

# SpaceWire Requirements

This document serves as a supporting artifact for the paper titled “*Temporal Logics Meet Real-World Software Requirements: A Reality Check*”, submitted for review at FormaliSE 2025. It includes the extracted requirements exhibiting temporal behaviour from SpaceWire [1], a standardized communication protocol widely employed in spacecraft and aerospace systems [2]. This artifact is also part of the replication package hosted on GitLab [3].

## References

- [1] European Cooperation for Space Standardization (ECSS), “ECSS-E-ST-50-12C Rev.1 – SpaceWire – Links, nodes, routers and networks,” May 2019. [Online]. Available: <https://ecss.nl/standard/ecss-e-st-50-12c-rev-1-spacewire-links-nodes-routers-and-networks-15-may-2019/>
- [2] S. Parkes and P. Armbruster, “Spacewire: A spacecraft onboard network for real-time communications,” in *14th IEEE-NPSS Real Time Conference, 2005*. IEEE, 2005, pp. 6–10.
- [3] “tlparser: Temporal Logic Parser,” Nov. 2024. [Online]. Available: <https://gitlab.com/FormaliSE25/tlparser>

## Contents

1	Requirement ID: 1001	6
2	Requirement ID: 1002	6
3	Requirement ID: 1003	7
4	Requirement ID: 1004	7
5	Requirement ID: 1005	8
6	Requirement ID: 1006	8
7	Requirement ID: 1007	8
8	Requirement ID: 1008	9

9 Requirement ID: 1009	9
10 Requirement ID: 1010	9
11 Requirement ID: 1011	10
12 Requirement ID: 1012	10
13 Requirement ID: 1013	10
14 Requirement ID: 1014	11
15 Requirement ID: 1015	11
16 Requirement ID: 1016	11
17 Requirement ID: 1017	12
18 Requirement ID: 1018	12
19 Requirement ID: 1019	12
20 Requirement ID: 1020	13
21 Requirement ID: 1021	13
22 Requirement ID: 1022	13
23 Requirement ID: 1023	14
24 Requirement ID: 1024	14
25 Requirement ID: 1025	15
26 Requirement ID: 1026	15
27 Requirement ID: 1027	16
28 Requirement ID: 1028	16
29 Requirement ID: 1029	17
30 Requirement ID: 1030	17

31 Requirement ID: 1031	18
32 Requirement ID: 1032	18
33 Requirement ID: 2001	18
34 Requirement ID: 2002	19
35 Requirement ID: 2003	19
36 Requirement ID: 2004	19
37 Requirement ID: 2005	20
38 Requirement ID: 2006	20
39 Requirement ID: 2007	20
40 Requirement ID: 2008	21
41 Requirement ID: 2009	21
42 Requirement ID: 2010	21
43 Requirement ID: 2011	22
44 Requirement ID: 2012	22
45 Requirement ID: 2013	22
46 Requirement ID: 2014	23
47 Requirement ID: 2015	23
48 Requirement ID: 2016	23
49 Requirement ID: 2017	24
50 Requirement ID: 2018	24
51 Requirement ID: 2019	24
52 Requirement ID: 2020	25

53 Requirement ID: 2021	25
54 Requirement ID: 2022	26
55 Requirement ID: 2023	26
56 Requirement ID: 2024	27
57 Requirement ID: 2025	27
58 Requirement ID: 2026	28
59 Requirement ID: 2027	28
60 Requirement ID: 2028	28
61 Requirement ID: 2029	29
62 Requirement ID: 2030	29
63 Requirement ID: 2031	29
64 Requirement ID: 2032	30
65 Requirement ID: 2033	30
66 Requirement ID: 2034	30
67 Requirement ID: 2035	31
68 Requirement ID: 2036	31
69 Requirement ID: 2037	31
70 Requirement ID: 3001	32
71 Requirement ID: 3002	32
72 Requirement ID: 3003	33
73 Requirement ID: 3004	33
74 Requirement ID: 3005	34

75 Requirement ID: 3006	34
76 Requirement ID: 3007	35
77 Requirement ID: 3008	35
78 Requirement ID: 3009	36
79 Requirement ID: 3010	36
80 Requirement ID: 3011	37
81 Requirement ID: 3012	37
82 Requirement ID: 3013	38
83 Requirement ID: 3014	38
84 Requirement ID: 3015	39
85 Requirement ID: 4001	39
86 Requirement ID: 4002	39
87 Requirement ID: 4003	40
88 Requirement ID: 5001	40
89 Requirement ID: 5002	41

## 1 Requirement ID: 1001

**Status:** OK

**Description:** A SpaceWire Port shall incorporate a flow control manager which manages the flow of data over the link, preventing data from being sent when there is no space for it in the receive FIFO.

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((RXFIFO \text{ full}) \rightarrow (\neg(send \ NChar))) \quad (1)$$

## 2 Requirement ID: 1002

**Status:** OK

**Description:** Transmit Enable, which enables the transmitter (character encoder, serialiser, data-strobe encoder and line driver) when asserted and resets it when de-asserted.

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \Box(((Transmit \ asserted) \rightarrow \\ (enable : (encoder \wedge serialiser \wedge data \ strobe \ encoder \\ \wedge line \ driver))) \vee ((Transmit \ de - asserted) \rightarrow \\ (reset : (encoder \wedge serialiser \wedge data \ strobe \ encoder \wedge line \ driver)))) \end{aligned} \quad (2)$$

### 3 Requirement ID: 1003

**Status:** OK

**Description:** Receive Enable, which enables the receiver (line receiver, data-strobe decoder, de-serialiser and character decoder) when asserted and resets it when de-asserted

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((Receive\ asserted) \rightarrow \\ & \quad (enable : (line\ receiver \wedge de - serialiser \wedge data - strobe\ decoder \wedge character\ decoder))) \\ & \vee ((Receive\ de - asserted) \rightarrow \\ & \quad (reset : (line\ receiver \wedge de - serialiser \wedge data - strobe\ decoder \wedge character\ decoder)))) \end{aligned} \quad (3)$$

### 4 Requirement ID: 1004

**Status:** OK

**Description:** The encoding layer shall inform the data link layer of changes in the encoding layer indicated by the following status flags: 1.Disconnect, which indicates that the link has been disconnected. 2.Receive error, which indicates that an error has been detected in a received symbol. 3.gotNull, which indicates that the first Null control code has been received without any parity errors

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((encoding\ layer\ disconnected) \rightarrow (Disconnect\ flag)) \\ & \Box((receive\ error) \rightarrow (receive\ error\ flag)) \\ & \Box((firstNull\ received) \rightarrow (gotNull\ flag)) \end{aligned} \quad (4)$$

