Copies of this document may be purchased from:
Global Engineering, 15 Inverness Way East,
Englewood, CO 80112-5704
Phone: (800) 854-7179 or (303) 792-2181 Fax: (303) 792-2192

X3.xxx-199x
X3T11/Project 959-D/Rev 3.3

FIBRE CHANNEL

SWITCH FABRIC

(FC-SW)

REV 3.3

NCITS working draft proposed
American National Standard
for Information Technology

October 21, 1997

Secretariat: Information Technology Industry Council

NOTE:

This is a working draft American National Standard of Accredited Standards Committee NCITS. As such this is not a completed standard. The T11 Technical Committee or anyone else may modify this document as a result of comments received anytime, or during a future public review and its eventual approval as a Standard. Use of the information contained herein is at your own risk.

POINTS OF CONTACT:

Roger Cummings (T11 Chairman)
Distributed Processing Technology
140 Candace Drive
Maitland, FL 32751
(407) 830-5522 x348
Fax: (407) 260-5366
EMail: cummings_roger@dpt.com

Ed Grivna (T11 Vice Chairman)
Cypress Semiconductor
2401 East 86th Street
Bloomington, MN 55425
(612) 851-5046
Fax: (612) 851-5087
E-Mail: elg@cypress.com

I. Dal Allan (FC Working Group Chairman)
ENDL
14426 Black Walnut Court
Saratoga, CA 95070
(408) 867-6630
Fax: (408) 867-2115
E-Mail: dal.allan@mcimail.com

Jeffrey Stai (Technical Editor)
Brocade Communications Systems, Inc.
15707 Rockfield Boulevard, Suite 215
Irvine, CA 92618
(714) 455-2908
Fax: (714) 455-9287
E-mail: stai@brocadecomm.com

Editor’s Notes, revision 3.3 (comment resolution, 1st letter ballot):

- In EFP, the value 0x01 is used for forcing a switch to become Principal, 0x02 is used for prior assign.
- An FC-SW compliant implementation can have one Switch Port (this is for FC-BB).
- Added another case for E_Port Isolation in 7.4.
- A few minor editorial items. I also finally ran spell check and picked up a few booboos.

Editor’s Notes, revision 3.2 (for 1st letter ballot):

- The RDI SW_ILS now has the ability to request more than one contiguous Domain_ID.
- The ELP SW_ILS has had the “Profile” specific information replaced with “E_Port Credit Parameters”.
- A couple of figures were added to 7.2 and 7.3, to hopefully clarify some of the processes.
- In Annex C, the term “Static Switch” is changed to “Link Switch”.

Editor’s Notes, revision 3.1:

- The “Domain_Map” has been changed to a “Domain_ID_List”. This change makes it unambiguous as to who owns an ID when the EFP is distributed.
- “Domain Address Manager” and “Area Address Manager” were mostly removed in favor of “Principal Switch” to reduce a bit of terminology glut.
- Added some more description to clause 5. Hope you like it.
- Clarification was added in several places to note that the Class F data field size may be increased via the ELP service.
- LSU, LSA, HLO were removed. They will be defined in SW-2.
- Added section on SW_ACC.
- Moved SEQ bit in ELP to match FC-PH convention.
- Added Receive Data Field Size for Class 1 parameters in ELP.
- DSCN and LOOPD got some extra definition.
- Switch Port initialize has been rewritten for clarity.
- Some corrections to Principal Switch selection.
- Some clarifications to Isolation.
- Annex A is now the Switch Profile annex.
- Annex B added to contain Switch Port initialization examples.
- Annex C added as per agreement with FC-AE group.

{Add Editor’s “thank you” section here!}

ANSI ®
X3.xxx-199x

draft proposed American National Standard
for Information Technology

Fibre Channel —
Switch Fabric (FC-SW)

Secretariat

Information Technology Industry Council

Approved ,199

American National Standards Institute, Inc.

Abstract

This standard describes the requirements for an interconnecting Fabric consisting of multiple Fabric Switch elements to support the ANSI X3.230-1994 Fibre Channel - Physical and Signaling Interface (FC-PH) standard.

PATENT STATEMENT

The developers of this Standard have requested that holder's of patents that may be required for the implementation of the Standard, disclose such patents to the publisher. However, neither the developers nor the publisher have undertaken a patent search in order to identify which, if any, patents may apply to this Standard.

As of the date of publication of this Standard and following calls for the identification of patents that may be required for the implementation of the Standard, no such claims have been made. No further patent search is conducted by the developer or the publisher in respect to any Standard it processes. No representation is made or implied that licenses are not required to avoid infringement in the use of this Standard.

Published by

American National Standards Institute
11 W. 42nd Street, New York, New York 10036

Printed in the United States of America

INSERT CODE HERE

Contents

Page

Figures

Tables

Page

draft proposed American National Standard
for Information Technology—

Fibre Channel —
Switch Fabric (FC-SW)

1 Introduction and scope

This American National Standard for FC-SW specifies tools and algorithms for interconnection and initialization of Fibre Channel switches to create a multi-switch Fibre Channel Fabric. This Standard defines an E_Port ("Expansion Port") that operates in a manner similar to an N_Port and F_Port, as defined in ANSI X3.230 FC-PH, with additional functionality provided for interconnecting switches.

This Standard also defines how ports that are capable of being an E_Port, F_Port, and/or FL_Port may discover and self-configure for their appropriate operating mode. Once a port establishes that it is connected to another switch and is operating as an E_Port, an address assignment algorithm is executed to allocate port addresses throughout the Fabric.

This Standard does not define credit models and management between E_Ports for the various Classes of Service other than Class F. Broadcast and multicast services are not defined. E_Ports conforming to this Standard support Class F, and also Class 1, Class 2, and/or Class 3; support for other Classes of Service are not defined by this Standard. The method by which routing of frames is established and effected is not described.

2 Normative references

The following Standards contain provisions which, through reference in the text, constitute provisions of this Standard. At the time of publication, the editions indicated were valid. All Standards are subject to revision, and parties to agreements based on this Standard are encouraged to investigate the possibility of applying the most recent editions of the Standards listed below.

Copies of the following documents can be obtained from ANSI: Approved ANSI Standards, approved and draft international and regional Standards (ISO, IEC, CEN/CENELEC), and approved foreign Standards (including BSI, JIS, and DIN). For further information, contact ANSI Customer Service Department at 212-642-4900 (phone), 212-302-1286 (fax) or via the World Wide Web at http://www.ansi.org.

Additional availability contact information is provided below as needed.

2.1 Approved references

[1] ANSI X3.230-1994, *Information Technology - Fibre Channel Physical and Signaling Interface (FC-PH)*.

[2] ANSI X3.297-1997, *Information Technology - Fibre Channel - Physical and Signalling Interface-2 (FC-PH-2)*

[3] ANSI X3.272-1996, *Information Technology - Fibre Channel - Arbitrated Loop (FC-AL).*

[4] ANSI X3.288-1996, *Information Technology - Fibre Channel - Generic Services (FC-GS).*

[5] ANSI X3.289-1996, *Information Technology - Fibre Channel - Fabric Generic (FC-FG)*.

2.2 References under development

At the time of publication, the following referenced Standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant Standards body or other organization as indicated.

NOTE – For more information on the current status of a document, contact the NCITS Secretariat at the address listed in the front matter. To obtain copies of this document, contact Global Engineering at the address listed in the front matter, or the NCITS Secretariat.

[6] ANSI X3.303-199x, *Fibre Channel - Physical and Signalling Interface-3 (FC-PH-3)*, T11/Project 1119D/Rev 9.3

[7] ANSI X3.xxx-199x, *Fibre Channel - Arbitrated Loop (FC-AL-2),* T11/Project 1133D/Rev 5.7

[8] ANSI X3.xxx-199x, *Fibre Channel - Backbone (FC-BB),* T11/Project 1238D/Rev ?.?

[9] ANSI X3.xxx-199x, *Fibre Channel - Generic Services-2 (FC-GS-2)*, T11/Project 1134D/Rev 4.0

[10] ANSI NCITS TR-20-199x, *Fibre Channel - Fabric Loop Attachment (FC-FLA)*, T11/Project 1235DT/Rev 2.7

2.3 Other references

All of the following profiles are available from the Fibre Channel Association (FCA), 12407 MoPac Expressway North 100-357, P. O. Box 9700, Austin, TX 78758-9700; (800) 272-4618 (phone); or via e-mail, FCA-Info@amcc.com.

[11] FCSI-101*, FCSI Common FC-PH Feature Sets Used in Multiple Profiles*, Rev 3.1

[12] *FCA N_Port to F_Port Interoperability Profile,* Rev 1.0

3 Definitions and conventions

For FC-SW, the following definitions, conventions, abbreviations, acronyms, and symbols apply.

3.1 Definitions

3.1.1 address assignment: A process whereby addresses are dispensed to Switches and Switch Ports.

3.1.2 address identifier: As defined in FC-PH (see reference [1])**,** an unsigned 24-bit address value used to uniquely identify the source (S_ID) and destination (D_ID) of Fibre Channel frames.

3.1.3 Address Manager: A logical entity within a Switch which is responsible for address assignment.

3.1.4 Area: As defined in FC-FG (see reference [5]), the second level in a three-level addressing hierarchy.

3.1.5 Area Identifier: As defined in FC-FG (see reference [5]), bits 15 through 8 of an address identifier.

3.1.6 byte: A group of eight bits.

3.1.7 Class F service: As defined in FC-FG (see reference [5]), a service which multiplexes frames at frame boundaries that is used for control and coordination of the internal behavior of the Fabric.

3.1.8 Class N service: A generic reference to a Class 1, Class 2, or Class 3 service, as defined in FC-PH (see reference [1]).

3.1.9 Domain: As defined in FC-FG (see reference [5]), the highest level in a three-level addressing hierarchy.

3.1.10 Domain Address Manager: A Principal Switch which is responsible for address assignment to other Switches outside of its Domain.

3.1.11 Domain Identifier: As defined in FC-FG (see reference [5]), bits 23 through 16 of an address identifier.

3.1.12 Domain_ID_List: A list in which each record contains a Domain_ID value and the Switch_Name of the Switch assigned the Domain_ID (see 6.1.4).

3.1.13 downstream Principal ISL: From the point of view of the local Switch, the downstream Principal ISL is the Principal ISL to which frames may be sent from the Principal Switch to the destination Switch. All Principal ISLs on the Principal Switch are downstream Principal ISLs. A Switch that is not the Principal Switch may have zero or more downstream Principal ISLs.

3.1.14 E_Port: As defined in FC-FG (see reference [5]), a Fabric “Expansion” Port which attaches to another E_Port to create an Inter-Switch Link.

3.1.15 E_Port Identifier: An address identifier assigned to an E_Port.

3.1.16 E_Port_Name: A Name_Identifier which identifies an E_Port for identification purposes. The format of the name is specified in FC-PH. Each E_Port shall provide a unique E_Port_Name within the Fabric.

3.1.17 Error_Detect_Timeout value: A time constant defined in FC-PH. In this Standard, the recommended value of this time constant is 2 seconds.

3.1.18 F_Port: As defined in FC-PH (see reference [1]). In this Standard, an F_Port is assumed to always refer to a port to which non-loop N_Ports are attached to a Fabric, and does not include FL_Ports.

3.1.19 Fabric: As defined in FC-FG (see reference [5]), an entity which interconnects various Nx_Ports attached to it and is capable of routing frames using only the D_ID information in an FC-2 frame header.

