
ii. Packet delivered to wrong destination or with wrong SpaceWire Protocol ID
iii. Packet received with data error
iv. Missing packet
v. Duplicate packet
vi. SpaceWire Error End of Packet Error (EEP)

5.4.2 CRC generation and checking:

a) The header CRC used in a scheduled network shall be identical to that used in an asynchronous network (see sub-clause 4.4.2)

b) The data CRC used in a scheduled network shall be identical to that used in an asynchronous network (see sub-clause 4.4.2)

5.4.3 Sending a DP

a) Each source channel buffer shall have a last sent sequence-number variable which shall hold the sequence-number of the last DP sent from that source channel buffer.

b) Each source channel buffer shall have a last acknowledged sequence-number variable which shall hold the sequence-number of the last DP that has been acknowledged.

c) After power up or reset the last sent and last acknowledged sequence-number variables shall be set to zero so that the sequence-number of the first DP sent from each channel shall be one.

d) When the source channel buffer has a UDS ready to send and there is room for it in the destination channel buffer, it shall request for it to be sent it with the next sequence-number for that channel i.e. the sequence-number will be one more (modulo 256) than that of the last sequence-number variable for that channel.

e) Whether the destination channel buffer has any room shall be determined from the last received SBFCT and knowledge of any DPs sent over the channel since then.

f) When a DP is sent, the last sent sequence-number shall be incremented so that it contains the sequence-number of the last DP sent.

5.4.4 Receiving a DP

a) When a DP is received without error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP:

   i. The UDS shall be extracted from the DP and placed in the destination channel buffer.

   ii. An acknowledgement shall be sent to the source of the DP containing:

      a. The logical address of the source that sent the DP being acknowledged,

      b. The logical address of the destination that received the DP and is sending the acknowledgement,

      c. The channel number from the DP,

      d. The sequence-number from the DP.

   iii. The last received sequence-number variable in the destination channel buffer shall be updated to the sequence-number of the DP just received.

5.4.5 Receiving a DP when the Destination Channel Buffer is Full

a) When a DP is received without error and there is no room in the destination channel buffer specified by the source logical address and channel number:

   i. The DP shall be discarded.

   ii. The error shall be logged as a flow control error.

5.4.6 Receiving a Packet Containing Errors

a) When a packet is received with a header error indicated by the header CRC, or by an EOP or EEP being received before the header CRC is received:

   i. The packet shall be discarded,

   ii. The error shall be logged as a header error.

b) When a packet is received with a correct header CRC but the destination logical address does not match a logical address accepted by the destination node:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid destination error.

c) When a packet is received with a correct header CRC, but the source logical address is not one that the destination node accepts data from:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid source error.

d) When a packet is received with a correct header CRC, but the combination of the source logical address and the channel number does not identify a destination channel buffer:

   i. The packet shall be discarded,

   ii. The error shall be logged as an invalid channel error.

e) When a packet is received with a correct header CRC, but the sequence-number is not equal to or one more than the value of the last received sequence-number :

   i. The packet shall be discarded,

   ii. The error shall be logged as an out of sequence error.

f) When a packet is received with a correct header CRC, but the sequence-number is equal to the value of the last received sequence-number :

   i. The packet shall be discarded,

   ii. The occurrence of a duplicate packet shall be logged.

g) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but a data CRC is detected after the UDS has been written to the destination channel buffer:

i. The DP shall be discarded,

ii. The UDS shall be removed from the destination channel buffer,

iii. The error shall be logged as a data CRC error.

h) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but an EOP occurs before the data CRC:

i. The DP shall be discarded,

ii. Any data written to the destination channel buffer from this DP shall be removed,

iii. The error shall be logged as an early EOP error.

i) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but an EEP occurs before or immediately after the data CRC:

i. The DP shall be discarded,

ii. Any data written to the destination channel buffer from this DP shall be removed,

iii. The error shall be logged as an EEP error.

j) When a DP is received without a header CRC error and its sequence-number is one more than the last received sequence-number for the destination channel buffer and there is room in the destination channel buffer specified by the source logical address and channel number in the DP but the data field is larger than specified in the data length field:

i. The DP shall be discarded,

ii. Any data written to the destination channel buffer from this DP shall be removed,

iii. The error shall be logged as a data length error.

