SpaceWire-R

DRAFT

SCDHA 151-0.~~3~~4
Issue 0.~~3~~4
13 August 2015

Takahiro Yamada
Japan Aerospace Exploration Agency (JAXA)
Institute of Space and Astronautical Science (ISAS)

CONTENTS

1. INTRODUCTION

1.1. PURPOSE

The purpose of this document is to specify a communications protocol called “SpaceWire-R” that provides onboard applications with reliable data transfer services over SpaceWire [A1] networks.

1.2. SCOPE

This document specifies SpaceWire-R in terms of the interfaces between SpaceWire nodes. It does not specify how these interfaces should be implemented with hardware or software.

1.3. APPLICABILITY

This document applies to development of spacecraft onboard subsystems and instruments and their ground support equipment for implementing interfaces that conform to SpaceWire-R.

1.4. REFERENCES

1.4.1. APPLICABLE DOCUMENTS

[A1] European Cooperation for Space Standardization (ECSS), “Space Engineering - SpaceWire - Links, nodes, routers and networks,” ECSS-E-50-12A, January 2003.

[A2] International Standard Organization (ISO), “Information Technology - Open Systems Interconnection - Basic Reference Model: The Basic Model,” International Standard, ISO/IEC 7498-1, 2nd ed., 1994.

[A3] International Standard Organization (ISO), “Information Technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services,” International Standard, ISO/IEC 10731, 1994.

[A4] European Cooperation for Space Standardization (ECSS), “Space Engineering - SpaceWire protocol identification,” ECSS-E-ST-50-51C, February 2010.

1.4.2. REFERENCE DOCUMENTS

[R1] Sandia National Laboratories, “Joint Architecture Standard (JAS) Reliable Data Delivery Protocol (RDDP) Specification,” Sandia Report SAND2011-3500, May 2011.

[R2] Goddard Space Flight Center, “GOES-R Reliable Data Delivery Protocol (GRDDP),” 417-R-RPT-0050, January 2008.

1.5. DOCUMENT STRUCTURE

This document is organized as follows:

a) Section 1 (this section) shows the purpose, scope, and applicability of the document, and lists the references, definitions, and notations used throughout the document;

b) Section 2 presents an overview of SpaceWire-R;

c) Section 3 defines the services provided by the protocol entity;

d) Section 4 specifies protocol data units and procedures employed by the protocol entity;

e) Section 5 presents a standard method for specifying the parameters of SpaceWire-R used for a particular SpaceWire network;

h) Appendix A lists all acronyms used in this document;

i) Appendix B shows some examples of exchanges of protocol data units (SpaceWire-R Packets) between a sender and a receiver; and

j) Appendix C shows an example of specifying the parameters of SpaceWire-R used for a particular SpaceWire network.

1.6. DEFINITIONS AND NOTATIONS

1.6.1. DEFINITIONS FROM SPACEWIRE

This document makes use of the following terms defined in “SpaceWire - Links, nodes, routers and networks” [A1]:

a) cargo;

b) destination identifier;

c) end-of-packet marker;

d) logical address;

e) network;

f) node;

g) (SpaceWire) packet; and

h) Time-Code.

1.6.2. DEFINITIONS FROM OSI BASIC REFERENCE MODEL

This document makes use of the following terms defined in “Open Systems Interconnection - Basic Reference Model: The Basic Model” [A2]:

a) (N)-entity (for brevity, this is called “entity” in this document);

b) flow control;

c) protocol data unit;

d) segmenting;

e) service; and

f) service data unit.

1.6.3. DEFINITIONS FROM OSI SERVICE DEFINITION CONVENTIONS

This document makes use of the following terms defined in “Information Technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services” [A3]:

a) indication;

b) primitive; and

c) request.

1.6.4. TERMS DEFINED IN THIS DOCUMENT

The following definitions apply throughout this document.

Receive TEP: The receiving end of a Transport Channel.

Transmit TEP: The transmitting end of a Transport Channel.

Transport Channel: A protocol-defined one-way logical data path between a transmit TEP and a receive TEP. There is only one Transport Channel between a Transmit TEP and a Receive TEP, but there can be multiple Transport Channels between any pair of nodes.

Transport End Point (TEP): A point in a node that transmits or receives application data using a Transport Channel over a SpaceWire network. A TEP is either a Transmit TEP or a Receive TEP.

1.6.5. NOTATIONS

The following notations apply throughout this document.

A paragraph after [Example] (or [Example *n*], where *n* is a positive integer) shows an example that helps to understand the specification, which is not part of the specification.

A paragraph after [Temporary Note] (or [Temporary Note *n*], where *n* is a positive integer) is an editorial note that will be removed from the final document.

A paragraph after [Note] (or [Note *n*], where *n* is a positive integer) contains an informative note that helps to understand the specification, which is not part of the specification.

1.6.6. CONVENTIONS

In this document, the following convention is used to identify each bit in an N-bit field. The first bit in the field to be transmitted (i.e., the most left justified when drawing a figure) is defined to be “Bit 0”; the following bit is defined to be “Bit 1” and so on up to “Bit N-1”. When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e., “Bit 0” (see Figure 1-1).

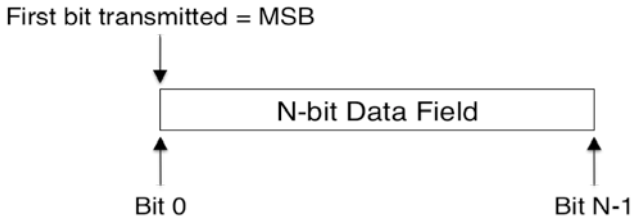

Figure 1-1: Bit Numbering Convention

In accordance with standard data-communications practice, data fields are often grouped into 8-bit “words” which conform to the above convention. Throughout this document, such an 8-bit word is called an “octet”. The numbering for octets within a data structure starts with 0.

2. OVERVIEW

This section provides an informative overview of SpaceWire-R, which will help the readers to understand the specification given in this document.

2.1. OVERALL PROTOCOL ARCHITECTURE

SpaceWire-R is a protocol that provides onboard applications with reliable data transfer services over SpaceWire [A1] networks.

The overall protocol architecture for SpaceWire networks when SpaceWire-R is used is shown in Figure 2-1. The basic SpaceWire protocol specified in [A1] is used at the bottom of the protocol stack. SpaceWire-R specified in this document is used by applications on top of the basic SpaceWire protocol.

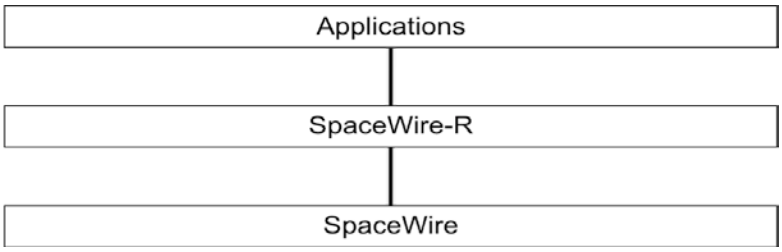

Figure 2-1: Overall Protocol Architecture

2.2. PROTOCOL FEATURES

2.2.1. GENERAL

The primary function of SpaceWire-R (SpW-R) is to transfer data reliably from a sending node to a receiving node over a SpaceWire network. Including this function, this protocol performs the following functions:

a) Multiplexing;

b) Segmentation;

c) Reliable transfer;

d) Flow control (optional); and

e) Heartbeat (optional).; and

f) Redundancy control (optional).

Each of these functions will be briefly explained in the following subsections.

2.2.2. MULTIPLEXING

SpW-R provides a mechanism for multiplexing independent flows of data from a sending end identified by a SpaceWire Logical Address (SLA) to a receiving end identified by another SLA. Each of such flows of data is called a Transport Channel. Each Transport Channel is controlled independently and can have different parameters than other Transport Channels for controlling data flow.

2.2.3. SEGMENTATION

If a data unit that the sending application wants to send (a service data unit) does not fit in a protocol data unit of SpW-R (SpW-R Packet) that can be transmitted with the underlying protocol of the SpaceWire network, SpW-R

can break the service data unit into smaller pieces so that each piece can be transmitted in a SpW-R Packet. At the receiving end, the original service data unit is reconstructed from a series of received SpW-R packets and delivered to the receiving application.

2.2.4. RELIABLE TRANSFER

The primary function of SpW-R is to guarantee transfer of data from the sending application to the receiving application without loss, without duplication, and in the same sequence as provided by the sending application. In order to do this, this protocol performs retransmissions of data when no confirmation of the reception of the data has been obtained from the receiver.

2.2.5. FLOW CONTROL

SpW-R provides an optional mechanism for the receiving entity of the protocol to tell the sending entity how many more data units it can receive so that the sending entity does not send data that cannot be received by the receiving entity.

2.2.6. HEARTBEAT

SpW-R provides an optional mechanism for the sending and/or receiving entities of the protocol to check whether the link and the other entity are still alive when there is no data to send or receive.

~~**2.3.0. REDUNDANCY CONTROL**~~

~~When the protocol finds that a route from the sending node to the receiving node is not functioning correctly, it can switch the route to a redundant one (if there is such a redundant route).~~

~~2.5.~~2.3. RELATIONSHIP WITH OTHER PROTOCOLS

This protocol has been developed based on a protocol called the Joint Architecture Standard (JAS) Reliable Data Delivery Protocol (RDDP) [R1] developed by the Sandia National Laboratories, which was based on a protocol called the GOES-R Reliable Data Delivery Protocol (GRDDP) [R2] developed by the Goddard Space Flight Center (GSFC) of NASA. The differences between the JAS RDDP and the GRDDP are described in [R1].

The technical differences between this protocol and the JAS RDDP are as follows:

1) The optional capability of sending urgent messages is removed because it can be realized with other methods.

2) The restriction on the size of the sliding window that it shall be a power of two was removed in order to enhance the flexibility of the protocol.

3) The Open/Reset Command was renamed the Open Command in order to simplify the terminology.

4) Ack Packets for Control Packets have a different Packet Type than those for Data Packets in order to differentiate Ack Packets acknowledging Control Packets from those acknowledging Data Packets with Sequence Number 0.

5) A Close Command can only be sent when an Open Command has been acknowledged because the Close Command is a mechanism to close an Open Transport Channel.

