Accredited Standards Committee
NCITS, National Committee for Information Technology Standards

Doc: FCW/98wcvr02
Date: February 16, 1998
Project:
Ref Doc.:
Reply to: John Scheible

To: T11 Membership
From: John Scheible

Subject: FCW 2/98 mailing cover letter

This mailing covers the February 9-13, 1998 FCW meeting in San Diego, CA. This mailing is available electronically as:

* 98wpdf02.zip = (1.13 MB) The PDF files of the mailing.
* 98wsrc02.zip = (0.32 MB) The source files (when available). This is not a complete set of the mailing.
* 98wpdf02.pdf = A single PDF of the mailing, this lacks the HTML linking of the .zip file.

These files are available on the SCSI FTP site (ftp://ftp.symbios.com/pub/standards/io/t11/pub/fc/futures/).

If the electronic form of 98wpdf02.zip file is unzipped into a directory, and the file 98wlml02.htm can be run to access all files via HTML code. Note that Adobe acrobat 3.0 or higher is needed to access the .PDF files.

Filename	Description	Author	Project	Pages	Page #
98wcvr02	02/1998 FCW mailing cover letter	John Scheible	ADMIN	1	1
98wmtg05	5/98 meeting announcement-Colorado Springs CO	John Lohmeyer	ADMIN	2	3
98wmtg04	4/98 meeting announcement - Palm Springs CA	Jeff Stai	ADMIN	1	5
98wmtg03	3/98 meeting announcement - San Diego CA	Skip Jones	ADMIN	1	7
98w133r0	Error detection mechanisms	Bob-Snively	PH	3	9
98w132r0	FC-AV - IP on Streams	Bent Stovhase	AV	4	13
98w131r0	T11.3 presentation	Rich Toburek	ADMIN	12	19
98w130r0	FCLC test period (Jan. 12 - 16 1998) test results	Dal Allan	AL2	7	29
98w129r0	FC-AL Loop Inititialization	Dal Allan	AL2	6	37
98w128r0	AL-2 Report	Horst Truestedt	AL2	1	43
98w127r0	FC-AL-2 ARB Detection	Ed Gardner	AL2	5	45
98w126r0	FCLC Jan 1998 interoperability testing results	Barry Reinhold	AL2	18	51
98w125r0	FCP/Class 2 Public Loop operation - Streaming	Dave Peterson	TAPE	12	69
98w124r0	RIP version 1.0	Dave Ford	AL3	34	81
98w123r0	FC-AL-3 2D Link Level Flow Control	Bent Stovhase	AL3	7	115
98w122r0	FC-AL-3 2D Loop Initialization	Bent Stovhase	AL3	11	123
98w121r0	FC-AL-3 2D Loop Identication	Bent Stovhase	AL3	12	135
98w119r0	New link error reporting	Jim Coomes	AL2	2	147
98w118r1	Multiple Circuit Mode	Jeff Williams	AL3	10	149
98w101r0	Jan 1998 FCW meeting minutes	Dal Allan	MINUTES	2	159
				Total	160

Sincerely,

John Scheible
Voice: (512) 823-8208
FAX: (512) 838-8378
Email: jpscheible@aol.com

Meeting Announcement

T10 Technical Committee (and friends)

MAY 4-8, 1998

HOSTED BY: Symbios Logic Inc.

LOCATION: Double Tree World Arena Hotel
1775 E. Cheyenne Mountain Blvd.
Colorado Springs, Colorado 80906
Phone: (719) 576-8900 Fax: (719) 576-4450

ROOM RATE: $110.00 + tax (single) $125.00 + tax (double)

GROUP NAME: **T10**

CUT-OFF DATE: **April 3, 1998**

DRIVING DIRECTIONS:

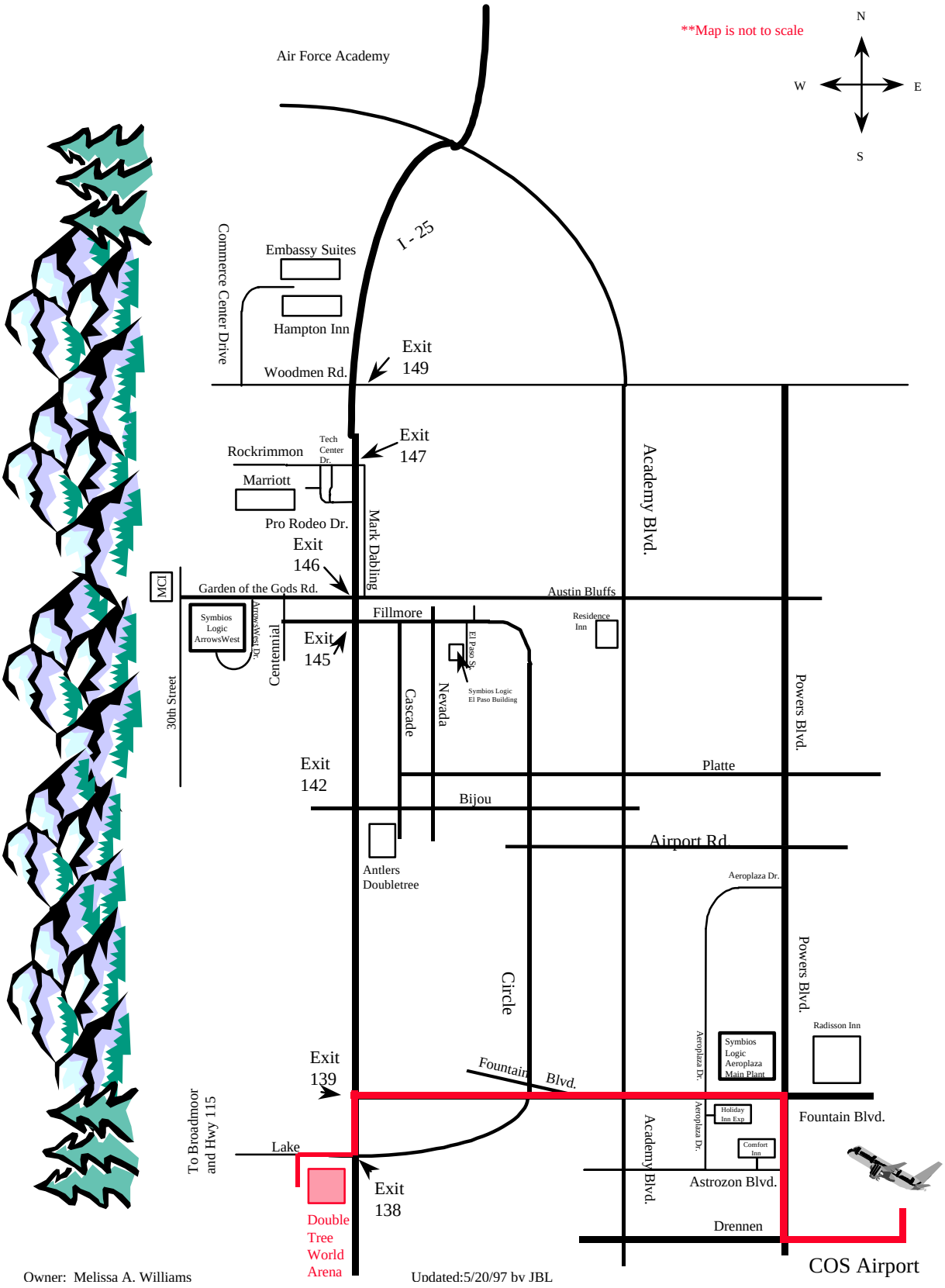
The Double Tree World Arena Hotel is located on the South side of Colorado Springs at I-25 (exit 138) and Circle Drive. A courtesy van is available from the Colorado Springs airport; call the hotel from the courtesy phone in baggage claim.

From the Colorado Springs airport, exit the airport (on Drennan Rd.). At the first stop light, turn right onto Powers Blvd. Proceed North to Fountain Blvd and turn left onto Fountain Blvd. Follow Fountain Blvd. West and bear left onto US-24 (which becomes an expressway). Continue West on US-24 to I-25. Go South (left) on I-25. Take the first exit (Circle Dr. exit 138). Turn right, then immediately left onto Cheyenne Mountain Blvd.

If you elect to drive from the Denver International Airport, exit the airport on Pena Blvd. and take I-70 West (about 2 miles) to I-225 South. Follow I-225 South around Denver (about 12 miles) to I-25. Follow I-25 South about 62 miles to Circle Dr. (exit 138). Turn right, then immediately left onto Cheyenne Mountain Blvd.

Weather: Anything goes in early May -- last year we had 22” of snow in late April. It most likely will be pleasant, but I’d recommend bringing a light jacket.

HOST CONTACT: John Lohmeyer (719-533-7560)



Owner: Melissa A. Williams

Updated:5/20/97 by JBL

Meeting Announcement
Accredited Standards Committee T11 Technical Committee
April 20-24, 1998

Hyatt Regency Suites Palm Springs
285 North Palm Canyon Drive
Palm Springs, CA 92262
760-322-9000 or 800-233-1234

Group Name: Brocade
Room Rate: $122/night, includes tax and parking
Your Host: Brocade Communications Systems
Cutoff Date: March 20, 1998

Brocade Communications Systems is pleased to be able to host the ASC-T11 TC meetings, to be held the week of April 20-24, 1998 in Palm Springs, California. This location is shuttle-close to Palm Springs Airport, and is freeway-close to Ontario Airport. If you cannot arrange a suitable connection to Palm Springs, I recommend using Ontario and driving into Palm Springs, a leisurely, scenic and slightly jurassic 60-90 minute drive depending on the lead content of your right foot (also, try and avoid the evening rush-hour or the first part of the drive is Real Ugly).

Some additional background may be in order. The Hyatt is located in the Palm Springs Village area. It is within walking distance of many shops and restaurants, and there is a casino in town, run by the local Native Americans (Agua Caliente tribe). With a little help from El Nino, the desert wildflowers will still be in bloom. There may perhaps be some of you who golf. If so, let the hotel know when you check in. They can set up tee times for you at Rancho Mirage Country Club, with special discount green fees. Other attractions include the Living Desert Museum, Desert Adventures Jeep Tours, the Indian canyons which contain the springs (yes, there are really springs!), Joshua Tree National Park, and the Palm Springs Aerial Tramway, which soars to 8516 feet suspended over jagged rocks. Enjoy!

Directions: Palm Springs is well served with jet service by several airlines to Palm Springs airport. To get to the hotel from Palm Springs Airport, you can call the hotel for a complementary shuttle (it's only three miles). To drive from the airport, exit the rental car lot by turning left on El Cielo Rd. At the stop sign turn right on Tahquitz. Go a few blocks and turn right on Indian Ave. This is a four lane, one way road and you need to get immediately over to the left. Turn left on Amado, and another left onto North Palm Canyon Drive. Hyatt Regency Suites is on the right at the corner of North Palm Canyon Drive and Amado Road. There is an entrance to an underground parking garage, or you may avail yourself of the fine valet service.

If you fly into Ontario airport, go north on Vineyard to I-10. Head east a long time on I-10 (do not be alarmed by the dinosaurs on your left) to state highway 111. Remain on highway 111 for about 11 miles (it will become North Palm Canyon Drive). Proceed to Amado Road and you're there.

You lucky few travelling from Orange County can hop state highway 91 and take it all the way out to state highway 60. Stay on 60 until it merges with I-10, proceeding as described above. San Diegans may use I-215 to connect to highway 60.

Kinko's: The closest Kinko's is in Cathedral City, about 7.5 miles from the hotel (sorry). Exit the hotel and go south on Palm Canyon Drive to Ramon Rd. Turn left and travel about 3 miles. Turn right on Gene Autry Trail and travel about 4 miles; note that the name of the road changes along the way and it veers southeast. Turn left on Date Palm Dr., it will be on your left; 35-325 Date Palm Dr., 760-770-2500. However, there is a Staples within walking distance near the corner of Palm Canyon and Ramon (350 South Palm Canyon, 760-778-4952) which is open until 9PM weekdays. This may meet your needs, unless you are like some people who do their copying at 2AM!

Web sites: Check out http://www.palm-springs.org/ and http://www.palmsprings.com/ and http://www.ceol.com/digitaldesert/ and http://www.nps.gov/jotr/ to start with.

Needless to say, if you have any problems, please contact me. I will beat up the appropriate people!

Jeff Stai
Brocade Communications Systems
714-455-2908 stai@brocadecomm.com
(Sharon Dudley 714-455-2926)

MEETING ANNOUNCEMENT

(T10, STA)

March 16 - 20, 1998

HOSTED BY

QLogic

LOCATION

Hyatt Islandia
(Mission Bay)
1441 Quivira Road, San Diego, CA 92109
Phone: (619) 224-1234
Fax: (619) 224-0348

ROOM RATE

$123.00 (single) or $143.00 (double), plus 10.5% tax

GROUP NAME

American National Standard Institute

CUT-OFF DATE

February 13, 1998

DRIVING DIRECTIONS

From San Diego Airport - I5 North to 8 West to Mission Bay Drive exit.
Right at exit and follow West Mission Bay Drive through the loop.
Left at light to Quivira Road, and an immediate right to front
driveway of the hotel.

HOST CONTACT

Skip Jones (714) 668-5058
sk_jones@qlc.com
Jill Huffman (714) 668-5059
j_huffman@qlc.com

www.qlc.com

Sun Microsystems

901 San Antonio Rd

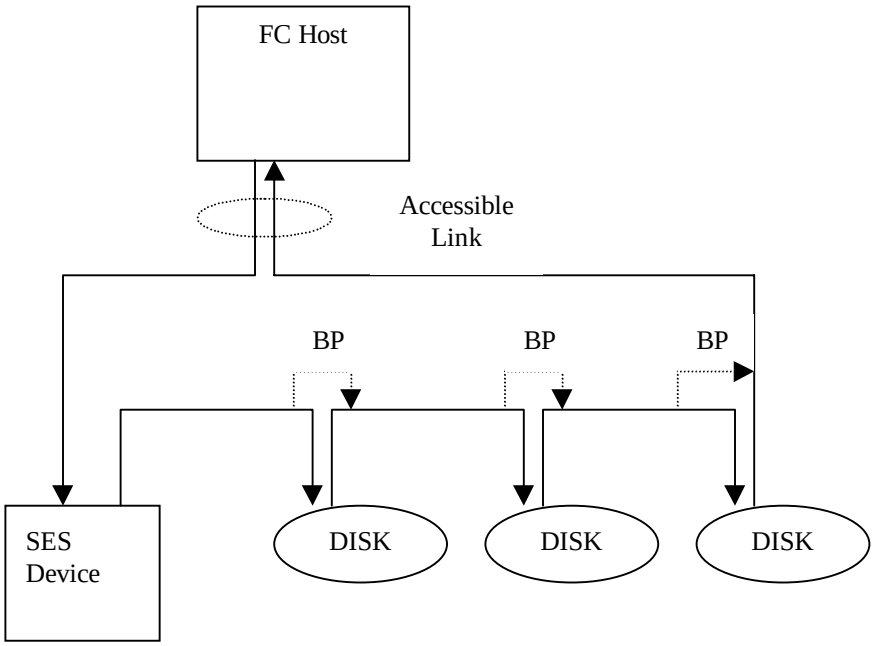
Palo Alto, CA 94303-4900

Doc:	T11/97-XXXr0
Date:	February 12, 1998
Project:	T11.2
Ref Doc.:	none
Reply to:	Bob Snively

To: T11 Membership
From: Bob Snively

Subject: Error detection mechanisms

This information is provided to T11.2 to provide the requested information about various error detection and presentation mechanisms provided in the higher levels of the Fibre Channel. An example of a typical Fibre Channel configuration is shown in the following figure. Note that the SES (SCSI Enclosure Services) device and the disks are typically placed in an enclosure such that data links are not accessible by anyone except the circuit designers. Mechanical interruption of the circuits at the printed wire or circuit level is typically required to study those internal links. The only link accessible for direct study and jitter injection is typically the link between the host and the enclosure. Bypass circuits in the enclosure are typically managed by the SES device, which may use both Fibre Channel and out-of-band communication (typically Ethernet) to control and sense the states of the bypass circuits.



There are at least four mechanisms defined in upper layers of the Fibre Channel standards that can be used to provide helpful error counting and error monitoring functions. Those mechanisms are described in the following paragraphs.

SCSI Command Failure error detection mechanism

Most SCSI drivers (and the corresponding TCP/IP programs) have mechanisms to transmit arbitrary data patterns across the link. These patterns can be selected very carefully to exacerbate known marginal jitter conditions. By carefully selecting the type of transmission (read or write) and the initiator and target selected, various combinations of the link elements of the complete loop can be verified for proper operation in the established jitter environment.

A SCSI command failure will have a very high probability that failures are associated with the data frames, since the number of bits transmitted during the data frames is much larger than the number of bits transmitted in other frames. SCSI command failures typically are actually counts of commands that have identified one or more missing frames associated with CRC failures or 8B10B code violations. The error messages will typically be either timeout indications or incomplete sequence indications.

The actual presentation mechanism for the SCSI failure is operating system dependent. The programs that are used to generate the test sequences are typically vendor specific benchmark and verification programs that exploit standard SCSI behavior.

SCSI Enclosure Services management

The SCSI enclosure services management and similar out-of-band manipulation of the loop configuration allows the testing of complete configurations by successive elimination or inclusion of links. The actual detection of the failures can be by any of the error detection mechanisms.

Intelligent hub

Intelligent hubs may have various types of error detection circuitry, including verification of loss of synchronization, loss of signal, detection of 8B10B encoding violations, and other errors. The actual error detection provided by a particular intelligent hub is vendor specific. Those errors are typically available through various types of communication links. The links may be in-band TCP/IP access to MIBs meeting SNMP requirements. The links may include out-of-band RS-232 links or Ethernet links accessing similar error information. The links may include out-of-band communication through a display panel or indicators on the intelligent hub. This form of error communication is helpful in isolating the behavior of individual links in a large configuration.

Extended Link Service interrogation of error counters

This is the most generic form of presenting detected error counts. The Extended Link Service (ELS) called “Read Link Error Status Block” (RLS) is defined in the FC-PH Fibre Channel standard in section 21.4.11. The ELS requests a payload containing an identifying code and a Link Error Status Block (LESB) which is defined in FC-PH section 29.8. The following information is contained in the ELS accept payload.

RLS Accept Payload, including LESB definition

Word (4 bytes)	Word of LSB	Function
0	n.a.	’02 00 00 00’ RLS Accept Payload code
1	0	Link Failure Count
2	1	Loss of Synch Count
3	2	Loss of Signal Count
4	3	Primitive Sequence Protocol Error
5	4	Invalid Transmission Word
6	5	Invalid CRC Count

The detailed definition of each of the counting mechanisms is vendor specific. The choice of which counters are implemented for a given node is vendor specific. Many low-cost devices that are normally not exposed to accessible links

do not implement the RLS Extended Link Service. The details of counting these errors are vendor specific. In particular, a short burst of errors is often related to a single error incident and may be counted as a single failure.

Loop attached devices can only count CRC if they are OPEN or OPENED. Devices in the monitoring state do not count CRC errors.

None of these 32-bit counters are reset. The error counter must be read and compared with the previous reading to determine the actual number of counted events since the last reading. The counter wraps to 0 after it reaches the value of 'FF FF FF FF'h.

Summary

The error detection and reporting capabilities are adequate to perform verification of proper operation of links in a loop or point-to-point environment. While most link elements are not available for the convenient insertion of measured amounts of jitter, error rates for those that are available can be determined reasonably accurately with the error information available at the system or link level.

Sincerely,

Bob Snively
Sun Microsystems
650-786-6694
bob.snively@sun.com

Serial Communications

FC–AV

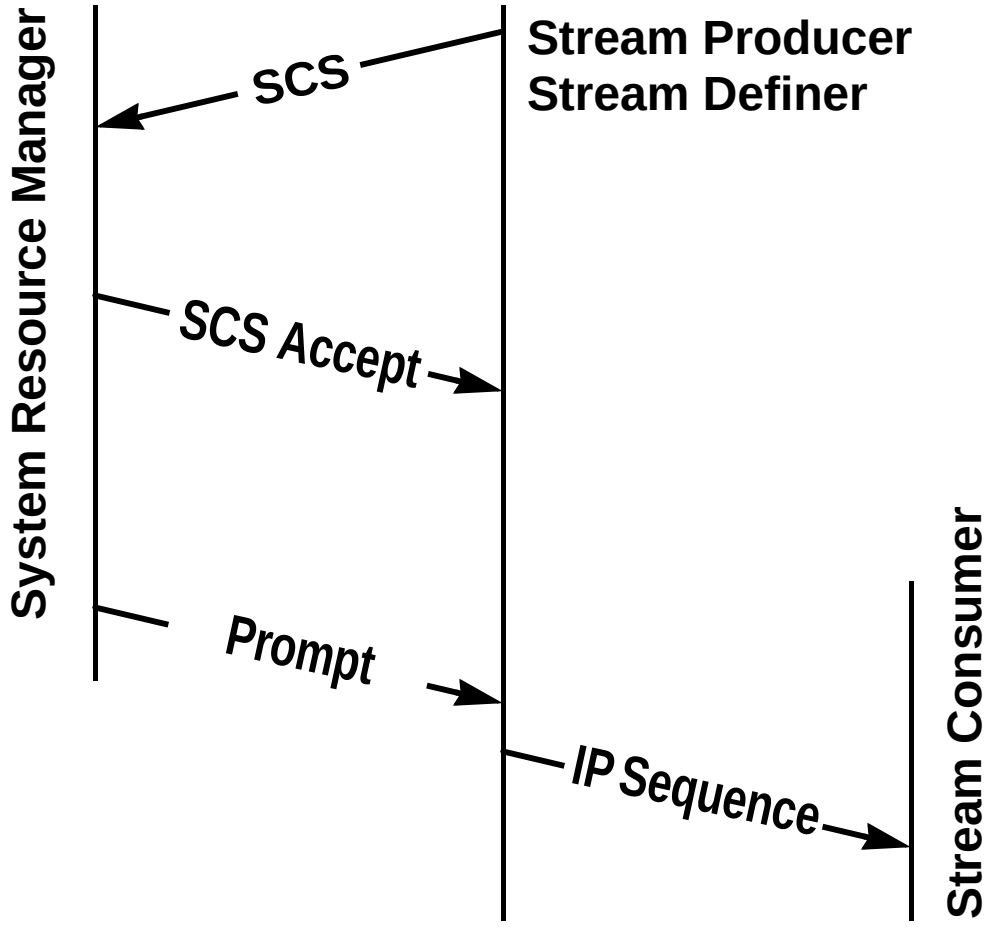
IP on Streams

Bent Støvhase (bent@inforamp.net)

FC–AV – IP on Streams

Class 2 & 3 Stream Environment

- All Fibre Channel traffic is required to be carried in a Stream, when used in a Class 2/3 Stream environment
  - IP traffic should be carried in an Asynchronous Stream
  - IP traffic may be carried on an Available Bit Rate Stream
  - IP should not be carried on a Real–Time Stream
    - Except for IP traffic integral to the Real–Time Stream
- IP flow example (Sequence level)



FC–AV – IP on Streams

IP flow example (Continued)

- The Stream Producer, as the Stream Definer, sends a Reserve Asynchronous Stream (RAS) request to the System Resource Manager (SRM) as part of a SCS (Stream Control Services) Extended Link Service command, when it desire to transfer an IP IU (Sequence) to a given Stream Consumer
  - A single RAS request may carry multiple IP IU requests for a single destination
- The SRM responds back with an acceptance (Accept Reserve Asynchronous Stream [ARAS]) reply in the SCS Accept Sequence
  - The ARAS response may be returned immediately, i.e in the first SCS Accept, or in any subsequent SCS Accept Sequences
  - The acceptance reply identify the Time Slot for Stream #0 (The Asynchronous Stream) at which time the IP IU is to be transferred
- The Prompt Extended Link Service Sequence broadcast, every 1/64 sec, by the SRM identify the current Time Slot value for each of the up to 256 Streams
- The Stream Producer sends the IP IU when the Time Slot value for the Asynchronous Stream (0) arrive
  - If multiple transferred are assigned to the same Time Slot, then they are serviced (transmitted) in the order assigned by the SRM, i.e. the order of the ARAS replies

FC–AV – IP on StreamsFC–AV – IP on Streams

Class 4 Stream Environment

- A mixture of Class 4 Streams and non–Streamed Class 2 or 3 may be carried in a Class 4 Stream environment
  - IP traffic integral to the Class 4 Stream may be carried in a Component Stream
    - A separate Component Stream should be defined for this purpose
  - IP traffic not integral to Stream operation should be carried in either Class 2, 3 or non–Streamed Class 4
- Class 1, 2, 3 or 6 Streams can not be used in a Class 4 Stream environment

T11.3

- Interconnect Schemes
  - Rich Taborek - G2 Networks - Chair
  - Jeff Stai - Brocade - Vice Chair
  - Secretary?

T11.3 Projects

- Focused projects (i.e. no 'bucket' projects)
  - Market driven
- Project leaders and editors required
  - in order to initiate and continue projects
  - should be from different companies
  - secretary (third individual) desirable

T11.3 Project Documents

- State machines required
  - State machines have precedence;
  - then tables; then figures.
  - Text is descriptive
- PICS required
- Executable code desirable
  - (e.g. C, Verilog, SES/workbench, etc.)
- Simulation of state machines is desirable

T11.3 Project Documents

- Agendas, presentations
  - fixed by midnight Wednesday prior to meeting
- Follow T11 electronic document procedures

T11.3 Comment Process

- Formal comment process starting at LB
  - Comments must include suggested remedy
  - Editor/WG writes proposed response
  - WG discusses and writes formal response
  - Editor updates document per proposed response

FC-AL-2

- **Hottest** T11. 3 Project
- John Scheible - IBM - Leader
- Horst Truestedt - ENDL - Editor
- Secretary?

FC-AL-3

- John Scheible - IBM - Leader
- Jeff Williams - Western Digital - Editor
- Secretary?

FC-SL

- Scott Darnell - Raytheon - Leader/Editor
- Joe Sigfried - Raytheon - Alternate (PMA)
- Secretary?

FC-SW-2

- Bill Martin - Gadzoox Networks - Leader
  - pending management approval
- Jeff Stai - Brocade - Editor
- Secretary?

FC-TAPE (FC-PLDA-2)

- Dale LaFollette - StorageTek - Leader
- Dave Peterson - StorageTek - Editor
- Steward Wyatt - HP - Secretary
  - pending management approval

Future Potential Projects

- FC-FLA-2

T11.4 FC-VIA

- Dave Ford - Orca Systems - Leader
- Rich Taborek - G2 Networks - Editor
- Charles Monia - Adaptec - Alternate Editor
  - pending management approval
- Secretary?

To: FC WG
From: Dal Allan

The following information is a summary of the test results (suitably sanitized to remove any company identification) at the FCLC interoperability testing in January.

FCLC TEST PERIOD (JAN. 12 – 16, 1998)

TEST RESULTS FORM

Phase I: Loop Stability

L_1 and L_2 are L_Ports. All combinations of L_Ports are to be tested.

Test 1:
Setup: L_1 and L_2 are powered off and connected in a Loop.
Procedure: Power on L_1 and wait until it is fully powered up. Power on L_2. Verify that the Loop is initialized.

Test 2:
Setup: Same as in Test 1.
Procedure: Power on L_2 and wait until it is fully powered up. Power on L_1. Verify that the Loop is initialized.

Test 3:
Setup: L_1 and L_2 are powered up, connected in a Loop and the Loop is initialized.
Procedure: Disconnect L_1 from the Loop and then reconnect it. Verify proper recovery of the Loop.

Test 4:
Setup: Same as in Test 3.
Procedure: Disconnect L_2 from the Loop and then reconnect it. Verify proper recovery of the Loop.

Test 5:
Setup: Same as in Test 3.
Procedure: Disconnect the receiver of L_1. Power cycle L_1. Reconnect the receiver after L_1 is fully powered up. Verify proper recovery of the Loop.

Test 6:
Setup: Same as in Test 3.
Procedure: Disconnect the receiver of L_2. Power cycle L_2. Reconnect the receiver after L_2 is fully powered up. Verify proper recovery of the Loop.

L_1	L_2	Hub	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
R	N	H	Pass	Pass	Pass	Pass	Pass	Pass
S	D		FAIL 1	FAIL 1	N/T	N/T	N/T	N/T
N	A		Pass	Pass	Pass	FAIL 2	Pass	Pass
N	F	U	Pass	Pass	Pass	FAIL 3	Pass	FAIL 3
J	R	H	Pass	FAIL 4	Pass	Pass	Pass	Pass
P	K	H	Pass	Pass	Pass	Pass	Pass	FAIL 5
S	D	H	Pass	Pass	Pass	Pass	Pass	FAIL 6
K	N	U	Pass	Pass	Pass	Pass	FAIL 5	Pass
A	T	H	FAIL 7	Pass	Pass	Pass	Pass	FAIL 7
D	P	H	Pass	Pass	Pass	Pass	FAIL 8	Pass
K	M	H	Pass	Pass	Pass	Pass	FAIL 5	Pass

K	D	H	Pass	Pass	Pass	Pass	FAIL 5	FAIL 9
J	P	B	FAIL 10	Pass	Pass	Pass	Pass	Pass
P	M		Pass	Pass	Pass	Pass	FAIL 11	Pass
S	T	H	Pass	Pass	Pass	Pass	Pass	FAIL 7
F	K		Pass	FAIL 12	Pass	Pass	FAIL 13	FAIL 14
J	T		Pass	Pass	Pass	Pass	FAIL 15	FAIL 7
T	K		Pass	Pass	Pass	Pass	FAIL 7	FAIL 5
T	P	U	Pass	Pass	Pass	Pass	FAIL 16	Pass
F	M	U	Pass	Pass	Pass	Pass	Pass	FAIL 17
T	D	U	Pass	Pass	Pass	Pass	FAIL 17	FAIL 18
E	K	H	Pass 19	Pass 19	FAIL 19	FAIL 19	FAIL 19	FAIL 19
G	K	H	Pass	Pass	Pass	Pass	Pass	FAIL 5
G	D	H	Pass 20	Pass	Pass	Pass	Pass	FAIL 21
J	G	H	Pass	Pass	Pass	Pass	FAIL 22	Pass
G	M	U	Pass	Pass	FAIL 22	Pass	Pass	FAIL 22
E	S	H	Pass	Pass	Pass	Pass	FAIL 23	Pass
T	G		Pass	Pass	Pass	Pass	FAIL 7	Pass
E	T	H	FAIL 24	FAIL 24	N/T	N/T	N/T	N/T
E	F	H	FAIL 25	FAIL 25	N/T	N/T	N/T	N/T
L	G		FAIL 22	FAIL 26	FAIL	FAIL	FAIL	FAIL
E	R		FAIL 22	FAIL 22	N/T	N/T	N/T	N/T
T	F		Pass	Pass	Pass	Pass	FAIL 7	Pass
S	K	B	FAIL 1	FAIL 26	N/T	N/T	N/T	N/T
G	F	U	Pass	FAIL 22	Pass	Pass	Pass	Pass
G	E		Pass	Pass	Pass	Pass	Pass	Pass
L	A		Pass	Pass	Pass	Pass	Pass	Pass
L	E	H	FAIL 22	FAIL 22	N/T	N/T	N/T	N/T
S	M	H	Pass	Pass	Pass	Pass	Pass	Pass
S	F		Pass	Pass	Pass	Pass	Pass	Pass
F	D		Pass	Pass	Pass	Pass	Pass	FAIL
N	L		Pass	Pass	Pass	Pass	Pass	Pass
E	J		Pass	Pass	Pass	Pass	Pass	Pass
D	Q		Pass	Pass	Pass	Pass	Pass	Pass
N	G	H	Pass	Pass	Pass	Pass	Pass	Pass
P	A	H	Pass	Pass	Pass	Pass	Pass	Pass
P	R		Pass	Pass	Pass	Pass	Pass	Pass
P	L		Pass	Pass	Pass	Pass	Pass	Pass
N	P		Pass	Pass	Pass	Pass	Pass	Pass
K	Q	H	Pass	Pass	Pass	Pass	FAIL 5	Pass
Q	M		Pass	Pass	Pass	Pass	Pass	Pass
S	Q		Pass	Pass	Pass	Pass	Pass	Pass
Q	N	U	Pass	Pass	FAIL 28	Pass	FAIL 28	FAIL 28
G	P	H	Pass	Pass	Pass	Pass	Pass	Pass
Q	P	U	Pass	Pass	Pass	Pass	Pass	Pass
S	P	H	Pass	Pass	Pass	Pass	Pass	Pass
D	M	H	Pass	Pass	Pass	Pass	FAIL 18	Pass
F	Q	U	FAIL 29	FAIL 14	Pass	Pass	FAIL 18	Pass
L	R		FAIL 22	FAIL 22	N/T	N/T	N/T	N/T
R	A	U	Pass	FAIL 22	Pass	Pass	FAIL 22	Pass
R	T	H	Pass	FAIL 22	FAIL 22	FAIL 22	FAIL 22	FAIL 8
N	M	H	FAIL 8	FAIL 8	N/T	N/T	N/T	N/T

Comments:

1. S transmitted LIP(F8,F7) forever.
2. N transmitted various LIPs e.g., LIP(F8,01), LIP(F1,01), LIP(F3,01).
3. N transmitted only LIP(F8,xx), while F transmitted LIP(F8,xx)/Idle.
4. After waiting 8 minutes, only Idle/garbage seen from J.
5. No light was detected from K.
6. Only garbage was detected from D.
7. No light was detected from T.
8. Only Idle/garbage was detected; no Loop reinitialization.
9. D transmitted only Idle/OLS.
10. Run 1: P transmitted LISM frames continuously. Run 2: J did not power up.
11. No Loop initialization; Idle transmitted. This passed once. Note: M cannot handle additional sense data and shuts off from the Loop, the OS hangs, and does not boot. P sends only basic sense data after check condition.
12. K could not initialize the Loop by itself.
13. No light was detected from F.
14. Only Idle/garbage was observed.
15. No light was detected from J.
16. P transmitted continuous Idle/garbage. No light was detected from T.
17. F did not select an AL_PA.
18. Only garbage was observed.
19. E “blue-screens”.
20. D initializes the Loop, but NT is unable to boot up completely and hangs.
21. No LIP or Loop initialization. Idle/garbage seen.
22. Continuous LISM frames observed.
23. Only S was observed to be on the Loop.
24. E crashes when T attempts PLOGI.
25. E crashes when F attempts PLOGI.
26. NOS, LRR, and Idles observed.
27. LIP(F8,F7) observed on the Loop.
28. N transmitted only LIP(F8,xx).
29. Q did not select an AL_PA.