## 5 Requirement ID: 1005

**Status:** OK

**Description:** The Data and Strobe signals shall be set to zero on power up reset

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((Power\ up\ Reset) \rightarrow (Data := 0 \wedge Strobe := 0)) \quad (5)$$

## 6 Requirement ID: 1006

**Status:** OK

**Description:** Null detection shall be enabled whenever the receiver is enabled

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((receiver\ enabled) \rightarrow (Null\ detection\ enabled)) \quad (6)$$

## 7 Requirement ID: 1007

**Status:** OK

**Description:** The gotNull condition shall only be cleared when the RX Enable flag is de-asserted

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((RX\ enable\ flag\ de\ asserted) \rightarrow (gotNull\ condition\ cleared)) \quad (7)$$



## 8 Requirement ID: 1008

**Status:** OK

**Description:** Parity detection shall be enabled whenever the receiver is enabled and the gotNull condition is asserted

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((receiver\ enabled) \wedge (gotNull\ asserted)) \rightarrow (Parity\ enable)) \quad (8)$$

## 9 Requirement ID: 1009

**Status:** OK

**Description:** ESC error detection shall be enabled while the gotNull condition is asserted

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((gotNull\ asserted) \rightarrow (ESC\ error\ detection\ enable)) \quad (9)$$

## 10 Requirement ID: 1010

**Status:** OK

**Description:** The data link layer shall indicate to the Network layer when it is able to accept another N-Char for sending

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((ready\ to\ accept\ N - Char) \rightarrow (indicate\ accept\ N - Char)) \quad (10)$$

## 11 Requirement ID: 1011

**Status:** OK

**Description:** The data link layer shall indicate to the Network layer when it has an N-Char ready for passing to the Network layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((NChar \text{ ready to pass}) \rightarrow (\text{indicate passing } NChar)) \quad (11)$$

## 12 Requirement ID: 1012

**Status:** OK

**Description:** The data link layer shall indicate to the Network layer when it has a broadcast code ready for passing to the Network layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((broadcast \text{ code ready}) \rightarrow (\text{indicate passing broadcast code})) \quad (12)$$

## 13 Requirement ID: 1013

**Status:** OK

**Description:** The data link layer shall control the operation of the encoding layer using the following control flags: 1. Transmit Enable, which enables the transmitter (character encoder, serialiser, data-strobe encoder and line driver) when asserted and reset it when de-asserted. 2. Receive Enable, which enables the receiver (line receiver, data-strobe decoder, de-serialiser and character decoder) when asserted and reset it when de-asserted

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((transmit \text{ enable}) \rightarrow (enable \text{ transmitter})) \\ &\Box(\neg(transmit \text{ enable}) \rightarrow (reset \text{ transmitter})) \\ &\Box((receive \text{ enable}) \rightarrow (enable \text{ receiver})) \\ &\Box(\neg(receive \text{ enable}) \rightarrow (reset \text{ receiver})) \end{aligned} \quad (13)$$

## 14 Requirement ID: 1014

**Status:** OK

**Description:** The data link layer shall respond to changes in the encoding layer indicated by the following status flags: 1.Disconnect, which indicates that the link has been disconnected. 2.Receive error, which indicates that an error has been detected in a received symbol. 3.gotNull, which indicates that the first Null character has been received without any parity errors

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((link\ disconnected) \rightarrow (indicate\ Disconnect)), \\ &\Box((error\ detected) \rightarrow (indicate\ Receive\ Error)), \\ &\Box((firstNull\ received) \rightarrow (indicate\ gotNull)) \end{aligned} \tag{14}$$

## 15 Requirement ID: 1015

**Status:** OK

**Description:** An FCT shall be sent from the data link layer to the encoding layer when the data link layer is ready to receive a further eight N-Chars

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((data\ link\ layer\ ready\ to\ receive) \rightarrow (FCT\ sent)) \tag{15}$$

## 16 Requirement ID: 1016

**Status:** OK

**Description:** The transmit credit count shall be set to zero whenever the link state machine enters the ErrorReset state

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((ErrorReset\ state) \rightarrow (set\ transmit\ credit := 0)) \tag{16}$$

## 17 Requirement ID: 1017

**Status:** OK

**Description:** A maximum of seven FCTs shall be outstanding at any time

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{Number of FCTs}) \leq 7) \quad (17)$$

## 18 Requirement ID: 1018

**Status:** OK

**Description:** If an FCT is received which causes the transmit credit counter to exceed its maximum value, a credit error shall be raised

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{transmit credit} \geq \text{max}) \rightarrow (\text{credit error})) \quad (18)$$

## 19 Requirement ID: 1019

**Status:** OK

**Description:** The data link layer shall keep a credit count of the number of N-Chars it has asked for by passing FCTs to the encoding layer and has yet to receive from the encoding layer (receive credit count) as follows: 1.Increment the receive credit count by eight each time an FCT is passed to the encoding layer. 2.Decrement the receive credit count by one each time an NChar is received from the encoding layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((\text{FCT passed}) \rightarrow (\text{receive credit} := \text{receive credit} + 8)) \\ &\Box((\text{Nchar received}) \rightarrow (\text{receive credit} := \text{receive credit} - 1)) \end{aligned} \quad (19)$$

## 20 Requirement ID: 1020

**Status:** OK

**Description:** The data link layer shall keep a credit count of the number of NChars it has been authorized to send (transmit credit count), as follows: 1.Each time the data link layer receives an FCT from the encoding layer it increments its transmit credit count by eight. 2.Whenever the data link layer sends an NChar to the encoding layer it decrements its transmit credit count by one

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((FCT\ received) \rightarrow (credit := credit + 1)) \\ &\Box((NChar\ sent) \rightarrow (credit := credit - 1)) \end{aligned} \tag{20}$$

## 21 Requirement ID: 1021

**Status:** OK

**Description:** The receive credit count shall be set to zero whenever the link state machine enters the ErrorReset state

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((ErrorReset\ state) \rightarrow (set\ receive\ credit := 0)) \tag{21}$$

## 22 Requirement ID: 1022

**Status:** OK

**Description:** An FCT shall only be sent when there is room in the data link layer to receive another eight more N-Chars from the encoding layer and when the receive credit count has a value of eight or more less than its maximum value

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((enough\ room\ for\ 8\ Nchar) \wedge ((receive\ credit := 8) \vee (receive\ credit \geq max)), \\ &\rightarrow (send\ FCT)) \end{aligned} \tag{22}$$

## 23 Requirement ID: 1023

**Status:** OK

**Description:** A credit error shall be detected when either of the following conditions occur: 1.An NChar is received when the receive credit counter is zero. 2.An FCT is received which causes the transmit counter to exceed its maximum permitted value

