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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

$$\begin{array}{l} \Box((encoding\ layer\ disconnected) \rightarrow (Disconnect\ flag)) \\ \Box((receive\ error) \rightarrow (receive\ error\ flag)) \\ \Box((firstNull\ \ received) \rightarrow (gotNull\ \ flag)) \end{array} \tag{4}$$

5 Requirement ID: 1005

Status: OK

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

Logic: INV

Formula:

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

6 Requirement ID: 1006

Status: OK

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

Logic: INV

Formula:

$$\Box((receiver\ \ enabled)\ \ \rightarrow\ \ (Null\ \ detection\ \ enabled)) \tag{6}$$

7 Requirement ID: 1007

Status: OK

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

Logic: INV

Formula:

$$\Box((RX\ \ enable\ \ flag\ \ de\ \ asserted) \rightarrow (gotNull\ \ condition\ \ cleared)) \tag{7}$$

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

Formula:

$$\Box(((receiver\ \ enabled) \wedge (gotNull\ \ asserted)) \rightarrow (Parity\ \ enable)) \tag{8}$$

9 Requirement ID: 1009

Status: OK

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

Logic: INV

Formula:

$$\Box((gotNull\ \ asserted) \rightarrow (ESC\ \ error\ \ detection\ \ enable)) \tag{9}$$

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

Formula:

$$\Box((ready\ \ to\ \ accept\ \ N-Char) \rightarrow\ \ (indicate\ \ accept\ \ N-Char)) \tag{10}$$

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

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

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

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

Formula:

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

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

Formula:

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

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

Formula:

$$\begin{array}{l}\Box((FCT\ passed) \rightarrow (receive\ credit := receive\ credit + 8)) \\ \Box((Nchar\ \ received) \rightarrow (receive\ \ credit := receive\ \ credit - 1))\end{array} \tag{19}$$

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

Formula:

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

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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 µs timer on entering the ErrorReset state. De-asserting the Receive Enable control flag causes the Encoding layer to clear the gotNull condition

Logic: INV

Formula:

$$\begin{aligned}&\Box((ErrorReset\ state) \rightarrow\\&((De-assert\ control\ \ flag) \wedge (De-assert\ Receive\ enable\ \ flag)\\&\wedge (set\ transnit\ credit{:=}0) \wedge (set\ receive\ credit := 0) \wedge (clear\ \ gotFCT)\\&\wedge (start\ \ timer_{6.4})))\end{aligned} \tag{27}$$

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

Formula:

$$\Box((simultaneous\ \ transition) \rightarrow (no\ \ lock - up)) \tag{28}$$

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

Formula:

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

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 µs 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

Formula:

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

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

Formula:

$$\Box((input\ port\ waiting) \leftrightarrow (output\ port\ waiting)) \tag{31}$$

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

Formula:

$$\Box((pocket\ transmission\ finished) \rightarrow (accept\ pocket\ available)) \tag{32}$$

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

37 Requirement ID: 2005

Status: OK

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

Logic: LTL

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

42 Requirement ID: 2010

Status: OK

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

Logic: LTL

Formula:

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

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

Formula:

$$\Box((Port\ Reset\ asserted) \rightarrow ((TX\ \ FIFO\ cleared)\ \wedge\ (RX\ \ FIFO\ cleared) \wedge \mathcal{X}(ErrorReset\ state))) \tag{43}$$

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

Formula:

$$\Box((transmit\ credit := 0) \rightarrow ((stop\ sending\ Nchar)\ \mathcal{U}(transmit\ credit := 8)), \Box((transmit\ \ credit := 0) \rightarrow (send\ \ (FCT \wedge Null \wedge codes))) \tag{44}$$

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

$$\square(((LinkDisabled\ asserted) \wedge (Ready\ state)) \rightarrow \mathcal{X}(\square(Ready\ state))) \tag{50}$$

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

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

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

Formula:

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

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

Formula:

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

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

Formula:

$$\begin{aligned}&\Box(\neg(N-Char\ accepted) \rightarrow \\ &((input\ port\ wait) \wedge (\neg(N-Char\ sent)\ \mathcal{U}(multicast\ output\ ports\ ready))))\end{aligned} \tag{63}$$

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

Formula:

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

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

Formula:

$$\begin{aligned}&\Box((BROADCAST-CODE\ received) \rightarrow \\ &\quad((Character\ \ sent) \rightarrow \mathcal{X}(broadcast\ \ code\ \ sent)))\end{aligned} \tag{65}$$

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

$$\begin{aligned}&\Box(((receiver\ enabled) \rightarrow \\ &\quad(\Diamond(firstNull\ received) \wedge \neg(error))) \rightarrow (gotNull\ passed))\end{aligned} \tag{69}$$

70 Requirement ID: 3001

Status: OK

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

Logic: MTLb

Formula:

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

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

Formula:

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

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 µs timer is elapsed, move to the Ready state.

Logic: MTLb

Formula:

$$\begin{aligned}&\Box(((LinkDisabled) \rightarrow \\ &\mathcal{X}(ErrorReset\ state)) \vee (Disconnect \rightarrow \mathcal{X}(ErrorReset\ state)) \\ &\vee ((Parity\ Error) \rightarrow \mathcal{X}(ErrorReset\ state)) \vee ((ESC\ Error) \rightarrow \\ &\mathcal{X}(ErroeReset\ state)) \vee (((FCT \vee NChar \vee broadcast\ code)\ received) \rightarrow \\ &\mathcal{X}(ErrorReset\ state)) \vee (\Diamond_{12.8}(Ready\ state)))\end{aligned} \tag{72}$$

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

Formula:

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

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 µs maximum (9 cycles of 10 MHz clock - 10).

Logic: MTLb

Formula:

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

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

Formula:

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

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

Formula:

$$\begin{aligned}&\Box(((Started\ state) \rightarrow \Diamond_{(0,14.33\mu s)}(ErrorReset\ state)) \rightarrow (no_error)),\\&\quad\Box((Connecting\ state) \rightarrow\\&\quad\Diamond_{(0,14.33\mu s)}\neg(FCT\ received) \rightarrow (ErrorReset\ state)))\end{aligned} \tag{76}$$

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

Formula:

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

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 µs after entering the Connecting state, move to the ErrorReset state.

Logic: MTLb

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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

Formula:

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

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 ±1 Mb/s.

Logic: STL

Formula:

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

86 Requirement ID: 4002

Status: OK

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

Logic: STL

Formula:

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

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

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: NOK

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

Logic: ?

Formula:

... (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: ?

Formula:

... (89)