5.4.7 Receiving an Acknowledgement:

a) A scheduled acknowledgement (SACK) shall contain the channel numbers and associated sequence-numbers for every packet sent in the last time-slot.

b) A SACK shall be sent shortly after all the DPs being transferred in one time-slot have been received.

c) There shall be one SACK covering all DPs transferred in one time-slot.

d) If the destination logical address in the SACK does not correspond to one or more valid channels;

   i. The acknowledgement shall be discarded

   ii. The error shall be logged as a SACK channel error.

e) When the SACK is received the individual acknowledgements for each channel covered by the SACK shall be extracted and passed to the appropriate source channel buffer as determined by the channel number.

f) If the sequence-number in an acknowledgement for a channel is greater than the last acknowledged sequence-number for that channel and less than or equal to the sequence-number of the last PD sent,

   i. The last acknowledged sequence-number variable for the channel shall be set to the sequence-number of the acknowledgement.

g) If the sequence-number in the acknowledgement for a channel is less than or equal to the last acknowledged sequence-number for that channel and less than or equal to the sequence-number of the last PD sent,

   i. The acknowledgement shall be discarded,

   ii. The occurrence of a duplicate acknowledgement shall be logged.

h) If the sequence-number in the acknowledgement for a channel is greater than the sequence-number of the last PD sent,

   i. The acknowledgement shall be discarded,

   ii. An invalid acknowledgement sequence-number shall be recorded.

5.5 Address Translation

a) The address translation used in a scheduled network shall be identical to that used in an asynchronous network (see sub-clause 4.5)

5.6 Encapsulation

5.6.1 Data PDU

a) The DP encapsulation function shall encapsulate a UDS and associated parameters into a SpaceWire packet.

Note: The DP encapsulation is illustrated in Figure 5-1.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type / Redundancy
Channel	Sequence Number	Data Length	Header CRC
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data CRC	EOP		

Last octet sent

Figure 5-1 DP Encapsulation

b) The DP encapsulation used in a scheduled network shall be identical to that used in an asynchronous network (see sub-clause **Error! Reference source not found.**).

5.6.2 Scheduled ACK

The encapsulation of the Scheduled ACK (SACK) control code is illustrated in Figure 5-2.

First octet sent

	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type = SACK
Channel Number	Sequence Number	Channel Number	Sequence Number
Channel Number	Sequence Number	Channel Number	Sequence Number
Channel Number	Sequence Number	Channel Number	Sequence Number
Header CRC	EOP		

Figure 5-2 Scheduled ACK Encapsulation

a) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the acknowledgement across the SpaceWire network back to the source (input node) of the DP that is being acknowledged.

b) The destination logical address shall be a one byte field containing the logical address of the source (input node) of the DP being acknowledged.

c) The destination logical address shall be copied from the source logical address field of the DP being acknowledged.

d) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

e) The source logical address shall be a one byte field that identifies the SpaceWire node sending the acknowledgement by its logical address.

f) The source logical address shall be copied from the destination field of the DP that is being acknowledged.

g) The type field shall be an eight bit field comprising the following sub-fields:

    i. The packet type shall be a 3-bit field containing the type of packet i.e. a SACK.

    ii. The redundancy field shall be a 2-bit field that is reserved.

    iii. The other 3-bits in the type field shall be reserved and set to zero.

h) The channel fields shall (together with the destination and source logical addresses) identify source channel buffers in the input node that sent the DPs being acknowledged.

i) There shall be a channel field for each channel being acknowledged.

j) Each channel field shall have a sequence field associated with it.

k) The sequence field shall be a one byte field containing the most recent in-sequence sequence-number received on the associated channel.

l) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

5.6.3 SBFCT

The encapsulation of the Scheduled BFCT (SBFCT) control code is illustrated in Figure 5-3.