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \Box(((Nchar\ received \wedge credit\ counter := 0) \vee (transmit\ counter \geq max)) \\ \rightarrow (credit\ error\ detected)) \end{aligned} \quad (23)$$

## 24 Requirement ID: 1024

**Status:** OK

**Description:** When in the ErrorWait state, the Link state machine shall initiate the following actions: 1.Start a 12,8 micro second timer on entering the ErrorWait state. 2.Deassert the Transmit Enable control flag. 3.Assert the Receive Enable control flag without storing any NChars received from the Encoding Layer in the RX FIFO and without registering any broadcast codes received from the Encoding Layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \Box((ErrorWait\ state) \rightarrow ((start\ timer_{12.8}) \wedge (De - assert\ transmit\ enable\ flag) \\ \wedge (assert\ Recieve\ enable\ flag) \wedge \neg(NChar\ stored) \wedge \neg(register\ broadcast\ code))) \end{aligned} \quad (24)$$

## 25 Requirement ID: 1025

**Status:** OK

**Description:** When in the Started state, the Link state machine shall initiate the following actions: 1.Start a 12,8 micro second timer on entering the Started state. 2.Assert the Transmit Enable control flag, but only pass Nulls to the Encoding Layer. 3.Assert the Receive Enable control flag without storing any N-Chars received from the Encoding Layer in the RX FIFO and without registering any broadcast codes received from the Encoding Layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \Box((started\ state) \rightarrow ((start\ timer_{12.8}) \wedge ((transmit\ control\ flag) \wedge (only\ pass\ Nulls)) \\ \wedge ((Receive\ control\ flag) \wedge \neg(storing\ NChar) \wedge \neg(register\ code)))) \end{aligned} \quad (25)$$

## 26 Requirement ID: 1026

**Status:** OK

**Description:** When in the Connecting state, the Link state machine shall initiate the following actions: 1.Start a 12,8 micro second timer on entering the Connecting state. 2.Assert the Transmit Enable control flag, but only pass FCTs and Nulls to the Encoding Layer, following the rules described in clause 5.5.6. 3.Assert the Receive Enable control flag without storing any N-Chars received from the Encoding Layer in the RX FIFO and without registering any broadcast codes received from the Encoding Layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \Box((connecting\ state) \rightarrow ((start\ timer_{12.8}) \wedge (transmit\ control\ flag) \wedge (only\ pass\ FCTs) \\ \wedge (Receive\ control\ flag) \wedge \neg(storing\ NChar) \wedge \neg(register\ code)))) \end{aligned} \quad (26)$$

## 27 Requirement ID: 1027

**Status:** OK

**Description:** When in the ErrorReset state, the link state machine shall initiate the following actions:  
 1.De-assert the Transmit Enable control flag. 2.De-assert the Receive Enable control flag.  
 3.Set the transmit credit counter to zero. 4.Set the receive credit counter to zero. 5.Clear the gotFCT condition. 6.Start a 6,4  $\mu$ s timer on entering the ErrorReset state. De-asserting the Receive Enable control flag causes the Encoding layer to clear the gotNull condition

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((ErrorReset\ state) \rightarrow \\ & ((De - assert\ control\ flag) \wedge (De - assert\ Receive\ enable\ flag) \\ & \wedge (set\ transmit\ credit := 0) \wedge (set\ receive\ credit := 0) \wedge (clear\ gotFCT) \\ & \wedge (start\ timer_{6,4}))) \end{aligned} \quad (27)$$

## 28 Requirement ID: 1028

**Status:** OK

**Description:** The SpaceWire receiver shall be tolerant of simultaneous transitions on the Data and Strobe lines. Being tolerant of simultaneous transitions means that there is no lock-up of the receiver when a simultaneous transition occurs. Simultaneous transitions on the Data and Strobe lines are not part of the normal operation of SpaceWire. They can occur, however, either when a SpaceWire cable is plugged in while the transmitter is trying to make a connection, or when the LVDS driver or receiver circuits are enabled while transmitter is trying to make a connection

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((simultaneous\ transition) \rightarrow (no\ lock - up)) \quad (28)$$



## 29 Requirement ID: 1029

**Status:** OK

**Description:** When a node supports sending of interrupt codes and the host requests an interrupt to be sent, an interrupt code with the interrupt identifier matching the interrupt shall be passed to the data link layer for sending

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \square((\text{interrupt codes supported}) \wedge (\text{interrupt send request}) \rightarrow \\ (\text{interrupt code passed})) \end{aligned} \tag{29}$$

## 30 Requirement ID: 1030

**Status:** OK

**Description:** When in the Connecting state, the Link state machine shall initiate the following actions: 1.Start a 12,8  $\mu$ s timer on entering the Connecting state. 2.Assert the Transmit Enable control flag, but only pass FCTs and Nulls to the Encoding Layer, following the rules described in clause 5.5.6. 3.Assert the Receive Enable control flag without storing any N-Chars received from the Encoding Layer in the RX FIFO and without registering any broadcast codes received from the Encoding Layer

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} \square((\text{Connecting state}) \rightarrow \\ ((\text{start timer}) \wedge (\text{assert transmit enable flag}) \\ \wedge ((\text{assert receive enable flag}) \wedge \neg(\text{NChar} \in \text{RX FIFO}) \\ \wedge \neg(\text{register broadcast code})))) \end{aligned} \tag{30}$$

### 31 Requirement ID: 1031

**Status:** OK

**Description:** If an input port is waiting for packet characters to arrive, the output port that it is connected to shall also wait. If an output port is waiting to transmit packet characters, the input port it is connected to shall also wait

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((input\ port\ waiting) \leftrightarrow (output\ port\ waiting)) \quad (31)$$

### 32 Requirement ID: 1032

**Status:** OK

**Description:** When the output port finishes transmission of a packet, it shall be available to accept a packet from another input port

**Logic:** INV

**Translation:**  $\rightarrow$  LTL (yes),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((pocket\ transmission\ finished) \rightarrow (accept\ packet\ available)) \quad (32)$$

### 33 Requirement ID: 2001

**Status:** OK

**Description:** A SpaceWire Port shall incorporate a transmit FIFO (TX FIFO) which stores N-Chars provided by the application via the SpaceWire port interface until they can be sent across the link.

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((Nchar\ provided) \rightarrow ((NChar \in TXFIFO) \mathcal{U} (NChar\ sent))) \quad (33)$$

## 34 Requirement ID: 2002

**Status:** OK

**Description:** A SpaceWire Port shall incorporate a receive FIFO (RX FIFO) which stores received N-Chars until they can be read by the application via the SpaceWire port interface.