6) The optional mechanism of flow control was added in order to enhance the capabilities of the protocol. To realize them, Packet Types called “Flow Control Packet” and “Flow Control Ack Packet” were added.

7) The optional mechanism of “heartbeat” was added in order to enhance the capabilities of the protocol.

To realize them, Packet Types called “Heartbeat Packet” and “Heartbeat Ack Packet” were added.

8) If the Receive TEP has found that the Transport Channel is inactive, it can close the Transport Channel.

If the features added to SpaceWire-R, which are described above, are not used, SpaceWire-R is compatible with JAS RDDP except for one thing: the Packet Type for Ack Packets that acknowledge Control Packets must be changed (see Table 4-2).

The style of the document has also been changed from those of [R1] and [R2], and it is now compatible with the style that the ISO and CCSDS protocol specifications are written with.

3. SERVICE DEFINITION

3.1. GENERAL

This section provides service definition in the form of primitives, which present an abstract model of the logical exchange of data and control information between the protocol entity of SpaceWire-R and applications that use SpaceWire-R. The definitions of primitives are independent of specific implementation approaches.

The parameters of the primitives are specified in an abstract sense and specify the information to be made available to applications. The way in which a specific implementation makes this information available is not constrained by this specification. In addition to the parameters specified in this section, an implementation may provide other parameters to applications (e.g., parameters for controlling the service, monitoring performance, facilitating diagnosis, and so on).

3.2. CHANNEL CONTROL SERVICE

3.2.1. OVERVIEW OF CHANNEL CONTROL SERVICE

The Channel Control Service provides applications with the capability of controlling the state machine of the Transmit TEPs and the Receive TEPs of Transport Channels (see 4.4.3 and 4.5.2).

3.2.2. CHANNEL CONTROL SERVICE PARAMETERS

3.2.2.1. Channel Number

The parameter Channel Number (see 4.2.3.6) specifies the Transport Channel whose state machine is to be controlled.

3.2.2.2. Directive Type

The parameter Directive Type specifies the type of Directive that the application delivers to the Transmit or Receive TEP with the ChannelControl.request primitive. The values taken by this parameter are specified in Table 3-1.

Table 3-1: Values for the Directive Type

Directive Type	Description	Note
Open Transport Channel	Direct the Transmit or Receive TEP to open the Transport Channel.	Used by the application at the sending and receiving ends of the Transport Channel.
Close Transport Channel	Direct the Transmit TEP to close the Transport Channel.	Used only by the application at the sending end of the Transport Channel.

3.2.2.3. Notification Type

The parameter Notification Type describes the event being notified by the Transmit or Receive TEP to the application with the ChannelControl.indication primitive. The values taken by this parameter are defined in Table 3-2.

Table 3-2: Values for the Notification Type

Notification Type	Description
Enabled	The Transmit or Receive TEP has transitioned to the ENABLED state.

Open	The Transmit or Receive TEP has transitioned to the OPEN state.
Closing	The Transmit or Receive TEP has transitioned to the CLOSING state.
Closed	The Transmit or Receive TEP has transitioned to the CLOSED state.

3.2.3. CHANNEL CONTROL SERVICE PRIMITIVES

3.2.3.1. General

The service primitives associated with this service are:

1) ChannelControl.request
2) ChannelControl.indication

The ChannelControl.request primitive is passed from the application to the Transmit or Receive TEP to request that the state of the Transmit or Receive TEP be transitioned.

The ChannelControl.indication primitive is passed from the Transmit or Receive TEP to the application to notify that the state of the Transmit or Receive TEP has transitioned.

3.2.3.2. ChannelControl.request

The ChannelControl.request primitive shall be used by the application for requesting the Transmit or Receive TEP to change the state of the Transmit or Receive TEP of the specified Transport Channel (see 4.4.3 and 4.5.2).

The ChannelControl.request primitive shall provide parameters as follows:

ChannelControl.request (Channel Number,
Directive Type)

3.2.3.3. ChannelControl.indication

The ChannelControl.indication primitive shall be used by the Transmit or Receive TEP for notifying the application that a state transition has occurred.

The ChannelControl.indication primitive shall provide parameters as follows:

ChannelControl.indication (Channel Number,
Notification Type)

3.3. DATA TRANSFER SERVICE

3.3.1. OVERVIEW OF DATA TRANSFER SERVICE

The Data Transfer Service provides applications with transfer of a sequence of service data units of variable length in a Transport Channel across a SpaceWire network. Transfer of service data units is unidirectional (one way) and asynchronous.

3.3.2. DATA TRANSFER SERVICE PARAMETERS

3.3.2.1. Service Data Unit

A Service Data Unit (SDU) is a variable-length, octet-aligned data unit, the format of which is unknown to the protocol entity. A Service Data Unit is provided by the sending application for a Transmit TEP, and delivered to the receiving application by the Receive TEP. A Service Data Unit is transmitted from the Transmit TEP to the

Receive TEP in the Application Data field (see 4.2.4.3) of a SpW-R Data Packet (or a series of SpW-R Data Packets).

The maximum length of Service Data Units shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

3.3.2.2. SDU ID

The Service Data Unit Identifier (SDU ID) is a sequence number provided by the sending application in the DataTransfer.request primitive. This number is used in subsequent DataTransferNotify.indication primitives to identify the Service Data Unit associated with them.

3.3.2.3. Channel Number

The parameter Channel Number (see 4.2.3.6) specifies the Transport Channel in which the Service Data Unit provided by the sending application is to be transmitted.

3.3.2.4. Notification Type

The parameter Notification Type describes the event being notified by the Transmit TEP to the sending application with the DataTransferNotify.indication primitive. The values taken by this parameter are defined in Table 3-3.

Table 3-3: Values for the Notification Type

Notification Type	Description
Accept Transfer	The Transmit TEP has accepted transfer of the Service Data Unit provided by the application.
Reject Transfer	The Transmit TEP has rejected transfer of the Service Data Unit provided by the application.
Transfer Confirmed	Transfer of the Service Data Unit has been confirmed by the Receive TEP.
Transfer Failure	Transfer of the Service Data Unit has not been confirmed by the Receive TEP.

3.3.2.5. Reason

The parameter Reason describes the reason for the event being notified by the Transmit TEP to the sending application with the DataTransferNotify.indication primitive when the Notification Type is “Reject Transfer” (see 3.3.2.4). The values taken by this parameter are defined in Table 3-4. When the Notification Type is not “Reject Transfer”, the value for this parameter is not defined.

Table 3-4: Values for the Reason

Reason	Description
SDU too long	The Service Data Unit exceeds the predetermined maximum length of Service Data Units (see 4.4.2).
Channel Not Open	The Transport Channel is not OPEN (see 4.4.3).

3.3.3. DATA TRANSFER SERVICE PRIMITIVES

3.3.3.1. General

The service primitives associated with this service are:

1) DataTransfer.request

2) DataTransferNotify.indication

3) DataTransfer.indication

The DataTransfer.request primitive is passed from the sending application to the Transmit TEP to request that a Service Data Unit be transferred to the receiving application through the specified Transport Channel.

The DataTransferNotify.indication primitive is passed from the Transmit TEP to the sending application to notify the application of an event associated with the transfer of a Service Data Unit.

The DataTransfer.indication primitive is passed from the Receive TEP to the receiving application to deliver a received Service Data Unit.

3.3.3.2. DataTransfer.request

The DataTransfer.request primitive shall be used by the sending application for requesting the Transmit TEP to send the Service Data Unit in the specified Transport Channel.

The DataTransfer.request primitive shall provide parameters as follows:

DataTransfer.request	(Service Data Unit, SDU ID, Channel Number)

3.3.3.3. DataTransferNotify.indication

The DataTransferNotify.indication primitive shall be used by the Transmit TEP for notifying the sending application of an event associated with the transfer of the Service Data Unit identified by the SDU ID.

The DataTransferNotify.indication primitive shall provide parameters as follows:

DataTransferNotify.indication	(SDU ID, Channel Number, Notification Type, Reason)

3.3.3.4. DataTransfer.indication

The DataTransfer.indication primitive shall be used by the Receive TEP for delivering a Service Data Unit received in the Transport Channel to the receiving application.

The DataTransfer.indication primitive shall provide parameters as follows:

DataTransfer.indication	(Service Data Unit, Transport Channel)

4. PROTOCOL SPECIFICATION

4.1. GENERAL

This section specifies SpaceWire-R (SpW-R) in terms of protocol data units exchanged between peer protocol entities (4.2) and procedures to be performed by the protocol entities (4.3, 4.4, 4.5, and 4.6).

4.2. SPW-R PACKETS

The protocol data unit (PDU) of SpaceWire-R is called the SpW-R Packet.

4.2.1. SPW-R PACKET TYPES

There are five types of SpW-R Packets:

a) Data Packets;

b) Control Packets;

c) Ack Packets;

d) Flow Control Packets; and

e) Heartbeat Packets.

4.2.1.1. Data Packets

Data Packets shall be used by the Transmit TEP to send Service Data Units to the Receive TEP of a Transport Channel.

4.2.1.2. Control Packets

Control Packets shall be used by the Transmit TEP to send Control Commands to the Receive TEP to control the state machine of the Receive TEP (see 4.5.2).

There are two Control Commands:

a) Open Command: Open Commands shall be used to change the state of the Receive TEP from the ENABLED state to the OPEN state; and

b) Close Command: Close Commands shall be used to change the state of the Receive TEP from either the ENABLED or OPEN state to the CLOSING state.

4.2.1.3. Ack Packets

Ack Packets shall be used by the Receive TEP to acknowledge receipt of Data, Control, and Heartbeat Packets to the Transmit TEP of a Transport Channel. Ack Packets shall also be used by the Transmit TEP to acknowledge receipt of Flow Control and Heartbeat Packets to the Receive TEP of a Transport Channel.

There are four kinds of Ack Packets:

a) Data Ack Packets: Ack Packets that acknowledge receipt of Data Packets;

b) Control Ack Packets: Ack Packets that acknowledge receipt of Control Packets;

c) Flow Control Ack Packets: Ack Packets that acknowledge receipt of Flow Control Packets; and

d) Heartbeat Ack Packets: Ack Packets that acknowledge receipt of Heartbeat Packets.