	First octet sent		
	Destination SpW Address	Destination SpW Address	Destination SpW Address
Destination Logical Address	SpW Protocol ID	Source Logical Address	Type = SBFCT
BFCT 0-1	BFCT 2-3	BFCT 4-5	BFCT 6-7
BFCT 8-9	BFCT 10-11	BFCT 12-13	BFCT 14-15
BFCT 16-17	BFCT 18-19	BFCT 20-21	BFCT 22-23
BFCT 24-25	BFCT 26-27	BFCT 28-29	BFCT 30-31
Header CRC	EOP		

Figure 5-3 Scheduled BFCT Encapsulation

a) The destination SpaceWire address shall be a variable length field that contains the SpaceWire path and/or regional logical address that routes the SBFCT across the SpaceWire network to the source (input node) of DPs for the destination channel buffer that is sending the SBFCT.

b) The destination logical address shall be a one byte field containing the logical address of the source (input node) of DPs for the destination channel buffer that is sending the SBFCT.

c) The protocol identifier shall be a one byte field containing the SpaceWire-T protocol identifier value (0x03).

d) The source logical address shall be a one byte field that identifies the SpaceWire node sending the SBFCT by its logical address.

e) The type field shall be an eight bit field comprising the following sub-fields:

   i. The packet type shall be a 3-bit field containing the type of packet i.e. a SBFCT.

   ii. The redundancy field shall be a 2-bit field that is reserved.

   iii. The other 3-bits in the type field shall be reserved and set to zero.

f) Each BFCT field shall contain two 4-bit BFCT codes which indicate the destination channel buffer status of four destination channel buffers.

g) The first BFCT field shall contain information about channel 0-1, the next BFCT field information about channels 2-3, and so on.

h) Only the BFCT fields for channels that are implemented shall be included.

i) The 4-bit BFCT code shall be:

i. 0 when there is no room in the destination channel buffer.

ii. 1 when there is room for 1 maximum sized DP in the destination channel buffer.

iii. 2 when there is room for 2 maximum sized DP in the destination channel buffer.

iv. 3 to14 when there is room for 3 to 14 maximum sized DP in the destination channel buffer, respectively.

v. 15 when there is room for 15 or more maximum sized DP in the destination channel buffer.

j) The header CRC shall be a one byte field containing an 8-bit CRC covering the header of the packet.

5.7 Priority

a) A separate channel shall be used for each level of priority to be supported for each destination.

b) Priority shall be associated with the channel number.

c) A DP shall be sent from a source channel buffer with a lower channel number before sending a DP from a source channel buffer with a high channel number.

d) The order of access to the media shall be dependent on two things: priority of data PDUs that are available to send and availability of space in the corresponding destination buffer.

e) The packet send next shall be the packet with highest priority that is both ready to send and for which there is space in the corresponding destination buffer.

5.8 Resource Reservation

5.8.1 Time-Slots

a) Time-slot boundaries shall be distributed using SpaceWire time-codes.

b) There shall be 64 time-slots each identified by the time value (0-63) in the SpaceWire time-code.

c) Each time-slot starts on receipt of a time-code and ends on the receipt of the next time-code.

d) Time-codes shall be sent out at a uniform rate determined by a managed parameter, the time-code rate.

e) An Epoch shall be defined as being 64 time-slots in length starting with time-slot 0 and ending with time-slot 63.