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((NChar \text{ received}) \rightarrow ((NChar \in RXFIFO) \mathcal{U} (NChar \text{ read}))) \quad (34)$$

## 35 Requirement ID: 2003

**Status:** OK

**Description:** Null is passed to the encoding layer by the data link layer whenever a link is not sending data or control symbols, to keep the link active and to support link disconnect detection.

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\neg((data \vee control \text{ symbol}) \text{ sent}) \rightarrow \mathcal{X}(Null \text{ passed}))) \quad (35)$$

## 36 Requirement ID: 2004

**Status:** OK

**Description:** When the SpaceWire output port is reset, it shall be a controlled reset avoiding simultaneous transitions of Data and Strobe signals. For example, after stopping transmission the Strobe signal can be reset first, followed by the Data signal. This prevents a simultaneous transition on the data and strobe signals which can cause some IEEE 1355-1995 devices to enter an unrecoverable fault state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} (output \text{ port reset}) \rightarrow & \\ & \Box \neg (data \text{ transition} \wedge Strobe \text{ signal transition}), \\ & \Box ((transmission \text{ stopped}) \\ & \rightarrow (\mathcal{X}(reset \text{ strobe signal}) \wedge \mathcal{X}\mathcal{X}(reset \text{ data signal}))) \end{aligned} \quad (36)$$

### 37 Requirement ID: 2005

**Status:** OK

**Description:** gotNull shall be asserted when the first Null after the receiver is enabled is detected

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((receiver\ enabled) \rightarrow (\neg(gotNull\ assert)\mathcal{U}(Null\ detected))) \quad (37)$$

### 38 Requirement ID: 2006

**Status:** OK

**Description:** The disconnect detection shall be enabled when the link state machine leaves the ErrorReset state and the first edge is detected on the Data or Strobe line

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((state\ machine\ leaves\ ErrorReset) \rightarrow \\ &\quad (\neg(disconnect\ detection\ enable)\mathcal{U}(data\ edge\ detected) \\ &\quad \vee (Strobe\ line\ edge\ detected))) \end{aligned} \quad (38)$$

### 39 Requirement ID: 2007

**Status:** OK

**Description:** An escape character (ESC) followed by ESC, EOP or EEP is an invalid sequence and when received shall produce an ESC error. ESC followed by FCT is a Null and ESC followed by a data character is a broadcast code

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box(((ESC\ received \wedge \mathcal{X}ESC\ received) \vee (ESC\ received \wedge \mathcal{X}EOP) \\ &\quad \vee (ESC\ received \wedge \mathcal{X}EEP\ received)) \rightarrow (ESC\ error)) \\ &\Box((ESC\ received \wedge \mathcal{X}FCT\ received) \rightarrow (Null), \\ &\Box((ESC\ received \wedge \mathcal{X}data\ received) \rightarrow (broadcast\ code)) \end{aligned} \quad (39)$$

## 40 Requirement ID: 2008

**Status:** OK

**Description:** The data link layer shall not send any N-Chars to the encoding layer until it has received one or more FCTs from the encoding layer to indicate that the data link layer at the other end of the link is ready to receive N-Chars

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(\neg(\text{send } NChar) \mathcal{U}(\text{FCT received})) \quad (40)$$

## 41 Requirement ID: 2009

**Status:** OK

**Description:** Parity Error, ESC Error, gotFCT, gotN-Char and gotBC are only enabled after the first Null has been received (i.e. gotNull asserted)

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{first Null recieved}) \rightarrow \mathcal{X}((\text{Parity Error enable}) \wedge (\text{ESC Error enable}) \wedge (\text{gotFCT enable}) \wedge (\text{gotN - Char enable}) \wedge (\text{gotBC enable}))) \quad (41)$$

## 42 Requirement ID: 2010

**Status:** OK

**Description:** Disconnect Error is only enabled after the first transition on the data or strobe line

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((\text{first data transition}) \vee (\text{first strobe line transition})) \rightarrow \mathcal{X}(\text{Disconnect Error enable})) \quad (42)$$

## 43 Requirement ID: 2011

**Status:** OK

**Description:** When Port Reset is asserted, the following actions shall occur: 1.The TX FIFO and RX FIFO are cleared 2.The link state machine enters the ErrorReset state

**Logic:** LTL

**Translation:**  $\rightarrow \text{INV (no)}, \rightarrow \text{MTLb (yes)}, \rightarrow \text{STL (conditional)}$

**Formula:**

$$\begin{aligned} & \Box((\text{Port Reset asserted}) \rightarrow \\ & ((\text{TX FIFO cleared}) \wedge (\text{RX FIFO cleared}) \wedge \mathcal{X}(\text{ErrorReset state}))) \end{aligned} \quad (43)$$

## 44 Requirement ID: 2012

**Status:** OK

**Description:** If the transmit credit count reaches zero, the data link layer shall cease sending NChars to the encoding layer until it receives another FCT from the encoding layer which increases the transmit credit count to eight. When the transmit credit count is zero, the data link layer continues to send FCTs, Nulls and broadcast codes to the encoding layer

**Logic:** LTL

**Translation:**  $\rightarrow \text{INV (no)}, \rightarrow \text{MTLb (no)}, \rightarrow \text{STL (conditional)}$

**Formula:**

$$\begin{aligned} & \Box((\text{transmit credit} := 0) \rightarrow ((\text{stop sending Nchar}) \mathcal{U}(\text{transmit credit} := 8)), \\ & \Box((\text{transmit credit} := 0) \rightarrow (\text{send (FCT} \wedge \text{Null} \wedge \text{codes}))) \end{aligned} \quad (44)$$

## 45 Requirement ID: 2013

**Status:** OK

**Description:** When the link is initialised or re-initialised, one FCT shall be sent for every eight N-Chars that can be held in the receive FIFO up to the maximum of seven FCTs

**Logic:** LTL

**Translation:**  $\rightarrow \text{INV (no)}, \rightarrow \text{MTLb (no)}, \rightarrow \text{STL (conditional)}$

**Formula:**

$$\begin{aligned} & \Box((\text{link state} : (\text{initialised} \vee \text{reinitialised})) \rightarrow \\ & (((8 \text{ NChar held}) \rightarrow \mathcal{X}(\text{one FCT sent})) \mathcal{U} (\text{Num sent FCT} \leq 7))) \end{aligned} \quad (45)$$

## 46 Requirement ID: 2014

**Status:** OK

**Description:** The Link state machine shall leave the Run state on one of the following conditions: 1. When LinkDisabled is asserted, move to the ErrorReset state. 2. When a disconnect occurs, move to the ErrorReset state. 3. When a parity error occurs, move to the ErrorReset state. 4. When an ESC error occurs, move to the ErrorReset state. 5. When a credit error occurs, move to the ErrorReset state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((LinkDisabled \vee (disconnect) \vee (parity\ error) \\ & \vee (ESCerror) \vee (crediterror)) \rightarrow \mathcal{X}(ErrorReset\ state))) \end{aligned} \quad (46)$$