Phase II: SCSI co-existence

I_1 and I_2 are SCSI initiators. T_1 and T_2 are SCSI targets. All combinations of initiators are to be tested. Though all combinations of targets are not required to be tested, all targets need to be included in this test.

Test 1:
Setup: All devices are powered off and connected in a Loop.
Procedure: Power on the targets and wait until they are fully powered up. Power on I_1 and wait for it to fully power up. Power on I_2. Verify SCSI connectivity.

Test 2:
Setup: As existing at the end of Test 1.
Procedure: Disconnect I_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 3:
Setup: As existing at the end of Test 1.
Procedure: Disconnect I_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 4:
Setup: As existing at the end of Test 1.
Procedure: Disconnect T_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 5:
Setup: As existing at the end of Test 1.
Procedure: Disconnect T_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

I_1	I_2	T_1	T_2	T_3	Hub	Test 1	Test 2	Test 3	Test 4	Test 5
E	T	G	J		H	FAIL 1	N/T	N/T	N/T	N/T
T	D	G	J		H	FAIL 2	FAIL 3	FAIL 3	FAIL 4	FAIL 4
E	Q	G	J		H	FAIL 10	FAIL 11	FAIL 11	FAIL 11	FAIL 11
D	Q	G	J		H	FAIL 12	FAIL 13	FAIL 14	FAIL 15	FAIL 16
K	M	G	J		H	Pass 22	FAIL 23	Pass 22	Pass 22	Pass 22
E	M	S	F	C		FAIL 27	FAIL 27	FAIL 27	FAIL 27	FAIL 27
M	D	S	F	C		FAIL 28	FAIL 28	FAIL 28	FAIL 28	FAIL 29
K	E	S	F	C		FAIL 31	N/T	N/T	N/T	N/T
T	Q	S	F	C		FAIL 32	FAIL 32	N/T	N/T	N/T
Q	M	P	A		H	Pass	Pass	Pass	Pass	Pass
Q	K	P	A		H	FAIL 33	Pass	Pass	Pass	Pass
K	T	P	A		H	Pass	FAIL 34	FAIL 34	FAIL 34	FAIL 34
M	T	P	A		H	Pass	FAIL 35	FAIL 35	Pass	FAIL 35
D	E	P	A		H	FAIL 36	N/T	N/T	N/T	N/T

Test 6:
Setup: All devices are powered off and connected in a Loop.
Procedure: Power on the targets and wait until they are fully powered up. Power on I_2 and wait for it to fully power up. Power on I_1. Verify SCSI connectivity.

Test 7:
Setup: As existing at the end of Test 6.
Procedure: Disconnect I_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 8:
Setup: As existing at the end of Test 6.
Procedure: Disconnect I_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 9:
Setup: As existing at the end of Test 6.
Procedure: Disconnect T_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.

Test 10:
Setup: As existing at the end of Test 6.
Procedure: Disconnect T_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

I_1	I_2	T_1	T_2	T_3	Hub	Test 6	Test 7	Test 8	Test 9	Test 10
E	T	G	J		H	N/T	N/T	N/T	N/T	N/T
T	D	G	J		H	FAIL 5	FAIL 6	FAIL 7	FAIL 8	FAIL 9
E	Q	G	J		H	N/T	N/T	N/T	N/T	N/T
D	Q	G	J		H	FAIL 17	FAIL 18	FAIL 19	FAIL 20	FAIL 21
K	M	G	J		H	FAIL 24	FAIL 3	FAIL 3	FAIL 25	FAIL 26
E	M	S	F	C		FAIL 27	FAIL 27	FAIL 27	FAIL 27	FAIL 27
M	D	S	F	C		FAIL 30	N/T	N/T	N/T	N/T
K	E	S	F	C		FAIL 31	N/T	N/T	N/T	N/T
T	Q	S	F	C		Pass	Pass	Pass	Pass	Pass
K	Q	P	A		H	Pass	Pass	Pass	Pass	Pass
T	K	P	A		H	Pass	N/T	N/T	Pass	N/T
T	M	P	A		H	Pass	Pass	Pass	Pass	Pass
E	D	P	A		H	FAIL 37	FAIL 38	FAIL 38	FAIL 38	FAIL 38

Comments:
1. After powering on E, E could see both Targets. However after powering on T, no light was detected from T.
2. After powering on T, T could see both Targets. However after powering on D, constant LISM frames were observed; the Loop did not initialize.
3. Constant LISM frames were observed.
4. Constant LISM frames and LIP(F7,F7) were observed.
5. After powering on D, constant LISM frames were observed; the Loop did not initialize. After powering on T, no light was observed from T.
6. No light observed from T, G transmits LR continuously, and D transmitted NOS and OLS continuously.
7. G transmitted OLS continuously, D transmitted LIP(F7,F7), OLS, NOS, and Idle continuously.
8. Constant LISM frames were observed. G transmitted OLS continuously.
9. Both D and G transmitted OLS continuously.
10. After powering on E, E could see both Targets. However, after on Q, LISM frames were seen often as the Loop initialized and reinitialized over and over.
11. LISM frames were seen often as the Loop initialized and reinitialized.

12. After powering on D, constant LISM frames were observed. After powering on Q, NOS was seen from Q, and Idle was seen from J.
13. J transmitted Idle, while Q transmitted OLS.
14. Q transmitted continuous LISM frames, while J transmitted Idle.
15. Idles and bad transmission words were observed; the Loop did not initialize.
16. Q transmitted LISM frames / LIP / OLS, while J transmitted LIP / OLS / NOS.
17. After powering on Q, J transmitted LR / Idle, while Q transmitted LIP / NOS. After powering on D, constant LISM frames were observed.
18. Q transmitted LISM frames / LIP / OLS, while J transmitted NOS.
19. Q transmitted Idle / OLS while J transmitted NOS.
20. Q transmitted LISM frames while J transmitted Idles.
21. Q transmitted LIP / OLS / LR while J transmitted NOS.
22. M saw the Targets OK, but K saw G twice.
23. K did not see J.
24. After powering on M, M could see the Targets OK. After powering on K, no light was seen from K, and constant LISM frames were observed.
25. J transmitted Idles / bad Transmission Words, while M transmitted LISM frames.
26. J transmitted Idles, while M transmitted LIP(F8,F7) / LIP(F7,F7).
27. M could not see Target S. Loop could not operate with F present; F removed for these tests.
28. M could not see Targets M or S. D was not operational (failed to boot).
29. Loop did not recover. LISM frames did not resolve.
30. Loop failed with both Initiators powered on.
31. Loop could not stabilize with K present.
32. T was unable to see the Targets (hung in Disk Administrator).
33. K could not see Target P; sent LS_RJT in response to PLOGI ACC from P.
34. K could not enter Disk Administrator (hung).
35. Only invalid transmission words on Loop; Loop was unusable.
36. E “blue screened” after receiving PLOGI; no further testing.
37. E “ blue screened” after receiving PLOGI.
38. Loop did not initialize; LISM frames did not resolve.

Phase III: Fabric operation

All combinations of SCSI initiators and SCSI targets are to be tested with each switch.

Test 1:
Setup: An initiator is connected to an F_Port on the switch. A target is connected to an FL_Port on the switch. Both initiator and target are powered off.
Procedure: Power on the target and wait until it is fully powered up. Power on the initiator. Verify SCSI connectivity.

Test 2:
Setup: As existing at the end of Test 1.
Procedure: Move the initiator to a different F_Port on the switch. Verify SCSI connectivity.

Test 3:
Setup: As existing at the end of Test 2.
Procedure: Move the target to a different FL_Port on the switch. Verify SCSI connectivity.

Test 4:
Setup: Same as in Test 1.
Procedure: Power on the initiator and wait until it is fully powered up. Power on the target. Verify SCSI connectivity.

Initiator	Target	Fabric	Test 1	Test 2	Test 3	Test 4
Q	P	R	Pass	FAIL 1	FAIL 2	Pass
D	P	R	Pass 3	Pass 3	Pass 3	FAIL 4
M	C	R	FAIL 5	N/T	N/T	FAIL 5
D	C	L	FAIL 6	FAIL 7	FAIL 6	FAIL 6
M	P	L	FAIL 8	FAIL 8	FAIL 8	FAIL 8
M	C	L	FAIL 9	FAIL 9	FAIL 9	FAIL 9
D	P	L	FAIL 10	FAIL 10	FAIL 11	FAIL 11
Q	P	L	Pass	Pass	FAIL 12	FAIL 12
D	C	R	Pass	Pass	FAIL 13	Pass

Comments:
1. Unable to move Q. Could not recover link after disconnect.
2. Q could not see Target P in the nameserver.
3. Initiator D was connected to an FL_PORT.
4. D could not see Target P.
5. Initiator M could not see Target C; both were registered with the nameserver.
6. Initiator D unable to see Target C.
7. Initiator D unable to reinitialize loop when moved.
8. Both devices logged in to Fabric; Initiator M could not see Target P.
9. Both devices logged in to Fabric; Initiator M could not see Target C.
10. Both devices logged in to Fabric; Initiator D “hung” in Disk Administrator initialization.
11. Initiator D could not see Target P.
12. Initiator Q could not see Target P.
13. Initiator D received Remote State Change Notification, but could not see Target C.

To: T11
From: Dal Allan
Date: 2-9-98
Subject: FC-AL Loop Initialization

An email I received from Barry Reinhold this morning contains information of value to us in the FC-AL-2 review which is on our current agenda. I have extracted the relevant material below, which consists of:

- some recommendations relative to the FC-AL-2 draft specifications
- a proposed starting point for a state diagram addressing loop initialization and recovery
- a set of comments on rev 5.8 (December 24, 1997) of the FC-AL-2 draft standard.

Inclusions from Barry's email:

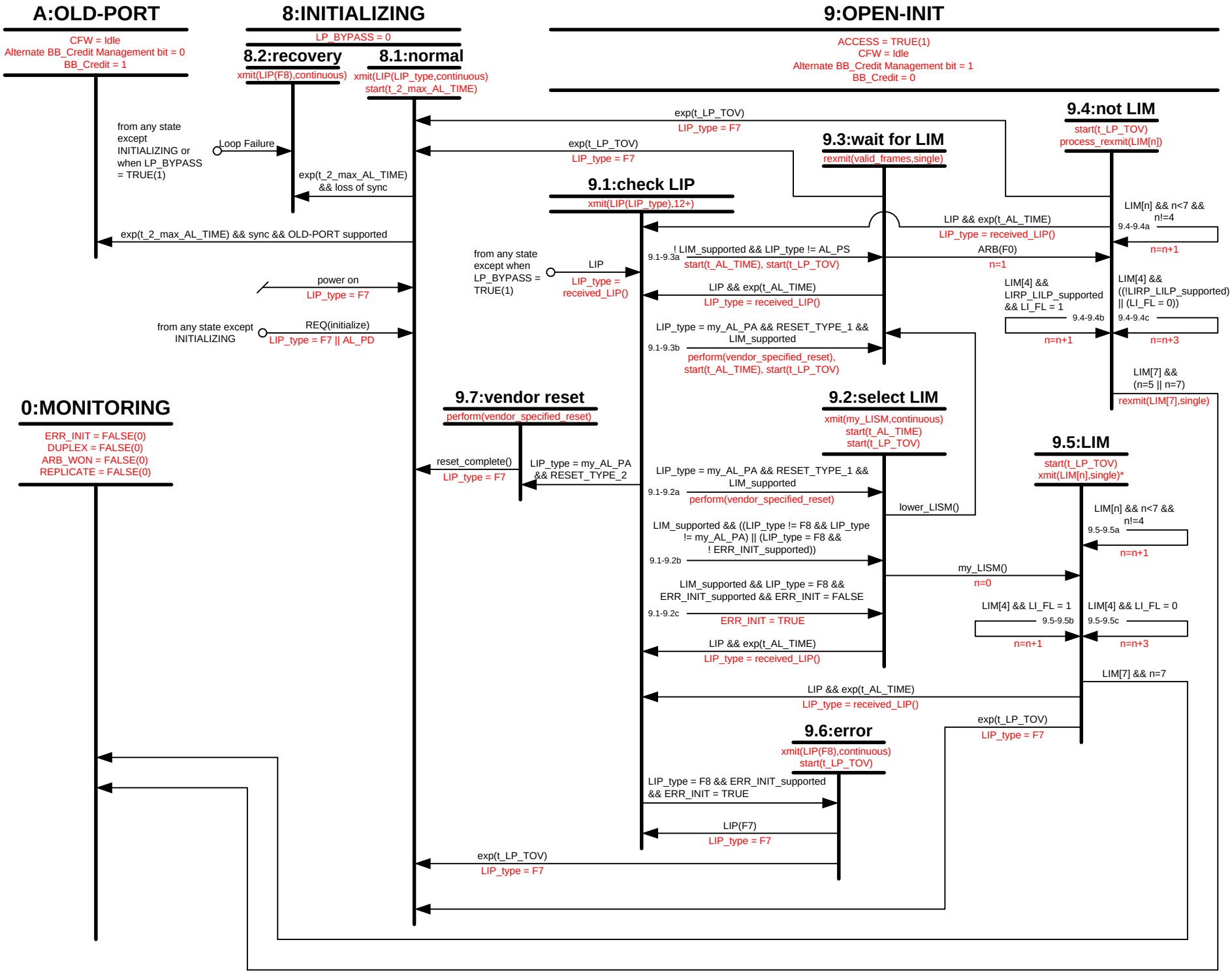
I (Barry Reinhold) would like to propose that the FC-AL-2 standards body address the following issues:

1. Document the proper external behavior of a L_Port during the loop initialization and recovery process using a normative state diagram with labeled transitions and a table of notes attached which references the transitions. The state diagram should be the highest level of priority in terms of conflict resolution with the transition notes being second and lose text the lowest priority. Examples of this nature are common and can be found in IEEE 1394 documentation, almost all IEEE 802 documentation as well as ANSI FDDI. A potential starting point for a state diagram is included in this message. The diagram is not an attempt to document the current state of FC-AL-2. After significant work we abandoned that effort as we felt the current draft was too ambiguous to document without significant changes in the external behavior. The provided state diagram makes a number of simplifications that we believe would significantly enhance the operation of arbitrated loops. However, our real goal is to achieve precise specification of external behavior.
2. Remove all references to Annex I.2.2 loop recovery. There are a number of reasons for this, based on the current state of the standard, current field experience with FC-AL products and the limited effectiveness of the trace operation in FDDI. Let me elaborate on each of these points:
   - The degree of specification of Annex I.2.2 is so loose that it is impossible to understand what proper behavior is. It is poorly integrated into the behavior of the LPSM. Actual use of this process in an arbitrated loop is very unlikely to be successful and could easily leave nodes in a bypassed state for no reason. The whole concept of removing devices from a loop using a set of AL_PAs that may no longer be valid is questionable – but if allowed should not be up to the discretion of an implementor but carefully documented by the standard.
   - To our knowledge no product has actually implemented this process and there is not a need to address backwards compatibility. Its existence causes needless complexity in an area where the standard should be simple and robust. The process of selecting a loop master in order to get a loop going is not that complex and very similar processes have been done in both Token Ring and FDDI. These solutions can't be adopted but the process can be simplified and made more robust.
   - Field experience has shown that the FDDI trace mechanism though well defined and documented, is seldom useful in resolving problems even though FDDI can easily bound the fault domain and has the ability to reach the faulting node through a second path. The basic problem is that when the station has gone bad, and not the physical connection itself, it has seldom responded to trace. In FC-AL we do not really have a good mechanism to isolate the fault domain and we do not have a second path to get to the fault. Our chances of actually recovering from a loop failure condition through the use of mechanism outlined in Annex I.2.2 are almost nil.
   - If Annex I.2.2 recovery is removed it is reasonable to eliminate all state transitions in INITIALIZING and OPEN-INIT associated with LPB and LPE. This would help simplify the standard significantly and not impact currently running loops. This would help in the robustness of loops as the use of LPB and LPE when the AL_PA is not solid can lead to unexpected behaviors.
3. Define the external behavior of a station when it enters the loop during power up. If this proves to be unacceptable at least limit the duration of time during which arbitrary behavior can take place. There are too many cases in the field where stations do not behave in a defined manner until disk power up cycling has completed or drivers loaded.

4. Although we realize that an effort has been made and rejected for establishing a standard for hubs there is still a need to bring legacy devices into a FC-AL loop in a manner that will allow stable operation. Concentrating devices that shield the loop from the destructive behavior of current L_Ports appear to offer a realistic avenue towards stable loop operation. The goals for the concentrating device should be basic:
    - Properly insert current legacy L_Ports into the loop. This can be done in any number of ways but the standard should provide a set of specifications that will ensure proper and consistent behavior during loop insertion.
    - A clear market segment has developed for concentrating devices and vendors are creating products without guidance of a standard. This is currently causing a new type of interoperability problem and as additional features are added the potential for new problems will grow.

The following section contains the state diagram and associated documentation:

General Notes – State Diagrams are expressed using vertical shafts to represent states and horizontal lines to denote transitions between states.



To maintain consistency with FC-AL 4.5 the INITIALIZING and OPEN-INIT states are broken into sub-states. When entering a sub-state the from outside of the parent state the entry actions for both the parent state and the sub-state are performed. The combined entry actions of the state and sub-state are assumed to start simultaneously. When entering a different sub-state within the same parent state only the entry actions of the sub-state are to be performed.

Transitions are illustrated with the triggering condition located above the horizontal line and any actions to be performed with the transitions below the line. All transitions are performed while remaining in the previous state, before entry to the new state.

All times are assumed to be global and are only reset by explicit action. An expired timer remains expired until it is reset with the start() procedure.

The following rules are associated with event processing:
1. All conditions are evaluated in the context of the current state.
2. If the conditions for a transition are satisfied then: (a) perform the actions associated with the transition in the current state, (b) enter the new state, (c) perform the entry actions for the new state, and (d) evaluate conditions for exiting the new state.
3. No two transitions in a given state may be true at the same time or the state diagram is in error.

State Variables
1. ERR_INIT – Set to true to indicate that the L_Port has already attempted to initialize the loop by sending LISM frames while receiving LIP F8.
2. ERR_INIT_supported – Set to true to indicate that the L_Port will send LIP F8 when receiving LIP F8 after an attempt has been made to bring up the loop by sending LISM frames. If set to false the L_Port will not send continuous LIP F8 frames.
3. LI_FL – The value of bit 8 of the 16 bit LI_FL field as defined in section 10.4.1.
4. LIM[0..7] – An array containing the different sequences that are used in the loop initialization process. This array is logically initialized using the following C programming language notation: LIM[ ] = {ARB F0, LIFA, LIPA,LIHA,LISA,LIRP,LILP,CLS}
5. LIM_supported – This is true if the L_PORT can perform the operations required of a loop master.
6. LIP_type – The type of LIP to be transmitted in the current or next state. May take on the value of F8, F7 or any valid AL_PD as defined in Annex K.
7. RESET_TYPE_1 – This is true if the implementation supports a type of reset that when externally observed would be identical to the external behavior of receiving LIP F7. The station shall continue in the OPEN-INIT sub state 9.3
8. RESET_TYPE_2 – This shall be the logical opposite of RESET_TYPE_1. It indicates that the vendor specific reset causes behavior different than that expected from a LIP F7
9. T_AL_TIME – A timer set to the value of AL_TIME.
10. T_LP_TOV – A timer set to the value of LP_TOV.

State Procedures (do not return a value)
1. Lower_LISM() – Returns true if (rx_D_ID < my_D_ID) or ((rx_D_ID = my_D_ID) && (rx_S_ID < my_S_ID)) or ((rx_D_ID = my_D_ID) && (rx_S_IS = my_S_ID) && (rx_WWN < my_WWN)).
2. My_LISM() – Returns true if the received LISM frame's D_ID, S_ID, and WWN match the D_ID, S_ID, and WWN of my LISM frame.
3. Perform(arg) – This procedure is a notational convenience that allows a complex action to be expressed. The actions being performed by this function do not alter that behavior of the LPSM.
4. Process_rexmit() – Performs the necessary actions on the received frame before retransmitting it. The actions are defined in section 10.4.3.
5. Rexmit(arg) – Transmit the last logical item received.
6. Start(arg) – This procedure starts the timer passed in as an argument. It is logically equivalent to starting a count down timer by loading the timer with the time value specified as an argument.
7. Xmit(arg1,arg2) – Transmit the first argument in the manner described by the second argument. The second argument may be either single, 12+, or continuous.

State Functions (return a value)
Expire() – This function is true when the timer associated with the argument reaches zero.

Receive_LIP() – Returns the type of LIP received in accordance with the FC-AL 3.1.22. This function can take on the values F7,F8 or a valid AL_PD as defined for the reset LIP.
Reset_complete() – True when the L_Port has completed the vendor specific reset process and is ready to reenter the INITIALIZING state.

Transitions – The following text provides additional information about particular transitions.

1. X-8.1 – The value of LIP_type is established by the LPSM as defined in section 7.8. It shall be either LIP F7 or LIPr.
2. 9.1-8.1a – The L_Port shall respond to the LIPr by performing a vendor specific reset. The station shall remain in state 9.7, Vendor Reset, until it is ready to reenter the initializing state. The external behavior of the station is as if it were bypassed until the station is ready to reenter the loop.
3. 9.1-9.2a - If the first character following the LIP is the AL_PA of this L_Port the L_Port is to perform a vendor specific reset procedure. The external behavior of the L_Port shall be the same as if a LIP F7 was received.
4. 9.1-9.3a and 9.1-9.3b – If the first character following the LIP is the AL_PA of this L_Port the L_Port is to perform a vendor specific reset procedure. The external behavior of the L_Port shall be the same as if a LIP F7 was received.
5. 9.4-0 – The L_Port transitions into monitoring only when CLS is received LIM[7]. This transitions is further constrained by checking to see that six or eight sequences have been sent. If these conditions are not met the station will time out on LP_TOV.
6. 9.4-9.4a – Each of the loop initialization sequences must be received in order to be processed by this transition. The exception to this rule is the LISA frame which is processed by 9.4-9.4b and 9.4-9.4c. If a loop initialization sequence is out of order the frame will not be forwarded and the LIM is expected to time out on LP_TOV.
7. 9.1-9.7 – This transition is taken if the reset process will cause external behavior different from that of receiving a LIP F7. This expected behavior of this type of reset is that of a power on reset.
8. 9.5-9.5a – When n=0 the xmit(LIM[n],single) function shall not transmit a single item but shall transmit ARB F0 continuously.

FC-AL-2 v5.8 comments

INITIALIZING state

- The cases of normal initialization and Loop recovery should be two separate states or substates. The behavior is different depending on which operation is performed.
- Annex I.2.2 should be removed from the standard. It causes needless complexity and ambiguity.
- The expected behavior for an L_Port detecting Loop Failure is unclear. Some interpretations of the behavior for a device powering up receiving no signal:
  1. Enter normal initializing section and transmit LIP(F7,F7) for up to 2*max AL_TIME. If Annex I.2.2 is supported, transmit LIP(F8,F7) for 2*AL_TIME followed by LBPyx for up to 2*AL_TIME for each AL_PA (recovery). Then what?
  2. Enter normal initializing section and transmit LIP(F7,F7) for up to 2*max AL_TIME. If Annex I.2.2 is not supported, transmit LIP(F8,F7) for 2*AL_TIME. Then what?
  3. Enter Loop recovery and transmit LIP(F8,F7) for 2*AL_TIME. Then what?
- If a device powers up receiving no signal, and it directly enters recovery, it cannot perform Annex I.2.2.
- The last paragraph of INITIALIZING states that if a device begins to either bypass or enable another port by transmitting LPB or LPE, it shall transmit the primitive until it is received or until the request has been dropped. By definition of what the port is transmitting, it is in the recovery section at this point. What if the port receives LIP? Should it continue to send the LPB or LPE or should LIP cause a transition into OPEN-INIT?

OPEN-INIT state

- The vendor specific reset mentioned in the first bullet point should not cause the external behavior of an L_Port to be undefined. The device then *may* continue with the initialization procedure. What happens if it does not? After the reset, the L_Port should go to the INITIALIZING state, as if from power up.
- It is not specified in the text what an L_Port should do if the received LIP is a reset LIP and y≠AL_PA of the L_Port. This behavior should be the same as for a LIP(F7).
- The third to last bullet point on page 41 (reception of ARB(F0)) should be clarified. What should happen if a device is transmitting its own LISM frames and receives ARB(F0) before receiving its LISM back? This should never happen, but if it does, shouldn't the L_Port ignore the ARB?
- The second to last bullet point on page 41 (reception of CLS) should be constrained. Receiving CLS at any point during the initialization process should not cause a transition into MONITORING. This transition should only be allowed after receiving LISA or LILP.
- On page 64, the last paragraph states "The frame header shall not be used to validate the Loop Initialization Sequences." What exactly does this mean?
- In section 10.3, P63, the last paragraph should have all but the first sentence should be removed. The entire paragraph should read, "The L_Port that is attempting to initialize shall make the transition to the INITIALIZING state (REQ(initialize)) (see 8.4.3, item 21)."[end]
- The second bullet point on page 41 states "…the L_Port shall ignore LIPs for AL_TIME" after receiving LIP(F7). Is this timer started upon entering the OPEN-INIT state or after the L_Port has finished transmitting LIP?
- The transmission and reception of LPB and LPE should not be allowed during the Loop initialization procedure (i.e., the L_Port is in the INITIALIZING or OPEN-INIT state).

MONITORING state

- On page 32, "If the L_Port detects Loop Failure, on its inbound fibre and LP_BYPASS is TRUE(1), the LPSM shall transmit LIP(F8), but remain in the MONITORING state." There is not a case when LP_BYPASS is FALSE(0). Although this transition exists in Table 4, it should be in the text as well. There should also be an explicit statement requiring the L_Port to stop transmitting LIP and continue retransmitting received transmission words once the Loop Failure condition has disappeared.

FC-AL-2 Report

FC-AL-2 letter ballot review was held in December, January, and February. All comments were addressed by the committee. Several changes were made during the review and a revised FCAL57LB.txt file will be created. FC-AL-2 rev 6.1 will be made available 17Feb98. Some of the changes include:

1. The L_bit during initialization has been modified. In FC-AL-1, the L_bit being set required a new login with every node that recognized the L_bit. The text has been changed to reflect the following:

   a) no reason for the L_bit being set is now specified in FC-Al-2

   b) if the L_bit is set in LISA (the only place it is now checked), the public NL_Ports are implicitly logged out with the fabric. Nothing is mentioned about what is required next. The L_bit being set is now treated similarly to NOS on an F_Port link.

2. A REQ(DHD) line has been added to all tables. If REQ(DHD) is activated, it is assumed that this L_Port knows that it and the other L_Port support DHD (consequently, checking the login bit has been removed from the tables and the text).

3. DUPLEX has been added to OPEN. It is set to 1 on entry and set to 0 when DHD has been transmitted.

4. LIPyx (reset) in OPEN-INIT has been changed. The last sentence of the paragraph now reads: "The L_Port shall continue with the initialization proced-ure by transmitting the LISM sequence as described in 10.4 or the L_Port shall set REPEAT to TRUE(1) and continue in the MONITORING state."

FC-AL-2 ARB Detection

FC-AL specifies the bit pattern transmitted for each primitive, but says nothing explicit about what received bit patterns are detected as specific primitives. While some have interpreted this as an implication that all 32 or 40 bits (4 bytes or characters) must be compared to detect a primitive, that implication has been missed by most readers and has not been implemented by existing FC-AL devices. Specific rules are being added to FC-AL-2 to address this confusion and/or problem.

The detection rules for most primitives have been non-controversial. For example, primitives that contain no parameter fields (e.g. RRdy, CLS) require checking all four characters or bytes (a so-called 40-bit comparison). Primitives such as OPN require checking three characters or bytes, the first two and the relevant address byte.

However the working group has been unable to reach consensus on the ARB primitive. We request that the plenary vote on the following to direct the document editor on what to include in FC-AL-2.

Note: Existing FC-AL devices from different vendors implement the following in conflicting ways. There is no set of rules consistent with all existing implementations (other than the current lack of rules). Reserved or invalid ARBs have been observed in loop testing, it is believed due to faulty transceivers with high error rates.

In the working group discussions, the consensus (compromise agreement?) on all of these detection rules is that they would be documented in FC-AL-2 as recommendations (“should”), with the intent (non-binding) that they become requirements in FC-AL-3. In each of the following I have asked for a separate motion on whether the detection rules should be a recommendation or requirement to obtain a recorded vote on this.

2. ARB(F0) detection (K28.5, D20.4, D16.7, D16.7)

ARB(F0) is used by the fairness algorithm. It is detected and responded to as a special case by several LPSM (Loop State Machine) transitions.

In the working group discussion yesterday, there was apparent consensus (no voiced objections) to recommending that all four characters or bytes must match exactly (a so-called 40-bit comparison) for detecting ARB(F0). The apparent consensus would imply voting Yes on motion 1a and No on motion 1b.

Motion 1a: Move that FC-AL-2 will either recommend ("should") or require ("shall") checking all four bytes or characters for an exact match to detect ARB(F0).

Vote Yes if you feel FC-AL-2 should give guidance on detecting ARB(F0).

Vote No if you feel FC-AL-2 should remain silent on this (as does FC-AL).

Motion 1b: Move that FC-AL-2 will require (rather than simply recommend) checking all four bytes or characters for an exact match to detect ARB(F0).

Vote Yes if you feel this shall be required to claim FC-AL-2 compliance.

Vote No if you feel that "old" FC-AL devices should be allowed to claim FC-AL-2 compliance, while giving guidance for new implementations.

3. Own ARB detection (K28.5, D20.4, AL_PA, AL_PA)

FC-AL ports detect their own ARB to win arbitration, plus a few special cases within the LPSM.