If flow control is used, Data Ack and Control Ack Packets shall also be used by the Receive TEP to inform the Transmit TEP of the maximum value of the Sequence Number it can accept (Maximum Acceptable Sequence Number or MASN) (see 4.5.3.5).

4.2.1.4. Flow Control Packets

If flow control is used, Flow Control Packets shall be used by the Receive TEP to inform the Transmit TEP of the maximum value of the Sequence Number it can accept (Maximum Acceptable Sequence Number, or MASN).

[Note] MASN can also be delivered to the Transmit TEP using Data Ack and Control Ack Packets (see 4.2.1.3).

4.2.1.5. Heartbeat Packets

Heartbeat Packets shall be used by the Transmit TEP and/or the Receive TEP to solicit a response from the Receive TEP and/or the Transmit TEP, respectively, of a Transport Channel to make sure that the link and the TEP at the other end are alive.

4.2.2. SPW-R PACKET STRUCTURE

4.2.2.1. SpW Packet Structure

An SpW-R Packet shall be sent as the cargo of a SpaceWire packet (see 9.2.1 of [A1]).

As specified in 9.2.1 and 9.2.2 of [A1], a list of zero or more destination identifiers shall precede an SpW-R Packet.

As specified in 9.2.1 and 9.2.3 of [A1], an end-of-packet marker shall follow an SpW-R Packet.

The structure of an SpW packet that carries an SpW-R Packet is shown in Figure 4-1.

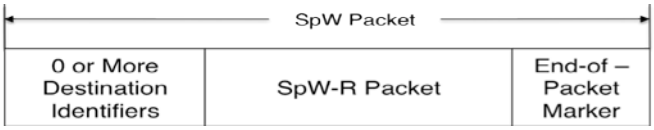

Figure 4-1: Structure of an SpW Packet Containing an SpW-R Packet

4.2.2.2. SpW-R Packet Structure

An SpW-R Packet shall consist of the following fields, positioned contiguously, in the following sequence:

a) Header (10+N octets);

b) Payload (X octets); and

c) Trailer (2 octets).

The structure of an SpW-R Packet is shown in Figure 4-2.

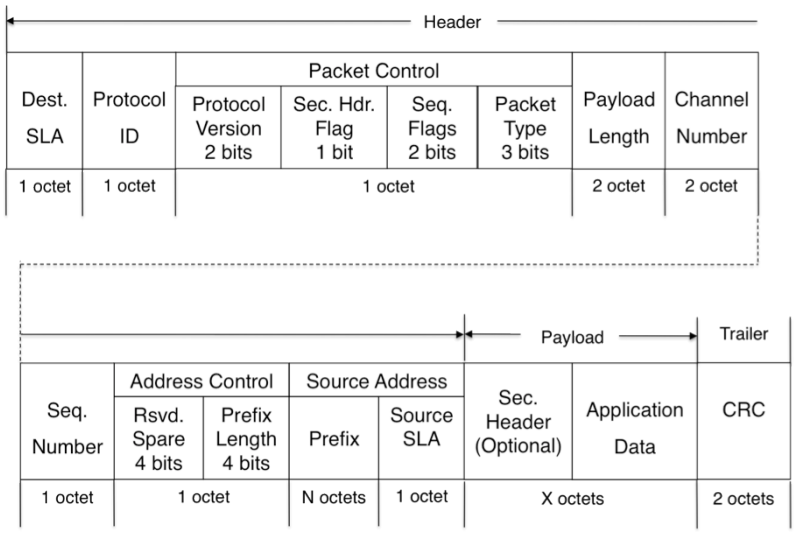

Figure 4-2: Structure of an SpW-R Packet

4.2.3. SPW-R PACKET HEADER

4.2.3.1. General

The Header of an SpW-R Packet shall consist of the following fields, positioned contiguously, in the following sequence:

a) Destination SLA (1 octet);

b) Protocol ID (1 octet);

c) Packet Control (1 octet);

d) Payload Length (2 octets);

e) Channel Number (2 octets);

f) Sequence Number (1 octet);

g) Address Control (1 octet); and

h) Source Address (N+1 octets).

For all the Packet types specified in 4.2.1, the same Header shall be used.

4.2.3.2. Destination SLA

As specified in 5.2.1 of [A4], the first octet (octet 0) of the Header shall contain the SpaceWire Logical Address (SLA) that identifies the node to which the Packet is being sent.

The values of SLA shall be assigned according to the rules on logical addresses specified in [A1].

4.2.3.3. Protocol ID

As specified in 5.2.2 of [A4], the second octet (octet 1) of the Header shall contain the Protocol ID of this protocol, which is to be assigned by the European Cooperation for Space Standardization (ECSS) at a later time.

4.2.3.4. Packet Control

4.2.3.4.1. General

Octet 2 of the Header shall contain the Packet Control field.

The Packet Control field shall consist of the following sub-fields, positioned contiguously, in the following sequence:

a) Protocol Version Number (2 bits);

b) Secondary Flag Header (1 bit);

c) Sequence Flags (2 bits); and

d) Packet Type (3 bits).

4.2.3.4.2. Protocol Version Number

Bits 0-1 of the Packet Control field shall contain the Protocol Version Number of the SpW-R protocol specification with which the format of the Packet complies. These bits shall be set to “01” for this version of the specification.

4.2.3.4.3. Secondary Header Flag

Bit 2 of the Packet Control field shall contain the Secondary Header Flag. Value “1” of this flag shall indicate the presence of a variable length secondary header in the Payload field of the Packet. Value “0” shall indicate the absence of a secondary header.

4.2.3.4.4. Sequence Flags

Bits 3-4 of the Packet Control field shall contain the Sequence Flags, which shall be used in a Data Packet to signal whether this Data Packet contains the first, intermediate, or final segment generated from a single Service Data Unit provided by the application.

Values for the Sequence Flags are shown in Table 4-1.

Table 4-1: Values for the Sequence Flags

Bit 3	Bit 4	Interpretation
0	1	Containing the first segment of a Service Data Unit
0	0	Containing a continuation segment of a Service Data Unit
1	0	Containing the last segment of a Service Data Unit
1	1	Containing one complete Service Data Unit (Stand-alone Packet)

The other types of Packets than Data Packets shall have their Sequence Flags set to “11” indicating that they are stand-alone Packets.

4.2.3.4.5. Packet Type

Bits 5-7 of the Packet Control field shall contain the Packet Type, which identifies the type of this Packet.

Values for the Packet Type are shown in Table 4-2.

Table 4-2: Values for the Packet Type

Value	Packet Type
0	Data Packet
1	Data Ack Packet
2	Control Packet (Open Command)
3	Control Packet (Close Command)
4	Heartbeat Packet
5	Heartbeat Ack Packet
6	Flow Control Packet and Flow Control Ack Packet (see Note)
7	Control Ack Packet

[Note] Packets of Packet Type 6 that are sent from the Receive TEP to the Transmit TEP of a Transport Channel are interpreted as Flow Control Packets, and Packets of Packet Type 6 that are sent from the Transmit TEP to the Receive TEP are interpreted as Flow Control Ack Packets.

4.2.3.5. Payload Length

Octets 3-4 of the Header shall contain the Payload Length, which specifies the number of octets in the Payload field. It shall not include any octets in the Header or Trailer but does include octets used in the Secondary Header if used (see 4.2.4.2).

4.2.3.6. Channel Number

Octets 5-6 of the Header shall contain the Channel Number, which specifies the Transport Channel to which this Packet belongs.

Channel Numbers shall be uniquely assigned to Transport Channels in a SpaceWire network.

4.2.3.7. Sequence Number

Octet 7 of the Header shall contain the Sequence Number, which identifies this Packet in a series of Packets sent in a Transport Channel modulo 256.

For Data and Heartbeat Packets, the Sequence Numbers shall be assigned with the procedure specified in sections 4.4.4.4 and 4.5.3.7.

For Control Packets, the value of the Sequence Number shall be 0.

For Ack Packets, the Sequence Numbers shall be assigned with the procedure specified in sections 4.5.3.4, 4.4.4.6, and 4.4.4.7.

For Flow Control Packets, the Sequence Numbers shall be assigned with the procedure specified in section 4.5.3.5.

4.2.3.8. Address Control

4.2.3.8.1. General

Octet 8 of the Header shall contain the Address Control field.

The Address Control field shall consist of the following sub-fields, positioned contiguously, in the following sequence:

a) Reserved Spare (4 bits); and

b) Prefix Length (4 bits).

4.2.3.8.2. Reserved Space

Bits 0-3 of the Address Control field shall contain the Reserved Spare, which is reserved for future use and shall be set to “0000”.

4.2.3.8.3. Prefix Length

Bits 4-7 of the Address Control field shall contain the Prefix Length (*N*), which specifies the number of octets contained in the Prefix (see 4.2.3.9.2) to be prepended to the Source SLA (see 4.2.3.9.3) to form a full variable-length source address.

If only logical addressing is used in a SpaceWire network, the Prefix Length (*N*) shall always be set to zero in that SpaceWire network.

4.2.3.9. Source Address

4.2.3.9.1. General

The *N*+1 Octets beginning with octet 9 of the Header shall contain the Source Address field.

The Source Address of the Packet, which shall identify the node from which this Packet is sent, shall consist of a variable-length Prefix (which may be absent) and the Source SpaceWire Logical Address (SLA).

The Source Address field shall consist of the following sub-fields, positioned contiguously, in the following sequence:

a) Prefix (*N* octets); and

b) Source SLA (1 octet).

The number of octets (*N*) contained within the Prefix field shall be specified in the Prefix Length field of the Address Control field (see 4.2.3.8.3). If *N* is equal to zero, the Prefix field shall be omitted. *N* shall not be greater than 15.

The contents of the Source Address field shall be assigned according to the rules addresses specified in [A1]. If only logical addressing is used in a SpaceWire network, the Prefix field shall always be omitted in that SpaceWire network.

[Note] SpW-R uses a variable-length source address to accommodate those networks that may require multiple address octets to uniquely identify an end point, such as in SpaceWire path addressing or regional addressing.

4.2.3.9.2. Prefix

If *N* is greater than zero, the first *N* octets of the Source Address field shall contain all the variable-length addressing octets except for the SLA.