f) A time-slot shall contain several time intervals as illustrated in Figure 5-4

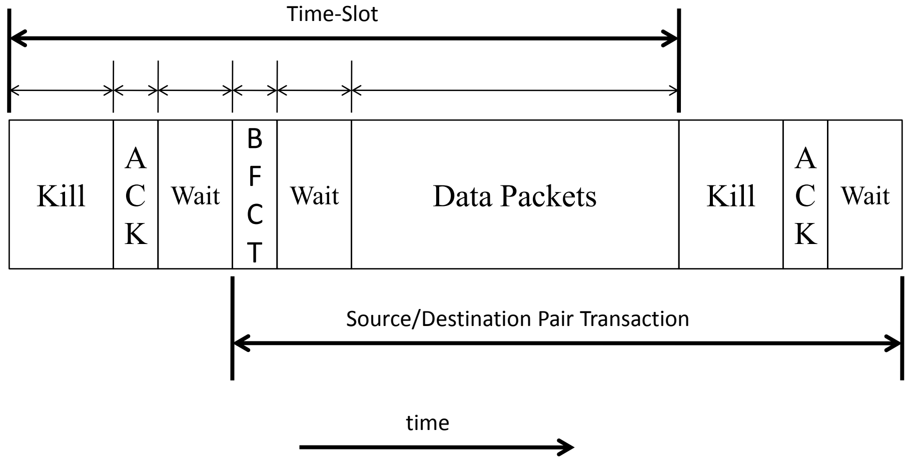

Figure 5-4 Time-Slot Timing

g) The time-slot shall begin with the “kill” interval during which any data PDUs still being sent are terminated by the source immediately with an EEP.

h) The destination shall not terminate DPs.

i) The ACK interval shall follow the “kill” interval.

j) During the ACK interval the scheduled acknowledgements shall be sent for the data PDUs transferred between the source/destination pair serviced in the previous time-slot.

k) A “wait” interval shall follow the ACK interval during which time any scheduled ACKs can propagate across the SpaceWire network.

l) After the “wait” interval following the ACK interval, the scheduled BFCT interval shall start during which time the scheduled BFCTs are sent from the destination back to the source.

m) A “wait” interval shall follow the BFCT interval during which time any scheduled BFCTs can propagate across the SpaceWire network.

n) After the “wait” interval following the BFCT interval, the data PDU interval shall start during which time data PDUs may be sent from the source to the destination.

o) Multiple source/destination pairs may operate in the same time-slot provided that they do not use anywhere the same SpaceWire link.

5.8.2 Resources and Channels

a) A channel shall comprise a source channel buffer in the input node, a destination channel buffer in the output node and a router or path through the SpaceWire network from the source to the destination.

b) The channel shall be uniquely identified by the logical address of the source, the logical address of the destination, and a channel number.

c) The channel number shall be an 8-bit value that identifies the source channel buffer in the source and the destination channel buffer in the destination for a particular channel.

d) The source channel buffer and destination channel buffer for a particular channel shall both have the same channel number.

5.8.3 Scheduled System

a) In a scheduled system the control of network traffic shall be done using time-slots.

b) In a time-slot a number of source/destination pairs shall be allowed to communicate concurrently provided that there is no conflicting use of resources, i.e. communication between the different source/destination pairs must not use the same SpaceWire link. Note that this includes SpaceWire links in the middle of a SpaceWire network.

c) Channels shall be assigned to time-slots in such a way that there is no conflicting use of resources.

d) The assignment of channels to time-slots shall be referred to as the schedule table.

e) When a specific time-slot starts, any source with a channel assigned to that time-slot may send a packet.

6 NETWORK CONFIGURATION PARAMETERS

This section has to be completed.

a) The network manager shall be responsible for network configuration.

b) The SpaceWire links shall be configured to AutoStart.

c) The routing tables in the SpaceWire router shall be configured by network management to provide the required paths for SpaceWire logical addresses.

d) Configuration of SpaceWire routers shall be performed using the SpaceWire RMAP protocol (ECSS-E50-11).

e) All SpaceWire links in the network shall be set to operate at the same data rate.

f) Automatic power-down on link silence may be enabled for SpaceWire links in the network to save power when the traffic over the link is expected to be sporadic.

7 Document Changes

The changes made this document are listed in this section.

Table 7-1: Changes to Document (v2.2 to V3.1)	
Section/Reference	**Change**
Entire document	Major changes throughout document