In both yesterday's and the previous working group discussion, there was apparent consensus (no voiced objections) to recommending that all four characters or bytes must match exactly (a so-called 40-bit comparison) for detecting the port's own ARB. The apparent consensus would imply voting Yes on motion 2a and No on motion 2b.

Motion 2a: Move that FC-AL-2 will either recommend ("should") or require ("shall") checking all four bytes or characters for an exact match for a port to detect its own ARB.

Vote Yes if you feel FC-AL-2 should give guidance on detecting a port's ARB.

Vote No if you feel FC-AL-2 should remain silent on this (as does FC-AL).

Motion 2b: Move that FC-AL-2 will require (rather than simply recommend) checking all four bytes or characters for an exact match for a port to detect its own ARB.

Vote Yes if you feel this shall be required to claim FC-AL-2 compliance.

Vote No if you feel that "old" FC-AL devices should be allowed to claim FC-AL-2 compliance, while giving guidance for new implementations.

5. Invalid ARB Processing

Later motions address the rules for detecting valid ARB primitives. These motions address the handling of any ordered set that begins with the characters K28.5, D20.4 and is not a valid ARB. I am calling these “invalid ARB primitives” here, the eventual term used in FC-AL-2 may be different.

There are two different behaviors that need to be addressed, since most implementations perform them with different logic. One is whether an invalid ARB may be deleted as a fill word for clock skew management. This typically takes place before the port’s elasticity buffer in logic operating on the recovered receive clock. The other is LPSM operation, which typically takes place after the elasticity buffer in logic operating on the port’s local clock.

In the working group discussion yesterday, there was apparent consensus (no voiced objections) to recommending that the LPSM should replace invalid ARB primitives with the CFW (Current Fill Word, typically the most recent valid ARB or Idle). There was apparent consensus that this implied that invalid ARB primitives should be considered deletable fill words, since the CFW being substituted could have been deleted. The apparent consensus would imply voting Yes on motions 3a and 3c and No on motions 3b and 3d. At present FC-AL appears to imply that invalid ARB primitives are “other ordered sets” that are passed through unaltered by the LPSM. That interpretation of FC-AL would likely be more emphatic if we otherwise clarify the rules for ARB detection.

Motion 3a: Move that FC-AL-2 will either recommend (“should”) or require (“shall”) that the LPSM substitute the CFW for invalid ARB primitives.

> Vote Yes if you feel FC-AL-2 devices should not propagate invalid ARB primitives.
>
> Vote No if you feel FC-AL-2 devices should forward invalid ARB primitives unaltered.

Motion 3b: Move that FC-AL-2 will require (rather than simply recommend) that the LPSM substitute the CFW for invalid ARB primitives.

> Vote Yes if you feel this shall be required to claim FC-AL-2 compliance.
>
> Vote No if you feel that “old” FC-AL devices should be allowed to claim FC-AL-2 compliance, while giving guidance for new implementations.

Motion 3c: Move that FC-AL-2 will either recommend (“should”) or require (“shall”) that ports consider all ordered sets beginning with K28.5, D20.4 (a so-called “20-bit comparison”, including both valid and invalid ARB primitives) as fill words that may be deleted for clock skew management.

Motion 3d: Move that FC-AL-2 will require (rather than simply recommend) that ports consider all ordered sets beginning with K28.5, D20.4 as fill words that may be deleted for clock skew management.

7. Valid ARB Detection

Ports detect valid ARB primitives for implementing arbitration and updating the CFW (Current Fill Word). Arbitration operates by comparison of the loop address in the received ARB against the port's own loop address. This comparison is considered valid and acted upon if and only if the received ARB is valid. One significant implication of this is that only valid ARB primitives may become the CFW (Current Fill Word). A single received ARB primitive, if it becomes the CFW, may be replicated and transmitted numerous times before it is replaced by a later fill word.

In the working group discussion yesterday, there was apparent consensus that an ARB primitive be detected as valid if its last two characters or bytes (character or byte 3 and 4) are equal. However subsequent discussion revealed that there were differing interpretations on how "equality" was determined. An early vote was taken before this misunderstanding was recognized; its results seem irrelevant.

Once this issue was understood, the following options were identified:

1) An ARB is valid if (and only if) the decoded (8-bit) values of bytes 3 and 4 are identical. This is most easily implemented by an equality comparison after the 8b/10b decoder.
2) An ARB is valid if (and only if) the decoded (8-bit) values of bytes 3 and 4 are identical, and the value is one of the neutral disparity code bytes. There are 134 neutral disparity code byte values, listed in FC-AL Table 1, which include AL_PA loop addresses plus the reserved and special AL_PA values. This is most easily implemented by an equality comparison of the encoded 10-bit characters before the 8b/10b decoder.
3) Allowing either 1 or 2 above as an implementation option (vendor unique behavior).

It should be noted that while choice 1 above is most easily implemented by a comparison in the 8b domain and choice 2 is most easily implemented by a comparison in the 10b domain, either can be implemented in the other domain with modest amounts of additional logic (n.b., there was considerable controversy over what constituted a "modest amount", I state that as my own opinion in the belief that there is no adjective that would be agreed to).

The principal arguments made in support of the choices were:

1. Advocates of both 1 and 2 felt that their preference was simpler in their respective implementations.
2. The majority of existing designs that implement such checking follow choice 1.
3. Choice 2 is simpler for FL_ports that do not otherwise have an 8b/10b decoder.

Three votes were taken, all with one vote per company. The first was taken shortly before lunch, at which time at least one voter complained that he needed to check with his office before voting. Regardless, the vote was taken with results 10 for choice 1, 8 for choice 2 and 3 for choice 3 (each company allowed to vote for one choice).

The other two votes were taken after lunch, approximately 20 to 30 minutes after the stated time for reconvening. The second vote allowed for each company to vote for one of the three choices, results were 7 for choice 1, 4 for choice 2 and 3 for choice 3. The third vote allowed for only choices 1 and 2, results were 9 for choice 1 and 5 for choice 2.

Motion 4a: Move that FC-AL-2 either recommend (“should”) or require (“shall”) that an ARB primitive is detected as K28.5, D20.4 and identical values in bytes 3 and 4. All 256 data byte values in bytes 3 and 4 are permitted and should/shall be detected as valid provided that the two bytes are identical.

Motion 4b: Move that FC-AL-2 either recommend (“should”) or require (“shall”) that an ARB primitive is detected as K28.5, D20.4, identical values in bytes 3 and 4 where that value is one of the 134 neutral disparity byte values listed in [FC-AL] Table 1.

Motion 4c: Move that FC-AL-2 either recommend (“should”) or require (“shall”) that an ARB primitive is detected as K28.5, D20.4 and identical values in bytes 3 and 4. FC-AL-2 ports may optionally choose to require, for detection of ARB, that the value in bytes 3 and 4 be one of the 134 neutral disparity byte values listed in [FC-AL] Table 1.

Motion 4d: Whichever of 4a to 4c is approved (if any), FC-AL-2 will require (as opposed to simply recommend) the result.

FCLC January 1998 Group Testing Period

- Overview of the Testing
- General Results
- Issues
- Proposed Directions

Overview of Testing

- Focus was on Arbitrated Loop Stability
  - Test all possible pairs of devices (2 days)
  - Test SCSI coexistence of multiple initiators
  - Test public loop operation
  - Study problems and issues in “large” loops

All L_Port Pair Testing

- Power on L_1 and wait until it is fully powered up. Power on L_2.
- Power on L_2 and wait until it is fully powered up. Power on L_1.
- Disconnect L_1 from the Loop and then reconnect it.
- Disconnect L_2 from the Loop and then reconnect it.
- Disconnect the receiver of L_1. Power cycle L_1. Reconnect the receiver after L_1 is fully powered up.
- Disconnect the receiver of L_2. Power cycle L_2. Reconnect the receiver after L_2 is fully powered up.

Pair Test Results

L_1	L_2	Hub	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
R	N	H	Pass	Pass	Pass	Pass	Pass	Pass
S	D		FAIL1	FAIL1	N/T	N/T	N/T	N/T
N	A		Pass	Pass	Pass	FAIL2	Pass	Pass
N	F	U	Pass	Pass	Pass	FAIL3	Pass	FAIL 3
J	R	H	Pass	FAIL4	Pass	Pass	Pass	Pass
P	K	H	Pass	Pass	Pass	Pass	Pass	FAIL 5
S	D	H	Pass	Pass	Pass	Pass	Pass	FAIL 6

Pair Test Results

- Test1 - 11/62 Pairs Failed (18%)
- Test2 - 16/62 Pairs Failed (26%)
- Test3 - 12/62 Pairs Failed (19%)
- Test4 - 12/62 Pairs Failed (19%)
- Test5 - 30/62 Pairs Failed (48%)
- Test6 - 27/62 Pairs Failed (44%)
- All tests passed by 19/62 Pairs (31%)

SCSI Coexistence Tests - 1

- Power on the targets and wait until they are fully powered up. Power on I_1 and wait for it to fully power up. Power on I_2. Verify SCSI connectivity.
- Disconnect I_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect I_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect T_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect T_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

SCSI Coexistence Test - 2

- Power on the targets and wait until they are fully powered up. Power on I_2 and wait for it to fully power up. Power on I_1. Verify SCSI connectivity.
- Disconnect I_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect I_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect T_1 and reconnect it after fifteen seconds. Verify SCSI connectivity.
- Disconnect T_2 and reconnect it after fifteen seconds. Verify SCSI connectivity.

SCSI Coexistence Results - 1

- Test1 - 10/14 Initiator Pairs Failed
- Test2 - 12/14 Initiator Pairs Failed
- Test3 - 11/14 Initiator Pairs Failed
- Test4 - 10/14 Initiator Pairs Failed
- Test5 - 11/14 Initiator Pairs Failed
- One Initiator pair passed all five tests

SCSI Coexistence Results - 2

- Test6 - 9/14 Initiator Pairs Failed
- Test7 - 10/14 Initiator Pairs Failed
- Test8 - 10/14 Initiator Pairs Failed
- Test9 - 9/14 Initiator Pairs Failed
- Test10 - 10/14 Initiator Pairs Failed
- Three Initiator pairs passed all five tests

Public Loop Testing Procedure

- Power on the target and wait until it is fully powered up. Power on the initiator. Verify SCSI connectivity.
- Move the initiator to a different F_Port on the switch. Verify SCSI connectivity.
- Move the target to a different FL_Port on the switch. Verify SCSI connectivity.
- Power on the initiator and wait until it is fully powered up. Power on the target. Verify SCSI connectivity.

Public Loop Test Results

- Test1 - 5/10 Pairs Failed
- Test2 - 6/10 Pairs Failed
- Test3 - 8/10 Pairs Failed
- Test4 - 8/10 Pairs Failed
- No pair passed all tests

Large Loop Test Procedure

- Mission - To examine the behavior of larger loops to see if additional failure modes can be isolated
- Isolated problems
  - A station would not respond to LIP while sending LISM frames
  - A station, if it hit a timing window during loop insertion, would send LIP every 2 to 3 seconds
  - A station would reinitialize the loop 10 milliseconds after an error burst.

Large Loop Test Procedure

- Unresolved observations
  - Steady state of OLS being transmitted, LSIM being received
  - CLS observed while LIM sending loop initialization frames
  - ARB F0 received while the LIM is waiting for a LISA frame
  - orphan sequences (ARB 01, CLS)
  - morphed LIP to LIPr
  - bad idles (typically B5)

Major Issues - The FC-AL 4.5 standard

- Is imprecise relative to loop initialization and recovery
  - fundamental reason for loop instability
- The INITIALIZATION and OPEN-INIT states are not robust
  - OPEN-INIT needs to have a time out mechanism
  - entry into OLD-PORT is poorly controlled
- Insertion process into loop uncontrolled
  - endless source of inconsistent behavior

The road to recovery - key #1

- Define simple precise behavior of an L_Port during loop initialization and recovery
  - use normative state machine and pseudo code
  - simplify - ignore LPB, LPE until MONITORING state has been reached
  - simplify - ignore LIPr until MONITORING has been reached
  - simplify - get rid of Annex I.2.2
  - enter OLD-PORT only on reception of OLS
  - do timers in OPEN-INIT (in FC-AL-2)

The road to recovery - key #2

- Define the behavior of hubs so that they may “gracefully” insert broken nodes
  - initialize L_Ports (bring to MONITORING state) and then reinitialize before insertion
  - standard needs to address market needs that impact interoperability
  - can resolve problems with current 4.5 equipment

Make conformance a requirement

- interoperability assumes conformance
- Can not base conformance on “working” in a particular application environment
- loop community should establish a “self-policing” metric for conformance and interoperability
- Group test periods are helpful but not sufficient

Fibre Channel Consortium On the Loop Program

- detailed conformance tests
- demonstrated loop stability in heterogeneous loop
- interoperability tested for conformant devices only
- public posting of products that pass testing (web only)
- see http://www.iol.unh.edu

4 FCP/Class 2 Public Loop Operation for Streaming Devices (FCW/98w125r0,2/10/98)

This document describes required and optional behavior of SCSI streaming devices in a Public Loop configuration using FCP and class 2 delivery service. This document specifies operation from a FCP perspective. Refer to the FC-PH document(s) for rules and guidelines pertaining to class 2 operation.

Class 2 delivery service provides end-to-end flow control and sequence/frame level error detection and recovery. The environment defined by this document does not ensure in-order delivery of frames.

4.1 N_Port Class 2 Service Parameters - Fabric Login

Table 3 lists Class 2 Service Parameters with usage defined by this document. The following legend is used for entries in this table:

'P' means the SCSI Initiator or Target is prohibited from using the specified feature
'R' means the SCSI Initiator or Target is required to support the specified feature.

Table 1 – Class 2 Service Parameters

Class 2 Service Parameter	SCSI Initiator	SCSI Target	Notes
Class validity = 1	R	R	
Service Options			
Intermix Mode = 0	R	R	
Stacked Connect Requests = '00'b	R	R	
Sequential Delivery = 0	A	A	
Dedicated Simplex = 0	R	R	
Camp-On = 0	R	R	
Buffered Class 1 = 0	R	R	
Initiator Control			
Sequence Initiator X_ID reassignment = '00'	R	R	
Initial Responder Process_Associator = '00'b	R	R	
ACK_0/ACK_N capable = '00'b ACK_0/ACK_N capable = '01'b ACK_0/ACK_N capable = '10'b ACK_0/ACK_N capable = '11'b	A P A P	A P A P	
ACK generation assistance = 0	R	R	
Initiator Data compression capable = 0	R	R	
Initiator Data compression History buffer size = '00'b	R	R	
Recipient Control			
ACK_0/ACK_N capable = '00'b ACK_0/ACK_N capable = '01'b ACK_0/ACK_N capable = '10'b ACK_0/ACK_N capable = '11'b	A P A P	A P A P	
X_ID interlock = 0	R	R	

Table 1 – Class 2 Service Parameters

Class 2 Service Parameter	SCSI Initiator	SCSI Target	Notes
Error Policy Supported			
Abort, discard multiple sequences (‘00’b)	A	A	
Abort, discard a single sequence (‘01’b)	R	R	
Process policy with infinite buffers (‘10’b)	P	P	
Discard multiple sequences w/immediate retrans (‘11’b)	P	P	
Categories per Sequence = ‘00’b (one)	R	R	
Data compression capable = 0	R	R	
Data compression History buffer size = ‘00’b	R	R	
Receive data field size (min)	256	256	
Concurrent Sequences > 0	R	R	
N_Port End-to-end Credit (min)	1	1	
Open Sequences per Exchange > 0	R	R	

4.2 Exchange Management

The following clauses define exchange management requirements and optional behavior for devices operating in accordance with this document.

4.2.1 Initiator Exchange Management

Each exchange shall be timed using the SCSI command timeout value specified for the command. This value shall be the ULP timeout value for the FCP exchange. If a valid FCP_RSP is received before the ULP timer expires the exchange is considered complete after the ACK is sent to the target. If the ULP timer expires before the FCP_RSP is received for the command the initiator shall abort the exchange using the Abort Sequence Protocol and return a proper error indication to the application.

The initiator shall maintain exchange information until the ACK for the FCP_RSP has been sent to the target.

4.2.2 Target Exchange Management

The target shall manage exchanges using the OX_ID/RX_ID pair and shall maintain exchange information until the ACK for the FCP_RSP has been received.

4.3 Sequence Management

The following clauses define sequence management requirements for devices operating in accordance with this document.

Sequence management shall be performed as specified in the FC-PH document for class 2 operation. Each sequence (frame?) shall be timed using the value of E_D_TOV. If an ACK for the sequence is not received within E_D_TOV a sequence timeout has occurred. If a sequence timeout occurs on the first sequence of the exchange (i.e. the FCP_CMD) the initiator shall issue a RES and the Abort Sequence Protocol shall be invoked.

4.3.1 FCP Sequence Delivery Confirmation

Since this document describes operations in a class 2 environment, acknowledgments are used to provide explicit confirmation of sequence delivery. Implicit delivery may be achieved by issuing the RES extended link service and checking the response or the status of the Sequence Initiative bit in the returned exchange status block. For class 2 the sequence recipient shall not consider Sequence Initiative to have been passed until the sequence which passes the Sequence Initiative has completed successfully and the ACK has been transmitted with the Sequence Initiative bit = 1.

5 Error Detection and Recovery

This clause describes the error actions to be taken by the SCSI Initiator and SCSI Target upon detection of an error condition.

5.1 Error Detection by SCSI Initiator

The Abort Sequence Protocol shall be invoked by a SCSI Initiator when any of the following conditions occur:

a) A Sequence error is detected.

b) ULP_TOV has expired and an Exchange has not completed.

5.2 Error Detection by SCSI Target

a) A Sequence error is detected.

When a SCSI Target detects a sequence error, it shall discard that Sequence only and perform the Abort Sequence Protocol. If the target holds sequence initiative it may optionally attempt to send a FCP_RSP with SCSI status and sense information.

5.3 Abort Sequence Protocol

The Abort Sequence Protocol is primarily used in this document to abort a single sequence within an exchange. The unit of error recovery for this document is a sequence not an exchange. A SCSI Initiator may also abort an exchange using ABTS with the LS bit = 1.

A SCSI Initiator and Target may transmit ABTS. ABTS may be transmitted even if Sequence Initiative is not held or there is no end-to-end credit available. Following the transmission of ABTS the port shall consider the status of the exchange indeterminate and shall not deliver any sequences or notification of delivery to the application until the BA_ACC is received, processed, and recovery (if any) is performed. The sender of the ABTS shall reset the E_D_TOV and R_A_TOV timers when the ABTS is transmitted.

Following receipt of the BA_ACC in response to an ABTS, and after $R\_A\_TOV_{SEQ\_QUAL}$ has elapsed, the ABTS Initiator shall transmit RRQ.

If a proper BA_ACC, BA_RJT, LOGO, or PRLO is not received from the recipient within E_D_TOV, second level error recovery as described below shall be performed.

5.3.1 ABTS Recipient Behavior

Upon transmission of any of the above responses, the recipient may reclaim any resources associated with the designated sequence after $R\_A\_TOV_{SEQ\_QUAL}$ has elapsed or a Reinstate Recover Qualifier (RRQ) extended link service request has been received.

SCSI Targets shall qualify ABTS based upon OX_ID and RX_ID, since the RX_ID is not guaranteed to be known by a SCSI Initiator.

5.3.2 Reinstate Recovery Qualifier (RRQ)

The RRQ link service is used to notify the destination port that the Recovery_Qualifier is available for reuse. A separate exchange shall be used to reinstate the Recovery_Qualifier. Resources associated with the OX_ID and RX_ID specified shall be released following reception of the ACC at the initiator and following transmission of the ACC at the recipient.

The recovery qualifier is a data structure provided by the recipient to the initiator in the BA_ACC to the ABTS. It describes a completely qualified exchange and sequence and a range of frames

The format of the RRQ is shown in figure 1.

Figure 1 – Reinstate Recovery Qualifier

	Field	Content
Frame Header	OX_ID	Identifier of a new exchange
	RX_ID	hex 'FFFF'
Payload	Originator S_ID	Source_ID of the Initiator
	OX_ID	OX_ID of XCHG that was previously aborted with ABTS
	RX_ID	RX_ID of XCHG that was previously aborted with ABTS

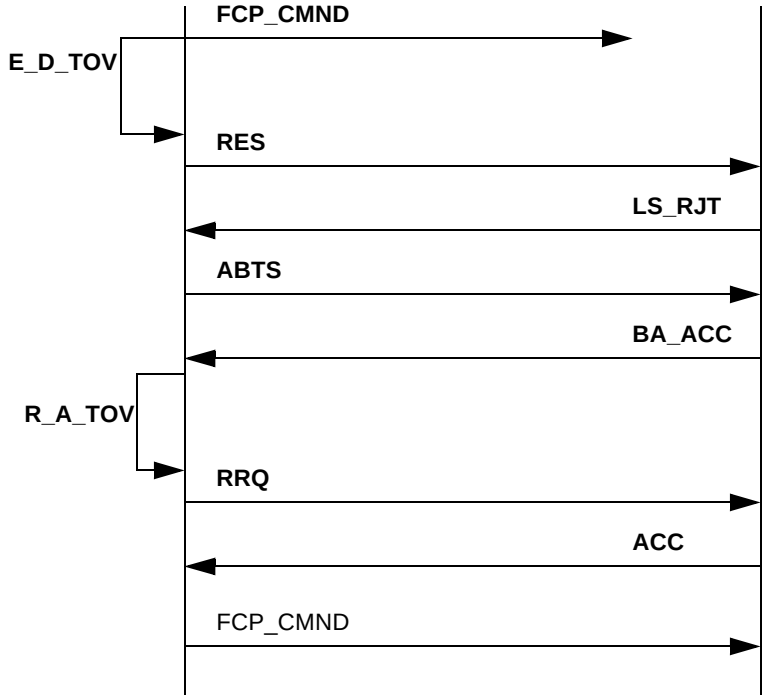
Following successful completion of the RRQ, the recipient shall respond with ACC.

5.3.3 Second-level error recovery

If a response to the ABTS is not received within E_D_TOV the Initiator shall send the ABTS again. This procedure shall continue until the ULP timer expires.

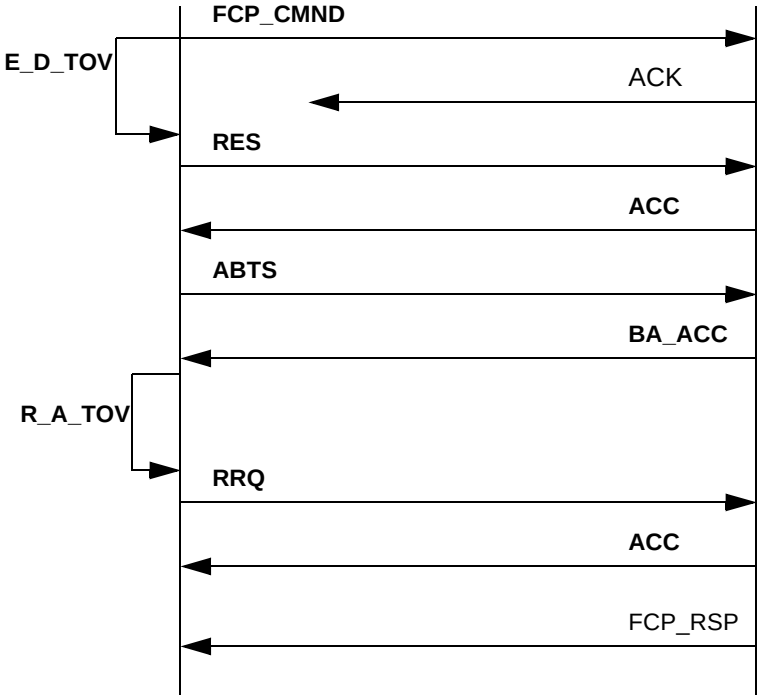
5.3.4 Error Detection and Recovery Examples

Figure 2 – Lost FCP_CMND



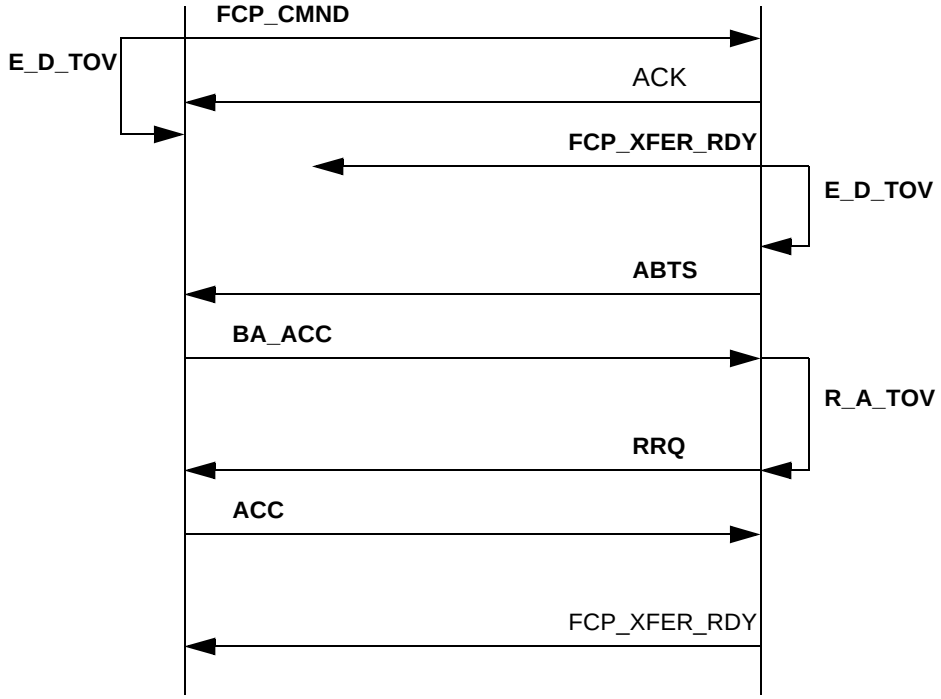
If a FCP_CMND is sent by the initiator and not received at the target the timer will expire and the initiator sends a RES. The target does not know of the exchange and returns a LS_RJT with an invalid OX_ID-RX_ID combination reason code. The initiator knows the command was not received so it performs the Abort Sequence Protocol and reissues the command.

Figure 3 – Lost ACK for FCP_CMND



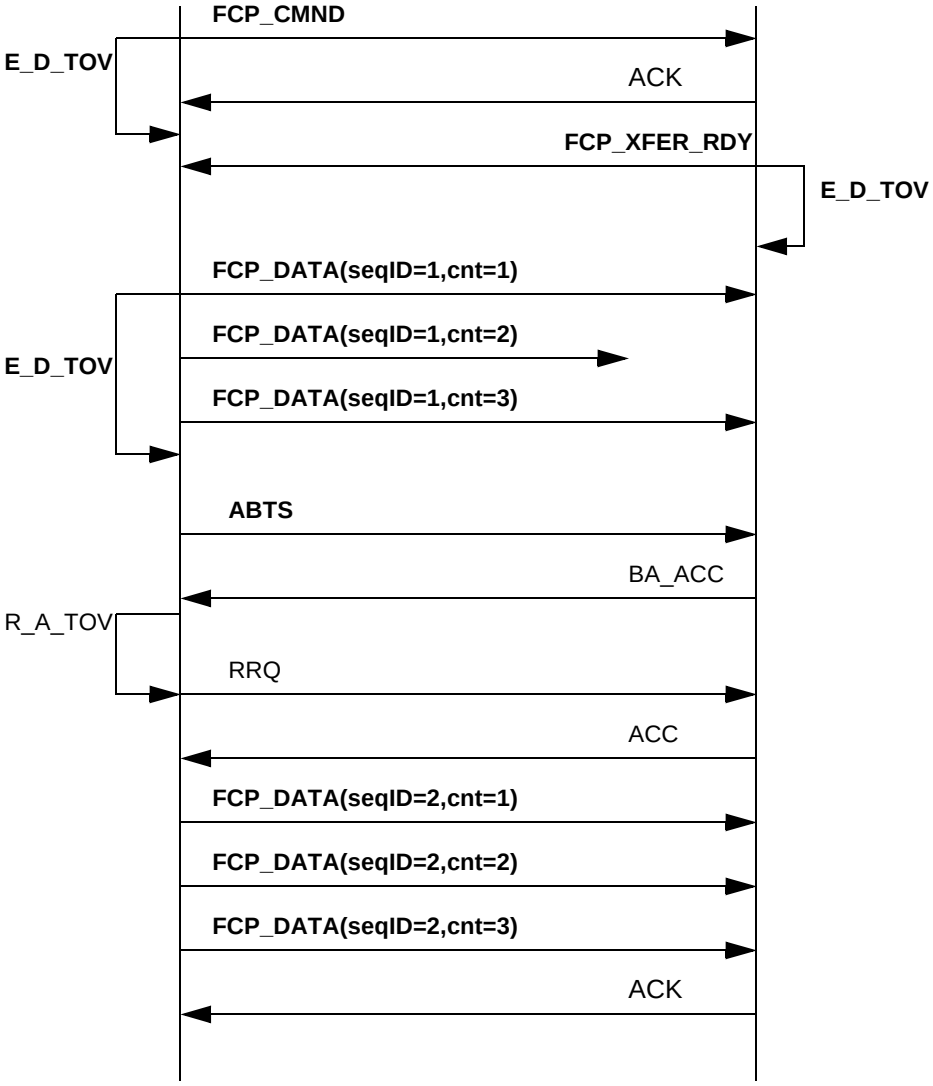
The FCP_CMND is sent by the initiator and received at the target. The target sends the ACK but it does not arrive at the initiator. The timer will expire and the initiator sends a RES. The target knows of the exchange and returns a ACC. The initiator knows the command was received and performs the Abort Sequence Protocol and maintains the exchange.

Figure 4 – Loss of FCP_XFER_RDY/FCP_RSP (or lost ACK for FCP_XFER_RDY/FCP_RSP)



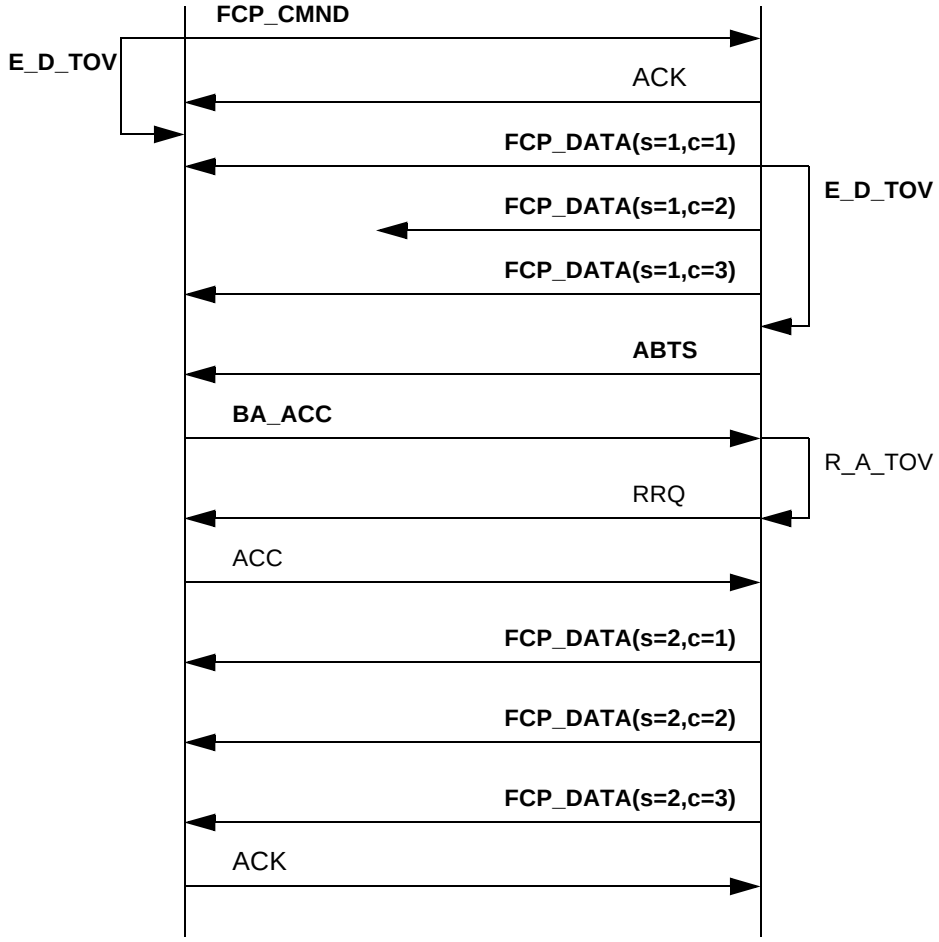
The FCP_XFER_RDY is sent by the target and not received at the initiator. The target does not see an ACK so the timer expires and it performs the Abort Sequence Protocol. After the ACC for the RRQ is received the target sends the FCP_XFER_RDY again.