4.2.3.9.3. Source SLA

The last octet of the Source Address field shall contain the Source SpaceWire Logical Address (SLA).

4.2.4. SPW-R PACKET PAYLOAD

4.2.4.1. General

The contents of the Payload field shall be as follows for each of the Packet types specified in 4.2.1.

a) For Data Packets, the Payload field shall consist of an optional Secondary Header (if it is used) and Application Data, positioned contiguously, in this sequence.

b) For Control Packets, Heartbeat Packets, Data Ack Packets (if flow control is not used), Control Ack Packets (if flow control is not used), Flow Control Ack Packets, and Heartbeat Ack Packets, the Payload field shall consist of only an optional Secondary Header (if it is used) or be nonexistent (if an optional Secondary Header is not used).

c) For Data Ack Packets (if flow control is used), Control Ack Packets (if flow control is used), and Flow Control Packets, the Payload field shall consist of an optional Secondary Header (if it is used) and the Maximum Acceptable Sequence Number (MASN) (see 4.5.3.5). The length of the MASN shall be one octet.

4.2.4.2. Secondary Header

The Secondary Header, if present, shall be the first octets of the Payload.

The contents, length, and format of the Secondary Header shall be determined by each project and documented in a project-specific document.

The presence of the Secondary Header is indicated by the Secondary Header Flag in the Packet Control field (see 4.2.3.4.3).

[Note] The Secondary Header can be used either for exchanging project-specific management information between the Transmit and Receive TEPs or for sending project-specific special data provided by the application.

4.2.4.3. Application Data

The Application Data filed shall follow the Secondary Header if it is present or the Header if the Secondary Header is absence, in Data Packets.

The Application Data filed shall contain a segment of a Service Data Unit or a complete Service Data Unit provided by the application.

The maximum length of the Application Data field shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

4.2.5. SPW-R PACKET TRAILER

The Trailer shall contain a 16-bit CCITT CRC.

The CRC shall be computed from the Destination SLA to the last octet of the Payload contained in the Packet.

The CRC bits shall be computed according to the following polynomial:

$$x^{16} + x^{12} + x^5 + 1$$

The initial value for the computation shall be set to all 1s before computing the CRC of each Packet.

4.3. STRUCTURE OF PROCEDURES

The procedures performed by protocol entities are classified into the following three categories:

a) Procedures at the Transmit TEP;

b) Procedures at the Receive TEP; and

c) Common Procedures at a Node.

Figure 4-3 shows the structure of procedures used to implement SpaceWire-R at a SpaceWire node.

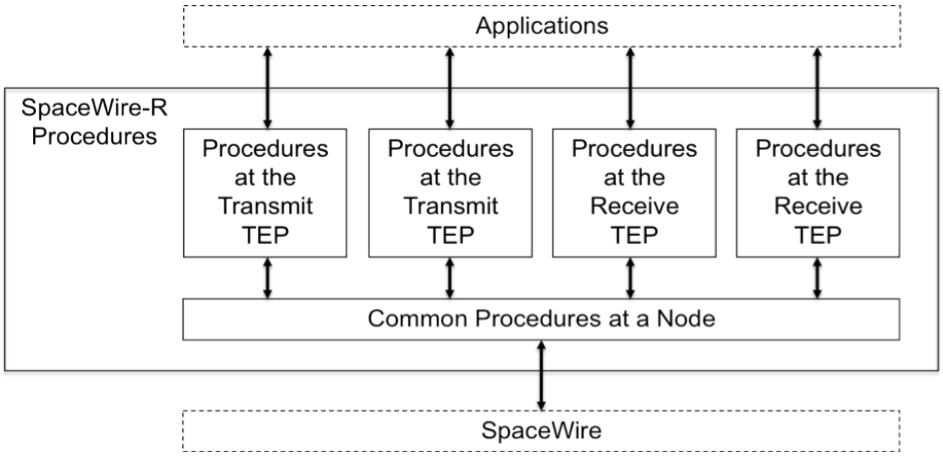

Figure 4-3: Structure of Procedures at a Node

There shall be an instance of the set of Procedures at the Transmit TEP (specified in 4.4) for each of the Transmit TEPs that exists on the node. There shall be an instance of the set of Procedures at the Receive TEP (specified in 4.5) for each of the Receive TEPs that exists on the node. There shall be one instance of the set of Common Procedures at a Node (specified in 4.6) at each node.

4.4. PROCEDURES AT THE TRANSMIT TEP

4.4.1. GENERAL

This section specifies procedures to be performed at the Transmit TEP.

[Note] The procedures specified in this subsection are presented in an abstract sense and are not intended to imply any particular implementation approach of a protocol entity.

The Transmit TEP shall perform the following three procedures:

a) Segmentation;

b) Transmission Control; and

c) Channel Control.

These procedures and the logical relationship among them are shown in Figure 4-4.

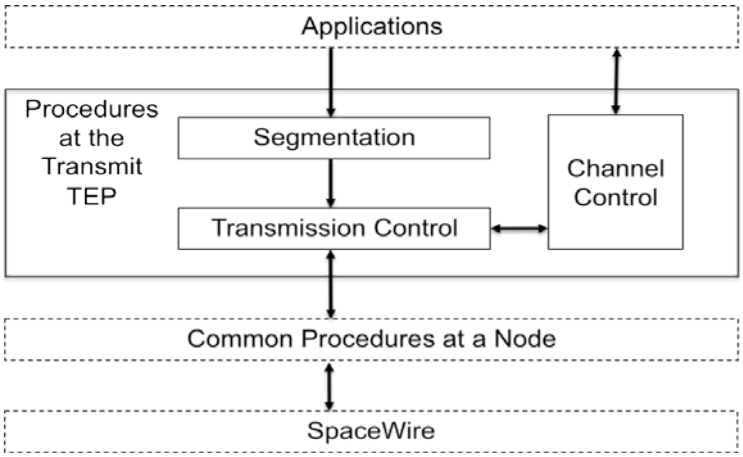

Figure 4-4: Procedures at the Transmit TEP

In what follows, Flow Control Packets shall only be processed if flow control is used. Heartbeat Packets shall only be processed if heartbeat is used.

4.4.2. SEGMENTATION

This procedure shall segment Service Data Units provided by the sending application with the TransferData.request primitive for transmission to the receiving application (see 3.3.3.2).

There shall be one instance of this procedure for each Transport Channel.

If a Service Data Unit provided by the sending application exceeds the predetermined maximum length of Service Data Units (see 3.3.2.1), the Transmit TEP shall reject the transmission of the Service Data Unit and notify the sending application using the TransferDataNotify primitive with the Notification Type “Reject Transfer” and the Reason “SDU too long” (see 3.3.3.3). If the Service Data Unit does not exceed the predetermined maximum length of Service Data Units, the Transmit TEP shall accept the transmission of the Service Data Unit, notify the sending application using the TransferDataNotify primitive with the Notification Type “Accept Transfer” (see 3.3.3.3), and apply the following procedure.

If a Service Data Unit provided by the sending application exceeds the predetermined maximum length of the Application Data field (see 4.2.4.3) of SpW-R Data Packets, this procedure shall divide it into segments that are compatible with insertion into the Application Data field. Then, the procedure shall insert the segments into the Application Data field of a series of SpW-R Data Packets, setting appropriate values to the Sequence Flags (see 4.2.3.4.4) of each SpW-R Data Packet.

The SpW-R Data Packets containing segments generated from a single Service Data Unit shall be transmitted consecutively without being interlaced with any other SpW-R Data Packet carrying (a segment of) another Service Data Unit.

If the Service Data Unit does not exceed the predetermined maximum length of the Application Data field of SpW-R Packets, this procedure shall insert the Service Data Unit into the Application Data field of an SpW-R Data Packet, setting appropriate values to the Sequence Flags.

4.4.3. CHANNEL CONTROL

4.4.3.1. General

This procedure shall control the state machine of the Transmit TEP.

There shall be one instance of this procedure for each Transport Channel.

Each Transmit TEP shall have the following four states.

a) CLOSED;

b) ENABLED;

c) OPEN; and

d) CLOSING.

The occurrence of any state transition shall be reported to the application with the ChannelControl.indication primitive (see 3.2.3.3).

A summary of the state machine is shown in Figure 4-5.

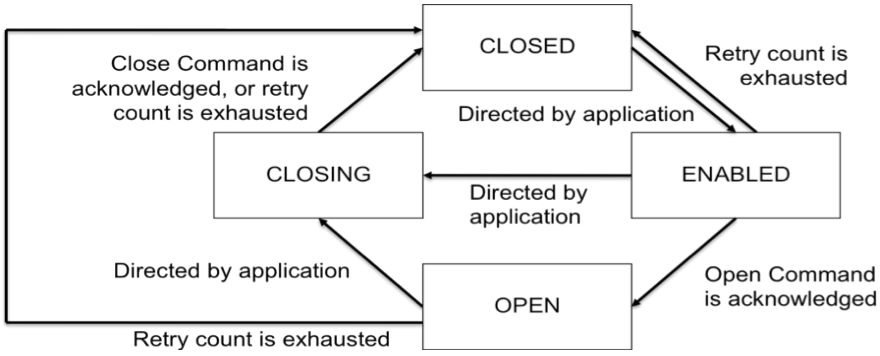

Figure 4-5: State Machine at the Transmit TEP

4.4.3.2. CLOSED

Upon initialization, the Transmit TEP shall be in the CLOSED state.

In this state, the Transmit TEP shall not send any Packets and shall not respond to any received Packets.

Upon reception of a direction from the application using the ChannelControl.request primitive with the Directive Type “Open” (see 3.2.3.2), the Transmit TEP shall transition to the ENABLED state. If a direction from the application using the ChannelControl.request primitive with another Directive Type than “Open” is received, it shall be discarded.

If the Transmit TEP has received a request to transmit a Service Data Unit from the sending application in this state with the TransferData.request primitive (see3.3.3.2), it shall reject it and notify the sending application using the TransferData.indication primitive with the Notification Type “Reject Transfer” and the Reason “Channel Not Open” (see 3.3.3.3).

4.4.3.3. ENABLED

Upon transitioning into this state, the Transmit TEP shall send an Open Command to the Receive TEP and wait for a Control Ack Packet for acknowledging the sent Open Command using the Transmission Control procedure (see 4.4.4). When the Open Command is acknowledged, the Transmit TEP shall transition to the OPEN state. When the retry count for the Open Command is exhausted (see 4.4.4.5), the Transmit TEP shall transition to the CLOSED state.