## 47 Requirement ID: 2015

**Status:** OK

**Description:** When the link state machine is in the Run state and sending of a broadcast code is requested, it shall be sent as soon as the data link layer has finished sending the current character or control code

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((link\ machine : Run\ state) \wedge (broadcast\ code\ requested)) \rightarrow \\ & ((current\ character\ finished) \rightarrow \mathcal{X}(broadcast\ codesent))) \end{aligned} \quad (47)$$

## 48 Requirement ID: 2016

**Status:** OK

**Description:** When sending of an FCT is requested, it shall be sent as soon as the current character or control code has been sent provided that: No broadcast code is waiting to be sent

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((FCT\ requested) \rightarrow (\neg(FCT\ sent) \mathcal{U} (current\ character\ sent))) \quad (48)$$

## 49 Requirement ID: 2017

**Status:** OK

**Description:** When the link state machine is in the Run state and an NChar is available in the transmit buffer, it shall be sent as soon as the current character or control code has been sent provided that: 1.No broadcast code is waiting to be sent. 2.No FCT is waiting to be sent. 3.The transmit control credit count is above zero

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((link\ machine\ state : Run) \wedge (NChar\ available)) \rightarrow \\ & \quad (\neg(send\ NChar) \mathcal{U} ((current\ Char\ sent) \\ & \quad \wedge (No\ waiting\ code) \wedge (No\ waiting\ FCT) \wedge (transmit\ credit \geq 0)))) \end{aligned} \quad (49)$$

## 50 Requirement ID: 2018

**Status:** OK

**Description:** When LinkDisabled is asserted in the Ready state, the link state machine remains in the Ready state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((LinkDisabled\ asserted) \wedge (Ready\ state)) \rightarrow \mathcal{X}(\Box(Ready\ state))) \quad (50)$$

## 51 Requirement ID: 2019

**Status:** OK

**Description:** When LinkDisabled is asserted in the Run state, the link state machine moves to the Error-Reset state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((LinkDisabled\ asserted) \wedge (Run\ state)) \rightarrow \mathcal{X}(ErrorReset\ state)) \quad (51)$$



## 52 Requirement ID: 2020

**Status:** OK

**Description:** The Link state machine shall leave the Ready state on one of the following conditions: 1. When LinkDisabled is asserted, move to the ErrorReset state. 2. When a disconnect occurs, move to the ErrorReset state. 3. When a parity error occurs, move to the ErrorReset state. 4. When an ESC error occurs, move to the ErrorReset state. 5. When an FCT, N-Char or broadcast code is received from the Encoding Layer, move to the ErrorReset state. 6. When LinkStart is asserted, move to the Started state. 7. When AutoStart is asserted and gotNull is asserted, move to the Started state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned}
 & \Box((\text{LinkDisabled}) \rightarrow \mathcal{X}(\text{ErrorReset})), \\
 & \Box((\text{disconnect}) \rightarrow \mathcal{X}(\text{ErrorReset})), \Box((\text{parity}) \rightarrow \mathcal{X}(\text{ErrorReset})), \\
 & \Box((\text{ESCError}) \rightarrow \mathcal{X}(\text{ErrorReset})), \\
 & \Box((\text{FCT received}) \vee (N - \text{char received}) \vee (\text{broadcast received})) \\
 & \rightarrow \mathcal{X}(\text{ErrorReset}), \\
 & \Box((\text{LinkStart asserted}) \\
 & \rightarrow \mathcal{X}(\text{Started state})), \\
 & \Box(((\text{AutoStart asserted}) \vee (\text{gotNull asserted})) \\
 & \rightarrow \mathcal{X}(\text{Started state}))
 \end{aligned} \tag{52}$$

## 53 Requirement ID: 2021

**Status:** OK

**Description:** This alternative behaviour allows the link state machine of the previous version of the SpaceWire standard ECSS-E-ST-50-12C (31 July 2008) to be used. The improvement in the current version (Revision 1) means that the link state machine immediately moves to and remains in the ErrorReset state with the receiver disabled, when Disable is asserted. The functions of Disable/Enable, Link Start and Autostart have been separated for clarity

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{receiver Disable asserted}) \rightarrow \mathcal{X}(\Box(\text{ErrorReset State}))) \tag{53}$$

## 54 Requirement ID: 2022

**Status:** OK

**Description:** When Port Reset is asserted, the Link Error Recovery state machine shall enter the Normal state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((PortReset\ asserted) \rightarrow \\ &\quad \mathcal{X}(Link\ Error\ Recovery\ state\ machine\ state : Normal)) \end{aligned} \tag{54}$$

## 55 Requirement ID: 2023

**Status:** OK

**Description:** The Link Error Recovery state machine shall leave the Normal state on the following conditions: 1. When the Link state machine is in the Run state and LinkDisabled is asserted, the Link Error Recovery state machine moves to the Recovery state. 2. When the Link state machine is in the Run state and a Disconnect, Parity Error, ESC Error, or Credit Error occurs, the Link Error Recovery state machine moves to the Recovery state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box(((Run\ State) \wedge (LinkDisabled\ asserted)) \rightarrow \mathcal{X}(Recovery\ state)), \\ &\Box(((Run\ state) \wedge ((Disconnect) \vee (Parity\ Error) \vee (ESC\ Error) \vee (Credit\ Error)))) \rightarrow \\ &\quad \mathcal{X}(Recovery\ state) \end{aligned} \tag{55}$$

## 56 Requirement ID: 2024

**Status:** OK

**Description:** When in the Recovery state, the Link Error Recovery state machine shall initiate the following actions: 1.If currently sending a packet, discard the remainder of the packet that has not yet been sent. 2.If the last character written to the receive FIFO was a data character, write an EEP to the receive FIFO. 3.When the last character written was an EOP or EEP, another EEP can be added to the receive FIFO. 4.If an EEP is pending, ready for writing to the receive FIFO and that FIFO is full preventing an EEP being written, wait until there is space in the receive FIFO and then write the EEP. 5.Record the cause of the error in a status register

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((\text{Recovery state}) \rightarrow ((\text{sending current packet}) \rightarrow ((a \text{ packet not sent}) \rightarrow \mathcal{X}(\text{discard remainder})))); \quad (56) \\ &\Box((\text{Recovery state}) \rightarrow ((\text{last character data}) \rightarrow \mathcal{X}(\text{EEP sent to receive FIFO}))), \\ &\Box((\text{Recovery state}) \rightarrow ((\text{last character } (EOP \vee EEP)) \rightarrow \mathcal{X}(\text{EEP sent to receive FIFO}))), \\ &\Box((\text{Recovery state}) \rightarrow ((EEP \text{ pending}) \wedge (FIFO \text{ full})) \rightarrow ((EEP \text{ wait}) \mathcal{U} (EEP \in \text{receive FIFO})) \rightarrow \mathcal{X}(\text{write EEP}))); \\ &\Box((\text{Recovery state}) \rightarrow ((\text{error}) \rightarrow \mathcal{X}(\text{Record error}))) \end{aligned}$$