3.1.20 Fabric Controller: 1. As defined in FC-FG (see reference [5]), the logical entity responsible for operation of the Fabric. 2. The entity at the well-known address hex ‘FF FF FD’.

3.1.21 Fabric Element: 1. As defined in FC-FG (see reference [5]), the smallest unit of a Fabric which meets the definition of a Fabric. From the point of view of an attached Nx_Port, a Fabric consisting of multiple Fabric Elements is indistinguishable from a Fabric consisting of a single Fabric Element.

3.1.22 Fabric F_Port: The entity at the well-known address hex ‘FF FF FE’. See reference [1].

3.1.23 FL_Port: An L_Port which is able to perform the function of an F_Port, attached via a link to one or more NL_Ports in an Arbitrated Loop topology (see FC-AL). The AL_PA of an FL_Port is hex’00’. In this Standard, an FL_Port is assumed to always refer to a port to which NL_Ports are attached to a Fabric, and does not include F_Ports.

3.1.24 Fx_Port: A Switch Port capable of operating as an F_Port or FL_Port.

3.1.25 Fabric_Stability_Timeout value: A time constant used to detect inactivity during Fabric Configuration. The value of this time constant shall be 5 seconds.

3.1.26 Interject: As defined in FC-FG.

3.1.27 Intermix: As defined in FC-FG.

3.1.28 Inter-Switch Link: A Link connecting the E_Port of one (local) Switch to the E_Port of another (remote) Switch.

3.1.29 Isolated: A condition in which it has been determined that no Class N traffic may be transmitted across an ISL.

3.1.30 L_Port: A port which contains Arbitrated Loop functions associated with the Arbitrated Loop topology.

3.1.31 Link: As defined in FC-PH.

3.1.32 local Switch: A Switch that can be reached without traversing any Inter-Switch Links.

3.1.33 Loop Fabric Address: An address identifier used to address a loop for purposes of loop management.

3.1.34 N_Port: As defined in FC-PH (see reference [1]). In this Standard, an N_Port is assumed to always refer to a direct Fabric-attached port, and does not include NL_Ports.

3.1.35 N_Port Identifier: An address identifier assigned to an N_Port.

3.1.36 Name_Identifier: As defined in FC-PH (see reference [1]), a 64-bit identifier.

3.1.37 NL_Port: An L_Port which is able to perform the function of an N_Port, attached via a link to one or more NL_Ports and zero or more FL_Ports in an Arbitrated Loop topology. In this Standard, an NL_Port is assumed to always refer to a loop-attached port, and does not include N_Ports.

3.1.38 non-zero Domain_ID_List: A Domain_ID_List which contains at least one record (see 7.2).

3.1.39 Nx_Port: A Port operating as an N_Port or NL_Port.

3.1.40 path: A route between a source and a destination.

3.1.41 path selection: A process whereby a path between a source and one or more destinations is discovered.

3.1.42 Port: 1. A generic reference to an N_Port, NL_Port, F_Port, FL_Port, or E_Port. 2. As defined in FC-FG (see reference [5]), the lowest level in a three-level addressing hierarchy.

3.1.43 Port Identifier: As defined in FC-FG (see reference [5]), bits 7 through 0 of an address identifier.

3.1.44 Port Mode: A generic reference to E_Port, F_Port or FL_Port operation.

3.1.45 Preferred Domain_ID: A Domain_ID previously granted to a Switch by the Domain Address Manager.

3.1.46 Principal ISL: An Inter-Switch Link that is used to communicate with the Principal Switch.

3.1.47 Principal Switch: A Switch which has been selected to perform certain duties.

3.1.48 remote Switch: A Switch that can be reached only by traversing one or more Inter-Switch Links.

3.1.49 Resource_Allocation_Timeout value: A time constant defined in FC-PH. In this Standard, the recommended value of this time constant is 10 seconds.

3.1.50 Router: An entity within a Switch responsible for routing of Class 2 and Class 3 frames.

3.1.51 routing: A process whereby the appropriate Switch Port(s) to deliver a Class 2 or Class 3 frame towards its destination is identified.

3.1.52 Switch: 1. A Fabric Element conforming to this Standard. 2. A member of the Fabric collective.

3.1.53 Switch Construct: An entity within a Switch responsible for transporting frames between Switch Ports.

3.1.54 Switch_Name: A Name_Identifier which identifies a Switch for identification purposes. The format of the name is specified in FC-PH. Each Switch shall provide a unique Switch_Name within the Fabric.

3.1.55 Switch Port: An E_Port, F_Port, or FL_Port.

3.1.56 Switch_Priority: A value used during Principal Switch selection to cause one Switch to be favored over another.

3.1.57 upstream Principal ISL: From the point of view of the local Switch, the upstream Principal ISL is the Principal ISL to which frames may be sent from the local Switch to the Principal Switch. A Switch that is not the Principal Switch always has exactly one upstream Principal ISL. The Principal Switch does not have an upstream Principal ISL.

3.1.58 zero Domain_ID_List: A Domain_ID_List which is empty (see 7.2).

3.2 Editorial conventions

In this Standard, a number of conditions, mechanisms, sequences, parameters, events, states, or similar terms that do not have their normal English meaning are printed with the following conventions:

- the first letter of each word in uppercase and the rest lowercase (e.g., Exchange, Class, etc.).
- a term consisting of multiple words, with the first letter of each word in uppercase and the rest lowercase, and each word separated from the other by an underscore (_) character. A word may consist of an acronym or abbreviation which would be printed in uppercase. (e.g., NL_Port, Transfer_Length, etc.).
- a term consisting of multiple words with all letters lowercase and each word separated from the other by a dash (-) character. A word may also consist of an acronym or abbreviation which would be printed in uppercase. (e.g., device-level, CUE-with-busy, etc.).

All terms and words not conforming to the conventions noted above have the normal technical English meanings.

Numbered items in this Standard do not represent any priority. Any priority is explicitly indicated.

In all of the figures, tables, and text of this Standard, the most significant bit of a binary quantity is shown on the left side. Exceptions to this convention are indicated in the appropriate sections.

The term “shall” is used to indicate a mandatory rule. If such a rule is not followed, the results are unpredictable unless indicated otherwise.

The fields or control bits which are not applicable shall be reset to zero.

If a field or a control bit in a frame is specified as not meaningful, the entity which receives the frame shall not check that field or control bit.

If a field or control bit is specified as reserved, it shall be filled with binary zeros by the source, and shall be ignored by the destination.

Temporary: Anything in “{ }” is an editor’s note indicating some unresolved issue.

3.2.1 Binary notation

Binary notation may be used to represent some fields. Single bit fields are represented using the binary values 0 and 1. For multiple bit fields, the binary value is enclosed in single quotation marks followed by the letter b. For example, a four-byte Process_Associator field containing a binary value may be represented as ‘00000000 11111111 10011000 11111010’b.

3.2.2 Hexadecimal notation

Hexadecimal notation may be used to represent some fields. When this is done, the value is enclosed in single quotation marks and preceded by the word hex. For example, a four-byte Process_ Associator field containing a binary value of ‘00000000 11111111 10011000 11111010’b is shown in hexadecimal format as hex’00 FF 98 FA’.

3.3 Abbreviations, acronyms, and symbols

Abbreviations and acronyms applicable to this International Standard are listed. Definitions of several of these items are included in 3.1. Abbreviations used that are not listed below are defined in FC-PH (see reference [1]).

3.3.1 Acronyms and abbreviations

Area_ID	Area Identifier
BLS	Basic Link Service
Domain_ID	Domain Identifier
E_D_TOV	Error_Detect_Timeout value
ELS	Extended Link Service
FAN	Fabric Address Notification Extended Link Service, see reference [10]
FC-AL	Fibre Channel Arbitrated Loop, reference [3]
FC-AL-2	Fibre Channel Arbitrated Loop-2, reference [7]
FC-BB	Fibre Channel Backbone, reference [8]
FC-FG	Fibre Channel - Fabric Generic, reference [5]
FC-FLA	Fibre Channel - Fabric Loop Attachment, reference [10]
FC-GS	Fibre Channel - Generic Services, reference [4]
FC-GS-2	Fibre Channel - Generic Services-2, reference [9]
FC-PH	Fibre Channel Physical and Signaling Interface, reference [1]
FC-PH-2	Fibre Channel Physical and Signaling Interface-2, reference [2]
FC-PH-3	Fibre Channel Physical and Signaling Interface-3, reference [6]
F_S_TOV	Fabric_Stability_Timeout value
ISL	Inter-Switch Link
IU	Information Unit
LAN	Local Area Network
LFA	Loop Fabric Address
Port_ID	Port Identifier
R	Reserved
R_A_TOV	Resource_Allocation_Timeout value
SI	Sequence Initiative
SW_ACC	Switch Fabric Link Service Accept
SW_LS	Switch Fabric Link Service
SW_RJT	Switch Fabric Link Service Reject
ULP	Upper Level Protocol
WKA	Well-Known Address
WWN	World Wide Name

3.3.2 Symbols

Unless indicated otherwise, the following symbols have the listed meaning.

||	concatenation

4 Structure and Concepts

This clause provides an overview of a Switch-based Fabric.

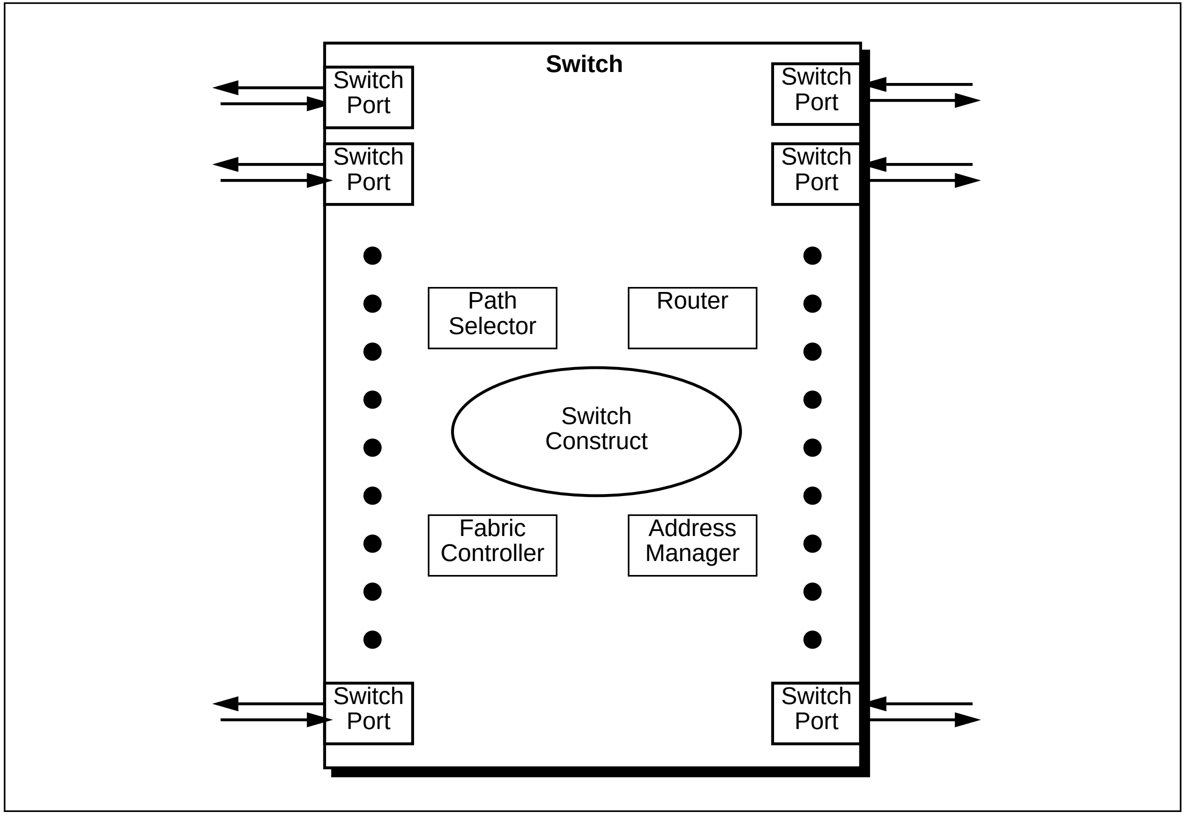
4.1 Fabric

A Fabric is a transport that provides switched interconnect between Nx_Ports. The general model of a Fibre Channel Fabric is defined in FC-FG, reference [5].