Figure 5 – Loss of FCP_DATA (Write)



The FCP_DATA sequence is sent by the initiator but all the data is not received at the target so no ACK will be sent. The timer will expire and the initiator performs Abort Sequence Protocol. After the ACC for the RRQ is received the FCP_DATA sequence is retransmitted using a new sequence ID. (Note: the Tachyon currently requires the FCP_XFER_RDY to be resent also).

Figure 6 – Loss of FCP_DATA (Read)



The FCP_DATA sequence is sent by the target but all the data is not received at the initiator so no ACK will be sent. The timer will expire and the target performs Abort Sequence Protocol. After the ACC for the RRQ is received the FCP_DATA sequence is retransmitted using a new sequence ID.

5.4 Notes

a) Should work with any ACK model (ACK 0 is preferred).

b) Need an indication that an exchange was aborted in LS_RJT response (versus lost ACK for FCP_RSP).

6 SCSI Stream Devices

This clause describes commands and features applicable to SCSI stream devices with usage defined by this profile.

6.1 Applicable Classes of Service

SCSI-3 stream devices conforming to this document shall use Class 2 service with parameters as described in Table 3.

6.2 Asynchronous Event Notification (AEN)

The use of AEN by stream devices conforming to this profile is prohibited.

6.3 Command Linking

Command Linking is allowed by all stream devices conforming to this profile. Support for command linking is identified in the Inquiry data. The Flag bit of the CDB shall be set to zero.

6.4 Sequential device commands

Command and features within commands, that are not listed are optional. Interoperability between SCSI Initiators and SCSI Targets is not guaranteed if optional commands or features are used. If support of a field is listed as Required without specifying a value for that field, it is assumed that all possible values of the bit or field shall be supported in accordance with the appropriate American National Standard. Any unlisted commands/features/settings are implicitly Invokeable by the SCSI Initiator, and Allowed by the SCSI Target.

Table 2 – SCSI Tape Device Commands

Feature	Initiator	Target	Doc	Notes
ERASE	I	R	SSC	
IMMED	I	R		
LONG	I	R		
SHORT	I	A		
FORMAT MEDIUM	I	A	SSC	
INQUIRY	I	R	SPC	
Standard INQUIRY data (bytes 0-35)	I	R		
EVPD=1	I	R		
Vital Product Data page codes:				
hex‘00’ (Supported vital product pages)	I	R		
hex‘80’ (Unit serial number page)	I	R		
hex‘81’ (Implemented operations definition page)	I	A		
hex‘82’ (ASCII implemented operations definition page)	I	A		
hex‘83’ (Device identification page)	I	A		
LOAD UNLOAD	I	R	SSC	
EOT = 1	I	A		
Immed = 1	I	R		
Load = 1	I	A		
Load = 0	I	R		
Reten = 1	I	A		

Table 2 – SCSI Tape Device Commands

Feature	Initiator	Target	Doc	Notes
LOCATE	I	R		
BT = 0	I	R		
BT = 1	I	A		
CP = 0	I	R	SSC	
CP = 1	I	A		
Immed=0	I	R		
Immed=1	I	R		
LOG SELECT	I	A		
SP = 0	I	R		
SP = 1	I	A		
PCR = 1 & PLL = 0	I	R	SPC	
PC = ‘00’b	I	A		
PC = ‘01’b	I	R		
PC = ‘10’b	I	A		
PC = ‘11’b	I	A		
LOG SENSE	I	R		
SP = 0	I	R		
SP = 1	I	A		
PPC = 0	I	R		
PPC = 1	I	A		
PC = ‘00’b	I	A		
PC = ‘01’b	I	R	SPC	
PC = ‘10’b	I	A		
PC = ‘11’b	I	A		
Page Select 00	I	R		
Page Select 02	I	R		
Page Select 03	I	R		
Page Select 06	I	R		
Page Select 0C	I	R		
MODE SENSE/MODE SELECT (6 and 10)	I	R	SPC	
DBD = 0	I	R		
DBD = 1	I	R		
PC = ‘00’b	I	A		
PC = ‘01’b	I	R		
PC = ‘10’b	I	A		
PC = ‘11’b	I	A		
Page 01	I	R	SPC	
Page 02	I	R	SPC	
Page 0A	I	R	SSC	
Page 10	I	R	SSC	
Pages 11-14	I	A	SSC	
MOVE MEDIUM	I	A	SMC	1
PERSISTENT RESERVE IN	I	R	SPC	
PERSISTENT RESERVE OUT	I	R	SPC	
PREVENT/ALLOW MEDIUM REMOVAL	I	A	SSC	

Table 2 – SCSI Tape Device Commands

Feature	Initiator	Target	Doc	Notes
READ	I	R		
Fixed = 1	I	R		
Fixed = 0 (Required if Read Block Limits values differ)	I	A	SSC	
SILI = 0	I	R		
SILI = 1 (Required if Read Block Limits values differ)	I	A		
READ BLOCK LIMITS	I	R	SSC	
READ BUFFER	I	A	SPC	
READ ELEMENT STATUS	I	A		1
Vol Tag = 0	I	R	SMC	1
Vol Tag = 1	I	A		1
READ POSITION	I	R		
BT = 0	I	R	SSC	
BT = 1	I	A		
READ REVERSE	I	A		
Fixed = 1	I	R		
Fixed = 0 (Required if Read Block Limits values differ)	I	A		
SILI = 0	I	R	SSC	
SILI = 1 (Required if Read Block Limits values differ)	I	A		
Byte Ord = 0	I	A		
Byte Ord = 1	I	A		
RECEIVE DIAGNOSTIC RESULTS	I	A	SPC	
RECOVER BUFFERED DATA	I	R		
Fixed = 1	I	R		
Fixed = 0 (Required if Read Block Limits values differ)	I	A		
SILI = 0	I	R	SSC	
SILI = 1 (Required if Read Block Limits values differ)	I	A		
FIFO	I	A		
LIFO	I	R		
RELEASE(6)	I	A		
3rd Party = 0	I	R	SPC	
3rd Party = 1	P	P		
RELEASE(10)	I	R		
3rd Party = 0	I	R	SPC	
3rd Party = 1	I	A		
REPORT DENSITY SUPPORT	I	R	SSC	
REPORT LUN	I	A	SPC	
RESERVE(6)	I	A		
3rd Party = 0	I	R	SPC	
3rd Party = 1	P	P		
RESERVE(10)	I	R		
3rd Party = 0	I	R	SPC	
3rd Party = 1	I	A		
REWIND	I	R		
Immed = 0	I	R	SSC	
Immed = 1	I	R		

Table 2 – SCSI Tape Device Commands

Feature	Initiator	Target	Doc	Notes
SEND DIAGNOSTIC	I	R		
ST = 1	I	R	SPC	
ST = 0	I	A		
SPACE	I	R		
Forward	I	R	SSC	
Reverse	I	R		
Code = ‘000’b (Blocks)	I	R		
Code = ‘001’b (Filemarks)	I	R		
Code = ‘010’b (Sequential Filemarks)	I	A		
Code = ‘011’b (End-of-Data)	I	A		
Code = ‘100’b (Setmarks)	I	A		
Code = ‘101’b (Sequential Setmarks)	I	A		
TEST UNIT READY	I	R	SPC	
VERIFY	I	A		
Fixed = 1	I	R		
Fixed = 0 (Required if Read Block Limits values differ)	I	A		
BytCmp = 0	I	R	SSC	
BytCmp = 1	I	A		
Immed = 0	I	R		
Immed = 1 & BytCmp = 0	I	R		
Immed = 1 & BytCmp = 1	I	A		
WRITE	I	R		
Fixed = 1	I	R	SSC	
Fixed = 0 (Required if Read Block Limits values differ)	I	A		
WRITE BUFFER	I	A		
Mode = ‘110’b (Download ucode w/ offsets)	I	R	SPC	
Mode = ‘111’b (Download ucode w/ offsets & save)	I	R		
WRITE FILEMARK	I	R		
Count = 0	I	R		
Count > 0	I	R	SSC	
Immed = 0	I	R		
Immed = 1	I	R		

NOTES:

1 Medium changer commands and features are disabled when Inquiry data indicates MCHGR = ‘0’b and enabled when the MCHGR = ‘1’b

Register Insertion Protocol for Fibre Channel Arbitrated Loop

1. Revision History

1.1 Rev 0.9 6/1/96

Initial version

1.2 Rev 1.0 12/15/96

Changed flow control mechanism from end-to-end (a.k.a. multipoint) to nearest neighbor.

Changed fairness mechanism from arbitration on starvation to the SAT algorithm.

Open issues:

⇒ Holding the SAT token to transmit while the Bypass FIFO is not empty requires depleting the upstream neighbor of available credit. This may result in a delay in replenishing credit after the L_Port is satisfied and the SAT token is released.
Solution: employ the Aaron method of receive buffering (common cut through and receive buffers)

⇒ Since RIP packets containing R_RDYs are not flow controlled, they may overflow the Bypass FIFO of a downstream L_Port when it is holding the SAT token.
Solution: Employ the Aaron method of embedding Primitive Signals within frames (RIP packets in the case of RIP).

2. Overview

This proposal documents a new operating mode for FC-AL, which is called the Register Insertion Protocol or RIP. As the name suggests, it is based on a modification to FC-AL which adds a new operating mode based on a register insertion mechanism. The primary motivation is to reduce the latency of acquiring access to and closing the Loop. RIP completely eliminates arbitration for Loop access.

When the Loop is idle, an L_Port may transmit one frame or one or more R_RDYs at any time. RIP encapsulates the frame or R_RDY(s) with a pair of Primitive Signals called Receive or RCV. The resulting string of Transmission Words is called a RIP packet. If another RIP packet is received and must be retransmitted (i.e., it is bound for another L_Port) while the L_Port is transmitting, the packet is temporarily buffered in a Bypass FIFO, which can accommodate a maximum size RIP packet. The buffered packet is retransmitted as soon as the L_Port has completed transmitting a single RIP packet.

The Bypass FIFO must be empty before an L_Port is allowed to transmit, since it may have to temporarily buffer part of all of a maximum length RIP packet after it begins transmitting.

A frame or a set of R_RDYs are routed on the Loop by prepending each with a new Primitive Signal, RCVyx, which contains the source and destination AL_PAs. The frame or set of R_RDYs has a RCVff Primitive Signal that marks the end of the transmission. The RCVyx header, encapsulated Transmission Words, and the RCVff trailer delineate a RIP packet. This packaging and routing scheme preserves much of the existing structure of FC-AL. It allows routing based solely on Primitive Signals and avoids the need to crack the frame header and examine the D_ID.

The RIP mode of operation, based on the register insertion protocol, results in the lowest possible latency, since frame transmission may usually begin at any time. Arbitrated Loop as defined in FC-AL X3, 272-1996, (hereafter referred to as FC-AL1), typically requires one half of a Loop round trip on average for arbitration before an L_Port may begin frame transmission. FC-AL1 also permits only a single point-to-point full duplex circuit per connection. It also requires one round trip to close the Loop. Other L_Ports which have frames to transmit must wait until the current circuit is terminated when the L_Ports relinquish control by closing. Thus, short command packets may be blocked behind large data transfers. In contrast, RIP mode allows multiple L_Ports to interleave frame transmission on the Loop. RIP allows short command packets to be interleaved with large data transfers.

RIP may allow improved class 2 performance, since ACK frames may be transmitted without arbitrating for the Loop. With FC-AL1, if the destination L_Port is closed immediately after class 2 frame reception, it may have to arbitrate and acquire Loop access just to return the ACK frame. Because of the nature of RIP, class 1 is not supported.

The goal of RIP is to minimize latency, which is often at odds with providing bandwidth and latency guarantees. In other words, the average latency of RIP is much lower than FC-AL1. RIP supports a variable packet or frame size. With a variable frame size and on-demand use of available bandwidth, RIP will efficiently utilize Loop resources. As a result, RIP will exhibit higher Loop utilization than FC-AL1.

RIP is designed to support asynchronous or dynamic workloads that typically exhibit a wide range of transfer length and bandwidth demand. It is also designed to support applications that frequently need to synchronize and communicate in a cluster or multi-computer environment. Such applications include Multi-Initiator Disk Arrays, On Line Transaction Processing (OLTP), Data Warehousing, Data Mining, large scale Network File Servers, scaleable Web servers, and parallel processing.

3. Register Insertion Protocol Definition

This proposal defines a new operational mode for Fiber Channel Arbitrated Loop. It is based on a register insertion mechanism, which is an extension for FC-AL2. The new operational mode is called the Register Insertion Protocol or RIP.

RIP mode is entered at the end of loop initialization when the temporary Loop Master discovers whether all L_Ports are capable and enabled for RIP mode. If true, all L_Ports are instructed to enter RIP mode. Once in RIP mode, all L_Ports remain in a Monitoring like state. Thus, L_Ports never enter the OPEN or OPENED states.

3.1 Basic Operation

Since the Loop is a closed, continuous circuit through all L_Ports, a routing mechanism must be provided for each frame or R_RDY Primitive Signal. For unicast operation, this is accomplished by immediately prepending each frame or one or more R_RDYs with a Receive (RCVyx) Primitive Signal. The frame or

R_RDYs have two Fill Words, a RCVff, and two additional Fill Words appended to it. This string of transmission words is called a RIP packet.

The destination L_Port whose AL_PA matches the “y” in the RCVyx Primitive Signal removes from the Loop the RCVyx, the frame or R_RDY(s) and all other Transmission Words between the RCVyx and RCVff inclusive. In addition, a RCVfx (f = hex ‘FF’) Primitive Signal provides support for broadcast, while a RCVef/RCVye pair (e = hex ‘FE’) supports multicast or selective replicate.

An L_Port may transmit one RIP packet whenever it has received at least six consecutive Idles and its Bypass FIFO is empty. During transmission of a RIP packet, if any frame or R_RDYs which are addressed to another L_Port or a MRKtx are received , they are stored in a special purpose receive buffer, called the Bypass FIFO. When the sender completes transmission of a single frame or one or more R_RDYs, it must immediately transmit any Transmission Words in the Bypass FIFO. The Bypass FIFO must be completely drained before any additional frames or R_RDYs may be transmitted. All transmitted frames and R_RDYs are packaged into RIP packets.

3.2 RIP L_Port Structure

The logical structure of a RIP capable L_Port is shown in Figure 1. A 3-to-1 mux connects three different sources to the transmitter output:

1) receiver input (“Idle” state)
2) Bypass FIFO output (“Drain” state)
3) Transmit (TX) FIFO output (“Inject” state)

The Receive (RCV) FIFO is shown for descriptive purposes only and is not required for RIP mode. It is sufficient to deliver the contents of a RIP packet directly to FC-2. The Transmit (TX) FIFO shall be at least as large as the largest RIP packet that the L_Port constructs.

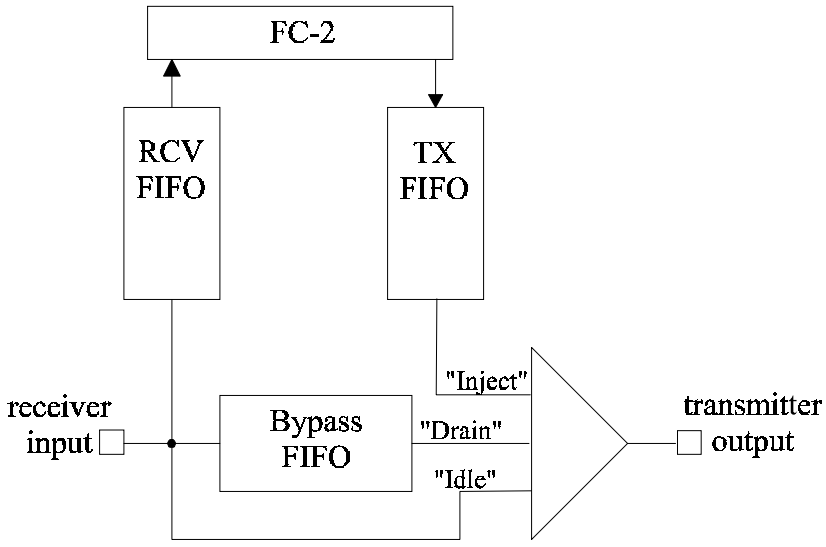


Figure 1: RIP L_Port Structure

When the L_Port is in the Idle state, it is repeating the receiver input bound for other L_Ports to the transmitter output, including any elasticity management. When the L_Port is transmitting a RIP packet the L_Port is in the Inject state and the TX FIFO output is connected to the transmitter output. If one or

more RIP packets or MRKtx Primitive Signals for other L_Ports are received while the L_Port is transmitting a RIP packet, then they are buffered in the Bypass FIFO. When the L_Port has emptied the TX FIFO, which shall only contain a single RIP packet, the L_Port enters the Drain state where the transmitter output is connected to the Bypass FIFO output if it is not empty. Otherwise, the receiver input is connected to the transmitter output and the L_Port enters the Idle state.

The Bypass FIFO output is connected to the transmitter output as long as it is not empty. If any subsequent RIP packets destined for other ports are received while the Bypass FIFO is connected to the transmitter output, the received Transmission Words are appended to the Bypass FIFO. When the Bypass FIFO is empty, the receiver input is connected to the transmitter output. Any frames or R_RDYs contained in RIP packets addressed to the L_Port are placed in the RCV FIFO and shall not be buffered in the Bypass FIFO.

3.3 RIP Primitive Signals

3.3.1 New Primitive Signals

RIP defines two new Primitive Signals. One is named Receive or RCV, while the second is named SATisfied or SAT. RCV is defined by the Ordered Set K28.5, Dxx.y, nd1, nd2, where each nd is a neutral disparity transmission character. SAT is defined by the Ordered Set K28.5, Dxx.y, Dxx.y, Dxx.y.

The RCV and SAT Primitive Signals shall follow the FC-PH rule for transmitting R_RDYs (i.e., two Fill Words shall precede and follow each RCV or SAT with at least six Primitive Signals between frames).

RCV provides seven different functions, depending on the values for nd[1, 2] in the third and fourth transmission characters of the RCV Ordered Set. See Table 1. A variation of SAT, SAT', is a special form of SAT originated by single port Nodes and Fabric ports in certain topologies.

Table 1 - RIP Primitive Signals

RIP Primitive Signal	nd1 (char. 3)	nd2 (char 4)	Function
RCVyx	<= EF	<= EF	Unicast Packet
RCVfx	FF	<= EF	Broadcast Replicate Packet
RCVex	FE	<= EF	Selective Replicate Packet
RCVye	<= EF	FE	Selective Replicate Receive
RCVff	FF	FF	RIP packet EOT
RCVbb	FB	FB	Negotiate RIP mode
RCVdd	FD	FD	Enter RIP mode
SAT	Dxx.y	Dxx.y	Satisfied
SAT'	Dxx.y	Dxx.y	Satisfied'

3.3.1.1 RCVyx (Unicast Packet)

RCVyx, where x <= hex 'FE' and y <= hex 'FE', is transmitted on a Loop by a participating L_Port to transmit RIP packet to a destination L_Port. The RCVyx shall contain the AL_PD (destination - y value) of the L_Port to receive the frame or R_RDY and the AL_PS (source - x value) of the L_Port transmitting the RCVyx.

An L_Port that receives the RCVyx, where y = AL_PA of the L_Port, shall remove all Transmission Words starting with the RCVyx and the next RCVff inclusive. The L_Port shall also store in the Receive FIFO the received frame or R_RDY(s) contained within the RIP packet.

An L_Port that receives the RCVyx, where y <> AL_PA of the L_Port and x = AL_PA of the L_Port, shall discard all Transmission Words between the RCVyx and the next RCVff inclusive.

3.3.1.2 RCVfx (Broadcast Replicate Packet)

RCVfx, where f = hex 'FF' and x <= hex 'FE', is transmitted on a Loop by a participating L_Port to transmit a single frame to all other L_Ports on the Loop. The RCVfx shall contain the AL_PS (source - x value) of the L_Port transmitting the RCVfx.

An L_Port that receives the RCVfx, where x <> AL_PA of the L_Port, shall retransmit all Transmission Words between the RCVfx and RCVff inclusive. The L_Port shall also store in the Receive FIFO the received frame contained within the RIP packet.

An L_Port that receives the RCVfx, where x = AL_PA of the L_Port, shall discard all Transmission Words between the RCVfx and the next RCVff inclusive.

3.3.1.3 RCVex/RCVye (Selective Replicate Packet)

RCVex, where e = hex 'FE' and x <= hex 'FE', is transmitted on a Loop by a participating L_Port to transmit a single frame to a subset or group of L_Ports on the Loop. The RCVex shall contain the AL_PS (source - x value) of the L_Port transmitting the RCVex.

In addition, the L_Port transmitting the RCVex shall transmit immediately following the RCVex one RCVye with appropriate Fill Words for each L_Port in the group. The RCVye, where e = hex 'FE', shall contain the AL_PD (destination - y value) of the L_Port to copy the frame.

An L_Port that receives the RCVex, where x <> AL_PA of the L_Port, shall retransmit all Transmission Words between the RCVex and the next RCVff inclusive, with the following exception. If the L_Port receives a RCVye, where y = AL_PA of the L_Port, it shall substitute the CFW; the L_Port shall also store in the Receive FIFO the received frame contained within the RIP packet.

An L_Port that receives the RCVex, where x = AL_PA of the L_Port, shall discard all Transmission Words between the RCVex and the next RCVff inclusive.

3.3.1.4 RCVff (RIP packet EOT)

RCVff, where f = hex 'FF', is transmitted on the Loop by a participating L_Port as the last Transmission Word of a RIP packet.

3.3.1.5 RCVbb (negotiate RIP mode)

RCVbb, where b = hex 'FB', is transmitted on the Loop in the Open-Init state by the temporary Loop Master if it is capable and enabled for RIP mode. If the temporary Loop master subsequently receives a RCVbb, it shall discard the RCVbb and then transmit a RCVdd. If the E_D_TOV timer expires before receiving a RCVbb, the temporary Loop Master shall transmit a CLS and make the transition to the Monitoring state.

If an L_Port in the OPEN-INIT state that is not the temporary Loop Master receives a RCVbb and it is capable and enabled for RIP mode, it shall retransmit the RCVbb. Otherwise, it shall be discarded.

3.3.1.6 RCVdd (enter RIP mode)

RCVdd, where d = hex 'FD', is transmitted on the Loop in the Open-Init state by the temporary Loop Master when it has sent and then receives a RCVbb Primitive Signal. If the temporary Loop master subsequently receives a RCVdd, it shall make the transition to the Idle state. If the E_D_TOV timer expires before receiving a RCVdd, the temporary Loop Master shall transmit a CLS and make the transition to the Initializing state.

If an L_Port in the OPEN-INIT state receives a RCVdd, it shall make the transition to the Idle state.

3.3.1.7 SAT (SATisfied)

SAT is retransmitted on a Loop by an L_Port if it has no RIP packets to originate, or if it has RIP packets to originate and it has reached it's HOLD quota since the last SAT visit. SAT is unconditionally retransmitted if the L_Port has originated it's IDLE quota for RIP packets since the last SAT visit. A SAT visit is defined as the next receipt of SAT after an L_Port has transmitted a SAT. In a cyclic path such as a loop, the SAT token will visit each node exactly once during a complete circulation around the loop.

3.3.1.8 SAT' (SATisfied')

For dual, counter rotating Loop topologies with single port Nodes and/or Fabric Nodes at either end of the loop, SAT is retransmitted on the Loop by single port or Fabric Node as SAT'. SAT' is re-transmitted on the Loop by a single port or Fabric Node as SAT. All dual port Nodes immediately retransmit SAT' without affecting any transmission quotas (Hold, Idle, Frame Counter).

3.3.2 Normal FC-AL Primitive Signals

The MRK tx Primitive Signal functions as defined in FC-AL, Rev. X3,272-1996.

3.3.3 Unused FC-AL Primitive Signals

The unused FC-AL Primitive Signals are ARB, OPN, CLS, and DHD. If received by an L_Port in RIP mode, the CFW shall be substituted.

3.4 RIP Characteristics

3.4.1 RIP Packet Structure

Figure 2 illustrates the 4 different types of RIP packets. The symbol {X} represents one or more occurrences of X (i.e., X, X, X ...). VDW is any Valid Data Word.

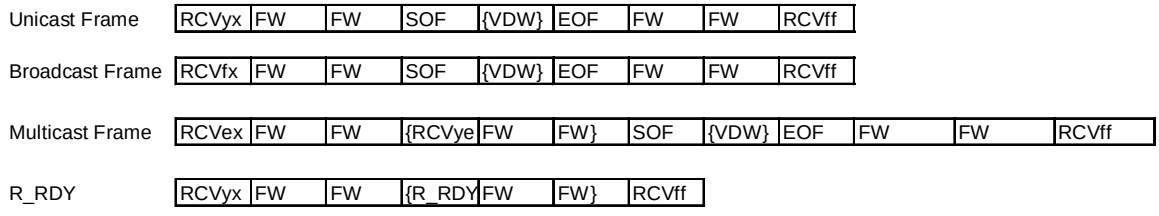


Figure 2: RIP Packet Structure

3.4.2 Bypass FIFO Size

The Bypass FIFO shall be large enough to hold one maximum length frame plus any required leading and trailing Fill Words and Primitive Signals. The maximum length of a transmitted RIP packet occurs when a full sized frame is selectively replicated to all 126 other L_Ports on the Loop (this is an unlikely occurrence, since a broadcast may be used instead). This is equal to 3688 bytes, as indicated in Figure 3.

Transmission Words	Byte Count
RCVex	4
2 Fill Words	8
126 * (RCVye, 2 Fill Words)	1492
SOF	4
Header	24
Data Field	2112
CRC	4
EOF	4
2 Fill Words	8
RCVff	4
6 Fill Words	24
Total:	3688

Figure 3: Bypass FIFO Size

3.4.3 Drain Behavior

The Bypass FIFO will empty at the rate at which Fill Words are received that are not between a RCVyx, RCVfx or RCVex and the next RCVff. For example, assume the Bypass FIFO has N Transmission Words buffered. If only Fill Words are received while in the Drain state then the Bypass FIFO will be emptied in N Transmission Word times. If however, a sequence of consecutive RIP packets are received while in the Drain state, then the Bypass FIFO cannot drain. It is loaded and unloaded on every Transmission Word. Thus the Bypass FIFO count will remain at N. If a mixture of RIP packets and Fill Words are received, the Bypass FIFO count is decremented for every Fill Word received that is not encapsulated by the RCVyx | RCVfx | RCVex and RCVff pair.

3.4.4 Port Delay

While an L_Port is in Drain mode, the length of the Loop and the delay through the L_Port is increased by the number of Transmission Words buffered in its Bypass FIFO. If N Transmission Words are buffered while in the Inject state, then the port delay will be a maximum of N+6 Transmission Words upon entering the Drain state. As the Bypass FIFO empties, the port delay decreases. Eventually the Bypass FIFO will drain and the delay through the port returns to a maximum of six Transmission Words per L_Port as specified by FC-AL1.

3.4.5 Loop Delay

The length of a Loop in Transmission Words temporarily expands by the combined contents of all Bypass FIFOs. In the worst case, the maximum length occurs when all L_Ports simultaneously transmit a full selective replicate RIP packet. For M L-Ports in a Loop, the transmission length of the Loop temporarily expands by M * 3688/4 or M * 922 Transmission Words.

3.5 RIP Initialization

Before the temporary Loop Master transmits a CLS to exit initialization, it shall determine if RIP mode is possible. If it is capable and enabled for RIP mode, the Loop Master shall transmit a RCVbb Primitive Signal, where b = hex 'FB'. If an L_Port that is not the temporary Loop Master receives RCVbb and it is capable and enabled for RIP mode, it shall retransmit the RCVbb. If that L_Port is neither capable nor enabled for RIP mode, it shall discard the received RCVbb Primitive Signal. If all L_Ports on the Loop are capable and enabled for RIP mode, the temporary Loop Master receives a RCVbb. The Loop Master shall discard the RCVbb and transmit a RCVdd, where f = hex 'FD'.

If an L_Port in the OPEN-INIT state that is not the temporary Loop Master receives a RCVdd it shall enter RIP mode and make the transition to the Idle state. If an L_Port receives a CLS, it shall make the transition to the Monitoring state. When the temporary Loop Master receives a RCVdd it shall discard the RCVdd and make the transition to the Idle state. If it receives a CLS, it shall make the transition to the Monitoring state. If the Loop Master does not receive a RCVbb within E_D_TOV and times out, it shall transmit a CLS and make the transition to the Initializing state, since some L_Ports may be in RIP mode and some may not.

If the temporary Loop Master is not RIP capable or not RIP enabled, it shall send a CLS and shall not send a RCVbb or RCVdd. The temporary Loop Master and all other L_Ports that have successfully completed initialization shall make the transition to the Monitoring state.

3.6 Transmission and Reception Rules

3.6.1 Unicast Frame or R_RDY

3.6.1.1 Transmission from Idle State

An L_Port may attempt to transmit whenever it is in the Idle state (i.e., the Bypass FIFO is empty) and it has originated less frames than its Idle Quota since the last SAT visit. In the Idle state the L_Port is in a selective repeat mode with the receiver input is connected to the transmitter output. A transmit request is indicated when any REQ(transmit) loop control is asserted to the LPSM. Transmission onto the Loop may begin after at least six consecutive Idles have been received. This rule prevents an L_Port from injecting its RIP packet in the middle of a cut through RIP packet.

The following conditions must be true before an L_Port begins RIP packet transmission:

- At least six consecutive Idles have been received
- Bypass FIFO is empty
- The Downstream Neighbor BB_Credit is greater than zero.
- Frame Counter <= Idle Quota & RIP packet contains a frame
- REQ(transmit) asserted

When the above conditions are true, the L_Port shall make the transition to the Inject state and the output of the Transmit FIFO shall be connected to the transmitter output on a Transmission Word boundary. The L_Port shall transmit a RCVyx Primitive Signal, where "x" equals the AL_PA of the transmitting L_Port

and “y” equals the AL_PA of the intended destination L_Port. Two Fill Words shall immediately follow the RCVyx. A single frame or one or more R_RDYs followed by two trailing Fill Words, a RCVff, and six additional Fill Words shall then be transmitted. The Inject state may be exited when the second Fill Word following the RCVff has been transmitted. If a frame was transmitted (i.e., it was not an R_RDY), the L_Port shall increment the Frame Counter.

For each transmission request, the L_Port shall transmit only a single frame or one or R_RDYs. If the L_Port has multiple frames to transmit (e. g., a sequence), it may transmit one or more RIP packets immediately if the conditions required for transmission are still true. If the L_Port has no more frames to transmit (i.e., REQ(transmit) is deasserted), and if the Bypass FIFO is empty, it shall make the transition to the Idle state. If the Bypass FIFO is not empty, the L_Port shall make the transition to the Drain state.

If a RCVyx Primitive Signal is received, where “y” is equal to the AL_PA of the L_Port, the following RIP packet is addressed to the L_Port. The RCVyx and the RIP packet are removed from the Loop and the frame or R_RDY is stored in a Receive FIFO. All Transmission Words between the RCVyx and the next RCVff inclusive are discarded. RIP packets addressed to the receiving L_Port shall not be stored in the Bypass FIFO.

If a RCVyx Primitive Signal is received, where “y” is not equal to the AL_PA of the L_Port and “x” is equal to the AL_PA of the L_Port, the RIP packet has not been removed by the addressed L_Port (“y” in the RCVyx) and it shall be removed from the Loop.