## 57 Requirement ID: 2025

**Status:** OK

**Description:** If one or more N-Chars have been sent since the link reached the Run state AND the last character sent was not an EOP or EEP, read the transmit FIFO and discard the characters read until an EOP or EEP has been read and discarded

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box(((\text{Run state started}) \wedge (\text{min one NChar sent}) \wedge \neg(\text{last char } (EOP \vee EEP))) \rightarrow \\ &\quad \mathcal{X}(((\text{read transmit FIFO}) \wedge (\text{discard characters})) \\ &\quad \mathcal{U} ((EOP \vee EEP \text{ read}) \wedge (EOP \vee EEP \text{ discarded})))) \end{aligned} \quad (57)$$

## 58 Requirement ID: 2026

**Status:** OK

**Description:** If the receive FIFO is full it is not possible for the transmitter to send an FCT, hence the link initialisation cannot be completed. The link state machine cycles through the ErrorReset, ErrorWait, Ready and Started state and then times out in the Connecting state because no FCT can be received. When there is room for at least eight N-Chars, at least one FCT can be sent so the link initialisation process is then able to complete successfully.

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((receive\ FIFO\ full) \rightarrow \neg\Diamond(send\ FCT)), \\ & \Box((8\ NChar\ room) \rightarrow \mathcal{X}(send\ FCT)) \end{aligned} \tag{58}$$

## 59 Requirement ID: 2027

**Status:** OK

**Description:** The Link Error Recovery state machine shall leave the Recovery state on the following conditions: When all error recovery actions, listed in 5.5.8.4.b are successfully completed, move to the Normal state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((all\ error\ recovery\ actions\ completed) \rightarrow \mathcal{X}(normal\ state)) \tag{59}$$

## 60 Requirement ID: 2028

**Status:** OK

**Description:** An output port shall not transmit any other packet until the packet that it is currently transmitting has finished being sent or has been terminated following an error

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(\neg(transmit\ packet)\mathcal{U}((current\ packet\ sent) \vee (error\ transmit))) \tag{60}$$

## 61 Requirement ID: 2029

**Status:** OK

**Description:** If the allocated output port is busy, the newly arrived packet shall wait at the input port until the allocated output port is free to transmit the new packet

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{allocated output port busy}) \rightarrow ((\text{arrived packet wait})\mathcal{U}(\text{output port free}))) \quad (61)$$

## 62 Requirement ID: 2030

**Status:** OK

**Description:** When the allocated output port becomes free, the input port connected to it after arbitration shall transfer one packet to the output port and then free the output port for subsequent arbitration and use by the same or another input port

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\text{output port free}) \rightarrow ((\text{packet transfer}) \wedge \mathcal{X}(\text{free output port}))) \quad (62)$$

## 63 Requirement ID: 2031

**Status:** OK

**Description:** If one or more of the output ports in the multicast set cannot accept a new N-Char during packet transfer, the input port waits, and the N-Char is not sent to any of the multicast output ports until they are all ready

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(\neg(N - \text{Char accepted}) \rightarrow ((\text{input port wait}) \wedge (\neg(N - \text{Char sent}) \mathcal{U}(\text{multicast output ports ready})))) \quad (63)$$

## 64 Requirement ID: 2032

**Status:** OK

**Description:** On receipt of the DISTRIBUTED-INTERRUPT.request primitive the SpaceWire end-point shall immediately send the interrupt code through its port

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((DISTRIBUTED - INTERRUPT \text{ received}) \rightarrow \mathcal{X}(\text{send interrupt code})) \quad (64)$$

## 65 Requirement ID: 2033

**Status:** OK

**Description:** On receipt of the BROADCAST-CODE.request primitive the port shall send the broadcast code immediately after the character current has finished being sent

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((BROADCAST - CODE \text{ received}) \rightarrow ((Character \text{ sent}) \rightarrow \mathcal{X}(\text{broadcast code sent}))) \quad (65)$$

## 66 Requirement ID: 2034

**Status:** OK

**Description:** The effect on receipt of the DISCONNECT.indication primitive by the data link layer shall be for data link layer to move to the ErrorReset state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((DISCONNECT \text{ received}) \rightarrow \mathcal{X}(\text{ErrorReset state})) \quad (66)$$

## 67 Requirement ID: 2035

**Status:** OK

**Description:** The effect on receipt of the RECEIVE-ERROR.indication primitive by the data link layer shall be for data link layer to move to the ErrorReset state

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((RECEIVE - ERROR \text{ received}) \rightarrow \mathcal{X}(ErrorReset \text{ state})) \quad (67)$$

## 68 Requirement ID: 2036

**Status:** OK

**Description:** The function of the gotNull.indication primitive shall be to indicate to the data link layer that the first Null has been received after the receiver has been enabled

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((receiver \text{ enabled}) \rightarrow \Diamond(Null \text{ received})) \rightarrow (gotNull \text{ indicated})) \quad (68)$$

## 69 Requirement ID: 2037

**Status:** OK

**Description:** The gotNull.indication primitive shall be passed to the data link layer, when the first Null is received without any errors after the receiver has been enabled

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box(((receiver \text{ enabled}) \rightarrow (\Diamond(firstNull \text{ received}) \wedge \neg(error))) \rightarrow (gotNull \text{ passed})) \quad (69)$$

## 70 Requirement ID: 3001

**Status:** OK

**Description:** The Link state machine shall leave the ErrorReset state on the following condition: When the 6,4  $\mu$ s timer is elapsed and LinkDisable is de-asserted, move to the ErrorWait state.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((\Diamond_{6.4\mu s}(LinkDisable \text{ deasserted}) \rightarrow \mathcal{X}(ErrorWait \text{ state})) \quad (70)$$

## 71 Requirement ID: 3002

**Status:** OK

**Description:** When a SpaceWire output port has been transmitting characters and the link state machine enters the ErrorReset state (see clause 5.5.7.2), the data and strobe signals shall be reset with a delay between the reset of the strobe followed by data signal or reset of the data followed by strobe signal.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} &\Box((output \text{ port transmitting}) \wedge (ErrorReset \text{ state}) \rightarrow \\ &\quad \Diamond_{(0, delay)}(reset \text{ (data} \wedge \text{ Strobe)})) \\ &\quad delay := (t_2 - t_1) \vee (t_1 - t_2), t_1 \\ &\quad := strobe \text{ reset}, t_2 \\ &\quad := data \text{ reset} \end{aligned} \quad (71)$$