4.2 Switch

A Switch is the smallest entity that can function as a Switch-based Fibre Channel Fabric. Figure 1 illustrates the conceptual model of a Switch.

Figure 1 – Switch Model



A Switch is composed of the following major components:

- One or more Switch Ports;
- a Switch Construct, capable of either multiplexed frame switching or circuit switching, or both;
- an Address Manager;
- a Path Selector, which performs path selection;
- a Router;

- and a Fabric Controller.

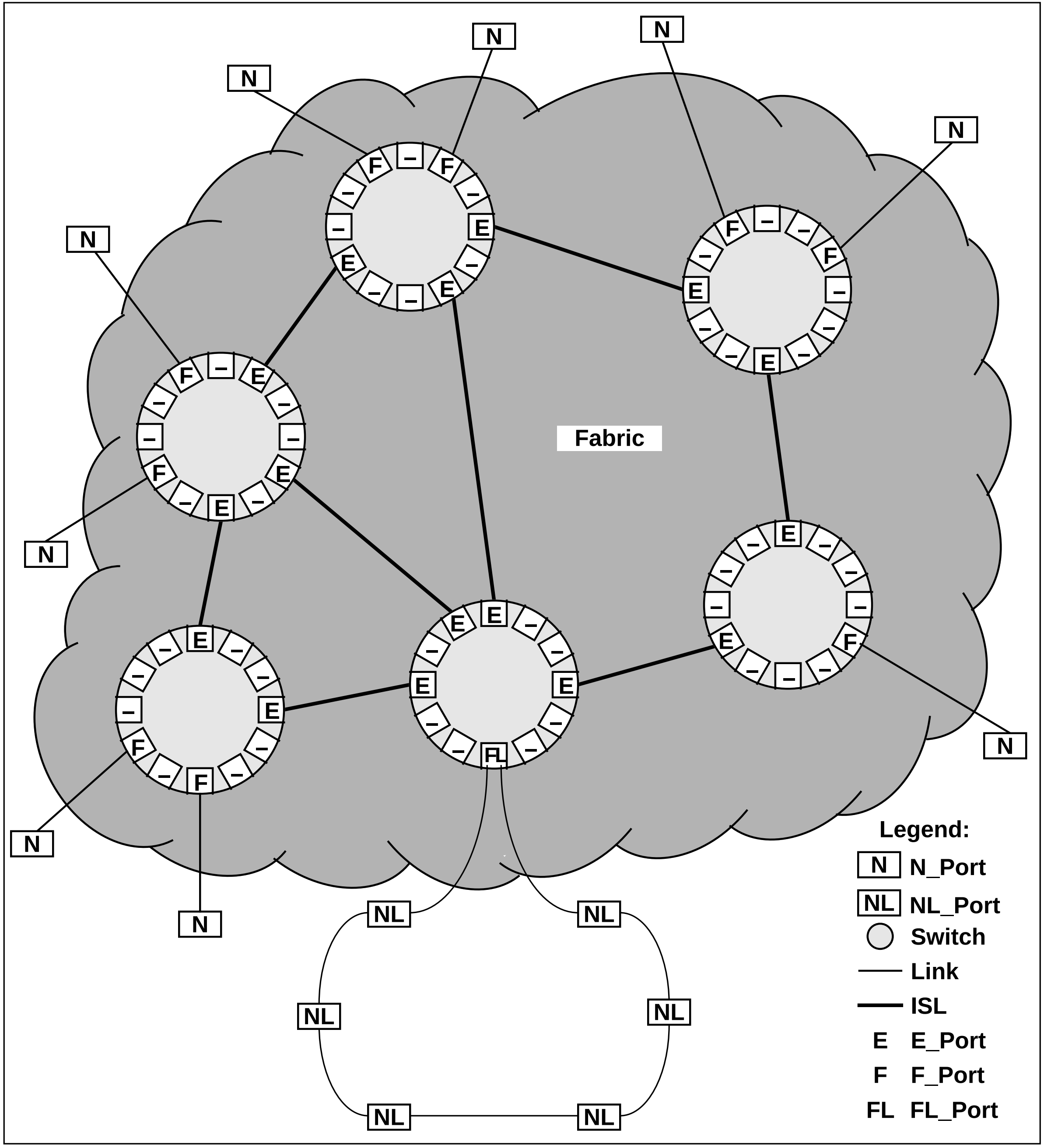
As defined, a Switch Port may be either an E_Port, an F_Port, or an FL_Port. A Switch Port that is capable of assuming more than one of these roles is called a multi-function Switch Port. Once a Switch Port assumes a role, via the Switch Port Initialization Procedure, it shall remain in that role until an event occurs that causes re-initialization.

The Link joining a pair of E_Ports is called an Inter-Switch Link (ISL). E_Ports conforming to this Standard use FC–PH compliant media, coding and data rates to form an ISL.

ISLs carry frames originating from the Node Ports and those generated within the Fabric. The frames generated within the Fabric serve as control, management and support for the Fabric.

Switches may be joined freely or in a structured fashion to form a larger Fabric, as illustrated in Figure 2.

Figure 2 – Multiple Switch Fabric Example



The structure of the Switch Construct in the Switch, as seen in figure 1, is undefined and beyond the scope of this Standard. It may support either or both circuit switching and multiplexed frame switching. It may be non-blocking, allowing concurrent operation of all possible combinations or it may be blocking, restricting operations. The Switch Construct may also contain redundancy, as may be required for high availability configurations.

The Address Manager is responsible for the assignment of addresses within some portion of the Fabric. Within the Switch, the Address Manager is responsible for acquiring a Domain and Area for the Switch, and allocating Port_IDs within the Domain and Area.

The Path Selector is a logical entity that establishes frame routing paths.

The Router is a logical entity that performs the routing of Class 2 and Class 3 frames to their final destination.

The Fabric Controller is a logical entity that performs the management of the Switch. The Fabric Controller has the characteristics of an N_Port, though it may or may not be attached to the Fabric via a Link.

4.3 Switch Topologies

Switch topologies are defined in FC-FG, reference [5].

4.4 Switching characteristics

Path, circuit, switching and frame routing within a Switch may occur synchronously or asynchronously to the current word alignment of the outbound fibre.

Synchronous switching guarantees retention of the established word alignment on the outbound fibre of the Switch Port. Asynchronous switching does not guarantee retention of word alignment on the outbound fibre of the Switch Port.

A Switch may employ either synchronous or asynchronous switching or a combination of the two (e.g., a Switch may use synchronous switching for Class F, Class 2 and Class 3, and asynchronous switching for Class 1). However, a Switch shall never mix the two within a given Class of Service.

A switching event occurs every time a connectionless frame is transmitted and when a connection based service is established, suspended or terminated. Frame Intermixing and Interjecting also constitute switching events.

4.4.1 Synchronous switching

Synchronous switching associated with connectionless frame routing and connection oriented Dedicated Connections or virtual connection Services shall guarantee the word alignment on the outbound fibre.

Switches shall ensure that synchronous switching only occurs between frames. Switches should use synchronous switching in support of Class 2, Class 3 and Class F service.

4.4.2 Asynchronous switching

Asynchronous switching may be performed any time Fill Words are being transmitted. Bit alignment and word alignment may be lost when an asynchronous switching event occurs. A recovery time that allows the attached Port time to regain synchronization shall be inserted before frame transmission resumes for the outbound fibre. Fill Words shall be transmitted during this recovery time. If conditions arise warranting transmission of a Primitive Sequence, then this should take precedence over transmission of Fill Words.

If a Switch or Node Port recognizes that it is linked to a Switch which employ asynchronous switching, and a permissible word realignment event occurs, then the Port may discount any resulting errors, i.e. not log errors resulting from the realignment event.

4.5 Switch Ports

A Switch shall have three or more Switch Ports. A Switch equipped only with F_Ports or FL_Ports forms a non-expandable Fabric. To be part of an expandable Fabric, a Switch shall incorporate at least one Switch Port capable of E_Port operation.

A Switch Port supports one or more of the following Port Modes: E_Port, F_Port, FL_Port. A Switch Port that is capable of supporting more than one Port Mode attempts to configure itself first as an FL_Port (as defined in FC-AL), then as an E_Port (as defined in this Standard), and finally as an F_Port (as defined in FC-PH), depending on which of the three Port Modes are supported by the Switch Port.

The detailed procedure is described in 7.1.

4.5.1 F_Port

An F_Port is the point at which all frames originated by an N_Port enter the Fabric, and all frames destined for an N_Port exit the Fabric. An F_Port may also be the Fabric entry point for frames originated by an N_Port destined for an internal Fabric destination, such as the Fabric Controller. Similarly, an F_Port may also be the Fabric exit point for frames originated internal to the Fabric and destined for an N_Port. Frames shall not be communicated across a Link between an F_Port and anything other than an N_Port.

F_Ports are described in detail in 5.2.

4.5.2 FL_Port

An FL_Port is the point at which all frames originated by an NL_Port enter the Fabric, and all frames destined for an NL_Port exit the Fabric. An FL_Port may also be the Fabric entry point for frames originated by an NL_Port destined for an internal Fabric destination, such as the Fabric Controller. Similarly, an FL_Port may also be the Fabric exit point for frames originated internal to the Fabric and destined for an NL_Port. Frames shall not be communicated across a Link between an FL_Port and anything other than an NL_Port.

FL_Ports are described in detail in 5.3.

4.5.3 E_Port

An E_Port is the point at which frames pass between the Switches within the Fabric. Frames with a destination other than the local Switch or any N_Port or NL_Port attached to the local Switch exit the local Switch through an E_Port. Frames that enter a Switch via an E_Port are forwarded to a local destination, or are forwarded towards their ultimate destination via another E_Port. Frames shall not be communicated across a Link between an E_Port and anything other than an E_Port.

E_Ports are described in detail in 5.4.

4.6 Fabric Addressing

Switches use the address partitioning model described in FC-FG (Annex A), and as described below. The 24-bit address identifier is divided into three fields: Domain, Area, and Port, as shown in figure 3.

Figure 3 – Domain, Area, and Port Address Partitioning

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Domain_ID								Area_ID								Port_ID							
Address Identifier																							

A Domain is one or more Switches that have the same Domain_ID for all N_Ports and NL_Ports within or attached to those Switches, except for Well-Known Addresses. If there is more than one Switch in the Domain, any Switch within the Domain shall be directly connected via an ISL to at least one other Switch in the same Domain.

An Area_ID shall apply to either of the following:

– One or more N_Ports or E_Ports within and attached to a single Switch, except for Well-Known Addresses; or,

– an Arbitrated Loop of NL_Ports attached to a single FL_Port.

A single Arbitrated Loop shall have exactly one Area_ID.