If a RCVyx Primitive Signal is received, where neither “y” nor “x” is equal to the AL_PA of the L_Port, the RIP packet is not for this L_Port. All Transmission Words between the RCVyx and the next RCVff inclusive shall be placed into the Bypass FIFO. Subsequent Fill Words until the next RCVyx | RCVfx | RCVex are discarded and shall not be placed in the Bypass FIFO.

If additional RCVyx primitives are received where neither “y” nor “x” is equal to the AL_PA of the L_Port, of if a RCVfx or RCVex is received, then all Transmission Words between the RCVyx | RCVfx | RCVex and the RCVff inclusive are placed into the Bypass FIFO.

For each MRKtx received, where x <> AL_PA of the L_Port, it is also stored in the Bypass FIFO, along with up to six trailing Fill Words. This insertion process into the Bypass FIFO may be repeated while in the Inject state. When the L_Port has finished transmitting the RIP packet, the L_Port shall make the transition to the Drain state if the Bypass FIFO is not empty. Otherwise, the L_Port shall make the transition to the Idle state.

In the Drain state, the output of the Bypass FIFO is connected to the transmitter output. Each Transmission Word is shifted out of the Bypass FIFO and transmitted onto the Loop. While in the Drain state, some received Transmission Words may be appended to the Bypass FIFO using the same set of rules as for the Inject state. Since all other received Transmission Words are not loaded into the Bypass FIFO, it will eventually empty and the L_Port shall then make the transition to the Idle state.

3.6.1.2 Transmission in the Drain State

An L_Port may attempt to transmit whenever it is in the Drain state (i.e., the Bypass FIFO is not empty), it has originated less frames than it’s Hold Quota since the last SAT visit, and it is currently holding the SAT token. A transmit request is indicated when any REQ(transmit) loop control is asserted to the LPSM. Before transmission can begin, the L_Port must exhaust the available credit to the Upstream Neighbor by withholding the transmission of R_RDYs, even if receive buffers are available. This enables an L_Port to originate a RIP packet even if the Bypass FIFO is full. Transmission onto the Loop may begin after the Upstream Neighbor BB_Credit is zero and at least six consecutive Idles have been *re-transmitted* by the requesting L_Port. The later rule prevents an L_Port from injecting a RIP packet in the middle of a RIP packet being re-transmitted from the Bypass FIFO.

The following conditions must be true before an L_Port begins RIP packet transmission in the Drain state:

- At least six consecutive Idles have been re-transmitted (from the Bypass FIFO)
- Frame Counter <= Hold Quota
- Holds the SAT token
- The Upstream Neighbor BB_Credit is zero.
- The Downstream Neighbor BB_Credit is greater than zero.
- REQ(transmit) asserted

When the above conditions are true, the L_Port shall make the transition to the Inject state and the output of the Transmit FIFO shall be connected to the transmitter output on a Transmission Word boundary. The L_Port shall transmit a RCVyx Primitive Signal, where "x" equals the AL_PA of the transmitting L_Port and "y" equals the AL_PA of the intended destination L_Port. Two Fill Words shall immediately follow the RCVyx. A single frame one or more R_RDYs followed by two trailing Fill Words, a RCVff, and six additional Fill Words shall then be transmitted.

After the RCVff trailer is transmitted, the L_Port shall increment the Frame Counter if the transmitted RIP packet contained a frame. If the Frame Counter is less than or equal to the Hold Quota and REQ(transmit) is still asserted (there are additional frames or R_RDYs to transmit), the L_Port shall re-enter the Inject state and transmit another frame or one or more R_RDYs. If the Frame Counter is greater than the Hold Quota, the L_Port shall reset the Frame Counter and retransmit the SAT token. It shall then transition to the Drain state. The L_Port shall also replenish the Upstream Neighbor BB_Credit if receive buffers are available.

3.6.2 Broadcast Replicate

3.6.2.1 Transmission in the Idle State

If an L_Port needs to send a frame to every other L_Port on the Loop, it shall indicate this request by asserting REQ(transmit fx) to the LPSM. If the Bypass FIFO is empty, the Downstream Neighbor BB_Credit is greater than zero, and the Frame Counter is less than or equal to the Idle Quota, the L_Port may begin transmission if at least six consecutive Idles have been received. The L_Port shall then transmit a RCVfx Primitive Signal, where f = hex 'FF' and x = AL_PA of the L_Port. The L_Port shall then transmit two Fill Words and a single frame, followed by two Fill Words, a RCVff Primitive Signal and six trailing Fill Words. The frame shall be transmitted using class 3 SOF delimiter and R_RDYs are suppressed for frames received via a RCVfx. The L_Port shall increment the Frame Counter if a frame was transmitted.

If the L_Port has additional frames to transmit, it may continue to transmit frames as long as the Bypass FIFO remains empty and it's Frame Counter is less than or equal to it's Idle Quota. Each frame is transmitted using the same Transmission Word sequence (i.e., a RCVfx, 2 Fill Words, one frame, a RCVff, and six Fill Words). If the Bypass FIFO is not empty at the end of frame transmission and the L_Port is not holding the SAT token, the L_Port shall make the transition to the Drain state before it transmits additional frames. If the Bypass FIFO is empty and REQ(transmit) is deasserted, the L_Port shall make the transition to the Idle state.

If the L_Port receives a RCVfx, where x = AL_PA of the L_Port and f = hex 'FF', the L_Port shall discard all Transmission Words between the RCVyx and the RCVff inclusive. If the L_Port is in the Idle state, it shall transmit the Current Fill Word for each Transmission Word removed from the Loop.

If the L_Port receives a RCVfx, where x <> AL_PA of the L_Port and f = hex 'FF', the L_Port is a receiver of a broadcast frame. The L_Port shall cause retransmission of the RCVfx. The L_Port shall also copy the encapsulated frame to the Receive FIFO and it shall also cause retransmission of the frame. The phrase "cause retransmission" has the following interpretation, depending on the current state of the LPSM:
- If the LPSM is in the Idle state, the RIP packet will be retransmitted immediately.
- If the LPSM is in the Inject state, the RIP packet will be stored in the Bypass FIFO for later

retransmission when the L_Port enters the Drain state.
- If the LPSM is in the Drain state, the RIP packet will be appended to the Bypass FIFO for retransmission.

If the L_Port is in the Idle state, it shall substitute the Current Fill Word when required.

3.6.2.2 Transmission in the Drain State

The L_Port may transmit a broadcast frame if it has posted broadcast frame request - REQ(transmit fx), it's Frame Counter is less than it's Hold quota, it is holding the SAT token, its Upstream Neighbor BB_Credit has been depleted to zero, and its Downstream Neighbor BB_Credit is greater than zero. These rules are identical for transmitting a Unicast frame (see Section 2.6.1.2).

3.6.3 Selective Replicate

3.6.3.1 Transmission in the Idle State

If an L_Port needs to send a frame to a subset (up to N -1) of L_Ports on the Loop, it shall indicate this request by asserting REQ(transmit ex, group) to the LPSM. The "group" is a list of AL_PAs (up to 126) that are destinations for the frame transmitted by the L_Port.

If the Bypass FIFO is empty, the Downstream Neighbor BB_Credit is greater than zero, and the Frame Counter is less than or equal to the Idle Quota, the L_Port may begin transmission if at least six consecutive Idles have been received. The L_Port shall then transmit a RCVex Primitive Signal, where x = AL_PA of the L_Port. The L_Port shall then transmit two Fill Words. For each AL_PA in the group, the L_Port shall transmit a RCVye, followed by two Fill Words, where y = AL_PA of each successive AL_PA in the group. After the last RCVye for the group has been transmitted (i.e., RCVye followed by two Fill Words), the L_Port shall transmit a single frame followed by two Fill Words, a RCVff and six trailing Fill Words.

If the L_Port has additional frames to transmit, it may continue to transmit frames as long as the Bypass FIFO remains empty, the Downstream Neighbor BB_Credit is greater than zero, and the Frame Counter is less than or equal to the Idle Quota. Each frame is transmitted using the same Transmission Word sequence (i.e., a RCVex, 2 Fill Words, up to N-1 (RCVye and 2 Fill Words), one frame, two Fill Words, a RCVff and six trailing Fill Words). If the Bypass FIFO is not empty at the end of any frame and the L_Port is not holding the SAT token, or if the L_Port's Frame Counter is greater than it's Hold quota, the L_Port shall make the transition to the Drain state before it transmits additional frames.

If the L_Port receives a RCVex, where x = AL_PA of the L_Port, the L_Port shall discard all Transmission Words between the RCVex and the RCVff inclusive. If the L_Port is in the Idle state, it shall transmit the Current Fill Word for each Transmission Word removed from the Loop.

If the L_Port receives a RCVex, where x <> AL_PA of the L_Port, then the L_Port shall cause retransmission of the RCVex.

If the L_Port receives a RCVye, where y = AL_PA of the L_Port, it shall cause retransmission of the RCVye. The L_Port shall copy the encapsulated frame to the Receive FIFO of the L_Port and it shall also cause retransmission of RIP packet. For any other RCVye (i.e., y <> AL_PA), the L-Port shall cause retransmission of the RIP packet (all transmissions words between the RCVye and RCVff inclusive.

If the L_Port is in the Idle state, it shall substitute the Current Fill Word when required.

3.6.3.2 Transmission in the Drain State

The L_Port may transmit a multicast frame if it has posted a multicast frame request - REQ(transmit ex, group), it's Frame Counter is less than it's Hold quota, and it is holding the SAT token. The rules are identical for transmitting a Unicast frame (see Section 2.6.1.2).

3.7 Fairness

An L_Port is blocked from transmitting a RIP packet when the Bypass FIFO is not empty. If an L_Port cannot empty the Bypass FIFO after an extended period of time, then it must suppress the injection of new RIP packets into the Loop by other L_Ports. This causes the transmission of Fill Words by other L_Ports, which allows its Bypass FIFO to drain. The SAT algorithm described in this section is employed to equally share loop bandwidth between originators and achieve fairness.

3.7.1 SAT

The origination of RIP packets is controlled by the SAT algorithm where SAT refers to Satisfied. The SAT algorithm helps balance between spatial reuse of the individual links and any L_Port being starved of bandwidth.

3.7.2 Terminology

The Hold Quota is used to determine how many RIP packets an L_Port may originate while holding cut through RIP packets in the Bypass FIFO. The Idle Quota is used to determine the number of frames an L_Port may originate if no cut through traffic exists (i.e., the Bypass FIFO is empty). The Reflection Flag is one of the factors used to instruct the L_Port to reflect SAT and SAT' Primitive Signals, breaking the loop or string into two SAT regions. The Frame Counter is a counter that counts the number of frames originated since the last time the SAT Primitive Signal was forwarded or reflected by the L_Port. The origination of R_RDYs does not change the value of the Frame Counter. Forwarding or reflecting a SAT Primitive Signal sets the Frame Counter to zero. Originated means that the trailing RCVff of an originating RIP packet has been sent. Satisfied refers to the condition of an L_Port when either of the two following conditions is met: 1) The L_Port has originated at least Hold Quota number of frames since it previously forwarded or reflected a SAT Primitive Signal; 2) The L_Port has no frames to originate when holding the SAT Primitive Signal.

3.7.3 Reflection versus forwarding

If the Reflection Flag is cleared, an L_Port shall forward SAT and SAT' Primitive Signals. If the Reflection Flag is set, an L_Port shall reflect SAT and SAT' Primitive Signals.

An L_Port shall forward a SAT Primitive Signal after receiving a SAT Primitive Signal by holding it according to the rules of the SAT algorithm and then originating a SAT Primitive Signal. A dual port Node shall unconditionally forward a SAT' Primitive Signal after receiving a SAT' Primitive Signal.

When the reflection flag is set, an L_Port shall reflect a SAT Primitive Signal after receiving a SAT Primitive Signal by holding it according to the rules of the SAT algorithm and then generating a SAT' Primitive Signal on the outbound link of the same L_Port that received the SAT Primitive Signal. A SAT' Primitive Signal is reflected unconditionally by generating a SAT Primitive Signal on the outbound link of the same L_Port that received the SAT' Primitive Signal.

3.7.4 The SAT algorithm

The SAT algorithm will operate correctly on the following topologies:

- single loop
- dual loops rotating in the same direction (Iso loops)
- dual loops rotating in the opposite direction (counter rotating)
- strings (counter rotating loops where the two end Nodes and single port Nodes)

The Node at the end of a String may be a single port Node, a dual port Node with one port not operational or a switch or Fabric Node (a dual port Node with both ports operational is not at the end of a string by definition).

3.7.4.1 Counter rotating loops

The SAT algorithm is described initially for a loop of dual port Nodes, with enhancements for a String shown in the next section. Frames entering a loop are controlled by a SAT Primitive Signal that circulates in the opposite direction to the frames that it controls, as shown in Figure 4. Circulating the SAT Primitive Signal in the opposite direction maximizes spatial reuse.

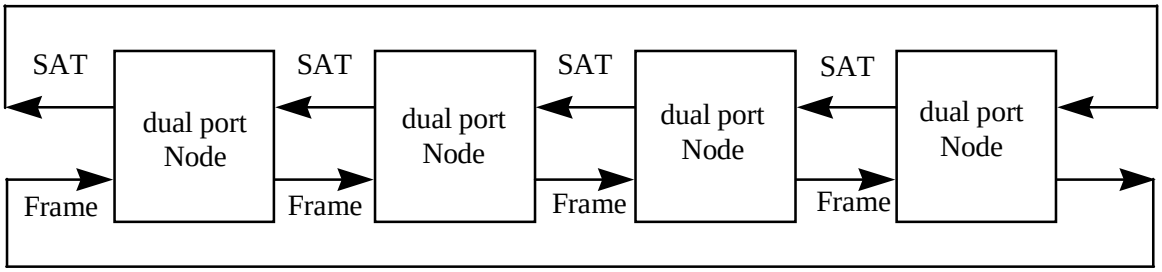


Figure 4: SAT operation on a dual, counter rotating loop

The SAT algorithm is applied independently to clockwise and counter-clockwise traffic. Therefore there are two SAT Primitive Signals circulating on the loop, one in each direction.

When an L_Port receives a SAT Primitive Signal it forwards or reflects the Primitive Signal via the other L_Port on the same Node to the next Node if it is Satisfied. Otherwise the L_Port holds the Primitive Signal until it is Satisfied. Then it forwards or reflects the SAT Primitive Signal and sets the Frame Counter to zero.

R_RDYs are not controlled by the SAT algorithm and, subject to the link protocol, they may be sent at any time. An L_Port may originate a frame, subject to the link protocol, under either of the following conditions: a) The L_Port is holding the SAT Primitive Signal and the L_Port is not Satisfied; b) There is no frame waiting to be forwarded from another L_Port, and the Frame Counter is less than the Idle Quota.

The ability of an L_Port to originate frames is broken into two cases (a and b), and one case (c) not able to originate frames: a) Frame Counter <= Hold quota. Before the SAT Primitive Signal arrives, the L_Port shall originate frames only when there are no cut through frames pending (the Bypass FIFO is empty). After the SAT Primitive Signal arrives, the L_Port shall hold RIP packets in the Bypass FIFO until Satisfied, then the SAT Primitive Signal is forwarded or reflected and the Frame Counter shall be set to zero; b) Frame Counter > Hold quota and Frame Counter <= Idle Quota. Before the SAT Primitive Signal arrives, the L_Port shall originate frames only when there are no cut through RIP packets pending (the Bypass FIFO is not empty). When the SAT Primitive Signal is received, it shall be forwarded or reflected and the Frame Counter shall be set to zero; c) Frame Counter > Idle Quota. The L_Port shall not originate any frames. When the SAT Primitive Signal is received, it shall be forwarded or reflected, the Frame Counter shall be set to zero and frames may be originated according to the SAT algorithm.

3.7.4.2 Strings

The extensions to the algorithm for a String are as follows: a) If an L_Port at the end of a String receives a SAT Primitive Signal then it reflects the SAT Primitive Signal as a SAT' Primitive Signal traveling in the opposite direction after it is Satisfied (see Figure 5). b) A dual port Node with both ports operational forwards SAT' received from either port unconditionally. If an L_Port at the end of a String receives a

SAT' Primitive Signal then it reflects the SAT' character as a SAT Primitive Signal traveling in the opposite direction immediately without resetting the Frame Counter.

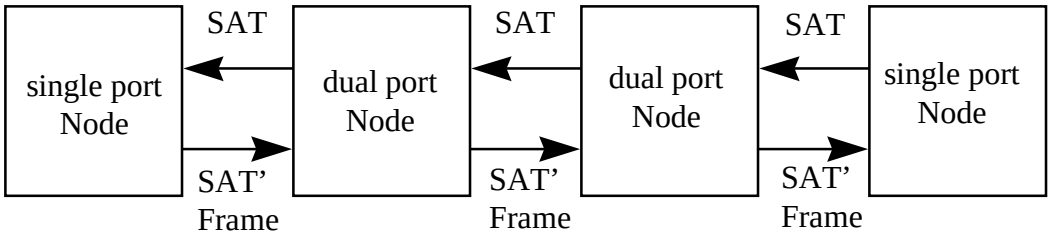


Figure 5: SAT/SAT' on a String

The reflection of SAT as SAT' Primitive Signal and SAT' as SAT Primitive Signal is dynamic. If a String of dual port nodes is connected into a closed loop then SAT' Primitive Signal are no longer generated. To avoid a previous SAT' Primitive Signal circulating indefinitely a dual port Node with both ports operational shall discard SAT' Primitive Signals as follows: a) a SAT' Primitive Signal received on port 1 is discarded if no SAT Primitive Signal has been received on port 2 since the previous SAT' Primitive Signal was received on port 1; b) a SAT' Primitive Signal received on port 2 is discarded if no SAT Primitive Signal has been received on port 1 since the previous SAT' Primitive Signal was received on port 2.

3.7.4.3 Single Loops

On single loop topologies, the SAT token must flow in the same direction as frames and RIP packets. All other aspects of the SAT algorithm operation are identical.

3.7.4.4 Dual Iso Loops

A dual, iso loop configuration is identical to a single loop topology with respect to operation of the SAT algorithm. The dual loops operate as independent single loops.

3.7.5 Timing requirements of the SAT algorithm

All ports shall incorporate a SAT TIME-OUT timer to detect loss of the SAT Primitive Signal due to a link error. The SAT TIME-OUT timer is set to zero when the port forwards a SAT Primitive Signal or reflects a SAT Primitive Signal as a SAT' Primitive Signal. If the SAT TIME-OUT timer counts 100 ms, then a SAT Primitive Signal is created and treated as if it has been received. Multiple SAT Primitive Signals circulating in the same direction result if several ports generate a new SAT Primitive Signal simultaneously. However the fairness algorithm still operates correctly and the Primitive Signals eventually merge into a single Primitive Signal as described below.

If an L_Port receives a SAT Primitive Signal while it is holding a SAT then it discards each new SAT Primitive Signal. Each of the following functions shall not introduce a delay exceeding six Transmission Word periods per node when the Bypass FIFO is empty: a) Forwarding a SAT Primitive Signal when the port is Satisfied, subject to the delay described in the paragraph below; b) Reflecting a SAT Primitive Signal as a SAT' Primitive Signal when the port is Satisfied; c) Forwarding a SAT' Primitive Signal; d) Reflecting a SAT' Primitive Signal as a SAT Primitive Signal.

For a loop or String with only a few nodes the SAT Primitive Signal rotates very rapidly if all Nodes are Satisfied. Therefore L_Ports shall perform the following two delays: a) An L_Port shall not forward a SAT Primitive Signal until at least 100 Transmission Word periods have elapsed since it forwarded the

previous SAT Primitive Signal; b) An L_Port shall not reflect a SAT Primitive Signal as a SAT' Primitive Signal until at least 100 Transmission Word periods have elapsed since it reflected the previous SAT Primitive Signal.

3.7.6 Configuring the quotas

The Idle Quota and Hold Quota values for each L_Port are allocated by the loop initialization master during loop initialization. The optimum values depend on the total number of Nodes in each String or loop and the traffic pattern. This clause describes how the values are determined. The SAT ROTATION time is the time taken by the SAT character to circulate once around a String or loop, as measured in Transmission Word periods. SAT ROTATION time is a minimum when all L_Ports are Satisfied and they forward the SAT Primitive Signal immediately (SAT' Primitive Signals are always forwarded immediately). SAT ROTATION time increases when the Network becomes heavily loaded and ports hold the SAT Primitive Signal.

The SAT DELAY time is the minimum time to propagate or reflect SAT or SAT' Primitive Signal through a single Node whose inbound port is Satisfied and the Bypass FIFO is empty, as measured in Transmission Word periods.

For a String with N nodes the minimum SAT ROTATION time is 2 * N * SAT DELAY Transmission Words. The corresponding value for a loop is only N * SAT DELAY Transmission Words since the propagation delay for SAT' Primitive Signal is eliminated. SAT ROTATION time is always greater than 100 Transmission Word periods. However after a link failure the loop may become a String and the minimum SAT ROTATION time doubles. Therefore it is advisable to always allow for a minimum SAT ROTATION time of 2 * N * SAT DELAY time.

For long Strings SAT DELAY time is a major contributor to SAT ROTATION. Consequently SAT DELAY time has been specified as six Transmission Words maximum. To avoid throttling when a single node wants to originate data frames, Idle Quota >= (SAT ROTATION time / FRAME LENGTH) >= (2 * N * SAT DELAY time / FRAME LENGTH). For the extreme case of a String with 240 nodes this requires a minimum Idle Quota of 9 frames.

In general there is a compromise between fairness and spatial reuse. Setting Idle Quota =Hold Quota guarantees perfect fairness but it limits spatial reuse. Choosing Idle Quota » Hold Quota permits maximum spatial reuse and therefore maximum total throughput. However in some scenarios this favors the active Node that is furthest upstream since cut through traffic normally has priority. Therefore the upstream Node is able to originate Idle Quota frames for each rotation of the SAT Primitive Signal, whereas each down-stream Node only originates Hold Quota frames. A reasonable compromise for most applications is Idle Quota = 4 * Hold Quota.

3.8 RIP Credit Model

RIP requires a new BB_Credit model that is different from the FC-AL1 BB_Credit model. Since there is no OPN and CLS between L_Ports to establish credit, BB_Credit between communicating L_Ports must be established in advance. One approach would require an L_Port to establish BB_Credit with all L_Ports it may communicate with. This would require a significant number of buffers in an L_Port. To reduce this amount of buffering, RIP employs a Nearest Neighbor credit model, where credit is established only with an L_Port's upstream neighbor Node. In effect, the Nearest Neighbor credit model may be envisioned as link level flow control with routing capability.

R_RDYs encapsulated in RIP packets never consume buffer space and therefore are not flow controlled. They always cut through all intermediate nodes between the source and destination. However, RIP packets containing R_RDYs may be temporarily buffered in the Bypass FIFO. They are processed in hardware at the destination node. As a result, RIP packets containing R_RDYs do not themselves generate an R_RDY

response from the DSN. This requires an L_Port to examine the contents of a RIP packet with a RCVyx header to determine whether it contains a frame or R_RDY(s)[1]. Note that a RIP packet with a RCVfx or RCVex can only contain frames - R_RDYs cannot be broadcast or multicast. If a RIP packet contains one or more R_RDYs the DSN shall not originate an R_RDY response to the USN.

Frames encapsulated in RIP packets are flow controlled. As a result all intermediate Nodes on the path between a source and destination Node shall generate a RIP packet containing one or more R_RDYs to it's USN. For example, on a loop with N Nodes, the average distance between a source and destination for a uniform traffic load is N/2. As a result, N/2 - 1 RIP packets containing R_RDYS will circulate the loop, while the RIP packet containing the frame will traverse only the links between the source and destination Nodes.

3.8.1 Single Loop Topology

This credit model is illustrated for a single loop in Figure 6. Before an L_Port may originate a RIP packet containing a frame it must have positive BB_Credit with it's Downstream Neighbor. The Downstream Neighbor or DSN is the Node at the receive end of an L_Port's output link. The Upstream Neighbor or USN is the Node at the transmit end of an L_Port's input link. When the DSN has freed a buffer for a RIP packet, it shall transmit a RIP packet encapsulating an R_RDY with a RCVyx header, where y = AL_PA of the USN. The AL_PA of the USN and DSN are acquired from the LIRP/LILP AL_PA position map transmitted during loop initialization.

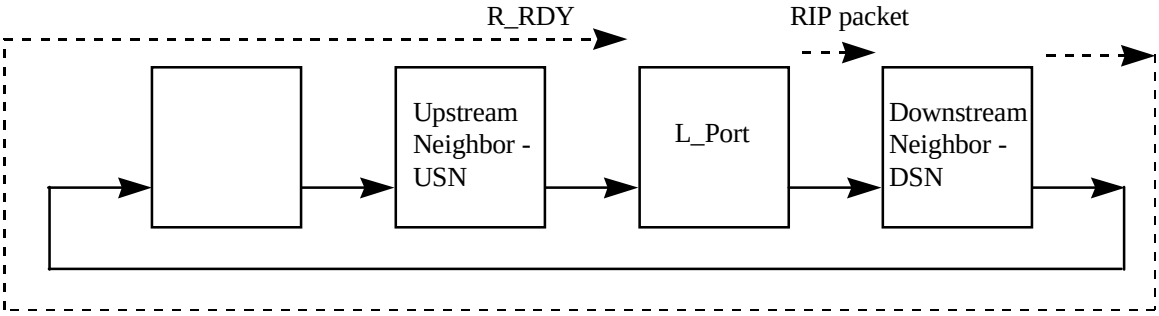


Figure 6: Nearest Neighbor Credit Model

3.8.2 Dual Loop Topologies

3.8.2.1 Iso Rotating Rings

On a dual loop topology with dual port Nodes, where both loops rotate in the same direction (i.e., iso), an R_RDY may be originated on either port by the DSN. Since both loops present an identical path between source and destination, there is no inherent performance advantage in choosing one path over the other.

[1] An alternative implementation may employ a different Primitive Signal to distinguish between a RIP packet containing frames or R_RDYs. For example, a RCVyx could employ two different second characters in the Primitive Signal to distinguish frames from R_RDYs.

3.8.2.2 Counter Rotating Rings

On a dual loop topology with counter rotating loops, an R_RDY may be originated from either port. However, by employing shortest path routing, an R_RDY will only traverse a single link to the L_Port's USN. This is illustrated in Figure 7.

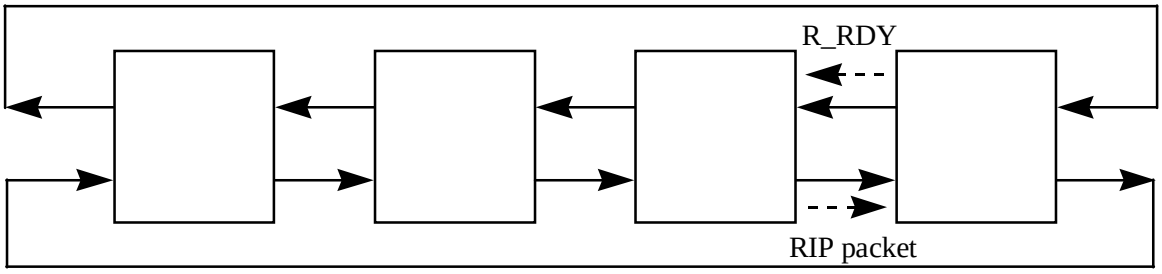


Figure 7: Flow control on counter rotating loops

3.9 RIP Changes to Loop Port State Machine (LPSM)

RIP requires three new states and seven state transitions in the LPSM, which are shown in Figure 8. After RIP mode has been negotiated all LPSMs enter the Idle state. A LPSM shall be in the Idle state when it has neither frames nor R_RDYs to transmit and the Bypass FIFO is empty.

From the Idle state, when the LPSM has a frame or R_RDY to transmit is shall assert REQ(transmit), and if at least six consecutive Idles have been received and the L_Port's Frame Counter is less than or equal to it's Idle Quota, the LPSM shall make the transition to the Inject state. If neither frames nor R_RDYs destined for other ports nor the MRKtx Primitive Signal are received while the LPSM is in the Inject state, the Bypass FIFO will remain empty and the LPSM shall make the transition to the Idle state if it has no more frames or R_RDYs to transmit. If the Bypass FIFO is empty and the LPSM has additional frames or R_RDYs to transmit (REQ(transmit) is reasserted), it may remain in the Inject state.

If any frames or R_RDYs destined for other ports are received while transmitting, the LPSM shall make the transition to the Drain state after the frame or R_RDY and six subsequent Fill Words have been transmitted in the Inject state. When the Bypass FIFO has been drained of all Transmission Words (i.e., it is empty), the LPSM shall make the transition to the Idle state if REQ(transmit) is no longer asserted; or if REQ(transmit) is asserted and the L_Port's Frame Counter is greater than it's Idle Quota; or if REQ(transmit) is asserted and the L_Port's Frame Counter is greater than it's Hold Quota and it is holding the SAT token.

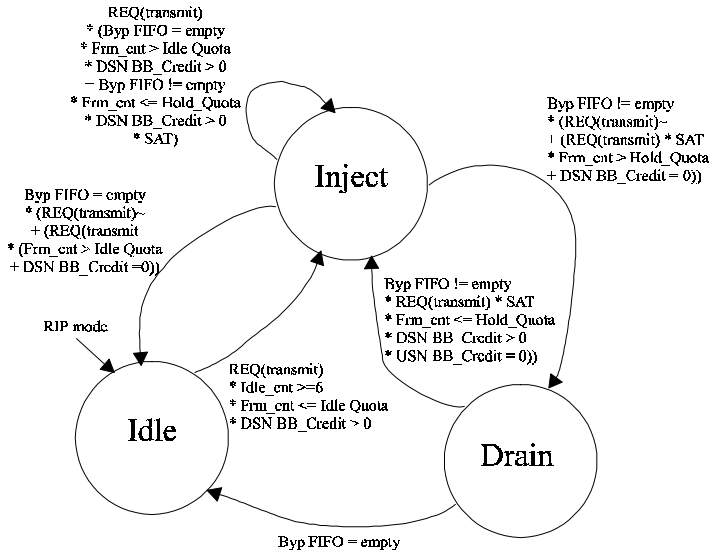


Figure 8: RIP LPSM

3.9.1 Idle State actions:

If Idle is received, the current Fill Word shall be set to Idle. The CFW shall be transmitted.

The LPSM shall make a transition to the Inject state when the L_Port requests transmission (REQ(transmit)), the Downstream Neighbor BB_Credit is greater than zero, the Frame Counter is less than or equal to the Idle Quota, and at least six consecutive Idles have been received.

If a Fill Word is to be transmitted, the current Fill Word shall be used.

If RCVfx is received, where x <> AL_PA of the L_Port, the LPSM shall retransmit the received RCVfx and all transmission words up to the RCVff inclusive (i.e., the L_Port shall retransmit the RIP packet). The LPSM shall also pass the encapsulated frame to the Receive FIFO.

If RCVfx is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVfx and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVex is received, where x <> AL_PA of the L_Port, the LPSM shall retransmit the RCVex.

If RCVex is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVex and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVye is received, with y <> AL_PA of the L_Port, then the LPSM shall retransmit the received RCVye. The LPSM shall also retransmit the encapsulated frame.

If RCVye is received, where y = AL_PA, the LPSM shall substitute the CFW. The L_Port shall also pass the encapsulated frame to the Receive FIFO and shall also retransmit the encapsulated frame.

If RCVyx is received, where y = AL_PA of the L_Port, then the LPSM shall pass the encapsulated frame or R_RDY(s) to the Receive FIFO. The LPSM shall remove all received Transmission Words between RCVyx and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word removed.

If RCVyx is received, where y <> AL_PA of the L_Port and x = AL_PA of the L_Port, then the LPSM shall discard all received Transmission Words between RCVyx and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If a RCVyx is received, where y <> AL_PA and x <> AL_PA, the LPSM shall retransmit all Transmission Words received between the RCVyx and the RCVff inclusive.

If MRKtx is received:
- if x = AL_PA of the L_Port, the LPSM shall transmit the current Fill Word; the MRKtx is discarded;
- if x <> AL_PA of the L_Port and the MK_TP and AL_PS match the expected values, the MRKtx shall be loaded into the Receive FIFO and the received MRKtx shall be retransmitted.