In this state, the Transmit TEP shall not send any other Packets and shall not respond to any other received Packets.

If the Transmit TEP has received a request to transmit a Service Data Unit from the sending application in this

state with the TransferData.request primitive (see 3.3.3.2), it shall reject it and notify the sending application using the TransferData.indication primitive with the Notification Type “Reject Transfer” and the Reason “Channel Not Open” (see 3.3.3.3).

Upon reception of a direction from the application using the ChannelControl.request primitive with the Directive Type “Close” (see 3.2.3.2), the Transmit TEP shall wait until the sent Open Command has been acknowledged and then transition to the CLOSING state. If a direction from the application using the ChannelControl.request primitive with another Directive Type than “Close” is received, it shall be discarded.

4.4.3.4. OPEN

In this state, the Transmit TEP shall send Data Packets to the Receive TEP when there are Data Packets generated by the Segmentation procedure (see 4.4.2), and wait for the corresponding Data Ack Packets using the Transmission Control procedure (see 4.4.4). It shall also optionally perform flow control and heartbeat as specified in the Transmission Control procedure (see 4.4.4).

When the retry count for a Data or Heartbeat Packet is exhausted (see 4.4.4.5), the Transmit TEP shall transition to the CLOSED state.

Upon reception of a direction from the application using the ChannelControl.request primitive with the Directive Type “Close” (see 3.2.3.2), the Transmit TEP shall transition to the CLOSING state. If a direction from the application using the ChannelControl.request primitive with another Directive Type than “Close” is received, it shall be discarded.

4.4.3.5. CLOSING

Upon transitioning into this state, the Transmit TEP shall send a Close Command to the Receive TEP and wait for a Control Ack Packet for acknowledging the sent Close Command using the Transmission Control procedure (see 4.4.4). When the Close Command is acknowledged, the Transmit TEP shall transition to the CLOSED state. When the retry count for the Close Command is exhausted (see 4.4.4.5), the Transmit TEP shall transition to the CLOSED state.

In this state, the Transmit TEP shall not send any other Packets and shall not respond to any other received Packets.

If the Transmit TEP has received a request to transmit a Service Data Unit from the sending application in this state with the TransferData.request primitive (see3.3.3.2), it shall reject it and notify the sending application using the TransferData.indication primitive with the Notification Type “Reject Transfer” and the Reason “Channel Not Open” (see 3.3.3.3).

If a direction from the application using the ChannelControl.request primitive is received, it shall be discarded.

4.4.4. TRANSMISSION CONTROL

4.4.4.1. General

This procedure shall be used to send Data, Control, and (optional) Heartbeat Packets to the Receive TEP, and receive Ack, (optional) Flow Control, and (optional) Heartbeat Packets from the Receive TEP. For controlling the transmission of Data Packets, a sliding window (see 4.4.4.2) shall be used. It shall perform retransmission of Data, Control, and Heartbeat Packets if no Ack Packet has been received for Packets. It shall also optionally perform flow control, heartbeat and redundancy control.

There shall be one instance of this procedure for each Transport Channel.

4.4.4.2. Packet Validation Check

The Transmit TEP shall examine all received Packets against a CRC error, and Packets with a CRC error shall be discarded.

The Transmit TEP shall examine all the fields of the Header (see Figure 4-2) of all received Packets, and Packets with an unexpected value in any field of the Header shall be discarded.

4.4.4.3. Sliding Window

The sliding window shall be a range of *k* consecutive Sequence Numbers (modulo 256). Data Packets with Sequence Numbers within the sliding window can be "outstanding" (that is, they have been transmitted but not yet been acknowledged).

The size of the sliding window (*k*) shall be less than or equal to 128. It shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

Upon transitioning to the OPEN state (see 4.4.3.4), the sliding window shall be set to the range from 1 to *k* inclusive.

When the sliding window is from *n* to *n*+*k*-1, it is advanced by 1 (that is, it becomes from *n*+1 to *n*+*k*) upon reception of the acknowledgement for the Data Packet with Sequence Number *n*. The acknowledgements for the Data Packets with consecutive Sequence Numbers *n*+1, …, *n*+*p* have already been received when the acknowledgement for the Data Packet with Sequence Number *n* is received, the sliding window shall become *n*+*p*+1 to *n*+*p*+*k*.

4.4.4.4. Packet Transmission

When a Service Data Unit is to be transmitted, it or its segments (see 4.4.2) shall be transmitted in Data Packets.

The first Data Packet after transitioning to the OPEN state shall have Sequence Number 1.

When the Data Packet previously transmitted had Sequence Number *m*, the next Data Packet to be transmitted shall have Sequence Number *m*+1. If *m*+1 is within the current sliding window, it shall be transmitted immediately. If not, its transmission shall be deferred until *m*+1 is included in the sliding window.

When an Open or Close Command is to be transmitted, it shall be transmitted immediately in a Control Packet with Sequence Number 0. After a Control Packet has been transmitted, no other Packets shall be transmitted and any incoming Packets (except for the Ack Packet that acknowledges the sent Control Packet) shall be discarded until the Control Packet is acknowledged.

When a Heartbeat Packet is to be transmitted (see 4.4.4.7), it shall be transmitted immediately. ~~with T~~the Sequence Number of ~~the last Data Packet that has been transmitted~~any Heartbeat Packet shall be set to 0.

~~4.4.4.6.~~4.4.4.5. Packet Retransmission

When– a Data, Control, or Heartbeat Packet is transmitted, a Transmit timer shall be started. It shall be started at the time when the last octet of the Packet is transmitted.

An independent Transmit timer shall be maintained for each outstanding Packet.

The initial value for the Transmit timers shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

When an Ack Packet acknowledging an outstanding Packet is received, the Transmit timer for that Packet shall

be cancelled.

When all of the Data Packets associated with a Service Data Unit have been acknowledged, the Transmit TEP shall notify the sending application of the success of the transfer of the Service Data Unit associated with the Packets using the DataTransferNotify.indication primitive with the Notification Type “Transfer Confirmed” (see 3.3.3.3).

If no Ack Packet has been received when the Transmit timer for a Packet expires, the Packet shall be retransmitted with the original Sequence Number until the maximum retry count is reached. The Transmit timer for that Packet shall be restarted.

The maximum retry count shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

When the maximum retry count has been exceeded for any Packet, the Transport Channel shall be declared to be “inactive,” and the Transmit TEP shall transition to the CLOSED state. If the failed Packet is a Data Packet, the Transmit TEP shall notify the sending application of the failure of the transfer of the Service Data Unit associated with the Packet using the DataTransferNotify.indication primitive with the Notification Type “Transfer Failure” (see 3.3.3.3). The Transmit TEP shall also notify the sending application that the Transport Channel has been closed using the ChannelControl.indication primitive (see 3.2.3.3).

If an Ack Packet has been received for a non-outstanding Packet (a Packet for which no Transmit timer is running), it shall be discarded.

~~4.4.4.7.~~4.4.4.6. Flow Control

Whether or not flow control is used shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

If flow control is used, the third paragraph in 4.4.4.4 shall be modified as follows.

When the Data Packet previously transmitted had Sequence Number *m*, the next Data Packet to be transmitted shall have Sequence Number *m*+1. If *m*+1 is less than or equal to the Maximum Acceptable Sequence Number (MASN) contained in the most recent Ack or Flow Control Packet, it shall be transmitted immediately. If not, its transmission shall be deferred until *m*+1 becomes less than or equal to the MASN.

When a Data Ack or Flow Control Packet is received, the Maximum Acceptable Sequence Number (MASN) contained in it shall be recorded.

When a Flow Control Packet is received, the Transmit TEP shall send a Flow Control Ack Packet having the same Sequence Number as that of the received Flow Control Packet. A Flow Control Ack Packet shall not be sent if the received Flow Control Packet has an error (see 4.4.4.2).

~~4.4.4.8.~~4.4.4.7. Heartbeat

Heartbeat can be used either (1) for the Transmit TEP to check whether the link and the Receive TEP are still alive when there is no data to send (Transmit Heartbeat), and/or (2) for the Receive TEP to check whether the link and the Transmit TEP are still alive when there is no data to receive (Receive Heartbeat). It is possible to use both Transmit Heartbeat and Receive Heartbeat simultaneously on a Transport Channel. Whether or not each of Transmit Heartbeat and Receive Heartbeat is used shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

(1) Transmit Heartbeat

If Transmit Heartbeat is used, a Transmit Heartbeat timer shall be used by the Transmit TEP for each Transport Channel as follows.

The initial value for the Transmit Heartbeat timer shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

Upon transitioning into the OPEN state, the Transmit Heartbeat timer shall be started. Whenever a Packet (either a Control, Data, or Ack Packet) is transmitted by the Transmit TEP, the Transmit Heartbeat timer shall be restarted.

When the Heartbeat Transmit timer expires, a Heartbeat Packet shall be transmitted using the Packet Transmission Procedure (see 4.4.4.4), and the Transmit Heartbeat timer shall be restarted.

(2) Receive Heartbeat

If Receive Heartbeat is used, upon reception of a Heartbeat Packet, the Transmit TEP shall send a Heartbeat Ack Packet having the same Sequence Number as that of the received Heartbeat Packet. A Heartbeat Ack Packet shall not be sent if the received Heartbeat Packet has an error (see 4.4.4.2).

~~**Redundancy Control**~~

~~Whether or not flow control is used shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.~~

~~If redundancy control is used, the following procedure shall apply.~~

~~When a Transport Channel has been declared to be "inactive" (see 4.4.4.5), an alternative Transport Channel using a different route (in the same SpaceWire network or in another SpaceWire network) to the receiving node (if it is registered) shall be opened and all the outstanding Data Packets shall be retransmitted using the new Transport Channel. The Transmit TEP shall notify the sending application that a new Transport Channel has been opened using the ChannelControl.indication primitive (see 3.2.3.3).~~

4.4.4.8. ~~For each Transport Channel, a set of alternative Transport Channels using different routes to the receiving node (if there are any) shall be pre-registered at the Transmit TEP.~~Generation of Secondary Header

If the Secondary Header (see 4.2.4.2) is present, its contents shall be generated with a procedure determined by each project, which shall be documented in a project-specific document

4.4.4.9. Redundancy Control Support

[Note] When a Transport Channel has been declared to be "inactive" (see 4.4.4.5), the Transmit TEP will notify the sending application that the Transport Channel has been closed using the ChannelControl.indication primitive (see 3.2.3.3). In such a case, the application may choose to open an alternative Transport Channel using a different route (in the same SpaceWire network or in another SpaceWire network) to the receiving node and retransmit all the outstanding Data Packets using the new Transport Channel.