A Port_ID shall apply to either of the following:

– a single N_Port or E_Port within a Domain/Area, except for Well-Known Addresses; or,

– the valid AL_PA of a single NL_Port or FL_Port on an Arbitrated Loop.

Address identifier values for this Standard are listed in table 1. Any value listed as Reserved is not meaningful within this Standard. Note also that some values defined below are different from the address allocations defined in Informative Annex A of FC-FG; these differences are noted in the table.

Table 1 – Address Identifier Values

Address Identifier (hex)			
Domain_ID	**Area_ID**	**Port_ID**	**Description**
00	00	00	Undefined (note 1)
00	00	AL_PA	E_Port: Reserved F_Port: Reserved FL_Port: Private Loop NL_Port (note 2)
00	00	non-AL_PA	Reserved
00	01 - FF	00 - FF	Reserved
01 - EF	00 - FF	00	E_Port: E_Port Identifier (note 4) F_Port: N_Port Identifier (note 4) FL_Port: Loop Fabric Address (note 3)
01 - EF	00 - FF	AL_PA	E_Port: E_Port Identifier (note 4) F_Port: N_Port Identifier (note 4) FL_Port: N_Port Identifier for Public Loop NL_Port (note 3)

Table 1 – Address Identifier Values

Address Identifier (hex)			
Domain_ID	Area_ID	Port_ID	Description
01 - EF	00 - FF	non-AL_PA	E_Port: E_Port Identifier (note 4) F_Port: N_Port Identifier (note 4) FL_Port: Reserved
F0 - FE	00 - FF	00 - FF	Reserved
FF	00 - FA	00 - FF	Reserved
FF	FB	00 - FF	Reserved for Multicast Group_ID (note 5)
FF	FC	00	Reserved (note 5)
FF	FC	01 - EF	N_Port Identifier for Domain Controller (note 6)
FF	FC	F0 - FF	Reserved (note 5)
FF	FD - FE	00 - FF	Reserved (note 5)
FF	FF	00 - EF	Reserved (note 5)
FF	FF	F0 - FC	Well-Known Address (note 7)
FF	FF	FD	N_Port Identifier for Fabric Controller (note 8)
FF	FF	FE	N_Port Identifier for Fabric F_Port
FF	FF	FF	Well-Known Address (note 7)

Notes:

1 This value is used by an N_Port requesting an address identifier during FLOGI.

2 See FC-AL for a definition of AL_PA and FC-FLA for a definition of Private Loop and FL_Port operation with Private Loop devices. In FC-FG, this range was reserved for other purposes.

3 See FC-FLA for the definition and use of Loop Fabric Address, and for a definition of Public Loop.

4 In FC-FG, the Area_ID range F0-FF was reserved for “Fabric Assisted Functions”, which were not defined in FC-FG.

5 In FC-FG, this range was reserved for other purposes.

6 A Domain Controller identifier may be used to address the Fabric Controller of a remote Switch that is not directly connected via an ISL to the originating Switch. The Port_ID field is set to the Domain_ID of the remote Switch.

7 The usage of Well-Known Addresses hex’FFFFF0’ through hex’FFFFFC’, and hex’FFFFFF’, are not defined by this Standard. FC-PH defines or reserves these values for Well-Known Addresses.

8 This address identifier has special usage depending on the originator. If the originator is an attached external N_Port or NL_Port (attached via an F_Port or FL_Port) then the destination of a frame sent to hex’FFFFFD’ is the Fabric Controller of the local Switch. If the originator is the Fabric Controller of the local Switch, then the destination of a frame sent to hex’FFFFFD’ via an ISL is the Fabric Controller of the remote Switch at the other end of the ISL.

4.7 Class F Service

Class F service is a connectionless service very similar to Class 2 that is used for internal control of the Fabric. Class F service as defined by this Standard differs in some ways from the definition in FC-FG. Class F service as used by this Standard is defined in 5.5.

4.8 Relationship Between this Standard and FC-FG

FC-FG defined the generic requirements for all Fabrics, independent of the specific type or topology. Many issues were appropriately left open for definition by later Fabric Standards specific to certain types and topologies.

In the process of defining the Switch Fabric, some items that were defined in FC-FG were found that required modification for use in this Standard.

In cases where this Standard and FC-FG conflict, this Standard shall take precedence.

5 Switch Ports

This clause defines the specific behaviors of all modes of Switch Port. Note that the models described below are defined for purposes of describing behavior. No implication is made as to whether the actual implementation of an element is in hardware or software. An element may be implemented on a per-Port basis, or may be a logical entity that is embodied in a single physical implementation shared by multiple ports.

A Switch Port may be able to operate in more than one mode, and configure itself to the appropriate mode during the initialization process (see 7.1). During initialization, the Switch Port can assume a mode for purposes of determining if that mode is appropriate. For example, a Switch Port operates in FL_Port mode to determine if it is attached to a loop of NL_Ports. If that is not successful, it then tries operating as an E_Port to see if another E_Port is attached. The Switch Port continues until it finds a mode in which to operate.

5.1 Model elements

Each Switch Port model described in this clause is made up of a set of elements. These elements are briefly defined below.

5.1.1 FC Transports

The FC-PH Transport includes all of the functionality described in FC-PH to construct and deconstruct a frame, to encode and decode the words that make up the frame, and to transmit and receive the frame on the physical media. The FC-AL Transport contains additional functionality to support the Arbitrated Loop protocols.

5.1.2 Switch Transport

The Switch Transport is an abstraction to show the “back end” of the Switch Port as it interacts with the Switch Construct and/or other Switch Ports within the Switch. The Switch Transport exists to move frames between the Switch Port and the rest of the Switch. No other implementation details are implied by this element.

5.1.3 Control Facilities

The Control Facilities are internal logical ports that receive and perform requests, and generate responses. Each Control Facility has associated with it an address identifier, and support for Classes of Service. The Control Facilities also manage the various Transport elements.

5.1.4 Link Services

The Link Services represent the various Link Services that are supported by the corresponding Control Facility.

5.2 F_Port Operation

An F_Port is the point at which an external N_Port is attached to the Fabric. It normally functions as a conduit to the Fabric for frames transmitted by the N_Port, and as a conduit from the Fabric for frames destined for the N_Port.

An F_Port shall support one or more of the following Classes of service: Class 1 service, Class 2 service, Class 3 service. An F_Port shall not intentionally transmit Class F frames on its outbound fibre.

NOTE – When a Class 1 Connection is ended, a Class F frame may be inadvertently transmitted by an F_Port. An N_Port that receives a Class F frame discards it, as required by FC-PH.

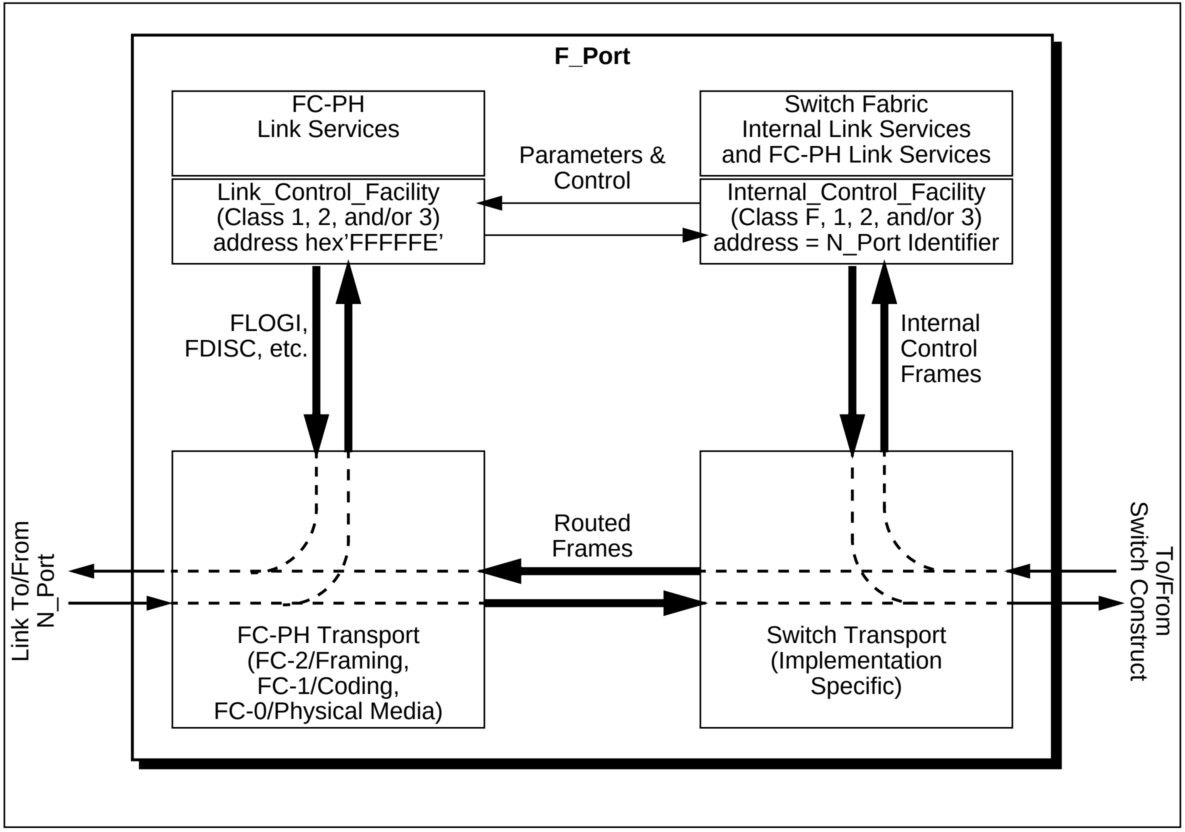
An F_Port shall not admit to the Fabric any Class F frames, any Primitive Sequences, or any Primitive Signals other than Idle, that the F_Port receives on its inbound fibre.

NOTE – Primitive Signals and Primitive Sequences are prohibited from entering the Fabric by FC-PH. For example, if an R_RDY was admitted to a Fabric, it could presumably propagate to another F_Port and be transmitted by that F_Port, disrupting credit on that Link.

5.2.1 Model

The model of an F_Port is shown in figure 4.

Figure 4 – F_Port Model



An F_Port contains an FC-PH Transport element through which passes all frames and Primitives transferred across the Link to and from the N_Port. Frames received from the N_Port are either directed to the Switch Construct via the Switch Transport element, or directed to the Link_Control_Facility.The Link_Control_Facility receives frames related to Link Services such as FLOGI, and transmits responses to those Link Service frames.

Frames received from the FC-PH Transport element that are destined for other ports are directed by the Switch Transport to the Switch Construct for further routing. Frames received from the Switch Construct by the Switch Transport are directed either to the FC-PH Transport for transmission to the

N_Port, or to the Internal_Control_Facility. The Internal_Control_Facility receives frames related to Switch Fabric Internal Link Services, and transmits responses to those Internal Link Services frames. Information is passed between the Internal_Control_Facility and the Link_Control_Facility to effect the control and configuration of the Transport elements.

5.2.2 Link Behavior

The F_Port Link is used by Switches to transmit and receive frames with a single Node. A Link to an F_Port always connects to exactly one N_Port.

An F_Port Link follows the FC-0, FC-1, and FC-2 protocols defined for point-to-point Links as defined in FC-PH.

5.3 FL_Port Operation

An FL_Port is the point at which one or more external NL_Ports are attached to the Fabric. It normally functions as a conduit to the Fabric for frames transmitted by the attached NL_Ports, and as a conduit from the Fabric for frames destined for the attached NL_Ports.