If SAT is received, the Frame Counter shall be set to zero and the SAT shall be retransmitted.

If SAT' is received and the Node is not a single port device, it shall be retransmitted.

If SAT' is received and the Node is a single port device, it shall discard the SAT' and transmit SAT.

If the LPSM requests to transmit a MRKtx (REQ(mark as tx)), the LPSM shall transmit one MRKtx at the next appropriate Fill Word (see clause 7), unless REQ(mark tx) is removed before MRKtx is transmitted.

If CLS is received, the current Fill Word shall be transmitted.

If OPN is received, the current Fill Word shall be transmitted.

If ARB is received, the current Fill Word shall be transmitted.

If LIP is received, the LPSM shall make the transition to the OPEN-INIT state.

IF LPByx is recognized:
- if x = AL_PA of the L_Port, the received LPByx shall be discarded and the Current Fill Word shall be transmitted;
- if y = AL_PA of the L_Port, or the L_Port requests to be bypassed (REQ(bypass L_Port)), the LPSM shall retransmit the LPByx, shall set the bypass circuit (if present), shall set LP_BYPASS to True (1), and shall go to nonparticipating mode in the Idle state;
- if y <> AL_PA of the L_Port, the LPByx shall be retransmitted.

If LPEyx is recognized:
- if x = AL_PA, the received LPEyx shall be discarded and the Current Fill Word shall be transmitted;
- if y <> AL_PA and y <> hex 'FF', retransmit the LPEyx;
- if y = AL_PA or y = hex 'FF', set LP_BYPASS = FALSE(0) and retransmit the LPEyx.

The LPSM shall retransmit all other received valid Transmission Words on the Loop.

Invalid Transmission Character substitution shall be performed as specified in 8.3.2.

If the LPSM detects a Loop failure on its inbound fibre and LP_BYPASS is set to FALSE (0), or the L_Port requests initialization (REQ(initialize)), the LPSM shall make the transition to the Initializing state.

If the L_Port requests not to participate on the Loop (REQ(nonparticipating)), the LPSM shall transmit at least 12 LIPs with the rightmost two characters equal to hex 'F7F7' to invoke Loop initialization. This allows another L_Port to acquire the relinquished AL_PA. The 12 LIPs are only transmitted once for each REQ(nonparticipating). The L_Port shall not participate further in Loop initialization until REQ(initialize) or REQ(participating) is set.

3.9.2 Inject State actions:

If Idle is received, the Current Fill Word shall be set to Idle.

If a Fill Word is to be transmitted, the Current Fill Word shall be used.

If a Fill Word is to stored in the Bypass FIFO, the Current Fill Word shall be used.

If a Fill Word is received that is not stored in the Bypass FIFO, it shall be discarded.

If the L_Port requests additional transmission (REQ(transmit) asserted), the Bypass FIFO is empty, the Downstream Neighbor BB_Credit is greater than zero, and the Frame Counter is less than or equal to the Idle Quota, the LPSM shall remain in the Inject state.

If the L_Port requests additional transmission (REQ(transmit) asserted), the Bypass FIFO is not empty, the Downstream Neighbor BB_Credit is greater than zero, the Frame Counter is less than or equal to the Hold Quota, and the SAT token is held, the LPSM shall remain in the Inject state.

After a RIP packet is transmitted by the L_Port (RCVff is transmitted), the L_Port shall increment the Frame Counter.

If RCVfx is received, where x <> AL_PA of the L_Port, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVfx and RCVff inclusive. The LPSM shall also pass the frame encapsulated in the RIP packet to the Receive FIFO.

If RCVfx is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVfx and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVex is received, where x <> AL_PA of the L_Port, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVex and RFVff inclusive.

If RCVex is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVex and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVye is received, where y = AL_PA, the LPSM shall substitute the CFW and load it into the Bypass FIFO. The L_Port shall also pass the encapsulated frame to the Receive FIFO.

If RCVyx is received, where y = AL_PA of the L_Port, then the LPSM shall pass the encapsulated frame or R_RDY(s) to the Receive FIFO. The LPSM shall remove all received Transmission Words between RCVyx and the subsequent RCVff inclusive.

If RCVyx is received, where y <> AL_PA of the L_Port and x = AL_PA of the L_Port, then the LPSM shall discard all received Transmission Words between RCVyx and the subsequent RCVff inclusive.

If a RCVyx is received, where y <> AL_PA and x <> AL_PA, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVyx and RFVff inclusive.

If CLS is received, it shall be discarded.

If OPN is received, it shall be discarded.

If ARB is received, it shall be discarded.

If SAT is received, it shall be loaded into the Bypass FIFO. The Frame Counter shall be set to zero.

If SAT’ is received and the Node is a dual port device, it shall be loaded into the Bypass FIFO.

If SAT' is received and the Node is a single port device, it shall be discarded and SAT shall be loaded into the Bypass FIFO.

If a MRKtx is received, where x = AL_PA of the L_Port, the LPSM shall discard the MRKtx.

If a MRKtx is received where MK_TP and AL_PS match the expected values, and x <> AL_PA, the MRKtx shall be processed. The MRKtx and up to 6 trailing Fill Words that are received shall be buffered in the Bypass FIFO. The MRKtx shall also be placed in the Receive FIFO.

If LIP is received, the LPSM shall make the transition to the OPEN-INIT state.

If LPByx is recognized:
- if x = AL_PA of the L_Port, the received LPByx shall be discarded;
- if y = AL_PA of the L_Port, or the L_Port requests to be bypassed (REQ(bypass L_Port)), the LPSM shall finish the current frame or R_RDY transmission; set LP_BYPASS to TRUE (1); make the transition to the Drain state if the Bypass FIFO is not empty; or make the transition to the Idle state if the Bypass FIFO is empty.

If the LPSM detects a Loop failure on its inbound fibre or the L_Port requests initialization (REQ(initialize)), the LPSM shall make the transition to the Initializing state.

If the LPSM requests to transmit a MRKtx (REQ(mark as tx)), the LPSM shall transmit one MRKtx at the next appropriate Fill Word (see clause 7), unless REQ(mark tx) request is removed before MRKtx is transmitted.

3.9.3 Drain State actions:

If Idle is received, the current Fill Word shall be set to Idle.

If a Fill Word is to be transmitted, the Current Fill Word shall be used.

If a Fill Word is to stored in the Bypass FIFO, the Current Fill Word shall be used.

If a Fill Word is received that is not stored in the Bypass FIFO, it shall be discarded.

If the L_Port requests transmission (REQ(transmit) asserted), the SAT token is held, the Frame Counter is less than or equal to the Hold Quota, the Upstream Neighbor BB_Credit is greater than zero, the Downstream Neighbor BB_Credit is zero, and six Idle characters have been transmitted from the head of the Bypass FIFO, the LPSM shall make the transition to the Inject state.

If RCVfx is received, where x <> AL_PA of the L_Port, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVfx and RFVff inclusive. The LPSM shall also pass the frame encapsulated in the RIP packet to the Receive FIFO.

If RCVfx is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVfx and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVex is received, where x <> AL_PA of the L_Port, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVex and RFVff inclusive.

If RCVex is received, where x = AL_PA of the L_Port, the LPSM shall discard all received Transmission Words between RCVex and the subsequent RCVff inclusive. The L_Port shall substitute the Current Fill Word for each Transmission Word discarded.

If RCVye is received, where y = AL_PA, the LPSM shall substitute the CFW and load it into the Bypass FIFO. The L_Port shall also load encapsulated frame into the Receive FIFO.

If RCVyx is received, where y = AL_PA of the L_Port, then the LPSM shall pass the encapsulated frame or R_RDY(s) to the Receive FIFO. The LPSM shall remove all received Transmission Words between RCVyx and the subsequent RCVff inclusive.

If RCVyx is received, where y <> AL_PA of the L_Port and x = AL_PA of the L_Port, then the LPSM shall discard all received Transmission Words between RCVyx and the subsequent RCVff inclusive.

If a RCVyx is received, where y <> AL_PA and x <> AL_PA, the LPSM shall load into the Bypass FIFO all Transmission Words between the RCVyx and RFVff inclusive.

If CLS is received, it shall be discarded.

If OPN is received, it shall be discarded.

If ARB is received, it shall be discarded.

If SAT is received, it shall be loaded into the Bypass FIFO. The Frame Counter shall be set to zero.

If SAT' is received and the Node is a dual port device, it shall be loaded into the Bypass FIFO.

If SAT' is received and the Node is a single port device, it shall be discarded and SAT shall be loaded into the Bypass FIFO.

If a MRKtx is received where x = AL_PA of the L_Port, the LPSM shall discard the MRKtx.

If a MRKtx is received where MK_TP and AL_PS match the expected values, and x <> AL_PA, the MRKtx shall be processed. The MRKtx and up to 6 trailing Fill Words that are received shall be buffered in the Bypass FIFO. The MRKtx shall be placed in the Receive FIFO.

If LIP is received, the LPSM shall make the transition to the OPEN-INIT state.

If LPByx is recognized:
- if x = AL_PA of the L_Port, the received LPByx shall be discarded;
- if y = AL_PA of the L_Port, or the L_Port requests to be bypassed (REQ(bypass L_Port)), the LPSM shall finish the current frame or R_RDY transmission; set LP_BYPASS to TRUE (1); make the transition to the Drain state if the Bypass FIFO is not empty; or make the transition to the Idle state if the Bypass FIFO is empty.

If the LPSM detects a Loop failure on its inbound fibre or the L_Port requests initialization (REQ(initialize)), the LPSM shall make the transition to the Initializing state.

If the LPSM requests to transmit a MRKtx (REQ(mark as tx)), the LPSM shall transmit one MRKtx at the next appropriate Fill Word (see clause 7), unless REQ(mark tx) request is removed before MRKtx is transmitted.

3.9.4 LPSM State Tables

State table legends:

RFW = Received Fill Word

CFW = Current Fill Word

DSN = Downstream Neighbor

USN = Upstream Neighbor

Table 2: Idle Transitions

Entry Actions			
COPY = 0	RETRANSMIT = 0	CFW = N/C	
LP_BYPASS = 0			
Input	**Output**	**Next State**	notes
loss of sync < R_T_TOV	IDLE	Idle	
loop failure			
LP_BYPASS = 0	none/inst.	Initializing	1
LP_BYPASS = 1	none/inst	Idle	1
Invalid trans. character	any valid trans. Character	Idle	1
running disparity at O.S.	CFW	Idle	
elasticity word required	CFW	Idle	
valid trans. word = O.S.:			
RCVyx: y = AL_PA	Copy = 1, Retransmit = 0		
	CFW	Idle	
RCVyx: y <> AL_PA & x <> AL_PA	Copy = 0, Retransmit = 1		
	same word	Idle	
RCVyx: y <> AL_PA & x = AL_PA	Copy = 0, Retransmit = 0		1
	CFW	Idle	
RCVfx: x <> AL_PA	Copy = 1, Retransmit = 1		
	same word	Idle	
RCVfx: x = AL_PA	Copy = 0, Retransmit = 0		
	CFW	Idle	
RCVex: x <> AL_PA	Retransmit = 1		
	same word	Idle	
RCVex: x = AL_PA	Retransmit = 0		

	CFW	Idle	
RCVye: y <> AL_PA			
Retransmit = 1	same word	Idle	
Retransmit = 0	CFW	Idle	
RCVye: y = AL_PA			
Retransmit = 1	Copy = 1		
	same word	Idle	
Retransmit = 0	CFW	Idle	
SOF | VDW | EOF | R_RDY			
Copy = 1	load Receive FIFO		
Retransmit = 1	same word	Idle	
Retransmit = 0	CFW	Idle	
RCVff	Copy = 0		
Retransmit = 1	Retransmit = 0		
	same word	Idle	
Retransmit = 0	CFW	Idle	
IDLE	CFW = IDLE		
	CFW	Idle	
OPN	CFW	Idle	1
CLS	CFW	Idle	1
ARB	CFW	Idle	
MRKtx			
x = AL_PA	CFW	Idle	
x <> AL_PA	same word	Idle	
SAT			
set Frame Counter = 0			
if dual port Node	SAT	Idle	
if single port Node	SAT’	Idle	
SAT’			
if dual port Node	SAT’	Idle	
if single port Node	SAT	Idle	
primitive sequences			
NOS	same word	Idle	
OLS	same word	Idle	
LR	same word	Idle	

LRR	same word	Idle	
LIP	same word	OPEN-INIT	
LPByx			
x = AL_PA	CFW	Idle	
y <> AL_PA	same word	Idle	
y = AL_PA	LP_BYPASS = 1		
	same word	Idle	
LPEyx | LPEfx			
x = AL_PA	CFW	Idle	
y <> AL_PA & y <> hex 'FF'	same word	Idle	
y = AL_PA | f = hex 'FF'	LP_BYPASS = 0		
	same word	Idle	
any other O.S.	same word	Idle	
L-Port controls			
REQ(monitor)	none/inst.	Idle	1
REQ(arbitrate as x)	none/inst.	Idle	1
REQ(transmit i): i = yx | fx | ex group | R_RDY & (DSN BB_Credit = 0 | Frame Counter > Idle Quota)	same word	Idle	
REQ(transmit i): i = yx | fx | ex group | R_RDY & DSN BB_Credit > 0 & Frame Counter <= Idle Quota			
FW_cnt < 6 Fill Words	same word	Idle	
FW_cnt >= 6 Fill Words	CFW = Idle	Inject	
REQ(open yx) f-d	none/inst.	Idle	1
REQ(open yy) h-d	none/inst.	Idle	1
REQ(open fr)	none/inst.	Idle	1
REQ(open yr)	none/inst.	Idle	1
REQ(close)	none/inst.	Idle	1
REQ(transfer)	none/inst.	Idle	1
REQ(old-port)	none/inst.	Idle	1
REQ(participating)	none/inst.	Idle	1
REQ(nonparticipating)	transmit 12 LIP(F7, F7)s	Idle	
REQ(mark as tx)	MRKtx at the next appropriate		
TW = appropriate Fill Word	MRKtx	Idle	
TW <> appropriate Fill Word	none/inst.	Idle	
REQ(bypass L_Port)	LP_BYPASS = 1	Idle	

REQ(bypass L_Port y)	LPByx at the next app. Fill Word	Idle	
REQ(enable L_Port)	none/inst.	Idle	1
REQ(enable all)	LPEfx at the next app. Fill Word	Idle	
REQ(enable L_Port y)	LPEyx at the next app. Fill Word	Idle	
REQ(initialize)	none/inst.	Initializing	

Notes:

1) error

Table 3: Inject Transitions

Entry Actions			
COPY = 0	RETRANSMIT = 0	LP_BYPASS = 0	
CFW = N/C			
Input	**Output**	**Next State**	notes
loss of sync < R_T_TOV	FC-2 FP/PSig/PSeq	Inject	
loop failure	none/inst.	Initializing	
Invalid trans. character	FC-2 FP/PSig/PSeq	Inject	
running disparity at O.S.	FC-2 FP/PSig/PSeq	Inject	
elasticity word required	n/a	Inject	
valid trans. word = O.S.:			
RCVyx: y = AL_PA	Copy = 1, Retransmit = 0		
	FC-2 FP/PSig/PSeq	Inject	
RCVyx: y <> AL_PA & x <> AL_PA	Copy = 0, Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Inject	
RCVyx: y <> AL_PA & x = AL_PA	Copy = 0, Retransmit = 0		1
	FC-2 FP/PSig/PSeq	Inject	
RCVfx: x <> AL_PA	Copy = 0, Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Inject	
RCVfx: x = AL_PA	Copy = 0, Retransmit = 0		
	FC-2 FP/PSig/PSeq	Inject	
RCVex: x <> AL_PA	Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Inject	
RCVex: x = AL_PA	Retransmit = 0		
	FC-2 FP/PSig/PSeq	Inject	
RCVye: y <> AL_PA			
Retransmit = 1	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Inject	
RCVye: y = AL_PA			

Retransmit = 1	Copy = 1, Load Bypass FIFO (CFW)		
	FC-2 FP/PSig/PSeq	Inject	
SOF | VDW | EOF | R_RDY			
Retransmit = 1	load Bypass FIFO		
Copy = 1	load Receive FIFO		
RCVff	Copy = 0		
Retransmit = 1	Retransmit = 0, Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Inject	
IDLE	CFW = IDLE		
	FC-2 FP/PSig/PSeq	Inject	
ARBx:	FC-2 FP/PSig/PSeq	Inject	1
OPN	FC-2 FP/PSig/PSeq	Inject	1
CLS	FC-2 FP/PSig/PSeq	Inject	1
MRKtx			
x = AL_PA	FC-2 FP/PSig/PSeq	Inject	
x <> AL_PA	load Bypass FIFO		
	load Receive FIFO		
	FC-2 FP/PSig/PSeq	Inject	
SAT			
set Frame Counter = 0			
if dual port Node	load SAT into Bypass FIFO	Idle	
if single port Node	load SAT' into Bypass FIFO	Idle	
SAT'			
if dual port Node	load SAT' into Bypass FIFO	Idle	
if single port Node	load SAT into Bypass FIFO	Idle	
primitive sequences			
NOS	FC-2 FP/PSig/PSeq	Inject	
OLS	FC-2 FP/PSig/PSeq	Inject	
LR	FC-2 FP/PSig/PSeq	Inject	
LRR	FC-2 FP/PSig/PSeq	Inject	
LIP	FC-2 FP/PSig/PSeq	Initializing	
LPByx			
x = AL_PA	FC-2 FP/PSig/PSeq	Inject	
y <> AL_PA	FC-2 FP/PSig/PSeq	Inject	
y = AL_PA			

Bypass FIFO = empty	LP_BYPASS = 1	Idle	
Bypass FIFO <> empty	LP_BYPASS = 1	Drain	
LPEyx | LPEfx			
x = AL_PA	FC-2 FP/PSig/PSeq	Inject	
y = AL_PA	FC-2 FP/PSig/PSeq	Inject	1
y <> AL_PA	FC-2 FP/PSig/PSeq	Inject	
y = ff	FC-2 FP/PSig/PSeq	Inject	
any other O.S.	FC-2 FP/PSig/PSeq	Inject	
L-Port controls			
REQ(monitor)	none/inst.	Monitoring	
REQ(transmit i): i = yx | fx | ex group | R_RDY & Bypass FIFO = empty & Frame Counter <= Idle Quota & DSN BB_Credit > 0	none/inst.	Inject	
REQ(transmit i): i = yx | fx | ex group | R_RDY & Bypass FIFO = empty & (Frame Counter > Idle Quota & DSN BB_Credit = 0)	none/inst.	Idle	
REQ(transmit i): i = yx | fx | ex group | R_RDY & Bypass FIFO <> empty & Frame Counter <= Hold Quota & SAT token held & DSN BB_Credit > 0 & USN BB_Credit = 0	none/inst.	Inject	
REQ(transmit i): i = yx | fx | ex group | R_RDY & Bypass FIFO <> empty & (Frame Counter > Hold Quota | DSN BB_Credit > 0)	none/inst.	Idle	
REQ(open yx) f-d	none/inst.	Inject	
REQ(open yy) h-d	none/inst.	Inject	
REQ(open fr)	none/inst.	Inject	
REQ(open yr)	none/inst.	Inject	
REQ(close)	none/inst.	Inject	
REQ(arbitrate as x)	none/inst.	Inject	1
REQ(transfer)	none/inst.	Inject	
REQ(old-port)	none/inst.	Inject	
REQ(participating)	none/inst.	Inject	1
REQ(nonparticipating)	transmit 12 LIP(F7, F7)s	Inject	
REQ(mark as tx)	none/inst.	Inject	
REQ(bypass L_Port)			

Bypass FIFO = empty	LP_BYPASS = 1	Idle	
Bypass FIFO <> empty	LP_BYPASS = 1	Drain	
REQ(bypass L_Port y)	none/inst.	Inject	
REQ(enable L_Port)	none/inst.	Inject	
REQ(enable all)	none/inst.	Inject	
REQ(enable L_Port y)	none/inst.	Inject	
REQ(initialize)	none/inst.	Initializing	

Notes:

1) error

Table 4: Drain Transitions

Entry Actions			
COPY = 0	RETRANSMIT = 0	LP_BYPASS = 0	
CFW = N/C			
Input	**Output**	**Next State**	notes
loss of sync < R_T_TOV	FC-2 FP/PSig/PSeq	Drain	
loop failure	none/inst.	Initializing	
Invalid trans. character	FC-2 FP/PSig/PSeq	Drain	
running disparity at O.S.	FC-2 FP/PSig/PSeq	Drain	
elasticity word required	n/a	Drain	
valid trans. word = O.S.:			
RCVyx: y = AL_PA	Copy = 1, Retransmit = 0		
	FC-2 FP/PSig/PSeq	Drain	
RCVyx: y <> AL_PA & x <> AL_PA	Copy = 0, Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Drain	
RCVyx: y <> AL_PA & x = AL_PA	Copy = 0, Retransmit = 0		1
	FC-2 FP/PSig/PSeq	Drain	
RCVfx: x <> AL_PA	Copy = 0, Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Drain	
RCVfx: x = AL_PA	Copy = 0, Retransmit = 0		
	FC-2 FP/PSig/PSeq	Drain	
RCVex: x <> AL_PA	Retransmit = 1		
	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Drain	
RCVex: x = AL_PA	Retransmit = 0		
	FC-2 FP/PSig/PSeq	Drain	
RCVye: y <> AL_PA			
Retransmit = 1	Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Drain	
RCVye: y = AL_PA			

Retransmit = 1	Copy = 1, Load Bypass FIFO (CFW)		
	FC-2 FP/PSig/PSeq	Drain	
SOF | VDW | EOF | R_RDY			
Retransmit = 1	load Bypass FIFO		
Copy = 1	load Receive FIFO		
RCVff	Copy = 0		
Retransmit = 1	Retransmit = 0, Load Bypass FIFO		
	FC-2 FP/PSig/PSeq	Drain	
IDLE	CFW = IDLE		
	ACCESS = TRUE (1)		
	FC-2 FP/PSig/PSeq	Drain	
ARB:	FC-2 FP/PSig/PSeq	Drain	1
OPN	FC-2 FP/PSig/PSeq	Drain	1
CLS	FC-2 FP/PSig/PSeq	Drain	1
MRKtx			
x = AL_PA	FC-2 FP/PSig/PSeq	Drain	
x <> AL_PA	load Bypass FIFO		
	load Receive FIFO		
	FC-2 FP/PSig/PSeq	Drain	
SAT			
if Frame Counter > Hold Quota set Frame Counter = 0			
if dual port Node	load SAT into Bypass FIFO	Drain	
if single port Node	load SAT' into Bypass FIFO	Drain	
if (Frame Counter <= Hold Quota & REQ(transmit)			
& DSN BB_Credit > 0) hold SAT token	FC-2 FP/PSig/PSeq	Drain	
SAT'			
if dual port Node	load SAT' into Bypass FIFO	Drain	
if single port Node	load SAT into Bypass FIFO	Drain	
primitive sequences			
NOS	FC-2 FP/PSig/PSeq	Drain	
OLS	FC-2 FP/PSig/PSeq	Drain	
LR	FC-2 FP/PSig/PSeq	Drain	
LRR	FC-2 FP/PSig/PSeq	Drain	
LIP	FC-2 FP/PSig/PSeq	Initializing	

LPByx			
x = AL_PA	FC-2 FP/PSig/PSeq	Drain	
y <> AL_PA	FC-2 FP/PSig/PSeq	Drain	
y = AL_PA			
Bypass FIFO = empty	LP_BYPASS = 1	Idle	
Bypass FIFO <> empty	LP_BYPASS = 1	Drain	
LPEyx | LPEfx			
x = AL_PA	FC-2 FP/PSig/PSeq	Drain	
y = AL_PA	FC-2 FP/PSig/PSeq	Drain	1
y <> AL_PA	FC-2 FP/PSig/PSeq	Drain	
y = ff	FC-2 FP/PSig/PSeq	Drain	
any other O.S.	FC-2 FP/PSig/PSeq	Drain	
L-Port controls			
REQ(monitor)	none/inst.	Drain	
REQ(arbitrate as x)	none/inst.	Drain	1
REQ(transmit i): i = yx | fx | fe group | R_RDY			
FW_cnt < 6 Fill Words	same word	Drain	
FW_cnt >= 6 Idles & Byp FIFO empty & Frame Counter <= Idle Quota & DSN BB_Credit > 0	none/inst.	Inject	
FW_cnt >= 6 Idles & Byp FIFO empty & Frame Counter > Idle Quota | DSN BB_Credit = 0	same word	Idle	
FW_cnt >= 6 Idles & Byp FIFO not empty & Frame Counter <= Hold Quota & DSN BB_Credit > 0 & USN BB_Credit = 0 & SAT token held	none/inst.	Inject	
FW_cnt >= 6 Idles & Byp FIFO not empty | Frame Counter > Hold Quota | DSN BB_Credit = 0 | USN BB_Credit != 0 | SAT token not held	FC-2 FP/PSig/PSeq	Drain	
REQ(open yx) f-d	none/inst.	Drain	
REQ(open yy) h-d	none/inst.	Drain	
REQ(open fr)	none/inst.	Drain	
REQ(open yr)	none/inst.	Drain	
REQ(close)	none/inst.	Drain	
REQ(transfer)	none/inst.	Drain	

REQ(old-port)	none/inst.	Drain	
REQ(participating)	none/inst.	Drain	1
REQ(nonparticipating)	none/inst.	Drain	
REQ(mark as tx)	none/inst.	Drain	
REQ(bypass L_Port)	none/inst.	Monitoring	
REQ(bypass L_Port y)	none/inst.	Drain	
REQ(enable L_Port)	none/inst.	Drain	
REQ(enable all)	none/inst.	Drain	
REQ(enable L_Port y)	none/inst.	Drain	
REQ(initialize)	none/inst.	Initializing	

Notes:

1) error

Serial Communications

FC–AL–3

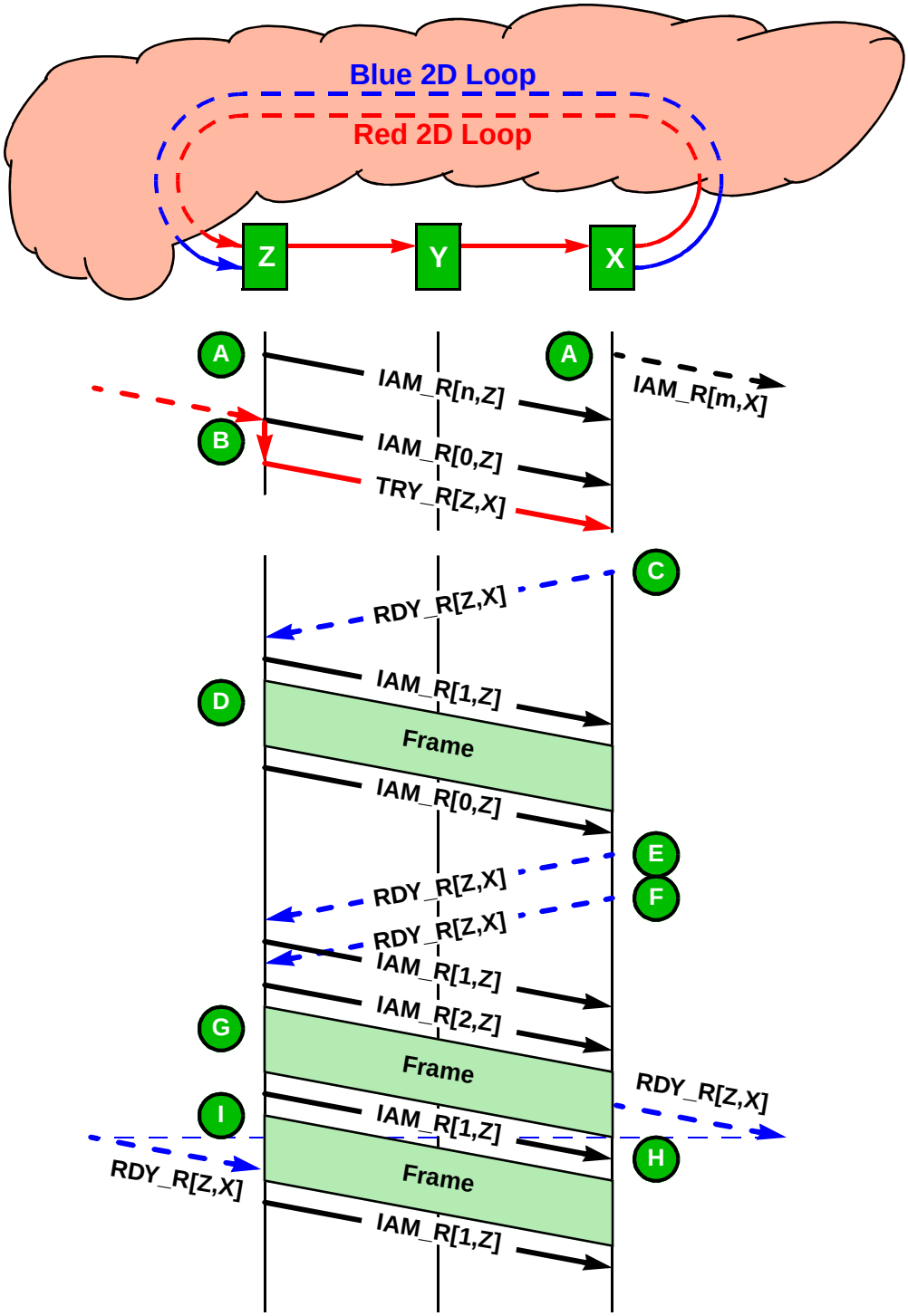
2D Loop
Link Level Flow Control

T11 9 Feb. 1998

Bent Støvhase (bent@inforamp.net)

FC–AL–3 – Link Level Flow Control

2D Loop Operation Scenario #1



Normal Link Level Flow Control on a 2D Loop

FC–AL–3 – Link Level Flow Control

2D Loop Operation Scenario #1

- **A** – The L_Ports ‘Z’ and ‘X’ on the red loop sends IAM_R[n,x] Primitive Signals as Fill Words
  - n = L_Port’s current Transmitter Credit
  - x = L_Port’s address (AL_PA)
  - The Observing L_Port ‘Y’ uses IDLE Primitive Signals as Fill Words
- **B** – L_Port ‘Z’ resets its Transmitter Credit when it receives a TRY_R[Z,X] Primitive Signal
  - X = ‘New’ Downstream neighbour for L_Port ‘Z’
  - The TRY_R[Z,X] Primitive Signal is preceded by 12 IAM_R[0,X] Primitive Signals before it is forwarded on the Red Loop
- **C** – L_Port ‘X’ inserts a RDY_R[Z,X] Primitive Signal to transfer a single credit to L_Port ‘Z’
  - This RDY_R[Z,X] Primitive Signal is routed along the shortest path from L_Port ‘X’ to L_Port ‘Z’
  - L_Port ‘Z’ adjusts its IAM_R[1,Z] Fill Word reflecting the reception of the RDY_R[Z,X] Primitive Signal
- **D** – L_Port ‘Z’ transmits a Frame on the ‘split’ link between L_Port ‘Z’ and L_Port ‘X’
  - Transmission of the Frame (SOF delimiter) consumes one credit
    - The loss of one credit is reflected in the modified IAM_R[0,Z] Fill Word used by L_Port ‘Z’

FC–AL–3 – Link Level Flow Control

2D Loop Operation Scenario #1

- **E** – L_Port ‘Z’ transmits a RDY_R[Z,X] Primitive Signal to transfer credit to L_Port ‘Z’, reflected in the adjusted IAM_R[1,Z] Fill Word used by L_Port ‘Z’
- **F** – L_Port ‘Z’ transmits a RDY_R[Z,X] Primitive Signal to transfer credit to L_Port ‘Z’, reflected in the adjusted IAM_R[2,Z] Fill Word used by L_Port ‘Z’
- **G** – L_Port ‘Z’ transmits a Frame, consuming one credit, reflected in the adjusted IAM_R[1,Z] Fill Word used by L_Port ‘Z’
- **H** – L_Port ‘Z’ transmits a RDY_R[Z,X] Primitive Signal to transfer credit to L_Port ‘Z’
  - The coincident transmission of a Frame while receiving a RDY_R[Z,X] Primitive Signal is reflected in the unchanged IAM_R[1,Z] Fill Word used by L_Port ‘Z’
- **I** – L_Port ‘Z’ transmits a Frame, reflected in the adjusted IAM_R[1,Z] Fill Word used by L_Port ‘Z’
  - The coincident transmission of a Frame while receiving a RDY_R[Z,X] Primitive Signal is reflected in the unchanged IAM_R[1,Z] Fill Word used by L_Port ‘Z’