[Note] There may be a case in which Data Packets were received correctly by the Receive TEP but their Ack Packets have been lost when a Transport Channel is declared to be "inactive". In such a case, the sending application is not informed of the fact that the Data Packets were actually received by the Receive TEP and may retransmit the same Data Packets using the new Transport Channel. The application shall be responsible for handling such unnecessary retransmissions.

4.5. PROCEDURES AT THE RECEIVE TEP

4.5.1. GENERAL

This section specifies procedures to be performed at the Receive TEP.

[Note] The procedures specified in this subsection are presented in an abstract sense and are not intended to imply any particular implementation approach of a protocol entity.

The Receive TEP shall perform the following three procedures:

a) Channel Control;

b) Reception Control; and

c) Reconstruction.

These procedures and the logical relationship among them are shown in Figure 4-6.

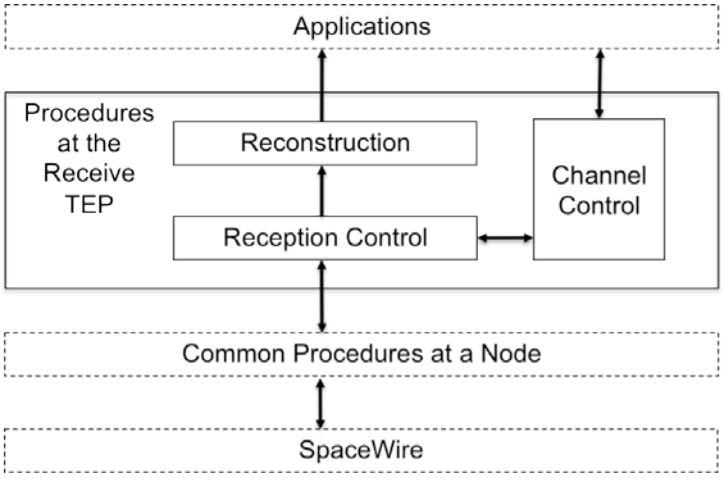

Figure 4-6: Procedures at the Receive TEP

In what follows, Flow Control Packets shall only be processed if flow control is used. Heartbeat Packets shall only be processed if heartbeat is used.

4.5.2. CHANNEL CONTROL

4.5.2.1. General

This procedure shall control the state machine of the Receive TEP.

There shall be one instance of this procedure for each Transport Channel.

Each Receive TEP shall have the following four states.

a) CLOSED;

b) ENABLED;

c) OPEN; and

d) CLOSING.

The occurrence of any state transition shall be reported to the application with the ChannelControl.indication primitive (see 3.2.3.3).

A summary of the state machine is shown in Figure 4-7.

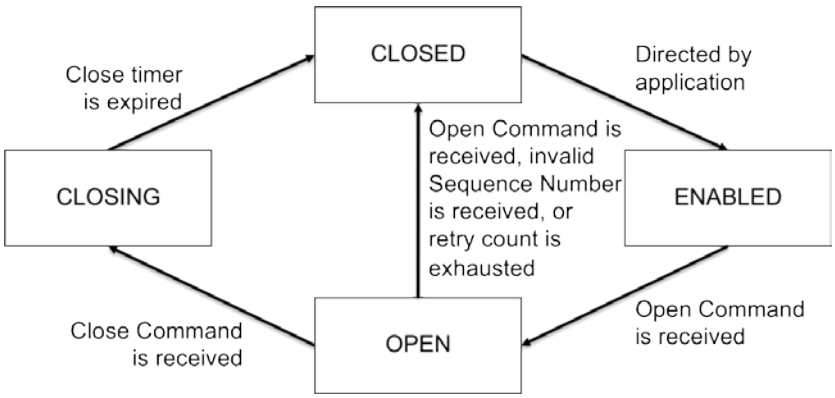

Figure 4-7: State Machine at the Receive TEP

4.5.2.2. CLOSED

Upon initialization, the Receive TEP shall be in the CLOSED state.

In this state, the Receive TEP shall not send any Packets and shall not respond to any received Packets.

Upon reception of a direction from the application using the ChannelControl.request primitive with the Directive Type “Open” (see 3.2.3.2), the Receive TEP shall transition to the ENABLED state. If a direction from the application using the ChannelControl.request primitive with another Directive Type than “Open” is received, it shall be discarded.

4.5.2.3. ENABLED

In this state, the Receive TEP shall wait for an Open Command to be received from the Transmit TEP. When it is received, the Receive TEP shall send back a Control Ack Packet for acknowledging the received Open Command using the Reception Control procedure (see 4.5.3.4) and transition to the OPEN state.

In this state, the Receive TEP shall not send any other Packets and shall not respond to any other received Packets.

If a direction from the application using the ChannelControl.request primitive is received, it shall be discarded.

4.5.2.4. OPEN

In this state, the Receive TEP shall wait for Data Packets to be received from the Transmit TEP and send back Data Ack Packets to the Transmit TEP using the Reception Control procedure (see 4.5.3.4). It shall also optionally perform flow control and heartbeat as specified in the Reception Control procedure (see 4.5.3.5 and 4.5.3.6, respectively).

If an Open Command is received before receiving any Data Packet, it shall be discarded but shall be acknowledged using the Reception Control procedure (see 4.5.3.4). If an Open Command is received after receiving one or more Data Packets, it shall be discarded, the Transport Channel shall be declared to be “inactive,” and the Receive TEP shall transition to the CLOSED state.

If a Data Packet with an invalid Sequence Number is received, the Receive TEP shall transition to the CLOSED state (see 4.5.3.4).

When the retry count for a Flow Control or Heartbeat Packet is exhausted, the Receive TEP shall transition to

the CLOSED state (see 4.5.3.7).

The Receive TEP shall also wait for a Close Command to be received from the Transmit TEP. When it is received, the Receive TEP shall send back a Control Ack Packet for acknowledging the received Close Command using the Reception Control procedure (see 4.5.3.4) and transition to the CLOSING state.

If a direction from the application using the ChannelControl.request primitive is received, it shall be discarded.

4.5.2.5. CLOSING

Upon transitioning into this state, the Receive TEP shall start a Close timer and wait for its expiration. When the Close timer expires, the Receive TEP shall transition into the CLOSED state.

The initial value for the Close timer shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

If a Close Command is received, it shall be discarded but shall be acknowledged using the Reception Control procedure (see 4.5.3.4).

The Receive TEP shall not send any other Packets and shall not respond to any other received Packets.

If a direction from the application using the ChannelControl.request primitive is received, it shall be discarded.

4.5.3. RECEPTION CONTROL

4.5.3.1. General

This procedure shall be used to receive Data, Control, and (optional) Heartbeat Packets from the Transmit TEP, and send Ack, (optional) Flow Control, and (optional) Heartbeat Packets to the Transmit TEP. For controlling the reception of Data Packets, a sliding window (see 4.5.3.3) shall be used. It shall also optionally perform flow control and heartbeat.

There shall be one instance of this procedure for each Transport Channel.

4.5.3.2. Packet Validation Check

The Receive TEP shall examine all received Packets against a CRC error, and Packets with a CRC error shall be discarded.

The Receive TEP shall examine all the fields of the Header (see Figure 4-2) of all received Packets, and Packets with an unexpected value in any field of the Header shall be discarded.

4.5.3.3. Sliding Window

The sliding window shall be a range of k consecutive Sequence Numbers (modulo 256). Data Packets with Sequence Numbers within the sliding window can be accepted.

The size of the sliding window (k) shall be the same as the size of the sliding window used at the Transmit TEP of the Transport Channel (see 4.4.4.3).

Upon transitioning to the OPEN state (see 4.5.2.4), the sliding window shall be set to the range from 1 to k inclusive.

When the sliding window is from n to $n+k-1$, it is advanced by 1 (that is, it becomes $n+1$ to $n+k$) upon reception of the Data Packet with Sequence Number n. The Data Packets with consecutive Sequence Numbers $n+1$, …, $n+p$ have already been received when the Data Packet with Sequence Number n is received, the sliding window shall become $n+p+1$ to $n+p+k$.

4.5.3.4. Packet Reception

When either a Control or Data Packet is received without an error (see 4.5.3.2), the Receive TEP shall process the received Packet with one of the following procedures.

(1) A received Data Packet shall be processed as follows according to its Sequence Number. It is assumed that the current sliding window is from *n* to *n*+*k*-1.

a) If the Sequence Number is in the current sliding window and the Packet has not been received yet, the Packet shall be accepted and acknowledged.

b) If the Sequence Number is in the current sliding window and the Packet has already been received, the Packet shall be acknowledged but discarded (because its Ack Packet may have been lost).

c) If the Sequence Number is in the range from *n*-*k* to *n*-1 inclusive, the Packet shall be acknowledged but discarded (because the Packet has already been received but its Ack Packet may have been lost).

d) If none of the above conditions hold, the Packet shall be discarded and not be acknowledged (because the Transmit TEP is not functioning correctly). The Transport Channel shall be declared to be "inactive," and the Receive TEP shall transition to the CLOSED state.

(2) A received Control Packet shall be processed as follows.

a) If an Open Command with Sequence Number 0 is received in the ENABLED state or a Close Command is received in the OPEN state, the Packet shall be accepted and acknowledged. A state transition shall be started using the Channel Control procedure (see 4.5.2)

b) If an Open Command with Sequence Number 0 is received in the OPEN state before receiving any Data Packet or a Close Command is received in the CLOSING state, it shall be acknowledged but discarded (because the Packet has already been received but its Ack Packet may have been lost).

c) If an Open Command with Sequence Number 0 is received in the OPEN state after receiving one or more Data Packets, it shall be discarded (because the Transmit TEP is not functioning correctly). The Transport Channel shall be declared to be "inactive," and the Receive TEP shall transition to the CLOSED state.

c) If the Sequence Number is not 0, the Packet shall be discarded and not acknowledged (because the Transmit TEP is not functioning correctly). The Transport Channel shall be declared to be "inactive," and the Receive TEP shall transition to the CLOSED state.