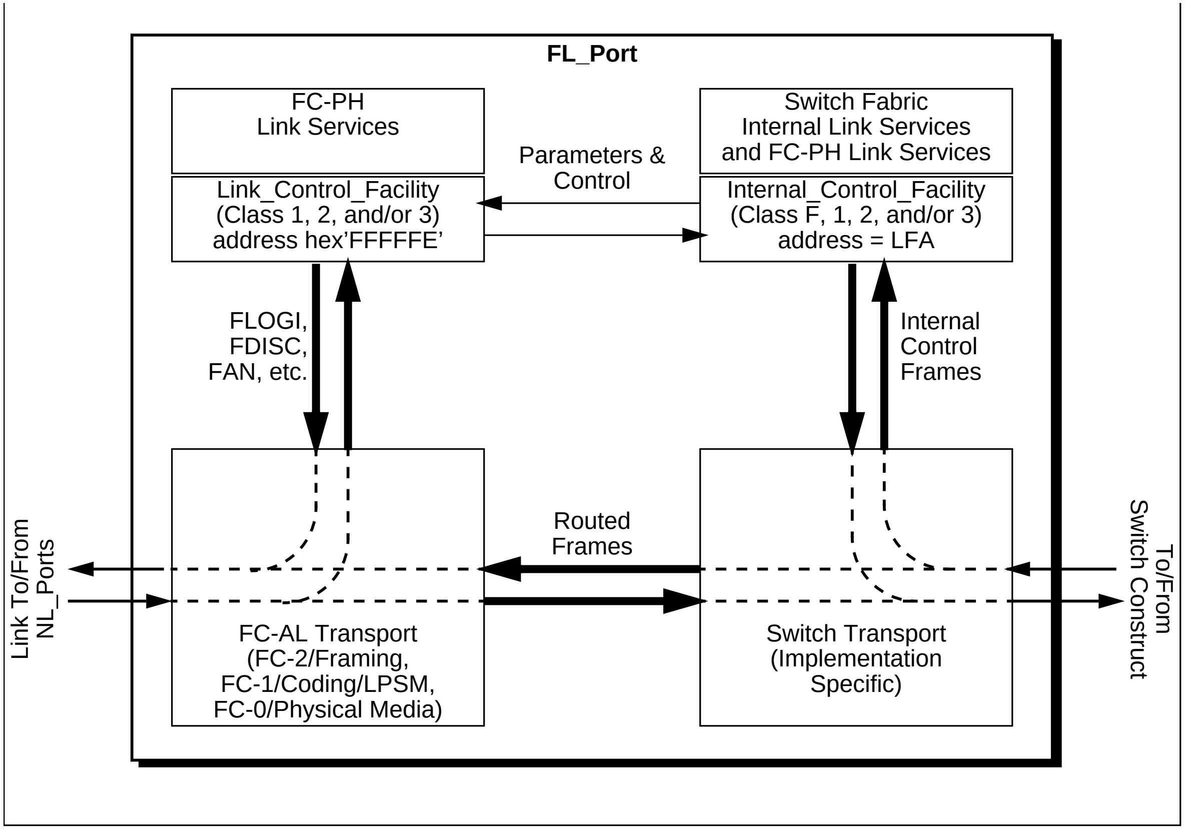
An FL_Port shall support one or more of the following Classes of service: Class 1 service, Class 2 service, Class 3 service. An FL_Port shall not intentionally transmit Class F frames on its outbound fibre. An FL_Port shall not admit to the Fabric any Class F frames, any Primitive Sequences, or any Primitive Signals other than Idle, that the FL_Port receives on its inbound fibre.

An FL_Port that conforms to this Standard shall conform to the FL_Port requirements defined in FC-FLA (reference [10]).

5.3.1 Model

The model of an FL_Port is shown in figure 5.

Figure 5 – FL_Port Model



An FL_Port contains an FC-AL Transport element through which passes all frames and Primitives transferred across the Link to and from the multiple NL_Ports. Frames received from the NL_Ports are either directed to the Switch Construct via the Switch Transport element, or directed to the Link_Control_Facility.The Link_Control_Facility receives frames related to Link Services such as FLOGI, and transmits responses to those Link Service frames. The Link_Control_Facility also transmits and receives Loop Initialization Sequences and transmits the FAN ELS.

Frames received from the FC-AL Transport element that are destined for other ports are directed by the Switch Transport to the Switch Construct for further routing. Frames received from the Switch Construct by the Switch Transport are directed either to the FC-AL Transport for transmission to the destination NL_Port, or to the Internal_Control_Facility. The Internal_Control_Facility receives frames related to Switch Fabric Internal Link Services and Loop management Extended Link Services (see FC-FLA), and transmits responses to those Link Services frames. Information is passed between the Internal_Control_Facility and the Link_Control_Facility to effect the control and configuration of the Transport elements.

5.3.2 Link Behavior

The FL_Port Link is used by Switches to transmit and receive frames with multiple Nodes. A Link to an FL_Port connects to one or more NL_Ports.

An FL_Port Link follows the FC-0, FC-1, and FC-2 protocols defined in FC-PH, with the additional Arbitrated Loop protocols defined in FC-AL.

5.4 E_Port Operation

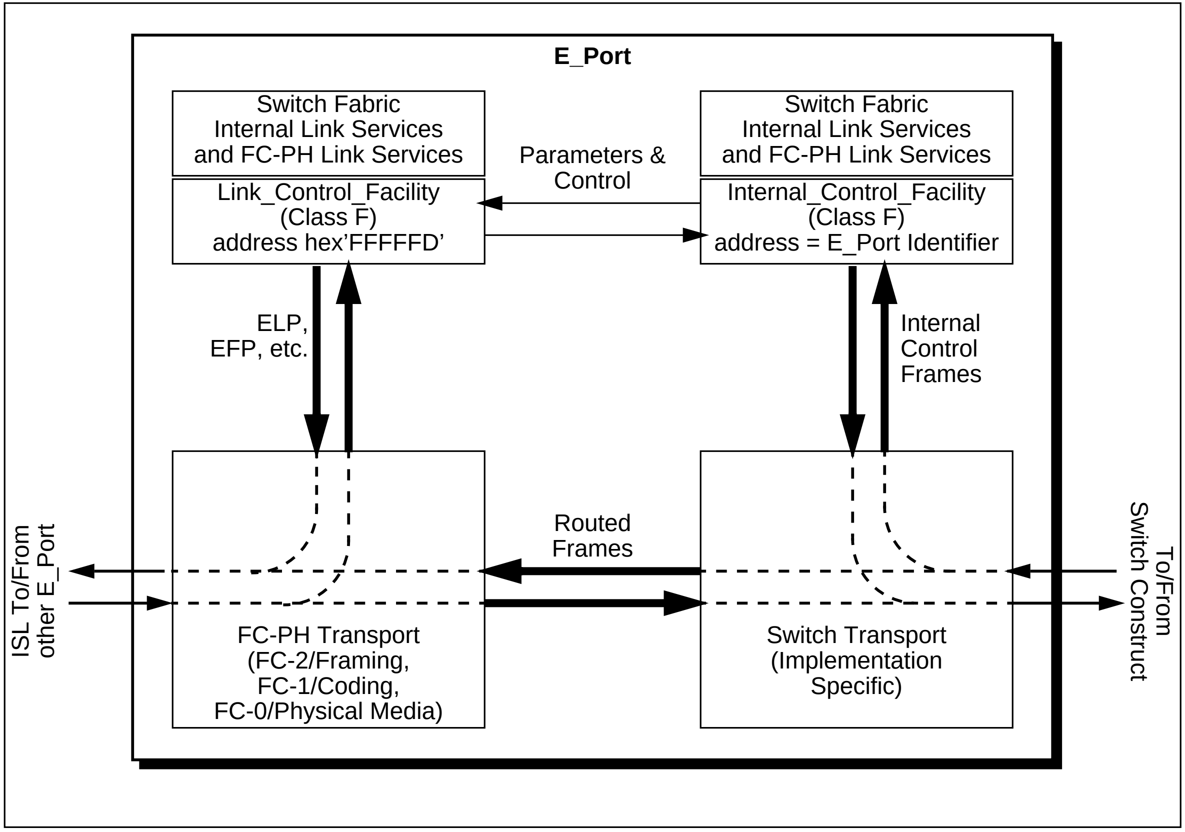
An E_Port is the point at which a Switch is connected to another Switch to create a Fabric. It normally functions as a conduit between the Switches for frames destined for remote N_Ports and NL_Ports. An E_Port is also used to carry frames between Switches for purposes of configuring and maintaining the Fabric.

An E_Port shall support the Class F service. An E_Port shall also be capable of routing one or more of the following Classes of service: Class 1 service, Class 2 service, Class 3 service. An E_Port shall not admit to the Fabric any Primitive Sequences, or any Primitive Signals other than Idle, that the E_Port receives on its inbound fibre.

5.4.1 Model

The model of an E_Port is shown in figure 4.

Figure 6 – E_Port Model


E_Port
Switch Fabric Internal Link Services and FC-PH Link Services
Link_Control_Facility (Class F) address hex'FFFFFD'
Parameters & Control
Switch Fabric Internal Link Services and FC-PH Link Services
Internal_Control_Facility (Class F) address = E_Port Identifier
ELP, EFP, etc.
Internal Control Frames
ISL To/From other E_Port
Routed Frames
To/From Switch Construct
FC-PH Transport (FC-2/Framing, FC-1/Coding, FC-0/Physical Media)
Switch Transport (Implementation Specific)

An E_Port contains an FC-PH Transport element through which passes all frames and Primitives transferred across the Link to and from the other E_Port. Frames received from the other E_Port are either directed to the Switch Construct via the Switch Transport element, or directed to the Link_Control_Facility. The Link_Control_Facility receives frames related to Switch Fabric Internal Link Services such as ELP, and transmits responses to those Link Service frames.

Frames received from the FC-PH Transport element that are destined for other ports are directed by the Switch Transport to the Switch Construct for further routing. Frames received from the Switch Construct by the Switch Transport are directed either to the FC-PH Transport for transmission to the other E_Port, or to the Internal_Control_Facility. The Internal_Control_Facility receives frames related to

Switch Fabric Internal Link Services, and transmits responses to those Internal Link Services frames. Information is passed between the Internal_Control_Facility and the Link_Control_Facility to effect the control and configuration of the Transport elements.

5.4.2 Inter-Switch Link Behavior

Inter-Switch Links (ISLs) are used by Switches to transmit and receive frames with other Switches. An ISL always connects exactly one E_Port on a Switch to exactly one E_Port on another Switch.

An ISL follows the FC-0, FC-1, and FC-2 protocols defined for point-to-point Links as defined in FC-PH, with the exception that Class F frames are allowed to transit the Link, as defined in FC-FG. The use of R_RDY shall be restricted to the management of buffer-to-buffer flow control of Class F frames on the ISL prior to the completion of the exchange of Link parameters (see 6.1.3 and 7.1); an alternate method of buffer-to-buffer flow control may be defined in that process. Flow control of Class N frames shall be managed by other means not defined in this Standard.

NOTE – It is expected that the various flow control models will be defined by a future standard.