FC–AL–3 – Link Level Flow Control

Rules – Summary

- An L_Port resets its Transmitter Credit when
  - It receives a TRY_x[Z,X] Primitive Signal of the correct colour, but only if Z = 'the L_Port's AL_PA' and X = 'a new downstream neighbouring L_Port'
    - The colour is correct if a TRY_R[Z,X] Primitive Signal is received for a L_Port on the red loop
    - The colour is correct if a TRY_B[Z,X] Primitive Signal is received for a L_Port on the blue loop
  - It receives a RDY_x[Z,X] Primitive Signal of the correct colour if Z = 'the L_Port's AL_PA' and X = 'an unexpected L_Port'
    - The colour is correct if a RDY_R[Z,X] Primitive Signal is received for a L_Port on the red loop
    - The colour is correct if a RDY_B[Z,X] Primitive Signal is received for a L_Port on the blue loop
- Transmission of a Class 2 or 3 Frame (SOF delimiter) shall consume one Transmitter Credit
  - Only L_Port's in the 2D state shall transmit Frame under Transmitter Credit flow control rules
    - An L_Port in the Joining state shall follow Transmitter Credit rules, once it starts to send IAM Fill Words, having inherited its upstream neighbouring L_Port's Transmitter Credit

FC–AL–3 – Link Level Flow Control

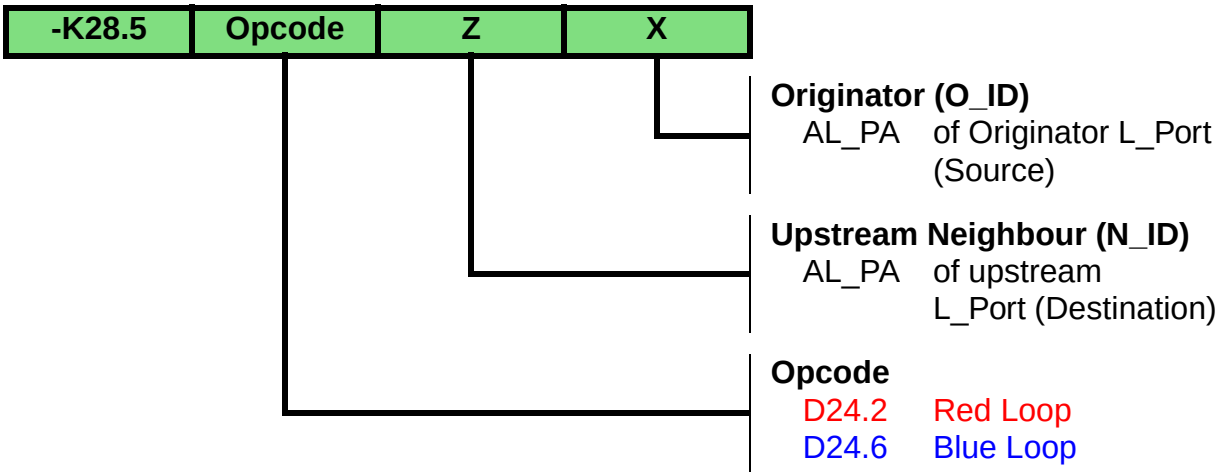
Rules – Summary

- Reception of a RDY_x[Z,X] Primitive Signal of the correct colour shall increase the L_Ports Transmitter Credit if Z = 'the L_Port's AL_PA' and X = 'the downstream neighbouring L_Port's AL_PA'
  - Transmitter Credit shall be limited to a maximum value of 3
    - RDY_x[Z,X] Primitive Signals received in excess shall be discarded, without affecting the value of the L_Port's Transmitter Credit
    - An L_Port's current Transmitter Credit is displayed in every IAM Fill Word sent by the L_Port
    - L_Port's are expected to limit Transmitter Credit to a value of 1
  - The colour is correct if a RDY_R[Z,X] Primitive Signal is received for a L_Port on the red loop
  - The colour is correct if a RDY_B[Z,X] Primitive Signal is received for a L_Port on the blue loop
- Transmission of a RDY_x[Z,X] Primitive Signal shall, for dual ported entities, be made on the port closest to the destination ('Z')
- Dual ported entities shall accept RDY_x[Z,X] Primitive Signal received on either port
- Transmission (forwarding) of Primitive Signals takes precedent (priority) over Frame transmission, so any RDY_x[Z,X] Primitive Signal not reflected in the IAM Fill Word is considered lost after 20 ms and shall be re-inserted by the originating L_Port ('X')
  - 20 ms > $126*(LinkLatency_{max} + FrameLatency_{max})$

FC–AL–3 – Link Level Flow Control

New Primitive Signals

- Ready (RDY) Primitive Signal



  - Passes Link level credit (BB_Credit) from a receiver (X) to a transmitter (Z)
    (Routed from source to destination)
    - An L_Port 'X' inserts a RDY_x[Z,X] when it is able to receive a full size Frame from its upstream neighbouring L_Port 'Z'
      - Z = L_Port receiving credit, the destination of the RDY_x[Z,X] Primitive Signal
      - X = L_Port issuing credit, the source of the RDY_x[Z,X] Primitive Signal
      - RDY_R[Z,X] shall be used to transfer credit on the red loop
      - RDY_B[Z,X] shall be used to transfer credit on the blue loop

Serial Communications

FC–AL–3

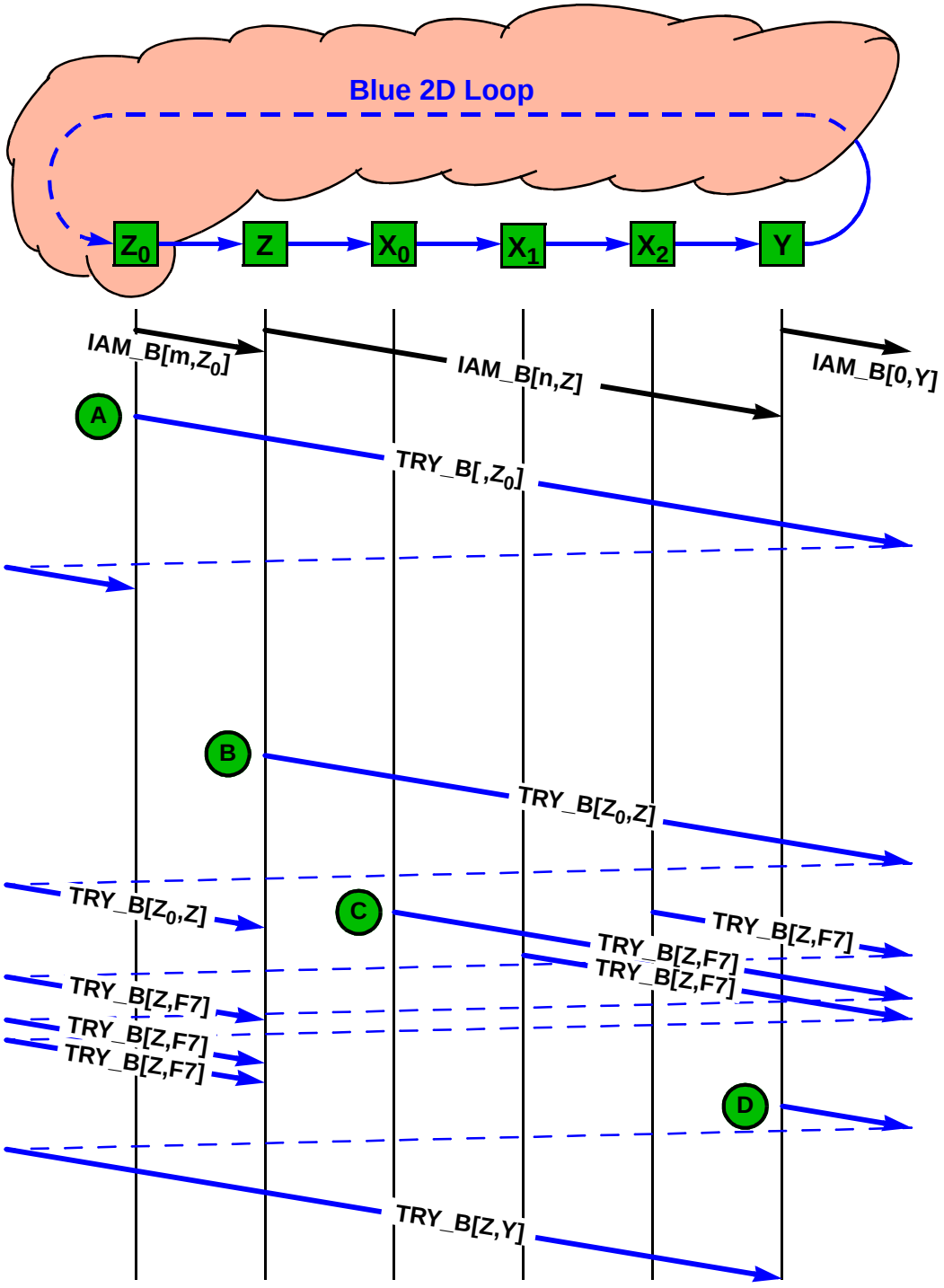
2D Loop
Loop Initialization

T11 9 Feb. 1998

Bent Støvhase (bent@inforamp.net)

FC–AL–3 – Loop Initialization

2D Loop Operation Scenario #1



Normal TRY propagation on a 2D Loop

FC–AL–3 – Loop Initialization

2D Loop Operation Scenario #1

- The Participating L_Ports 'Z_0', 'Z' and 'Y' sends IAM_B[n,X] Primitive Signals as Fill Words
  - The IAM Primitive Signal identify the sending L_Port to its immediate downstream neighbour
  - IAM Primitive Signals are not passed through Participating L_Ports
  - The Observing (Non-Participating) L_Ports 'X_0', 'X_1' and 'X_2' uses IDLE Primitive Signals as Fill Words
    - IAM Primitive Signals are passed through Observing L_Ports
    - IAM Primitive Signals may be removed for clock skew purposes
- **A** – L_Port 'Z_0' transmits a TRY_B[,Z_0] Primitive Signal
  - The TRY[,Z_0] Primitive Signal is extracted from the loop by L_Port 'Z_0', once it has traversed the loop

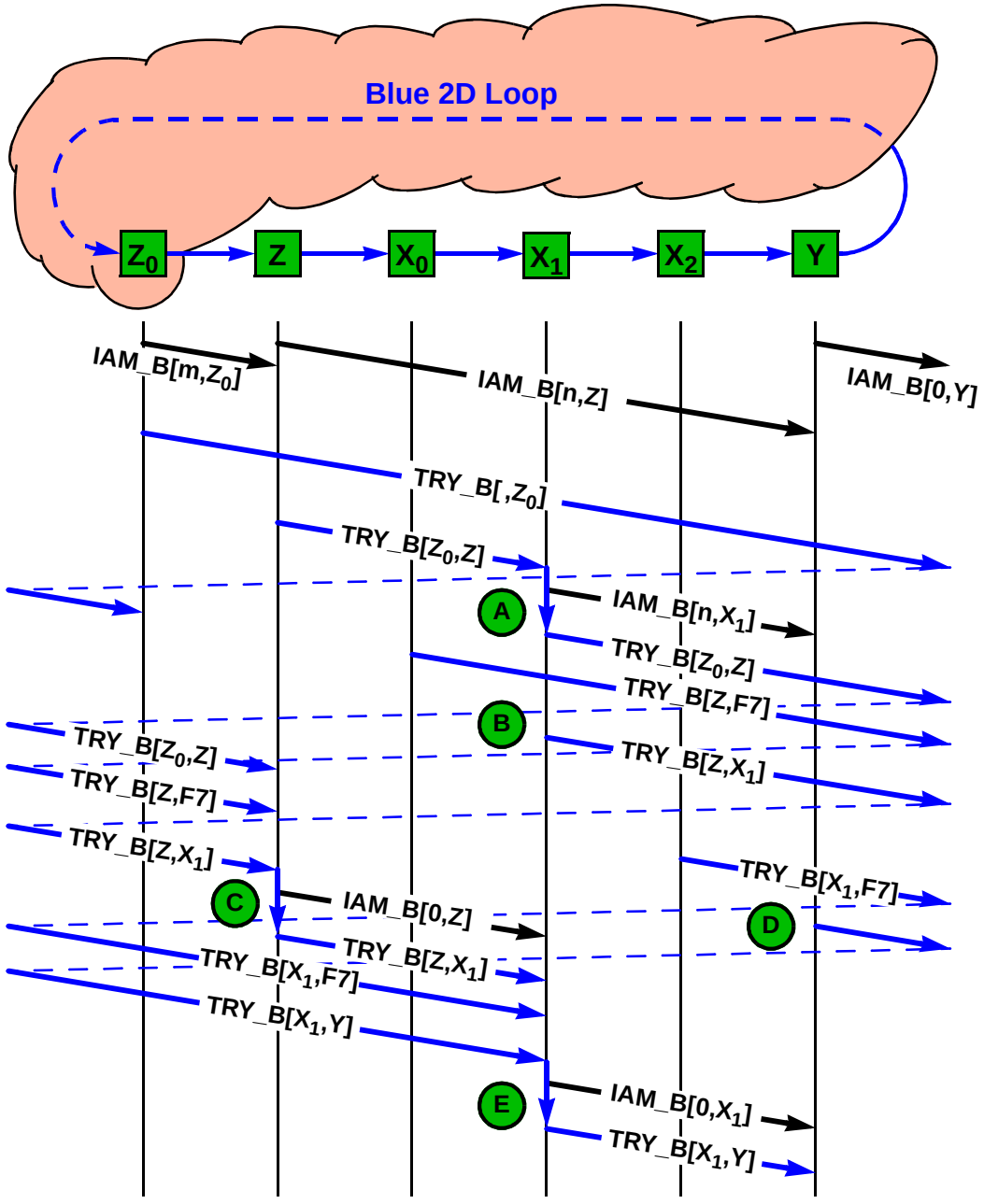
FC–AL–3 – Loop Initialization

2D Loop Operation Scenario #1

- **B** – L_Port 'Z' transmits a TRY_B[Z_0,Z] Primitive Signal, announcing its position on the loop
  - Z_0 = AL_PA address of the L_Port upstream to the originating L_Port
  - Z = AL_PA address of the originating L_Port
  - L_Port 'Z' transmits the TRY_B[Z_0,Z] Primitive Signal after a delay of between 80 and 100 µs from the receipt of the TRY_B[,Z_0] Primitive Signal
    - All L_Ports are required to send a delayed TRY Primitive Signal, once a TRY Primitive Signal originated by its upstream neighbouring L_Port is received
  - The TRY_B[Z_0,Z] Primitive Signal is extracted from the loop by L_Port 'Z', once it has traversed the loop
- **C** – L_Port 'X_0', 'X_1' and 'X_2' each inserts, after a 40 to 50 µs delay, a TRY_B[Z,F7] Primitive Signal
  - This identify the number and position of Observing L_Ports on the loop
  - The TRY_B[Z,F7] Primitive Signals are extracted from the loop by L_Port 'Z'
- **D** – L_Port 'Y' inserts its TRY_B[Z,Y] Primitive signal, 80 to 100 µs, after the receipt of the TRY_B[Z_0,Z] Primitive Signal
  - The TRY_B[Z,Y] Primitive Signal is extracted from the loop by L_Port 'Y', once it has traversed the loop

FC–AL–3 – Loop Initialization

Initialization Scenario #1 (Warm Start)



Joining an operating 2D Loop

FC–AL–3 – Loop Initialization

Initialization Scenario #1 (Warm Start)

- The Participating L_Ports 'Z_0', 'Z' and 'Y' sends IAM_B[n,X] Primitive Signals as Fill Words
  - The Observing L_Ports 'X_0', 'X_1' and 'X_2' uses IDLE Primitive Signals as Fill Words
- **A** – The Joining L_Port 'X_1' changes it its Fill Word to IAM_B[n,X_1] when it receives the TRY_B[Z_0,Z] Primitive Signal
  - The transmitter credit value 'n' is copied (inherited) from the received IAM_B[n,Z] Fill Word
  - L_Port 'X_1' precede the received TRY_B[Z_0,Z] with 12 IAM_B[n,X_1] Primitive Signals, when the TRY_B[Z_0,Z] is forwarded
- **B** – L_Port 'X_1' inserts a TRY_B[Z,X_1] Primitive Signal, announcing its position on the loop
  - L_Port 'X_1' inserts the TRY_B[Z,X_1] Primitive Signal between 80 and 100 µs after the receipt of the TRY_B[Z_0,Z] Primitive Signal
- **C** – The upstream L_Port 'Z' withhold forwarding the TRY_B[Z,X_1] Primitive Signal until it have reset its transmitter credit
  - The TRY_B[Z,X_1] Primitive Signal is preceded by 12 IAM_B[0,Z] Fill Words before it is forwarded

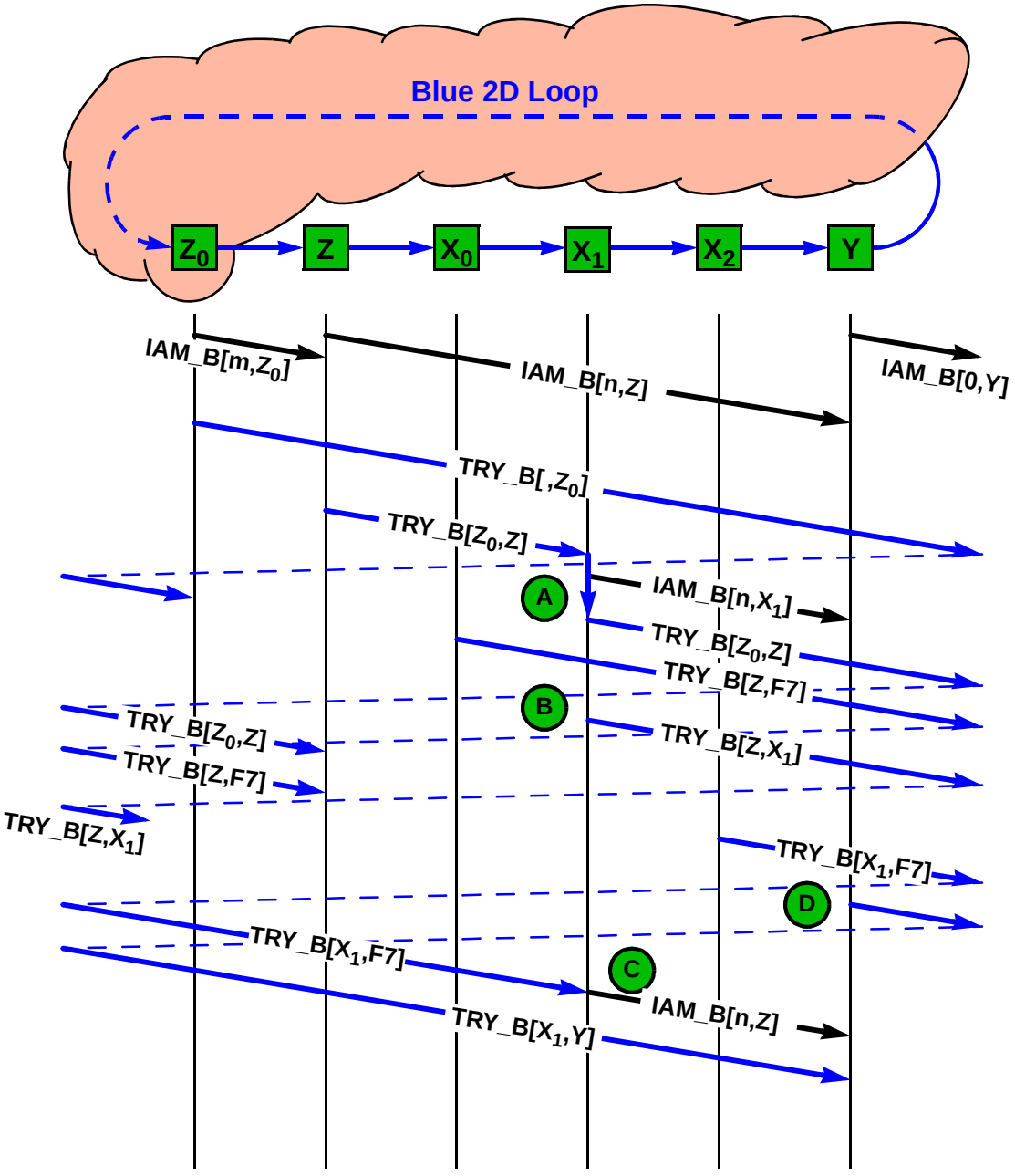
FC–AL–3 – Loop Initialization

Initialization Scenario #1 (Warm Start)

- **D** – L_Port 'Y' inserts a TRY_B[X_1,Y] Primitive Signal, announcing its 'new' position on the loop
  - L_Port 'Y' inserts the TRY_B[Z,X_1] Primitive Signal between 80 and 100 µs after the receipt of the TRY_B[Z,X_1] Primitive Signal
- **E** – The upstream L_Port 'X_1' withhold forwarding the TRY_B[X_1,Y] Primitive Signal until it have reset its transmitter credit
  - The TRY_B[X_1,Y] Primitive Signal is preceded by 12 IAM_B[0,X_1] Fill Words before it is forwarded

FC–AL–3 – Loop Initialization

Initialization Scenario #2 (Warm Start)



Joining failure on an operating 2D Loop

FC–AL–3 – Loop Initialization

Initialization Scenario #2 (Warm Start)

- The Participating L_Ports 'Z_0', 'Z' and 'Y' sends IAM_B[n,X] Primitive Signals as Fill Words
  - The Observing L_Ports 'X_0', 'X_1' and 'X_2' uses IDLE Primitive Signals as Fill Words
- **A** – The Joining L_Port 'X_1' changes it its Fill Word to IAM_B[n,X_1] when it receives the TRY_B[Z_0,Z] Primitive Signal
  - The transmitter credit value 'n' is copied (inherited) from the received IAM_B[n,Z] Fill Word
  - L_Port 'X_1' precede the received TRY_B[Z_0,Z] with 12 IAM_B[n,X_1] Primitive Signals, when the TRY_B[Z_0,Z] is forwarded
- **B** – L_Port 'X_1' inserts a TRY_B[Z,X_1] Primitive Signal, announcing its position on the loop
  - L_Port 'X_1' inserts the TRY_B[Z,X_1] Primitive Signal between 80 and 100 µs after the receipt of the TRY_B[Z_0,Z] Primitive Signal
  - The TRY_B[Z,X_1] Primitive Signal is blocked (discarded) or lost somewhere between L_Port 'Y' and L_port 'Z_0'

FC–AL–3 – Loop Initialization

Initialization Scenario #2 (Warm Start)

- **C** – L_Port 'X_1' discover that it failed to obtain an address on the operating 2D loop, when it receive the TRY_B[X_1,F7] Primitive Signal
  - The TRY_B[Z,X_1] Primitive Signal would have preceded the TRY_B[X_1,F7] Primitive Signal
  - L_Port 'X_1' starts forwarding the IAM_B[n,Z] Fill Word sent by L_Port 'Z', when it returns to the Observing state
- **D** – L_Port 'Y' inserts a TRY_B[X_1,Y] Primitive Signal, announcing its 'new' position on the loop
  - L_Port 'Y' inserts the TRY_B[Z,X_1] Primitive Signal between 80 and 100 µs after the receipt of the TRY_B[Z,X_1] Primitive Signal

FC–AL–3 – Loop Initialization

Rules – Summary

- L_Ports in the 2D state shall extract TRY_x[Z,X] Primitive Signal when
  - X = L_Port's AL_PA
  - Z = L_Port's AL_PA and X = 'F7' (≥ 'F0')
- A Joining L_Port shall claim an address (AL_PA) not observed in the preceding TRY Primitive Signal cycle
  - The interval between insertions of TRY_x[Z,F7] Primitive Signals
- A Joining L_Port shall insert 12 of its own IAM Primitive Signals before forwarding a TRY Primitive Signal from its upstream neighbour
- A Joining attempt have failed if
  - A FLT Primitive Signal is received
  - Loss of Sync or Loss of Signal is detected
  - A TRY_x[Z,X] Primitive Signal is received where Z = Joining L_Ports address
  - A TRY_x[Z,X] Primitive Signal is received where X = Upstream L_Ports address if the Joining L_Port have inserted its own TRY_x[Z,X] Primitive Signal
  - An L_Port which Joining attempts has failed shall return to the Observing state

Serial Communications

FC–AL–3

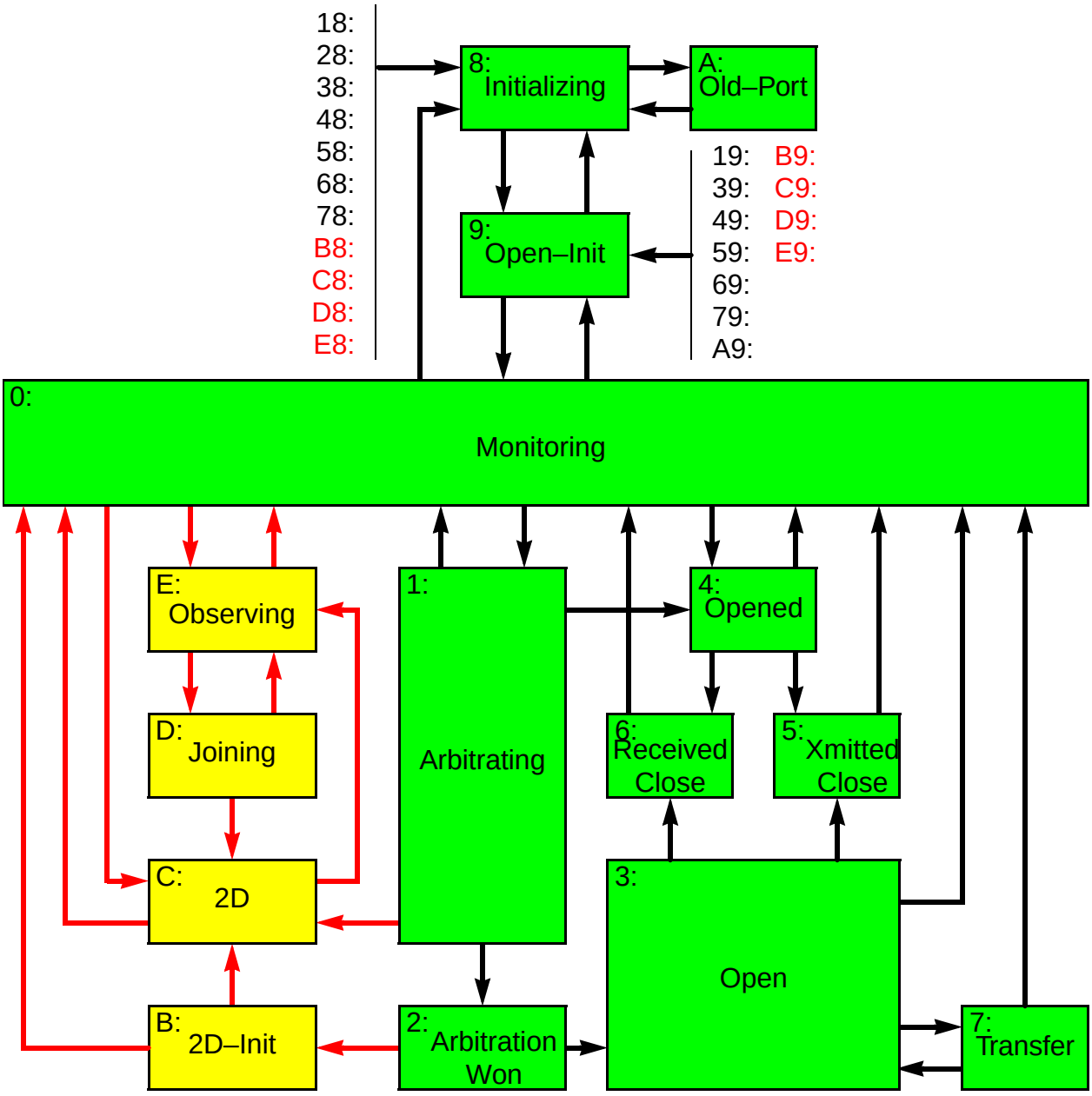
2D Loop
Loop Identification

T11 9 Feb. 1998

Bent Støvhase (bent@inforamp.net)

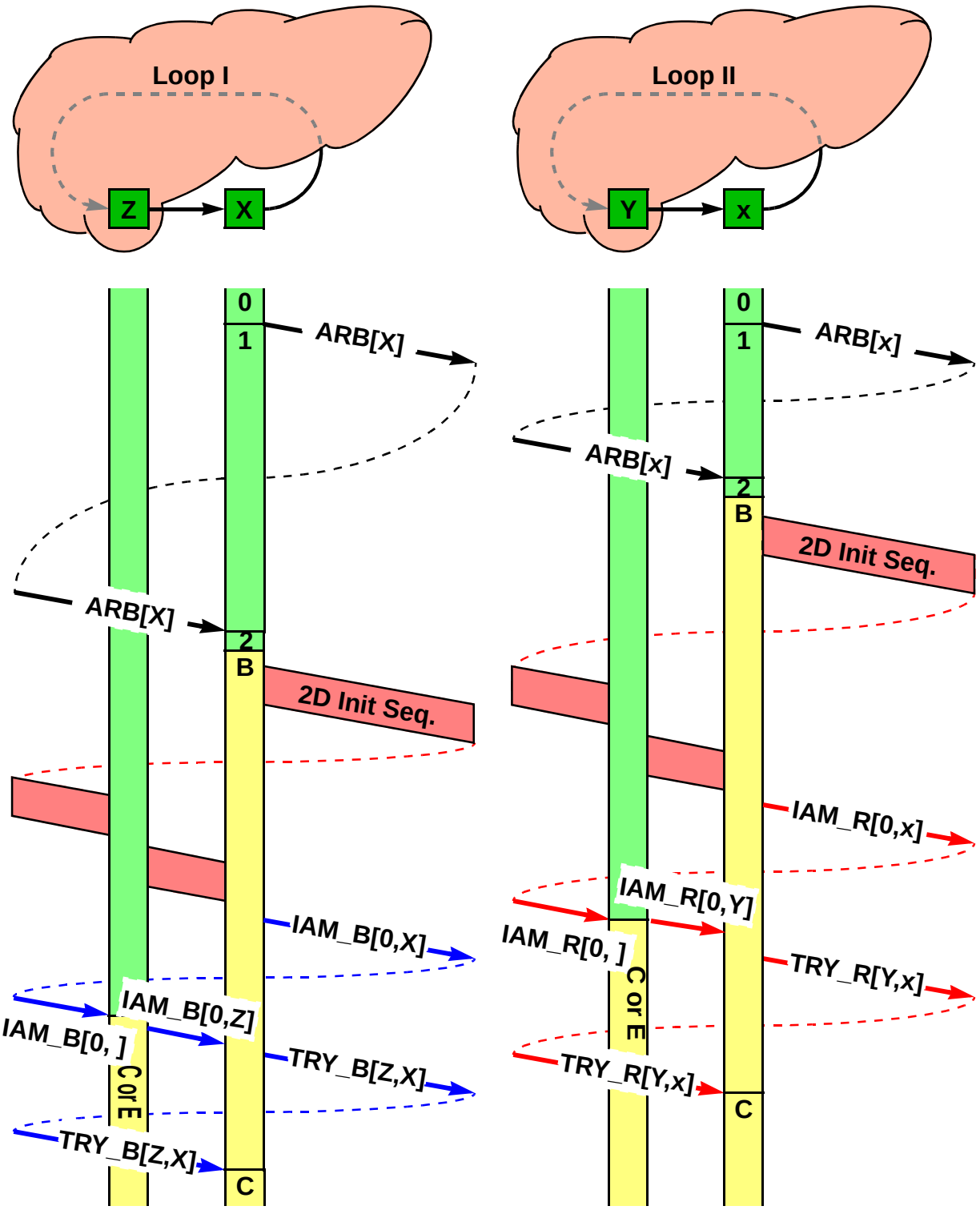
FC–AL–3 – Loop Identification

2D Loop State Diagram



State Diagram

FC–AL–3 – Loop Identification



Successful 2D Loop Initialization Example

FC–AL–3 – Loop Identification

2D Loop Initialization

- Only dual ported entities may request entry to the 2D mode
  - The requesting dual ported entity shall, before it proceed to attempt to activate the 2D mode
    - Successfully complete FC–AL–2 loop initialization on both ports
    - Attempt Fabric Login on both loops
      - If a Fabric is present then the requesting entity shall not proceed before it successfully completes Fabric Login
    - Ensure both loops are on the same Fabric if both ports are on a Public Loop
      - If the Fabric Names are different, then operation in 2D mode shall not be requested
- The requesting entity shall Arbitrate on both loops
  - The requesting entity shall send the single frame '2D Initialization Sequence' when it wins Arbitration and enters the 2D–Init state on that port
    - Initialization shall proceed independently on the two loops
  - The requesting L_Port shall select an IAM_x[,X] Primitive Signal as its Fill Word, once the '2D Initialization Sequence' returns to the entity requesting 2D Loop operation
    - X = AL_PA of the sending L_Port
    - IAM_R[–,X] shall be used for the Red Loop
      - The loop with the lower valued addresses
    - IAM_B[–,X] shall be used for the Blue Loop
      - The loop with the higher valued addresses