For acknowledging a received Data or Control Packet, the Receive TEP shall send a Data Ack or Control Ack Packet, respectively, having the same Sequence Number as that of the received Packet.

An Ack Packet shall not be sent for any received Packet that has an error (see 4.5.3.2).

When the Receive TEP has transitioned to the CLOSED state in this procedure, it shall notify the receiving application that the Transport Channel has been closed using the ChannelControl.indication primitive (see 3.2.3.3)

4.5.3.5. Flow control

Whether or not flow control is used shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

If flow control is used, any Data Ack or Control Ack Packet shall carry the maximum value of the Sequence Number that the Receive TEP can accept at that time (Maximum Acceptable Sequence Number or MASN, see

4.2.4.1).

[Note 1] When a Transport Channel is opened, the initial value of MASN is carried in the Control Ack Packet that acknowledges the received Open Command.

[Note 2] There is no meaning in carrying a value of MASN in a Control Ack Packet that acknowledges a received Close Command. Therefore, any default value of MASN can be carried in a Control Ack Packet that acknowledges a received Close Command.

The MASN shall be within the current sliding window.

When the value of MASN has changed and there is no Data Ack Packet to be transmitted, the Receive TEP shall transmit a Flow Control Packet carrying the new value of MASN using the Packet Transmission procedure (see 4.5.3.7). The Sequence Number of a Flow Control Packet shall be the same as that of the Packet transmitted from the Receive TPE most recently. When a Flow Control Packet has been transmitted, no other Flow Control Packet (except for a retransmission of the same Flow Control Packet) shall be transmitted until the transmitted Flow Control Packet is acknowledged (see 4.5.3.7).

If flow control is used, the second paragraph in 4.5.3.4 shall be modified as follows.

A received Data Packet shall be processed as follows according to its Sequence Number. It is assumed that the current sliding window is from n to $n+k-1$.

a) If the Sequence Number is in the current sliding window and less than or equal to the MASN, and the Packet has not been received yet, the Packet shall be accepted and acknowledged.

b) If the Sequence Number is in the current sliding window and less than or equal to the MASN, and the Packet has already been received, the Packet shall be acknowledged but discarded (because its Ack Packet may have been lost).

c) If the Sequence Number is in the range from $n-k$ to $n-1$ inclusive, the Packet shall be acknowledged but discarded (because the Packet has already been received but its Ack Packet may have been lost).

d) If none of the above conditions hold, the Packet shall be discarded and not be acknowledged.

4.5.3.6. Heartbeat

Heartbeat can be used either (1) for the Transmit TEP to check whether the link and the Receive TEP are still alive when there is no data to send (Transmit Heartbeat), and/or (2) for the Receive TEP to check whether the link and the Transmit TEP are still alive when there is no data to receive (Receive Heartbeat). It is possible to use both Transmit Heartbeat and Receive Heartbeat simultaneously on a Transport Channel. Whether or not each of Transmit Heartbeat and Receive Heartbeat is used shall be determined for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

(1) Transmit Heartbeat

If Transmit Heartbeat is used, upon reception of a Heartbeat Packet from the Transmit TEP, the Receive TEP shall send a Heartbeat Ack Packet having the same Sequence Number as that of the received Heartbeat Packet. A Heartbeat Ack Packet shall not be sent if the received Heartbeat Packet has an error (see 4.5.3.2).

(2) Receive Heartbeat

If Receive Heartbeat is used, a Receive Heartbeat timer shall be used by the Receive TEP for each Transport Channel as follows.

The initial value for the Receive Heartbeat timer shall be determined for each Transport Channel and specified in

a project-specific document using the table provided in 5.2 of this document.

Upon transitioning into the OPEN state, the Receive Heartbeat timer shall be started. Whenever a Packet (either an Ack or Flow Control Packet) is transmitted by the Receive TEP, the Receive Heartbeat timer shall be restarted.

When the Receive Heartbeat timer expires, a Heartbeat Packet shall be transmitted using the Packet Transmission procedure (see 4.5.3.7), and the Receive Heartbeat timer shall be restarted. The Sequence Number of a Heartbeat Packet shall be ~~the same as that of the Ack Packet transmitted most recently~~set to 0.

4.5.3.7. Packet Transmission

The Receive TEP shall transmit Flow Control Packets (if flow control is used) and Heartbeat Packets (if Receive Heartbeat is used) using this procedure.

When either a Flow Control Packet or Heartbeat Packet is to be transmitted, it shall be transmitted immediately and a Transmit timer shall be started. It shall be started at the time when the last octet of the Packet is transmitted.

An independent Transmit timer shall be maintained for each outstanding Packet.

The initial value for the Transmit timers shall be the same as that used by the Transmit TEP of the same Transport Channel (see 4.4.4.5).

When an Ack Packet acknowledging an outstanding Packet is received, the Transmit timer for that Packet shall be cancelled.

If no Ack Packet has been received when the Transmit timer for a Packet expires, the Packet shall be retransmitted with the original Sequence Number until the maximum retry count is reached. The Transmit timer for that Packet shall be restarted.

The maximum retry count shall be the same as that used by the Transmit TEP of the same Transport Channel (see 4.4.4.5).

When the maximum retry count has been exceeded for any Packet, the Transport Channel shall be declared to be “inactive,” and the Receive TEP shall transition to the CLOSED state. The Receive TEP shall also notify the receiving application that the Transport Channel has been closed using the ChannelControl.indication primitive (see 3.2.3.3).

If an Ack Packet has been received for a non-outstanding Packet (a Packet for which no Transmit timer is running), it shall be discarded.

4.5.3.8. Processing of Secondary Header

If the Secondary Header (see 4.2.4.2) is present, its contents shall be processed with a procedure determined by each project, which shall be documented in a project-specific document

4.5.3.9. Redundancy Control Support

When a Transport Channel has been declared to be “inactive” (see 4.4.4.5) at the Transmit TEP, the sending application may choose to open an alternative Transport Channel using a different route to the receiving node and retransmit all the outstanding Data Packets using the new Transport Channel (see 4.4.4.9). If such a redundancy control mechanism is implemented by the sending application, the Receive TEP shall use Receive Heartbeat (see 4.4.4.7 and 4.5.3.6) in order to close any inactive Transport Channel.

4.5.4. RECONSTRUCTION

This procedure shall reconstruct received Service Data Units and deliver them to the receiving application.

There shall be one instance of this procedure for each Transport Channel.

This procedure shall extract Service Data Units (or their segments) from the Application Data field (see 4.2.4.3) of received SpW-R Data Packets, reconstruct the original Service Data Units using the values of the Sequence Flags (see 4.2.3.4.4) of each SpW Data Packet if necessary, and deliver the reconstructed Service Data Units to the receiving application.

4.6. COMMON PROCEDURES AT A NODE

4.6.1. GENERAL

This section specifies procedures commonly used by the TEPs and REPs of all the Transport Channels that exist on a node.

[Note] The procedures specified in this subsection are presented in an abstract sense and are not intended to imply any particular implementation approach of a protocol entity.

There shall be two procedures commonly used at a node:

a) Channel Multiplexing; and

b) Channel De-multiplexing.

4.6.2. CHANNEL MULTIPLEXING

This procedure shall multiplex outgoing Packets delivered by the Transmission Control and Reception Control procedures (see 4.4.4 and 4.5.3) of all the Transport Channels to generate a single stream of Packets to be transmitted on the outgoing SpaceWire link.

There shall be one instance of this procedure at each node.

In the stream of Packets mentioned above, Packets shall be transmitted based, firstly, on the priority levels assigned to the Transport Channels (if they are assigned) and, secondly, on the priority levels assigned to the Packet Types. The priority level may be assigned for each Transport Channel and specified in a project-specific document using the table provided in 5.2 of this document.

Each Packet Type has a priority level specified as follows. This list means that if an Ack Packet is available for transmission, it shall be transmitted before Packets of the other Types in a Transport Channel.

1) Ack Packet;

2) Control Packet;

3) Data Packet;

4) Flow Control Packet;

5) Heartbeat Packet.

4.6.3. CHANNEL DEMULTIPLEXING

This procedure shall de-multiplex incoming Packets of all the Transport Channels and deliver them to the instances of the Transmission Control and Reception Control procedure (see 4.4.4 and 4.5.3) of the appropriate Transport Channels based on the value of the Channel Number contained in the SpW-R Packet Header (see

4.2.3.6).

There shall be one instance of this procedure at each node.

5. HOW TO SPECIFY PARAMETERS FOR EACH NEWORK

5.1. GENERAL

This section presents methods for specifying the parameters of SpaceWire-R (SpW-R) used in each SpaceWire network.

The values of the parameters of SpW-R used for a SpaceWire network shall be specified by filling the table shown in the following sub-sections.

5.2. PARAMETERS USED FOR EACH TRANSPORT CHANNEL

The parameters used for each Transport Channel shall be specified by filling Table 5-1.

Table 5-1: Table for Specifying the Values of Parameters Used for Each Transport Channel

Option	Reference	Value	Unit or Options
SLA of Transmit TEP	4.2.3.2 and 4.2.3.9.3		N/A
Length of the Prefix to be prepended to the SLA of Transmit TEP	4.2.3.8.3		Octet
Prefix to be prepended to the SLA of Transmit TEP (if the Prefix is present)	4.2.3.9.2		N/A
SLA of Receive TEP	4.2.3.2 and 4.2.3.9.3		N/A
Length of the Prefix to be prepended to the SLA of Receive TEP	4.2.3.8.3		Octet
Prefix to be prepended to the SLA of Receive TEP (if the Prefix is present)	4.2.3.9.2		N/A
Maximum Length of Service Data Unit	3.3.2.1		Octet
Maximum Length of Application Data Field	4.2.4.30		Octet
Sliding Window Size (*k*)	4.4.4.3		Integer
Transmit Timer Initial Value	4.4.4.5		Millisecond
Maximum Retry Count	4.4.4.5		Integer
Flow Control	4.4.4.6 and 4.5.3.5		Used or Not Used
Transmit Heartbeat	4.4.4.7		Used or Not Used
Transmit Heartbeat Timer Initial Value	4.4.4.7		Millisecond
~~Redundancy Control~~	~~4.4.4.8~~		~~Used or Not Used~~
Receive Heartbeat	4.5.3.6		Used or Not Used
Receive Heartbeat Timer Initial Value	4.5.3.6		Millisecond
Close Timer Initial Value	4.5.2.5		Millisecond
Priority Level	4.6.1		Integer

[Example] An example of specifying the values of the parameters of SpW-R used for a Transport Channel is shown in Appendix C.