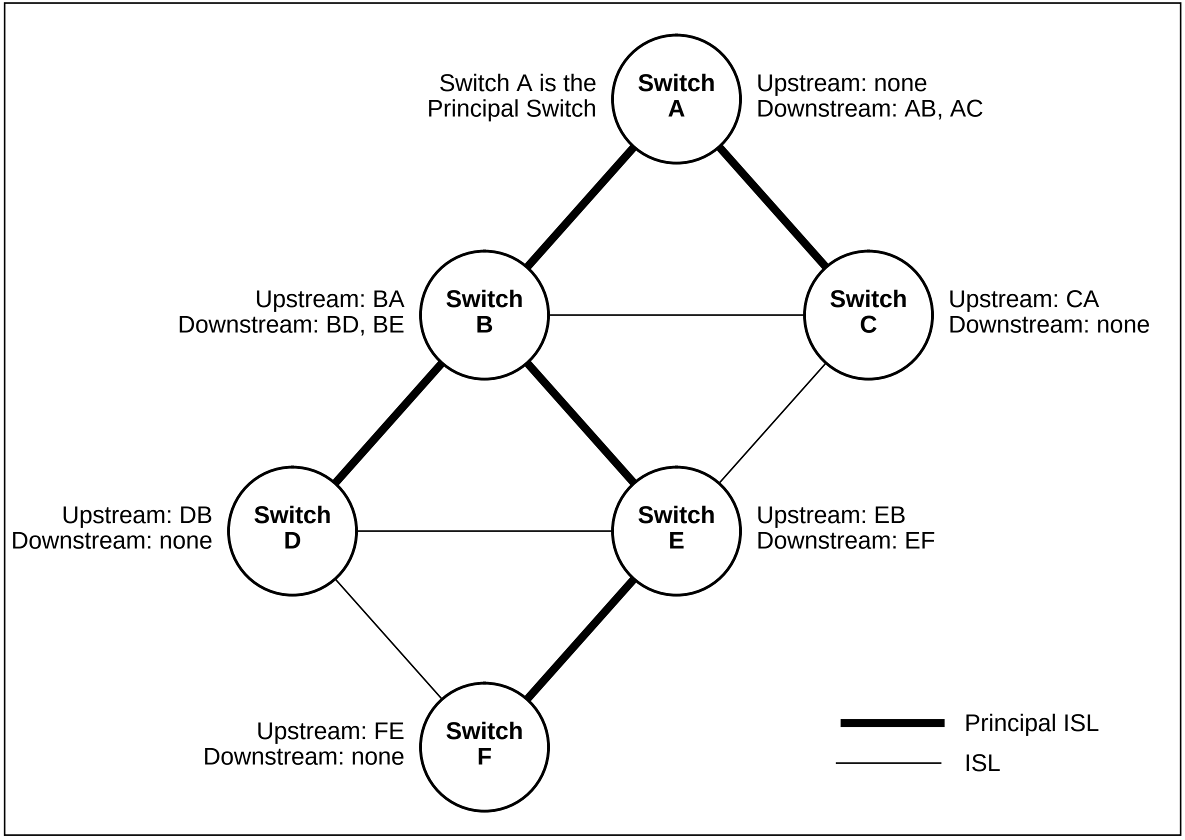
For purposes of defining and maintaining the Fabric Configuration, an ISL may be designated as a Principal ISL. The Principal ISL is a path that is used during configuration and address assignment to route Class F configuration frames, and is therefore a known path between two Switches. If a Principal ISL is lost, there may be no other available paths between the two affected Switches, so as a result the Fabric Configuration is possibly broken and must be rebuilt (by issuing the BF SW_ILS, see 6.1.7). If a non-Principal ISL is lost, at least one other path is known to be available between the Switches (i.e., the Principal ISL), therefore the lost ISL can be resolved via a routing change.

A Switch discovers the Principal ISL(s) during the process of Principal Switch Selection (see 7.2) and Address Distribution (see 7.3). During this process, the Switch identifies two kinds of Principal ISL. The Principal ISL that leads towards the Principal Switch is called the upstream Principal ISL. All frames from the Switch to the Principal Switch are sent via the upstream Principal ISL. The Principal Switch has no upstream Principal ISL; all other Switches have exactly one upstream Principal ISL.

A Principal ISL that leads away from the Principal Switch is called the downstream Principal ISL. Any frame sent by the Switch to another Switch as a result of a frame received on the upstream Principal ISL is sent via the downstream Principal ISL that leads towards that Switch. The Principal Switch may have one or more downstream Principal ISLs; all other Switches may have zero or more downstream Principal ISLs.

Principal ISLs are further illustrated in figure 7.

Figure 7 – Principal Inter-Switch Links



5.5 Class F Service

Class F Service is a connectionless service with notification of non-delivery between E_Ports, used for control, coordination, and configuration of the Fabric. Class F Service is defined by this Standard for use by Switches communicating across Inter-Switch Links. This definition of Class F for Inter-Switch Links supersedes the definitions of Class F for Inter-Element Links in FC-FG.

5.5.1 Class F Function

A Class F Service is requested by an E_Port on a frame by frame basis. The Fabric routes the frame to the destination E_Port. If the E_Port transmits consecutive frames to multiple destinations, the Fabric demultiplexes them to the requested destinations. Class F delimiters are used to indicate the requested service and to initiate and terminate one or more Sequences as described in FC-PH.

5.5.2 Class F Rules

To provide Class F Service, the transmitting and receiving E_Ports and the Fabric shall obey the following rules:

a) Except for some Switch Fabric Internal Link Service protocols, an E_Port is required to have exchanged Link parameters (see 6.1.3 and 7.1) with the associated destination with which it intends to communicate **(Login)**.

b) The Fabric routes the frames without establishing a Dedicated Connection between communicating E_Ports. To obtain Class F service, the E_Port shall use Class F delimiters as defined in 5.5.3. **(Connectionless service)**

c) An E_Port is allowed to send consecutive frames to one or more destinations. This enables an E_Port to demultiplex multiple Sequences to a single or multiple destinations concurrently. **(demultiplexing)**

d) A given E_Port may receive consecutive frames from different sources. Each source is allowed to send consecutive frames for one or more Sequences. **(multiplexing)**

e) A destination E_Port shall provide an acknowledgment to the source for each valid Data frame received. The destination E_Port shall use ACK_1 for the acknowledgment. If a Switch is unable to deliver the ACK_1 frame, the Switch shall return an F_BSY or F_RJT. **(Acknowledgment)**

f) The Sequence Initiator shall increment the SEQ_CNT field of each successive frame transmitted within a Sequence. However, the Switches may not guarantee delivery to the destination in the same order of transmission. **(non-sequential delivery)**

g) An E_Port may originate multiple Exchanges and initiate multiple Sequences with one or more E_Ports. The E_Port originating an Exchange shall assign an X_ID unique to the Originator called OX_ ID and the Responder of the Exchange shall assign an X_ID unique to the responder called RX_ID. The value of OX_ ID or RX_ID is unique to a given E_Port. The Sequence Initiator shall assign a SEQ_ID, for each Sequence it initiates, which is unique to the Sequence Initiator and the respective Sequence Recipient pair while the Sequence is Open. **(concurrent Exchanges and Sequences)**

h) Each E_Port exercises buffer-to-buffer flow control with the E_Port to which it is directly attached. End-to-end flow control is performed by communicating E_Ports. ACK_1 frames are used to perform end-to-end flow control and R_RDY is used for buffer-to-buffer flow control. **(dual flow control)**

i) If a Switch is unable to deliver the frame to the destination E_Port, then the source is notified of each frame not delivered by an F_BSY or F_RJT frame with corresponding D_ ID, S_ID, OX_ID, RX_ID, SEQ_ID, and SEQ_CNT from the Switch. The source is also notified of valid frames busied or rejected by the destination E_Port by P_BSY or P_RJT. **(non-delivery)**

j) A busy or reject may be issued by an intermediate E_Port or the destination E_Port with a valid reason code. **(busy/reject)**

k) If a Class F Data frame is busied, the sender shall retransmit the busied frame up to the ability of the sender to retry, including zero. **(retransmit)**

l) The Credit established during the ELP protocol by interchanging Link Parameters shall be honored. Class F shall not share Credit with any other Class of service. **(Credit)**

m) Effective transfer rate between any given E_Port pair is dependent upon the number of E_Ports a given E_Port is demultiplexing to and multiplexing from. **(bandwidth)**

n) Frames within a Sequence are tracked on a Sequence_Qualifier and SEQ_CNT basis. **(tracking)**

o) An E_Port shall be able to recognize SOF delimiters for Class F, Class 1, Class 2, and Class 3 service, whether or not all Classes of service are supported by the Port. An E_Port shall accept frames for all FC-PH service Classes. **(invalid Class)**

p) An E_Port receiving a Vendor Unique Class F frame may discard the frame without notification. A Vendor Unique Class F frame is indicated by an R_CTL field value of hex'F0'. **(vendor unique)**

q) An E_Port shall support insertion of Class F frames onto an established Class 1 Dedicated Connection. However, this insertion shall not cause loss of any Class 1 frames. A Switch may abort (**EOFa**) or discard an Intermixed Class 2 or Class 3 frame in progress if its transmission of a Class F frame interferes. A Switch shall not abort an Inserted Class F frame. **(Class F intermix)**

r) An E_Port shall use R_RDY and FC-PH buffer-to-buffer flow control with the E_Port to which it is directly attached, until after the exchange of Link parameters (see 6.1.3 and 7.1). The BB_Credit prior to the exchange of Link parameters shall be 1. E_Ports may agree to use an alternate buffer-to-buffer credit model for Class F following the successful exchange of Link parameters. **(alternate credit models)**

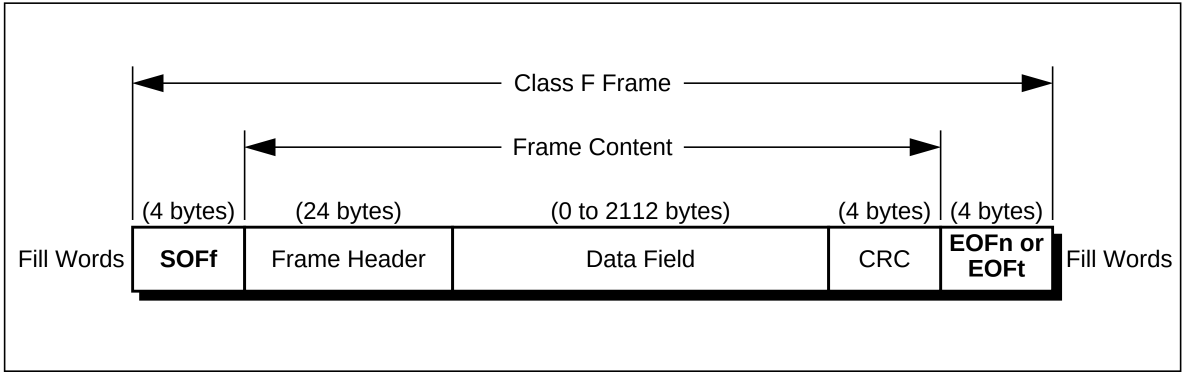
s) Since the SOFf delimiter does not indicate whether a frame is the first frame of a Sequence, the starting SEQ_CNT of every Sequence shall be zero. **(Sequence reassembly)**

5.5.3 Class F Frame Format

Class F frames shall use the Frame_Header defined in Clause 18 of FC-PH. The Class F frame format is shown in figure 8. The Start_of_Frame Fabric (**SOFf**) delimiter shall precede the frame content of all Class F frames. The Data Field size of all Class F frames shall be less than or equal to 256 bytes prior to the successful completion of Exchange Link Parameters (see 6.1.3; Exchange Link Parameters establishes the maximum receive frame size for Class F frames). All Class F frames shall include the CRC defined in Clause 17 of FC-PH. The End_of_Frame Normal (**EOFn**) delimiter shall immediately follow the CRC of all normally completed Class F Data frames and all normally completed Class F Link_Control frames except the last frame of a Sequence. The End_of_Frame Terminate (**EOFt**) delimiter shall immediately follow the CRC of all Class F Link_Control frames that indicate the last frame of a Sequence which is normally terminated. A Class F frame is preceded and followed by Primitive Signals as defined in FC-PH.