## 72 Requirement ID: 3003

**Status:** OK

**Description:** The Link state machine shall leave the ErrorWait state on one of the following conditions which are evaluated in the order given: 1. When LinkDisabled is asserted, move to the ErrorReset state. 2. When a disconnect occurs, move to the ErrorReset state. 3. When a parity error occurs, move to the ErrorReset state. 4. When an ESC error occurs, move to the ErrorReset state. 5. When an FCT, N-Char or broadcast code is received from the Encoding Layer, move to the ErrorReset state. 6. When the 12,8  $\mu$ s timer is elapsed, move to the Ready state.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((LinkDisabled) \rightarrow \\ & \quad \mathcal{X}(ErrorReset\ state)) \vee (Disconnect \rightarrow \mathcal{X}(ErrorReset\ state)) \\ & \quad \vee ((Parity\ Error) \rightarrow \mathcal{X}(ErrorReset\ state)) \vee ((ESC\ Error) \rightarrow \\ & \quad \mathcal{X}(ErrorReset\ state)) \vee (((FCT \vee NChar \vee broadcast\ code)\ received) \rightarrow \\ & \quad \mathcal{X}(ErrorReset\ state)) \vee (\Diamond_{12,8}(Ready\ state))) \end{aligned} \quad (72)$$

## 73 Requirement ID: 3004

**Status:** OK

**Description:** The delay between the reset of the Strobe signal and the Data signal shall be between 500 ns (the period of slowest permitted transmit bit rate, 2 Mbps) and the period of the fastest transmit time for the particular transmitter which is dependent upon implementation.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\Box((reset\ strobe\ signal) \rightarrow \Diamond_{(p, 500ns)}(reset\ data\ signal)) \quad (73)$$

## 74 Requirement ID: 3005

**Status:** OK

**Description:** A disconnect shall occur when the length of time since the last transition on either the Data or Strobe line is longer than 727 ns (8 cycles of 10 MHz clock + 10 ) and 1  $\mu$ s maximum (9 cycles of 10 MHz clock - 10 ).

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(\neg(\text{data transmitted})\mathcal{S}_{727\text{ ns}} \\ & ((\text{Data} \vee \text{Strobe}) \text{ transmitted}) \rightarrow (\text{disconnect})) \end{aligned} \quad (74)$$

## 75 Requirement ID: 3006

**Status:** OK

**Description:** A packet shall be regarded by a port as being stuck when the following conditions are all fulfilled: 1.It has started. 2.It has not yet ended. 3.The time since the last data character was sent from the input port to the output port is longer than the port time-out period.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((\text{packet started}) \wedge \neg(\text{packet ended}) \wedge (\neg(\text{data sent})\mathcal{S}_{(0,t)} \\ & (\text{last data sent}))) \rightarrow (\text{packet stuck})), \\ & t = \text{time} - \text{out period} \end{aligned} \quad (75)$$

## 76 Requirement ID: 3007

**Status:** OK

**Description:** The transition from the Started state to the ErrorReset state has “after 12,8 μs” in black (dark text) which is intentional as this transition is normal operation and not an error condition. It occurs when this end of the line is trying to start and the other end is disabled. The transition from the Connecting state to the ErrorReset state has “after 12,8 μs” in red (lighter text) because it is an error: although a Null has been received indicating that the other end of the link is active, no FCT is received within 12,8 μs so it is a fault.

**Logic:** MTLb

**Translation:** → INV (no), → LTL (yes), → STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((Started\ state) \rightarrow \Diamond_{(0,14.33\mu s)}(ErrorReset\ state)) \rightarrow (no\_error)), \\ & \Box((Connecting\ state) \rightarrow \\ & \Diamond_{(0,14.33\mu s)} \neg (FCT\ received) \rightarrow (ErrorReset\ state))) \end{aligned} \quad (76)$$

## 77 Requirement ID: 3008

**Status:** OK

**Description:** The Link state machine shall leave the Started state on one of the following conditions: 1. When LinkDisabled is asserted, move to the ErrorReset state. 2. When a disconnect occurs, move to the ErrorReset state. 3. When a parity error occurs, move to the ErrorReset state. 4. When an ESC error, move to the ErrorReset state. 5. When an FCT, N-Char or broadcast code is received from the Encoding Layer, move to the ErrorReset state. 6. When enable is asserted, at least one Null has been sent and gotNull is asserted, move to the Connecting state. 7. 12,8 μs after entering the Started state, move to the ErrorReset state.

**Logic:** MTLb

**Translation:** → INV (no), → LTL (yes), → STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(((LinkDisabled\ asserted) \rightarrow \mathcal{X}(Error\ Reset\ state)) \\ & \vee ((disconnect) \rightarrow \mathcal{X}(ErrorReset\ state)) \vee ((parity\ error) \rightarrow \mathcal{X}(ErrorReset\ state)) \\ & \vee ((ESC\ error) \rightarrow \mathcal{X}(ErrorReset\ state)) \\ & \vee (((FCT \vee NChar \vee broadcast\ code)received) \rightarrow \mathcal{X}(ErrorReset\ state)) \\ & \vee (((enable\ asserted) \wedge (one\ null\ sent) \wedge (gotNull\ asserted)) \rightarrow \mathcal{X}(Connecting\ state)) \\ & \vee (Started\ state) \rightarrow \Diamond_{12.8\mu s}(ErrorReset\ state)) \end{aligned} \quad (77)$$

## 78 Requirement ID: 3009

**Status:** OK

**Description:** The Link state machine shall leave the Connecting state on one of the following conditions:  
 1. When LinkDisabled is asserted, move to the ErrorReset state. 2. When a disconnect occurs, move to the ErrorReset state. 3. When a parity error occurs, move to the ErrorReset state. 4. When an ESC error occurs, move to the ErrorReset state. 5. When enable is asserted, at least one FCT has been sent and an FCT has been received (gotFCT asserted), move to the Run state. 6. When an N-Char or broadcast code is received from the Encoding Layer, move to the ErrorReset state. 7. 12,8  $\mu$ s after entering the Connecting state, move to the ErrorReset state.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned}
 & \Box(((LinkDisabled \text{ asserted}) \rightarrow \mathcal{X}(Error \ Reset \ state)) \\
 & \quad \vee ((disconnect) \rightarrow \mathcal{X}(ErrorReset \ state)) \vee ((parity \ error) \rightarrow \mathcal{X}(ErrorReset \ state)) \\
 & \quad \vee ((ESC \ error) \rightarrow \mathcal{X}(ErrorReset \ state)) \\
 & \quad \vee (((enable \ asserted) \wedge (one \ FCT \ sent) \wedge (one \ FCT \ received)) \rightarrow \mathcal{X}(Run \ state)) \\
 & \quad \vee (((NChar \ received) \vee (broadcast \ code \ received)) \rightarrow \mathcal{X}(ErrorReset \ state)) \\
 & \quad \vee ((Connecting \ state) \rightarrow \Diamond_{12.8\mu s}(ErrorReset \ state)))
 \end{aligned} \tag{78}$$