APPENDIX A ACRONYMS

This appendix lists all acronyms used in this document.

CCSDS	Consultative Committee for Space Data Systems
CRC	Cyclic Redundancy Check
ECSS	European Cooperation for Space Standardization
ESA	European Space Agency
GRDDP	GOES-R Reliable Data Delivery Protocol
GSFC	Goddard Space Flight Center
JAS	Joint Architecture Standard
JRDDP	JAS Reliable Data Delivery Protocol
MASN	Maximum Acceptable Sequence Number
NASA	National Aeronautics and Space Administration
PDU	Protocol Data Unit
RDDP	Reliable Data Delivery Protocol
SDU	Service Data Unit
SpW	SpaceWire
SLA	SpaceWire Logical Address
TBD	To Be Determined
TEP	Transport End Point

APPENDIX B EXAMPLES OF PDU EXCHANGES

B.1 GENERAL

This appendix shows some examples of exchanges of SpW-R Packets between the Transmit TEP and the receive TEP of a Transport Channel.

This appendix is not a part of the specification.

B.2 NORMAL CASE

An example of an exchange of SpW-R Packets when there is no error is shown in Figure B-1. The number shown in the parentheses in Figure B-1 is the Sequence Number of each SpW-R Packet.

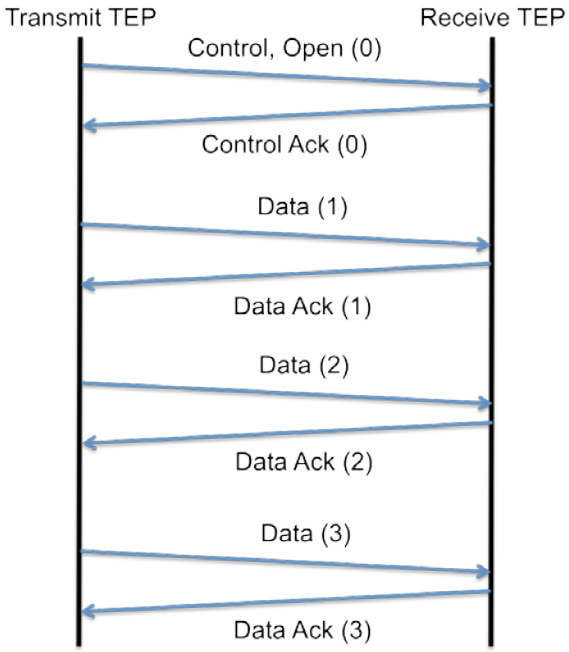

Figure B-1: Exchange of SpW-R Packets in a normal case

B.2 DATA ERROR CASE

An example of an exchange of SpW-R Packets when there is an error in the transmission of a Data Packet is shown in Figure B-2. The number shown in the parentheses in Figure B-2 is the Sequence Number of each SpW-R Packet.

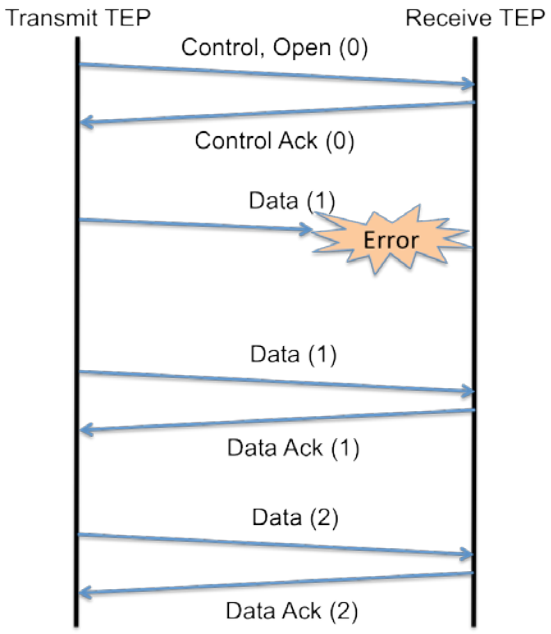

Figure B-2: Exchange of SpW-R Packets when there is an error in a Data Packet

B.3 ACK ERROR CASE

An example of an exchange of SpW-R Packets when there is an error in the transmission of an Ack Packet is shown in Figure B-3. The number shown in the parentheses in Figure B-3 is the Sequence Number of each SpW-R Packet.

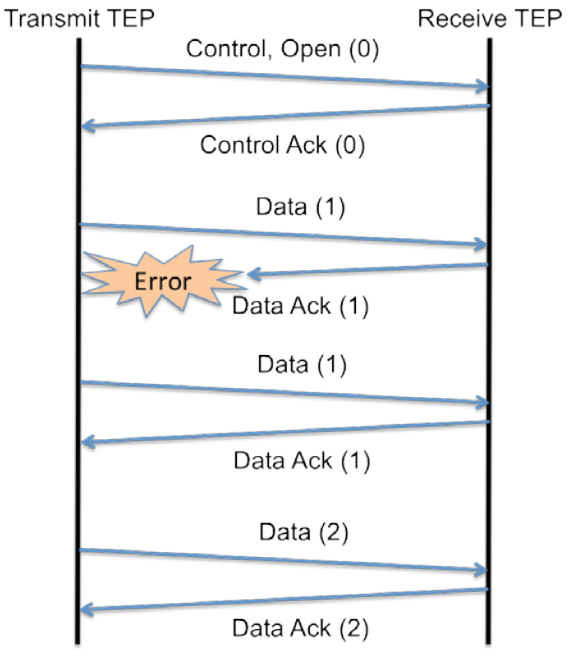

Figure B-3: Exchange of SpW-R Packets when there is an error in an Ack Packet

B.4 FLOW CONTROL CASE

An example of an exchange of SpW-R Packets when the optional flow control is used is shown in Figure B-4. For the SpW-R Packets sent from the Transmit TEP to the Receive TEP, the number shown in the parentheses in

Figure B-4 is the Sequence Number of each SpW-R Packet. For the SpW-R Packets sent from the Receive TEP to the Transmit TEP, the numbers shown in the parentheses are the Sequence Number and the Maximum Acceptable Sequence Number (MASN) of each SpW-R Packet.

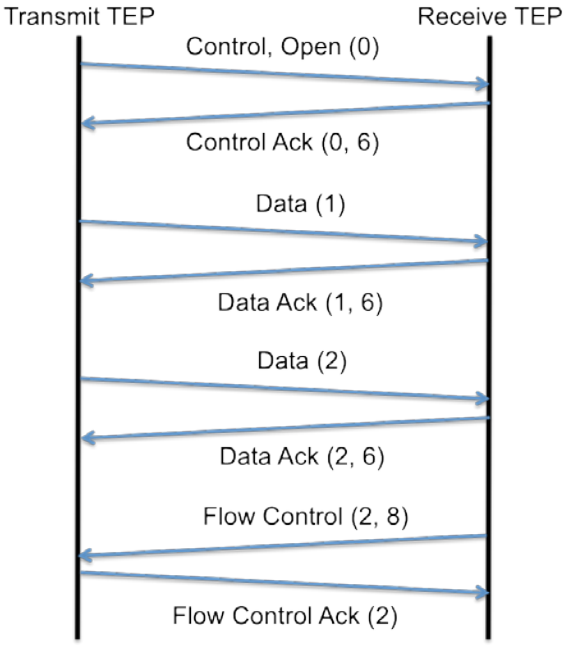

Figure B-4: Exchange of SpW-R Packets when flow control is used

APPENDIX C EXAMPLE OF PARAMETERS FOR A NETWORK

C.1 GENERAL

This appendix shows an example of specifying the parameters of SpaceWire-R used in a SpaceWire network.

This appendix is not a part of the specification.

C.2 OPTIONS USED FOR EACH TRANSPORT CHANNEL

An example of specifying the values of the parameters of SpW-R used for a Transport Channel is shown in Table C-1.

Table C-1: Example of Specifying the Values of Parameters Used for a Transport Channel

Option	Reference	Value	Unit or Options
SLA of Transmit TEP	4.2.3.2 and 4.2.3.9.3	65	N/A
Length of the Prefix to be prepended to the SLA of Transmit TEP	4.2.3.8.3	0	Octet
Prefix to be prepended to the SLA of Transmit TEP (if the Prefix is present)	4.2.3.9.2	N/A	N/A
SLA of Receive TEP	4.2.3.2 and 4.2.3.9.3	66	N/A
Length of the Prefix to be prepended to the SLA of Receive TEP	4.2.3.8.3	0	Octet
Prefix to be prepended to the SLA of Receive TEP (if the Prefix is present)	4.2.3.9.2	N/A	N/A
Maximum Length of Service Data Unit	3.3.2.1	2048	Octet
Maximum Length of Application Data Field	4.2.4.3~~0~~	256	Octet
Sliding Window Size (*k*)	4.4.4.3	8	Integer
Transmit Timer Initial Value	4.4.4.5	500	Millisecond
Maximum Retry Count	4.4.4.5	3	Integer
Flow Control	4.4.4.6 and 4.5.3.5	Used	Used or Not Used
Transmit Heartbeat	4.4.4.7	Used	Used or Not Used
Transmit Heartbeat Timer Initial Value	4.4.4.7	2000	Millisecond
~~Redundancy Control~~	~~4.4.4.8~~	~~Not Used~~	~~Used or Not Used~~
Receive Heartbeat	4.5.3.6	Used	Used or Not Used
Receive Heartbeat Timer Initial Value	4.5.3.6	2000	Millisecond
Close Timer Initial Value	4.5.2.5	500	Millisecond
Priority Level	4.6.1	2	Integer