An E_Port or Switch may invalidate or discard without notification any incorrectly formed Class F frame, or any Class F frame with a code violation or CRC error.

Figure 8 – Class F Frame Format



5.5.4 Class F Flow Control

Class F service uses both buffer-to-buffer and end-to-end flow controls. R_RDY is used for buffer-to-buffer flow control. R_RDY is transmitted by the E_Port at one end of the ISL, to the E_ Port at the other end of the ISL, to indicate that a buffer is available for further frame reception by the first E_Port. This process operates in both directions on the ISL.

ACK_1 frames are used to perform end-to-end flow control. ACK_1 frames shall begin with an **SOFf** delimiter. The ACK_1 frame shall be terminated by an **EOFn** or **EOFt** delimiter. The ACK_0 and ACK_N Link Control frame shall not be used for Class F service.

After the successful exchange of Link Parameters, an alternate method of buffer-to-buffer flow control may be established on an ISL (see 7.1). This alternate method of buffer-to-buffer flow control remains in effect until a Link Offline or Link Failure occurs, or a new set of Link Parameters is successfully exchanged between the E_Ports.

6 Switch Fabric Services

This clause describes services provided for use by Switch Fabrics.

6.1 Switch Fabric Internal Link Services (SW_ILS)

This clause describes Link Services that operate internal to the Fabric between Switches. All SW_ILS frames shall be transmitted using the FT-1 frame format via the Class F service. The following defines the header fields of all SW_ILS frames:

- R_CTL: This field shall be set to hex'02' for all request frames, and to hex'03' for all reply frames.
- CS_CTL: This field shall be set to hex'00'.
- D_ID and S_ID: Set as indicated for the specific SW_ILS.
- TYPE: This field shall be set to hex'22', indicating Fibre Channel Fabric Switch Services.

All other fields shall be set as appropriate according to the rules defined in FC-PH.

The first word in the payload specifies the Command Code. The Command Codes are summarized in table 2.

Table 2 – SW_ILS Command Codes

Encoded Value (hex)	Description	Abbr.
01 00 00 00	Switch Fabric Internal Link Service Reject	SW_RJT
02 xx xx xx	Switch Fabric Internal Link Service Accept	SW_ACC
10 00 00 00	Exchange Link Parameters	ELP
11 xx xx xx	Exchange Fabric Parameters	EFP
12 00 00 00	Announce Address Identifier	AAI
13 00 xx xx	Request Domain_ID	RDI
17 00 00 00	Build Fabric	BF
18 00 00 00	Reconfigure Fabric	RCF
20 00 00 00	Disconnect Class 1 Connection	DSCN
21 00 00 00	Detect Queued Class 1 Connection Request Deadlock	LOOPD
others	Reserved	
70 00 00 00 to 7F 00 00 00	Vendor Unique	

6.1.1 Switch Fabric Internal Link Service Accept (SW_ACC)

The Switch Fabric Internal Link Service Accept reply Sequence shall notify the transmitter of an SW_ILS request that the SW_ILS request Sequence has been completed. The first word of the Payload shall contain hex ‘02 xx xx xx’. The remainder of the Payload is unique to the specific SW_ILS request.

Protocol: SW_ACC may be sent as a reply Sequence to any SW_ILS request.

Format: FT–1

Addressing: The S_ID field shall be set to the value of the D_ID field in the SW_ILS request. The D_ID field shall be set to the value of the S_ID field in the SW_ILS request.

Payload: The Payload content following the first word is defined within individual SW_ILS requests.

6.1.2 Switch Fabric Internal Link Service Reject (SW_RJT)

The Switch Fabric Internal Link Service Reject shall notify the transmitter of an SW_ILS request that the SW_ILS request Sequence has been rejected. A four-byte reason code shall be contained in the Data_Field. SW_RJT may be transmitted for a variety of conditions which may be unique to a specific SW_ILS request.

Protocol: SW_RJT may be sent as a reply Sequence to any SW_ILS request.

Format: FT–1

Addressing: The S_ID field shall be set to the value of the D_ID field in the SW_ILS request. The D_ID field shall be set to the value of the S_ID field in the SW_ILS request.

Payload: The format of the SW_RJT reply Payload is shown in table 3.

Table 3 – SW_RJT Payload

Item	Size Bytes
hex ‘01 00 00 00’	4
Reserved	1
Reason Code	1
Reason Code Explanation	1
Vendor Unique	1

Reason Code: The Reason Codes are summarized in table 4.

Table 4 – SW_RJT Reason Codes

Encoded Value (Bits 23-16)	Description
0000 0001	Invalid SW_ILS command code
0000 0010	Invalid revision level
0000 0011	Logical error
0000 0100	Invalid payload size
0000 0101	Logical busy
0000 0111	Protocol error
0000 1001	Unable to perform command request
0000 1011	Command not supported
others	Reserved
1111 1111	Vendor Unique error

Invalid SW_ILS command code: The Command Code is not recognized by the recipient.

Invalid revision level: The recipient does not support the specified revision level.

Logical error: The request identified by the Command Code and the Payload content is invalid or logically inconsistent for the conditions present.

Invalid payload size: The size of the Payload is inconsistent with the Command Code and/or any Length fields in the Payload.

Logical busy: The recipient is busy and is unable to process the request at this time.

Protocol error: An error has been detected that violates the protocol.

Unable to perform command request: The recipient cannot perform the request.

Command not supported: The command code is not supported by the recipient.

Vendor Unique Error: The Vendor Unique field indicates the error condition.

Reason Code Explanation: The Reason Code Explanation is summarized in table 5.

Table 5 – SW_RJT Reason Code Explanation

Encoded Value (Bits 15-8)	Description
0000 0000	No additional explanation
0000 0001	Class F Service Parameter error
0000 0011	Class N Service Parameter error
0000 0100	Unknown Flow Control code
0000 0101	Invalid Flow Control Parameters
0000 1101	Invalid Port_Name
0000 1110	Invalid Switch_Name
0000 1111	R_A_TOV or E_D_TOV mismatch
0001 0000	Invalid Domain_ID_List
0001 1001	Command already in progress
0010 1001	Insufficient resources available
0010 1010	Domain_ID not available
0010 1011	Invalid Domain_ID
0010 1100	Request not supported
0010 1101	Link Parameters not yet established
0010 1110	Contiguous Domain_IDs not available
0010 1111	E_Port is Isolated
others	Reserved

Vendor Unique: This field is valid when the Reason Code indicates a Vendor Unique error.

6.1.3 Exchange Link Parameters (ELP)

The Exchange Link Parameters Switch Fabric Internal Link Service requests the exchange of Link Parameters between two E_Ports connected via an ISL. The exchange of Link Parameters establishes the operating environment between the two E_Ports, and the capabilities of the Switches that are connected by the E_Ports. When an ELP is received by an E_Port, any Active or Open Class F Sequences between the two E_Ports, and any Dedicated Connections, shall be abnormally terminated prior to transmission of the SW_ACC reply Sequence.

Use of the ELP SW_ILS for Switch Port initialization is described in 7.1. Other uses of ELP are not defined by this Standard.

Protocol:

Exchange Link Parameters (ELP) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Switch Port initialization, the S_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the originating Switch; the D_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the destination Switch.

Payload: The format of the ELP request Payload is shown in table 6.

Table 6 – ELP Request Payload

Item	Size Bytes
hex ‘10 00 00 00’	4
Revision	1
Reserved	3
R_A_TOV	4
E_D_TOV	4
Requester E_Port_Name	8
Requester Switch_Name	8
Class F Service Parameters	16
Class 1 E_Port Parameters	4
Class 2 E_Port Parameters	4
Class 3 E_Port Parameters	4
Reserved	20
ISL Flow Control Mode	2
Flow Control Parameter Length (N)	2
Flow Control Parameters	N

Revision: This field denotes the revision of the protocol. The first revision has the value of 1.

R_A_TOV: This field shall be set to the value of R_A_TOV required by the Switch.

E_D_TOV: This field shall be set to the value of E_D_TOV required by the Switch.

NOTE – The Value of R_A_TOV and E_D_TOV may be established by Profile or other means.

E_Port_Name: The E_Port_Name is an eight-byte field which identifies an E_Port for identification purposes. The format of the name is specified in FC-PH. Each E_Port shall provide a unique E_Port_Name within the Fabric.

Switch_Name: The Switch_Name is an eight-byte field which identifies a Switch for identification purposes. The format of the name is specified in FC-PH. Each Switch shall provide a unique Switch_Name within the Fabric.

Class F Service Parameters: This field contains the E_Port Class F Service Parameters. The format of the Parameters is shown in table 7.

Table 7 – E_Port Class F Service Parameters

Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	VAL	Reserved															Reserved															
1	R		XII	Reserved													Receive Data Field Size															
2	Concurrent Sequences																End-to-End Credit															
3	Open Sequences per Exchange																Reserved															

The Class F Service Parameters are defined as follows:

- VAL (Class Valid): This bit shall be set to one.

- XII (X_ID Interlock): This bit when one indicates that the E_Port supplying this parameter requires that an interlock be used during X_ID assignment in Class F. In X_ID assignment, the Sequence Initiator shall set the Recipient X_ID value to hex'FFFF' in the first Data frame of a Sequence, and the Recipient shall supply its X_ID in the ACK frame corresponding to the first Data frame of a Sequence. The Sequence Initiator shall not transmit additional frames until the corresponding ACK is received. Following reception of the ACK, the Sequence Initiator continues transmission of the Sequence using both assigned X_ID values.

- Receive Data Field Size: This field shall specify the largest Data Field size in bytes for an FT-1 frame that can be received by the E_Port supplying the Parameters as a Sequence Recipient for a Class F frame. Values less than 256 or greater than 2112 are invalid. Values shall be a multiple of four bytes.

- Concurrent Sequences: This field shall specify the number of Sequence Status Blocks provided by the E_Port supplying the Parameters for tracking the progress of a Sequence as a Sequence Recipient. The maximum number of Concurrent Sequences that can be specified is 255. A value of zero in this field is reserved. In Class F, the value of SEQ_ID shall range from 0 to 255, independent of the value in this field. An E_Port is allowed to respond with P_BSY to a frame initiating a new Sequence if E_Port resources are not available.

- End-to-End Credit: End-to-end credit is the maximum number of Class F Data frames which can be transmitted by an E_Port without receipt of accompanying ACK or Link_Response frames. The minimum value of end-to-end credit is one. The end-to-end credit field specified is associated with the number of buffers available for holding the Data_Field of a Class F frame and processing the contents of that Data_Field by the E_Port supplying the Parameters. Bit 15 of this field shall be set to zero. A value of zero for this field is reserved.

- Open Sequences per Exchange: The value of the Open Sequences per Exchange shall specify the maximum number of Sequences that can be Open at one time at the Recipient between a pair of E_Ports for one Exchange. This value plus two shall specify the number of instances of Sequence Status that shall be maintained by the Recipient for a single Exchange in the Exchange Status Block. This value is used for Exchange and Sequence tracking. The value in this field limits the link facility resources required for error detection and recovery (see FC-FG).