## 79 Requirement ID: 3010

**Status:** OK

**Description:** When operating in interrupt mode, there should be a minimum interval between a node sending one interrupt code with a particular value and sending the next interrupt with that same value which is greater than the maximum propagation time of an interrupt code across the network.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned}
 & \Box((interrupt \ mode) \rightarrow ((send \ interrupt \ code \ A) \rightarrow \Diamond_{(0,t)}(sent \ interrupt \ code \ A))), \\
 & \quad t \geq maximum \ propagation \ time
 \end{aligned} \tag{79}$$

## 80 Requirement ID: 3011

**Status:** OK

**Description:** If an interrupt is used in interrupt mode and the host system requests to send an interrupt too fast after the previous identical interrupt was sent, i.e. before the corresponding interrupt time-out timer in the routing switches has expired, the new interrupt code that the node sends is discarded by a router.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((\text{interrupt mode}) \rightarrow \\ & ((\text{interrupt code sent}) \rightarrow \\ & \Diamond_I(\text{interrupt code sent}) \rightarrow \\ & (\text{new interrupt code discard}))), \\ & I = \text{corresponding interrupt time - out} \end{aligned} \tag{80}$$

## 81 Requirement ID: 3012

**Status:** OK

**Description:** When operating in interrupt acknowledgement mode, there shall be a minimum interval between a node sending one interrupt code with a particular value and sending the next interrupt with that same value which is greater than the maximum propagation time of a interrupt code across the network and the maximum time for the interrupt acknowledgement code to be generated and return across the network.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((\text{interrupt ack mode}) \rightarrow \\ & ((\text{send interrupt code } A) \rightarrow \\ & \Diamond_{(0,t)}(\text{sent interrupt code } A))), \\ & t \geq \text{maximum propagation time} \end{aligned} \tag{81}$$

## 82 Requirement ID: 3013

**Status:** OK

**Description:** There shall be a delay between the interrupt code arriving and the interrupt acknowledgement being generated, which is greater than the propagation time of the interrupt code across the network.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((\text{interrupt code arriving}) \rightarrow \Diamond_{(0,t)}(\text{interrupt} \\ & \quad \text{acknowledgement})), \\ & \quad t \geq \text{propagation interrupt code time} \end{aligned} \tag{82}$$

## 83 Requirement ID: 3014

**Status:** OK

**Description:** The delay between the interrupt code arriving and the interrupt acknowledgement being generated shall be less than the maximum time determined for a node to generate an interrupt acknowledgement code

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((\text{interrupt code arriving}) \rightarrow \Diamond_{(0,t)}(\text{interrupt ack} \text{ generated}), \\ & \quad t \leq \text{maxinterrupt ack time} \end{aligned} \tag{83}$$

## 84 Requirement ID: 3015

**Status:** OK

**Description:** If the host system is too slow in sending an interrupt acknowledgement after a corresponding interrupt code has been received, i.e. the interrupt acknowledgement code is sent after the corresponding interrupt time-out timer in the routing switches has expired, the interrupt acknowledgement code is discarded by the first router.

**Logic:** MTLb

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (yes),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box((\text{interrupt code received}) \rightarrow \\ & \quad (\neg \Diamond_{(0,t)}(\text{interrupt ack received}) \\ & \quad \rightarrow (\text{Discard interrupt ack code}))), \\ & \quad t = \text{corresponding interrupt time - out} \end{aligned} \tag{84}$$

## 85 Requirement ID: 4001

**Status:** OK

**Description:** After a reset or disconnect (see clause 5.4.8) an output port shall start operating at a data signalling rate of  $10 \pm 1$  Mb/s.

**Logic:** STL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (conditional),  $\rightarrow$  MTLb (conditional)

**Formula:**

$$\Box((\text{reset} \vee \text{disconnect}) \rightarrow \mathcal{X}(9 \leq S_{data}(t) \leq 11)) \tag{85}$$

## 86 Requirement ID: 4002

**Status:** OK

**Description:** The SpaceWire output port shall operate at  $10 \pm 1$  Mb/s until set to operate at a different data signalling rate.

**Logic:** STL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (conditional),  $\rightarrow$  MTLb (conditional)

**Formula:**

$$\Box((9 \leq S_{data}(t) \leq 11) \mathcal{U} (\text{set different rate})) \tag{86}$$

## 87 Requirement ID: 4003

**Status:** OK

**Description:** Once in the Run state it is possible to change the output port data signalling rate from the initial data signalling rate to the intended operating data signalling rate

**Logic:** STL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  LTL (conditional),  $\rightarrow$  MTLb (conditional)

**Formula:**

$$\begin{aligned} & \Box((Run\ state) \rightarrow \\ & (S_{data}(t) := |x| \vee S_{data}(t) := |y|) \wedge \neg(S_{data}(t) := |x| \wedge S_{data}(t) := |y|)), \\ & x := initial\ data\ signaling\ rate \\ & y := intended\ data\ signaling\ rate \end{aligned} \tag{87}$$

## 88 Requirement ID: 5001

**Status:** OK

**Description:** Received characters and control codes shall be passed to the data link layer in the order in which they are received.

**Logic:** LTL

**Translation:**  $\rightarrow$  INV (no),  $\rightarrow$  MTLb (no),  $\rightarrow$  STL (conditional)

**Formula:**

$$\begin{aligned} & \Box(\bigwedge_{j=1}^n \bigwedge_{i=1}^n (Receive_i \rightarrow \mathcal{X} \Diamond Receive_j) \rightarrow (Send_i \wedge \mathcal{X} Send_j)), \\ & n = number\ of\ characters \end{aligned} \tag{88}$$



## 89 Requirement ID: 5002

**Status:** NOK

**Description:** If the host system tries to send an interrupt acknowledgement too soon after a corresponding interrupt code has been received, i.e. before the interrupt code has propagated across the entire network, the result is indeterminate for that specific interrupt. The new interrupt acknowledgement code that the node sends can either be discarded by a router, or repeatedly propagated through the network if the network has circular connections.

**Logic:** ?

**Translation:**  $\rightarrow \text{INV (no)}, \rightarrow \text{LTL (no)}, \rightarrow \text{MTLb (no)}, \rightarrow \text{STL (no)}$

**Formula:**

... (89)