FC–AL–3 – Loop Identification

2D Loop Initialization

- The requesting L_Port shall transmit FLT_x[FF,FF] Primitive Signals continuously until the port either receive a FLT_x[FF,FF] Primitive Signal or 5 ms have lapsed, if the requesting L_Port receive a FLT_x[X,X] Primitive Signal or if it continuously receives an IAM_x Primitive of the wrong colour
  - The requesting L_Port shall return to the Monitoring state once it cease to transmit the FLT_x[FF,FF] Primitive Signal
  - FLT_R shall be used for the Red Loop
  - FLT_B shall be used for the Blue Loop
- The requesting L_Port shall transmit a TRY_x[Z,X] Primitive Signal once it receive a IAM_x Primitive Signal, valid for the loop
  - X = AL_PA of the L_Port originating the TRY Primitive Signal
  - Z = AL_PA of the originating L_Ports upstream neighbour
  - TRY_R[Z,X] shall be used for the Red Loop
  - TRY_B[Z,X] shall be used for the Blue Loop
- The requesting L_Port shall enter the 2D state, when it receives and discard the transmitted TRY_x[Z,X] Primitive Signal
  - The requesting entity shall continue its attempts to establish 2D loop operation on the adjoining loop if this isn't already done
  - Another dual ported entity may complete or have completed the transition to 2D Loop operation on the adjoining loop

FC–AL–3 – Loop Identification

2D Loop Initialization

- Timeouts
  - The entity attempting to establish a working 2D loop complex, shall, if it have not succeeded within LP_TOV, restore Arbitrated Loop operation
    - The LP_TOV starts from the time an L_Port on the requesting entity first wins arbitration
    - The L_Port shall transmit FLT_x[FF,FF] Primitive Signals to restore Arbitrated Loop operation
      - FLT_R shall be used for the Red Loop
      - FLT_B shall be used for the Blue Loop
      - Transmission of FLT_x[FF,FF] Primitive Signals shall cease when FLT_x[FF,FF] Primitive Signals are received by the originator or after 5 ms

- 2D compatible L_Ports shall promiscuously receive, validate and forward the single frame ‘2D Initialization Sequence’
  - Both Participating and Non–Participating entities shall perform the described actions
    - Non–Participating L_Port on an operational 2D Loop partake in Primitive Signal processes, but are prohibited form extracting and originating Data Frames

FC–AL–3 – Loop Identification

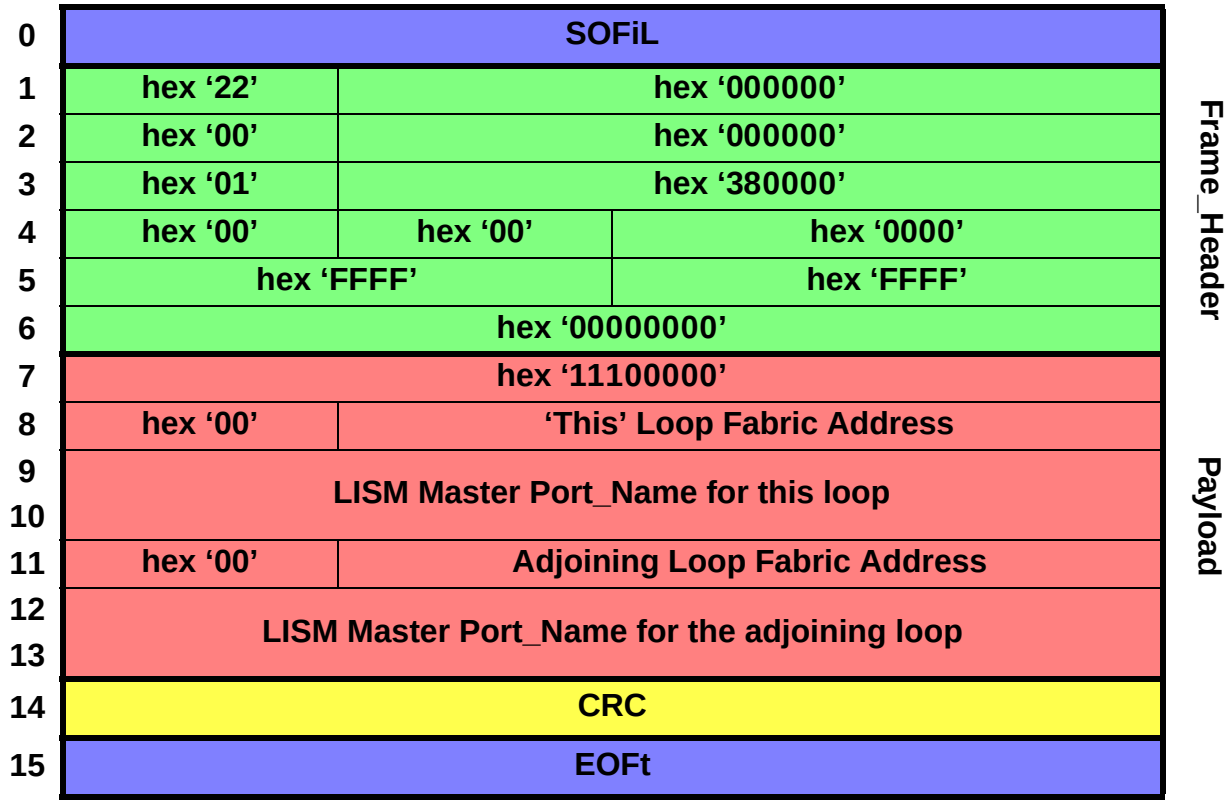
2D Loop Initialization

- The ‘2D Initialization Sequence’ shall only be forwarded if it is validated successfully
  - The ‘LISM Master Port_Name for this loop’ field shall match the receiving L_Ports value for this field
  - The ‘‘This’ Loop Fabric Address’ field shall, for L_Port’s having successfully completed Fabric Login, match the receiving L_Port’s value for this field
  - The ‘LISM Master Port_Name for the adjoining loop’ field should, for dual ported entities, match the receiving L_Ports value
  - The ‘Adjoining Loop Fabric Address’ field shall, for dual ported entities, match the receiving L_Ports value, if that port have completed Fabric Login successfully
  - The L_Port shall transmit a FLT_R[AL_PA,AL_PA] Primitive Signal if the port elects to discard the single frame ‘2D Initialization Sequence’
    - Non–Participating L_Ports shall transmit a FLT_RF7,F7] Primitive Signal
- L_Ports shall delay forwarding the ‘2D Initialization Sequence’ for at most 1 ms
- Participating L_Port shall enter the 2D state when they receive an IAM_x[–,X] Primitive Signal
  - If the L_Port have not received and validated the ‘2D Initialization Sequence’ then it should insert a FLT_x[AL_PA,AL_PA] Primitive Signal, matching the colour of the received IAM
- A Non–Participating L_Port shall enter the Observing state when they receive an IAM_x[–,X] Primitive Signal and set its Fill Word to Idle

FC–AL–3 – Loop Identification

2D Loop Initialization

2D Initialization Sequence



- Frame Delimiters, as specified in FC–AL–2 for Loop Initialization
- Frame Header, as specified in FC–AL–2, for Loop Initialization by an FL_Port
- CRC field, as specified by FC–PH

FC–AL–3 – Loop Identification

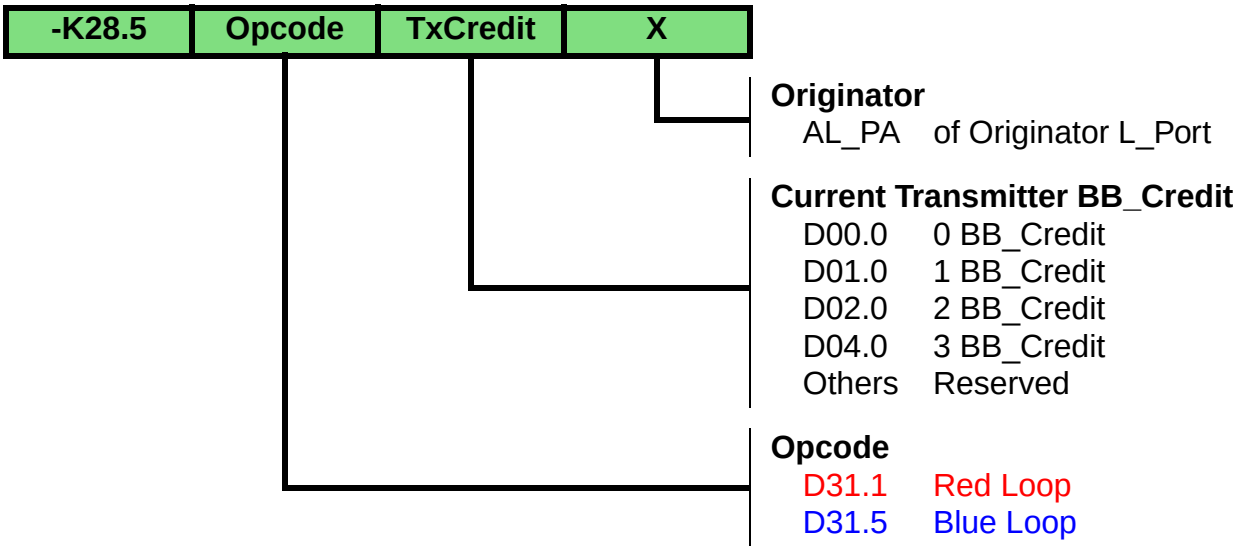
2D Loop Initialization

- 2D Initialization Sequence fields
  - ‘This’ Loop Fabric Address
    - The Loop Fabric Address for the loop on which the 2D Initialization Sequence is sent
      - hex ‘000000’ if both loops are Private and this loop’s LISM Master Port_Name is the lower valued name
      - hex ‘000100’ if both loops are Private and this loop’s LISM Master Port_Name is the higher valued name
  - LISM Master Port_Name for this loop
    - The Port_Name of the entity which won LISM mastership for the loop on which the 2D Initialization Sequence is sent
  - Adjoining Loop Fabric Address
    - The Loop Fabric Address for the companion loop
  - LISM Master Port_Name for the adjoining loop
    - The Port_Name of the entity which won LISM mastership for the companion loop
      - hex ‘000000’ if both loops are Private and the companion loop’s LISM Master Port_Name is the lower valued name
      - hex ‘000100’ if both loops are Private and the companion loop’s LISM Master Port_Name is the higher valued name

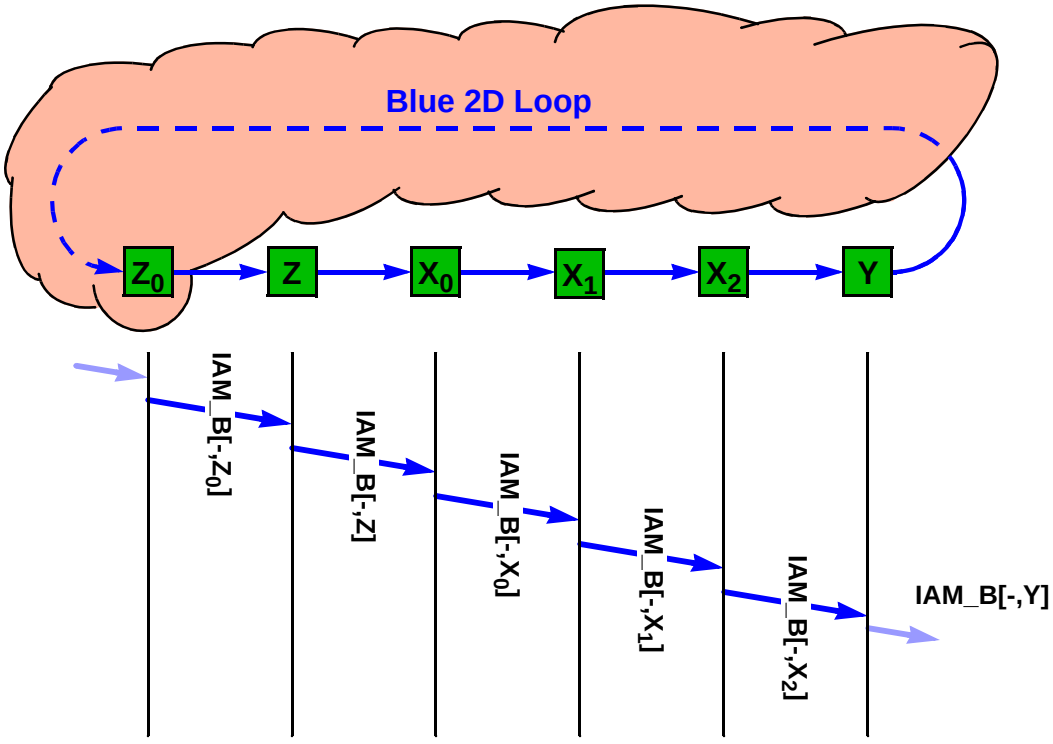
FC–AL–3 – Loop Identification

New Primitive Signals

- IAM Primitive Signal



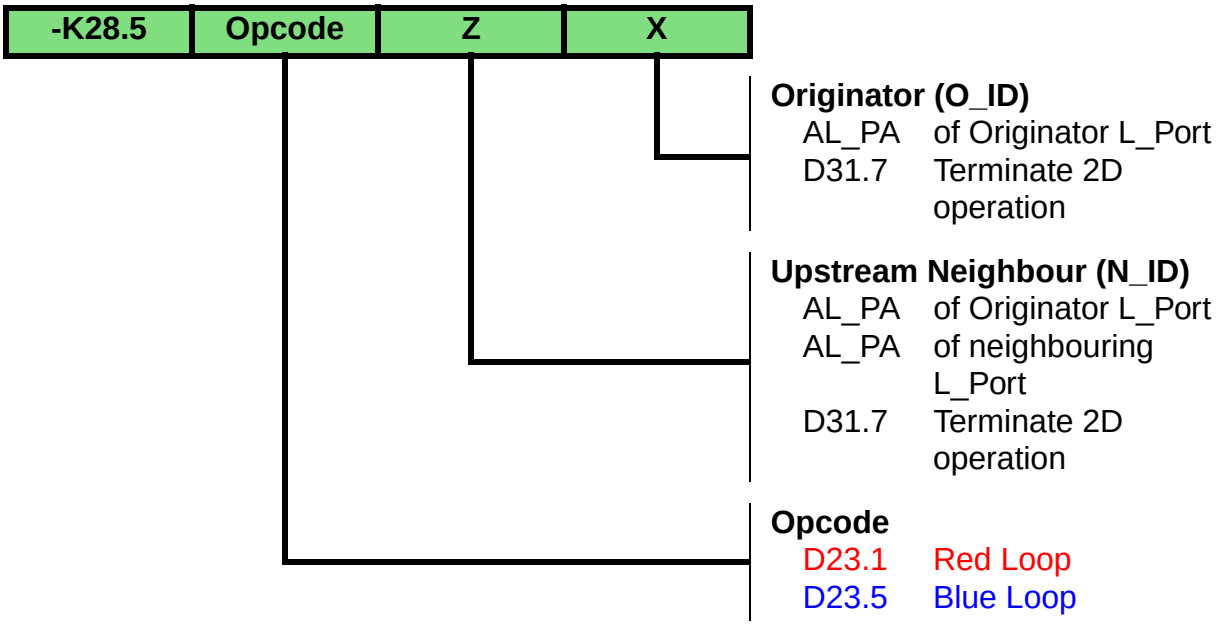
- Fill Word for Participating L_Ports (traverses a single 'split' link only)



FC–AL–3 – Loop Identification

New Primitive Signals

- Fault (FLT) Primitive Signal

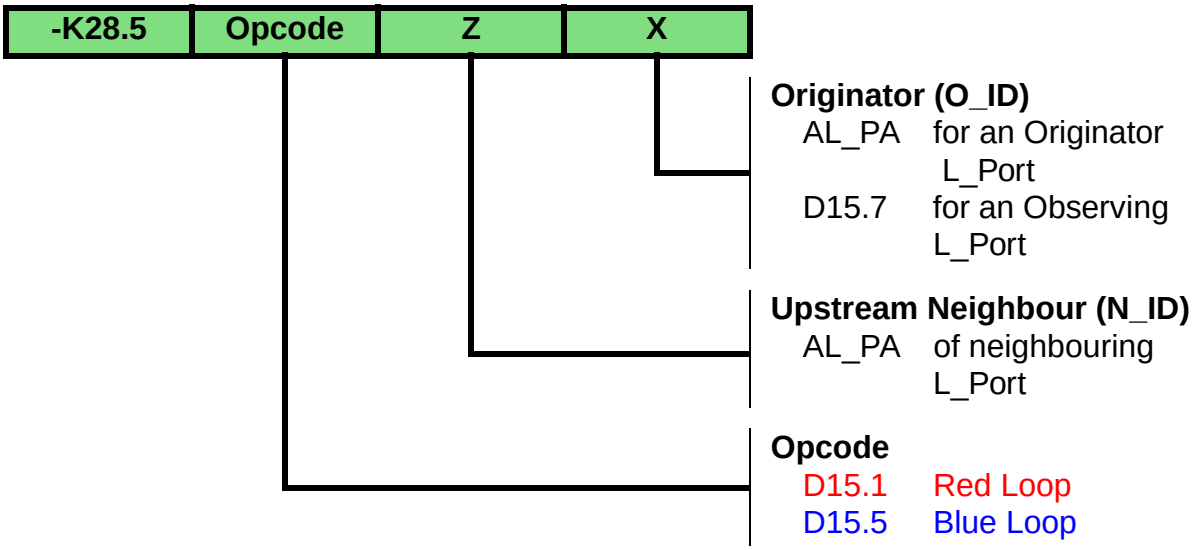


  - Announces error conditions on the loop (traverses the entire loop)
    - L_Ports shall insert a FLT_x[Z,X] every 8 to 10 ms in response to ‘Loss of Signal’ or ‘Loss of Sync.’ for more than R_T_TOV
      - FLT_R[Z,X] shall be used for the red loop
      - FLT_B[Z,X] shall be used for the blue loop
    - L_Ports shall insert a FLT_R[X,X] Primitive Signal to signify its rejection and discarding of the ‘2D Initialization Sequence’
    - L_Ports shall insert FLT_x[FF,FF] Primitive Signals to restore the Loop to Arbitrated Loop mode
      - FLT_R[FF,FF] shall be used for the red loop, or if the colour of the loop is unknown
      - FLT_B[FF,FF] shall be used for the blue loop

FC–AL–3 – Loop Identification

New Primitive Signals

- TRY Primitive Signal



- Announces the originator's location on the loop (traverses the entire loop)
  - An L_Port shall insert its TRY_x[Z,X] Primitive Signal at the most once every 4 to 5 ms (Repetition limit)
    - If one or more insertions are delayed by the repetition limit then a single TRY_x[Z,X] Primitive Signal shall be sent when the repetition timeout limit is satisfied
  - An L_Port shall insert its TRY_x[Z,X] Primitive Signal, in response to receipt of a TRY_x[Z,X] or FLT_x[Z,X] Primitive Signal originated by the upstream neighbour
    - The insertion shall be delayed 40 to 50 µs for L_Ports in the Observing state
    - The insertion shall be delayed 80 to 100 µs for L_Ports in the Joining or 2D state, subject to the 5 ms repetition limit
  - An L_Port in the 2D state shall insert its TRY_x[Z,X] Primitive Signal at least once every 16 to 20 ms
  - TRY_R[Z,X] shall be used on the red loop
  - TRY_B[Z,X] shall be used on the blue loop

Accredited Standards Committee
NCITS, National Committee for Information Technology Standards

Doc: FCW/98w119r0
Date: January 30, 1998
Project:
Ref Doc.:
Reply to: Jim Coomes

To: T11 Membership
From: Jim Coomes

Subject: New link error reporting

A new method for reporting link errors has been proposed for FC and particularly FC-AL. This proposal is for a method to report these errors while running test or ULP protocol in a test mode. The proposal calls for a new primitive signal with an error count field.

First the problems:

The definition of a new primitive is loaded with complexity:

- for ports in Monitoring state, these ports would have to detect frame boundaries and process the frame content. This is not required today and was met with stiff opposition in the FCL Error Recovery SSWG;
- to be even partly reliable, the primitives could not be deleted for clock skew management. If different ports on a loop detect different errors within the same frame, more than one error primitive would follow the frame. If these primitives replace fill words, the ability to perform clock skew management would be broken. If the error primitives are inserted with the elasticity FIFO expanded, the depth of FIFOs would have to increase;
- the delivery of the new error primitive would be unreliable with any present FC-AL devices on the loop. Ports in the "Open" states would remove the primitives;
- the requirement that a port scrub the primitives it originates needs a fail-over to cover the cases of the originating port being removed or the primitive being corrupted; and
- the complexity of future devices would increase if they are required to buffer received error primitives while they are in an Open state and retransmit them.

There is no need for a new way to report link errors. FC-PH already defines a method, the Link Error Status Block for ports to log and report link errors. The problem to date is that FC-PH allows the implementation and degree of support to be optional. If the goal of this new proposal is to provide a link/loop monitor the capability to capture link error information, the LESB function should be enhanced.

Sure a link monitor that wants to gather error information has to initiate the Read Link Status ELS to recover the LESBs. But, the monitor is the device that should bare the burden of the function. The complexity should not be forced on all devices.

Where the real problem is:

- Implementations of the LESB have varied in the implementation due to the lack of consensus on what errors should be counted. A clear definition is needed of what errors may be meaningfully and reasonably counted.
- FC-PH does not include a method for resetting the count fields. They are just allowed to wrap. This means the link monitor must keep the previous readings to determine if the count is increasing.
- Bit error rate testers have a big advantage in detecting and quantifying link errors. They know exactly what was transferred and can compare this expected data to what was received. This allows them to isolate the length of errors.

- When functional information is being transferred on the link, transmitted and received information are quite different. A receiving port does not know what information to expect. It may detect an error in the first character of a frame. This error could be a single bit error and cause running disparity to be wrong for all the remaining
- characters in the frame as running disparity is only recovered between frames. This single bit error may cause 536 Invalid -Transmission-Words for a full size frame before running disparity is restored. But after four consecutive Invalid -Transmission-Words a Loss-of Synchronization occurs.

Thus, determining a bit error rate in a functional system is just not possible. Determining the relative quality of the links is an achievable goal.

The suggested approach:

Support of a clearly defined LESB should required. This would allow link monitors to obtain what information is presently available in LESBs and take advantage of more comprehensive error data when it becomes available.

Jim Coomes
Jim_Coomes@notes.seagate.com

Multiple Circuit Mode Overview

- Credit is maintained in the same manner as FC_AL and FC_AL2
- Arbitration is used to enter multiple circuit mode
- Devices which wish to maintain more than one circuit at a time may by applying the resources required (additional credit counters)
- Existing primitives are used
- SAT (as defined in Torn) is used
- Fully backward compatible with FC_AL and FC_AL2
- Uses many of the concepts from the Torn proposal and RIP proposals

Western Digital

MCM Differs From Other Proposals

- RIP did not use end point to end point flow control (used nearest neighbor)
- Torn did not use destination addressed credit (source address only used)
- MCM uses fully addressed, end point to end point flow control
- MCM is entered and exited on the fly rather than at loop initialization time
- MCM does not require enabled frame counting
- MCM does not require a scrubber since the source can watch for its own frames

Western Digital

MCM Is Similar To Other Proposal

- MCM uses the RCVyx and RCVff concept of RIP
- MCM uses buffer insertion as defined in Torn and RIP
- MCM does not require arbitration per connection allowing for spatial reuse
- MCM uses the credit concepts created in FC-AL

Western Digital

Entering Multiple Circuit Mode

- NL_Port wishing the loop to enter multiple circuit mode arbitrates as normal
- When arbitration is won, a new primitve called Enter Multiple Circuit Mode is sent by winning NL_Port
  - A lost EMCM primitive only causes devices which do not see primitve to not enter the mode
  - Any device which sees the EMCM will enter multiple channel mode and can communicate with any other device on the loop which understands MCM
- When a NL_Port detects the EMCM primitive, it may send initiate connections using MCM

Western Digital

Exiting Multiple Circuit Mode

- When a NL_Port detects an arbitration, all circuits are closed using normal close protocol from FC_AL
  - ARB may be generated by any port that wishes to use FC-AL protocol
    - May be an old port that does not know MCM
    - May be an MCM port that wished to talk to a non-MCM port
  - ARB may be generated by NL_Port which did not see the EMCM or lost the context for some other reason
- When all circuits are closed on a NL_Port, the NL_Port forwards the ARB
- When all devices which understand MCM have closed all cicuits, the arbitration will complete

Western Digital

Managing Circuits

- In order to begin communicating between two NL_Ports, an RCVyx/RCVff encapsulated OPNxy is sent using the two NL_Port AL_PAs
- Credit is maintained as it is in FC_AL and FC_AL2 except
  - RRDY is addressable with the source and destination AL_PA (RCVyx RCVff encapsulated)
  - Login BB credit need is diminished and should be removed
- If credit balance is lost or zero for a long time, a close followed by an open will restore credit balance
- Frames sent within MCM must be RCVyx/RCVff encapsulated to identify the source and destination

Western Digital

Managing Circuits (cont)

- Closing a circuit is performed by sending a RCVyx/RCVff encapsulated CLS which works exactly like FC_AL and FC_AL2 CLS
- The detection of an arbitration forces the requirement to close all MCM circuits as soon as possible
  - When ARB is received, a CLS should be sent at the best time (end of a sequence)
  - Once the corresponding CLS is received, the ARB may be forwarded
  - Once the ARB is forwared, no MCM circuits may be started until EMCM is seen
- Circuits may be kept open as long as desired

Western Digital

New Primitives Required

Primitive Name	Purpose
EMCM	Enter Multiple channel mode
RCVyx	Unicate packet - send from AL_PA x to AL_PA y
RCVfx	Broadcast replicate packet - send from AL_PA x to all
RCVex	Selective replicate packet - send from AL_PA x to selective group determined by the RCVye
RCVye	Selective replicate receive - AL_PA y should accept the selective replicate packet
RCVff	End of RIP packet - denotes the end of a RIP packet

Advantages of MCM

- Provides spatial reuse
- Provides a simpler buffer to buffer credit model
- High traffic devices can maintain a connection and credit for a long time
- Large number of connections can be supported by applying appropriate resources
- Credit reclamation is v ery easy
- Credit model is similar to FC-AL enabling smoother migration
- Backward compatible
- FC-AL and MCM devices can coexist on the same loop

Western Digital

Issues

- How does a N_Port determine that another N_Port supports MCM mode?
  - N_Port login parameter
  - Sending it and if it returns, then the N_Port did not support it
- If both the source and destination N_Ports disappear after a frame is started, will that frame propogate forever or will leaving MCM mode and going back to FC-AL mode get rid of it?
- Should all FC-AL primitives be RCVyx / RCVff encapsulated for simplicity?

Western Digital

Accredited Standards Committee
NCITS, National Committee for Information Technology Standards

Doc: FCW/98w101r0
Date: January 31, 1998
Project:
Ref Doc.:
Reply to: Dal Allan

To: T11 Membership
From: Dal Allan

Subject: January 1998 FCW meeting minutes

The working group was held 1/13-15 at the Hilton in Sunnyvale instead of the scheduled location at the Sheraton in Milpitas. Norm Harris apologized for the confusion which caused the relocation.

Horst Truestedt had completed Rev 5.9 of FC-AL-2.

Unfinished business from the last working group included a decision on how many bits had to be checked on an ARB primitive. The ARB is used to make a decision on forwarding and to win arbitration.

To forward, it was agreed that only 30 bits have to be checked. A poll of 13:4 recommended a footnote that byte 3 should be priority and that byte 4 may be used for some purpose other than serving as a duplicate in a future standard.

Matt Wakeley had encountered a situation in which the third and fourth bytes were different, and caused a false winning of arbitration. A poll of 17:1 decided that 40 bits had to be checked to win arbitration.

An extended debate on the merits or otherwise of allowing bytes 3 and 4 to be different at some time in the future led to a poll of 7:6 in favor.

THIS ISSUE WILL BE DISCUSSED AT THE FEBRUARY WORKING GROUP AND A DECISION MADE AT THE PLENARY.

Horst provided copies of his responses to comments from the letter ballot and the rest of the working group time was spent covering these. The review was not completed, because several improvements were made to the language of FC-AL-2 based on the comments and the responses. Although this matter occupied most of the working group time, a comment by comment description in not included in these minutes. Detailed information on the decisions made can be found in the document prepared by Horst on comment resolutions.

Brian Smith presented his view of the decision matrix that existed for tape in the private and public loop, generic fabric and with/without command queueing. He expressed confidence that the approach documented by Crossroads for the private loop can be expanded to fabrics and command queueing.

Dave Peterson had prepared three sets of documentation, two Class 3 (Crossroads and his own) plus one on Class 2. The Class 2 copies were mislaid by the airline so this approach was not discussed.

Dave's summary of the differences between the three proposals involved discussion of what was needed to come up with a single approach. Rob Basham insisted that if the FCP_CONF in Dave's proposal was integrated into the Crossroads proposal that this would solve all known PLDA issues.

Agreement was reached that PLDA-2 will be based on:

- The Crossroads Class 3 proposal
- The addition of FCP_CONF as defined by Dave Peterson

- The use of FCP_CONF will be negotiated at Process Login

Suggestions that Class 3 can be extended to cross fabrics met resistance from Dave, who felt Class 2 is essential. The issue raised in the plenary by Jeff Stai that PLDA-2 be expanded to include public operation was discussed.

A poll found everyone present agreeing that completion of a standard is essential to the tape industry, and if PLDA-2 it is to be completed quickly. it cannot define public operation. Dave indicated his willingness to edit whatever was needed next and issues of public operation will continue to be discussed in the working groups.

Attendees:

D. Allan	ENDL
D. Baldwin	Emulex
R. Basham	IBM
C. Binford	Symbios Logic
J. Coomes	Seagate
R. Cummings	DPT
N. Harris	Adaptec
E. Kohlwey	Ancor
B. Martin	Gadzoox
E. Moyle	Cypress Semi
J. Nelson	Northrop Grumman
D. Peterson	Network Systems
J. Scheible	IBM
P. Seto	Quantum
B. Smith	Crossroads
R. Snively	Sun Microsystems
J. Stai	Brocade
A. Stone	Unisys
R. Taborek	G2 Networks
H. Truestedt	ENDL
M. Wakeley	Hewlett Packard
N. Wanamaker	Crossroads
G. Weston-Lewis	Symbios Logic
S. Wyatt	Hewlett Packard
D. Ybarra	Adaptec

Sincerely,

Dal Allan