Class N E_Port Parameters: E_Port Parameters indicate that the E_Port is capable of transporting the indicated Class of Service, and the conditions under which it can transport the Class. One word of the ELP Payload is allocated for each Class.

Class 1 E_Port Parameters: This field contains the Class 1 E_Port Parameters. The format of the Parameters is shown in table 8.

Table 8 – Class 1 E_Port Parameters

Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	VAL	IMX	XPS	LKS	Reserved												Receive Data Field Size															

The Class 1 E_Port Parameters are defined as follows:

- VAL (Class Valid): This bit is set to one if the E_Port supports Class 1. If this bit is zero, all other Class 1 E_Port Parameters shall be invalid.

- IMX (Intermix): This bit is set to one if the E_Port can perform Intermix as defined in FC-PH. Intermix shall be functional only if both E_Ports indicate support for this feature.

- XPS (Transparent Mode Stacked Connect Request): This bit is set to one if the E_Port can perform Transparent Mode Stacked Connect Requests as defined in FC-PH. Transparent Mode Stacked Connect Requests shall be functional only if both E_Ports indicate support for this feature. A Switch shall not indicate support for both XPS and LKS.

- LKS (Lock-down Mode Stacked Connect Request): This bit is set to one if the E_Port can perform Lock-down Mode Stacked Connect Requests as defined in FC-PH. Lock-down Mode Stacked Connect Requests shall be functional only if both E_Ports indicate support for this feature. A Switch shall not indicate support for both XPS and LKS.

- Receive Data Field Size: This field shall specify the largest Data Field size in bytes for an FT-1 frame that can be received by the E_Port supplying the Parameters for a Class 1 frame. Values less than 256 or greater than 2112 are invalid. Values shall be a multiple of four bytes.

Class 2 E_Port Parameters: This field contains the Class 2 E_Port Parameters. The format of the Parameters is shown in table 9.

Table 9 – Class 2 E_Port Parameters

Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	VAL	R			SEQ	Reserved											Receive Data Field Size															

The Class 2 E_Port Parameters are defined as follows:

- VAL (Class Valid): This bit shall be set to one if the E_Port supports Class 2. If this bit is zero, all other Class 2 E_Port Parameters shall be invalid.

- SEQ (Sequential Delivery): If this bit is set to one by an E_Port, it is indicating that the Switch is able to guarantee sequential delivery (as defined in FC-PH) of Class 2 frames. Sequential Delivery shall be functional only if both E_Ports indicate support for this feature.

- Receive Data Field Size: This field shall specify the largest Data Field size in bytes for an FT-1 frame that can be received by the E_Port supplying the Parameters for a Class 2 frame. Values less than 256 or greater than 2112 are invalid. Values shall be a multiple of four bytes.

Class 3 E_Port Parameters: This field contains the Class 3 E_Port Parameters. The format of the Parameters is shown in table 10.

Table 10 – Class 3 E_Port Parameters

Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	VAL	R			SEQ	Reserved											Receive Data Field Size															

The Class 3 E_Port Parameters are defined as follows:

- VAL (Class Valid): This bit shall be set to one if the E_Port supports Class 3. If this bit is zero, all other Class 3 E_Port Parameters shall be invalid.

- SEQ (Sequential Delivery): If this bit is set to one by an E_Port, it is indicating that the Switch is able to guarantee sequential delivery (as defined in FC-PH) of Class 3 frames. Sequential Delivery shall be functional only if both E_Ports indicate support for this feature.

- Receive Data Field Size: This field shall specify the largest Data Field size in bytes for an FT-1 frame that can be received by the E_Port supplying the Parameters for a Class 3 frame. Values less than 256 or greater than 2112 are invalid. Values shall be a multiple of four bytes.

ISL Flow Control Mode: This field contains a code which specifies the Flow Control method supported by the E_Port. Values of hex’0000’ and hex’FFFF’ are reserved. Values of hex’8000’ through hex’FFFE’ are Vendor Unique. All other values are reserved for future standardization.

Flow Control Parameter Length: This field specifies the length in bytes of the Flow Control Parameters that follow. Values shall be a multiple of four. A value of zero indicates no parameters follow.

Flow Control Parameters: These parameters contain information used to configure Flow Control for the ISL.

NOTE – Different Switch implementations may use different methods for managing flow control of user frames across an ISL. These parameters are intended to provide a Switch-specific way to indicate these flow control parameters. A Switch will not be prohibited from supporting more than one method. See annex A for more information.

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the ELP command
Accept (SW_ACC)
Signifies acceptance of the ELP request.

- Accept Payload

Payload: The format of the ELP Accept Payload is shown in table 11.

Table 11 – ELP Accept Payload

Item	Size Bytes
hex ‘02 00 00 00’	4
Revision	1
Reserved	3
R_A_TOV	4
E_D_TOV	4
Responder E_Port_Name	8
Responder Switch_Name	8
Class F Service Parameters	16
Class 1 E_Port Parameters	4
Class 2 E_Port Parameters	4
Class 3 E_Port Parameters	4
Reserved	20
ISL Flow Control Mode	2
Flow Control Parameter Length (N)	2
Flow Control Parameters	N

The fields in table 11 are the same as defined for table 6.

6.1.4 Exchange Fabric Parameters (EFP)

The Exchange Fabric Parameters Switch Fabric Internal Link Service requests the exchange of Fabric Parameters between two E_Ports connected via an ISL. The exchange of Fabric Parameters is used to establish the address allocation within the Fabric. When an E_Port receives EFP from another E_Port, all Active or Open Class F Sequences and Dedicated Connections shall be unaffected.

Use of the EFP SW_ILS for Fabric Configuration is described in 7.2 and 7.3. Other uses of EFP are not defined by this Standard.

Protocol:

Exchange Fabric Parameters (EFP) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Fabric Configuration, the S_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the originating Switch. The D_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the destination Switch.

Payload: The format of the EFP request Payload is shown in table 12.

Table 12 – EFP Request Payload

Item	Size Bytes
Command code = hex ‘11’	1
Record length = hex ‘10’	1
Payload length	2
Reserved	3
Principal Switch_Priority	1
Principal Switch_Name	8
Domain_ID_List	N

Record Length: This field contains an 8-bit unsigned binary integer that specifies the total length of each record in the Payload (see below). The value shall be hex ‘10’.

Payload Length: This field contains a 16-bit unsigned binary integer that specifies the total length of the Payload. The rightmost two bits shall be zero. The value specified shall be greater than or equal to 16, and less than or equal to 65532.

Principal Switch_Priority: This field shall specify the priority level of the Switch that the transmitting Switch believes is the Principal Switch. Values for this field are summarized in table 13.

Table 13 – Switch_Priority Field Values

Value (hex)	Description
00	Reserved
01	Highest priority value. (note 1)
02	The Switch was the Principal Switch prior to sending or receiving BF. (note 2)
02 to FE	Higher to lower priority values. (note 3)
FF	The Switch is not capable of acting as a Principal Switch.

Notes:

1 This value allows the system administrator to establish which Switch becomes the Principal Switch.

2 This allows the same Switch to become Principal Switch if it is still part of the Fabric after sending and/or receiving the Build Fabric SW_ILS.

3 The Switch_Priority value for a given Switch is established by means not defined by this Standard.

Principal Switch_Name: This field shall specify the Switch_Name of the Switch that the transmitting Switch believes is the Principal Switch.

Domain_ID_List: This field shall contain a list of records which specify the Domain_ID and corresponding Switch_Name of the Switch that has been granted the Domain_ID by the Principal Switch. The Domain_ID_List shall contain a record for each value of Domain_ID which has been assigned. If no Switch has been assigned a Domain_ID, the Domain_ID_List shall contain no records. The format of a Domain_ID_List record is shown in table 14.

Table 14 – Domain_ID_List Record Format

Item	Size Bytes
Record_Type	1
Domain_ID	1
Reserved	2
Reserved	4
Switch_Name for Domain_ID	8

Record_Type: This field shall specify the type of record. Values for this field are summarized in table 15.

Table 15 – Record_Type Field Values

Value (hex)	Description
00	Reserved
01	Domain_ID_List record
all others	Reserved (note 1)
Notes: 1 A second record type may be defined for Alias groups in a future standard.	

Domain_ID: This field shall specify the Domain_ID assigned by the Principal Switch.

Switch_Name for Domain_ID: This field shall specify the Switch_Name of the Switch that has been assigned the Domain_ID by the Principal Switch.

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the EFP command
Accept (SW_ACC)
Signifies acceptance of the EFP request.
– Accept Payload

Payload: The format of the EFP Accept Payload is shown in table 16.

Table 16 – EFP Accept Payload

Item	Size Bytes
Command code = hex ‘02’	1
Page length = hex ‘10’	1
Payload length	2
Reserved	3
Principal Switch_Priority	1
Principal Switch_Name	8
Domain_ID_List	N

The fields in table 16 are the same as defined for table 12.

6.1.5 Announce Address Identifier (AAI)

The Announce Address Identifier Switch Fabric Internal Link Service communicates the address identifier of the E_Port to another E_Port. This communication establishes that the E_Port has been assigned an address identifier, and that the Recipient may request an address identifier from the Originating E_Port.

Use of the AAI SW_ILS for Fabric Configuration is described in 7.3. Other uses of AAI are not defined by this Standard.

Protocol:

Announce Address Identifier (AAI) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Fabric Configuration, the S_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the originating Switch. The D_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the destination Switch.

Payload: The format of the AAI request Payload is shown in table 17.

Table 17 – AAI Request Payload

Item	Size Bytes
hex ‘12 00 00 00’	4
Originating Switch_Name	8
Reserved	1
Originating Address identifier	3

Originating Switch_Name: This field shall contain the Switch_Name of the Originating E_Port.

Originating Address identifier: This field shall contain the address identifier of the Originating E_Port.

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
  Signifies the rejection of the AAI command
Accept (SW_ACC)
  Signifies acceptance of the AAI request.
– Accept Payload

Payload: The format of the AAI Accept Payload is shown in table 18.

Table 18 – AAI Accept Payload

Item	Size Bytes
hex '02 00 00 00'	4
Responding Switch_Name	8
Reserved	1
Responding Address identifier	3

Responding Switch_Name: This field shall contain the Switch_Name of the Responding E_Port.

Responding Address identifier: This field shall contain the address identifier of the Responding E_Port.

6.1.6 Request Domain_ID (RDI)

The Request Domain_ID Switch Fabric Internal Link Service is sent by a Switch to request a Domain_ID from the Domain Address Manager. RDI shall not be sent by a Switch unless the Switch has received an AAI SW_ILS since the last reconfiguration event.

Use of the RDI SW_ILS for Fabric Configuration is described in 7.3. Other uses of RDI are not defined by this Standard.

Protocol:

Request Domain_ID (RDI) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Fabric Configuration, the S_ID field shall be set to hex'FFFFFD', indicating the Fabric Controller of the originating Switch. The D_ID field shall be set to hex'FFFFFD', indicating the Fabric Controller of the destination Switch.

Payload: The format of the RDI request Payload is shown in table 19.

Table 19 – RDI Request Payload

Item	Size Bytes
hex ‘13’	1
Reserved	1
Payload Length	2
Requesting Switch_Name	8
Reserved	3
Requested Domain_ID #1	1
Reserved	3
Requested Domain_ID #2	1
...	
Reserved	3
Requested Domain_ID #n	1

Payload Length: This field contains a 16-bit unsigned binary integer that specifies the total length of the Payload. The rightmost two bits shall be zero. The value specified shall be greater than or equal to 16, and less than or equal to 968.

Requesting Switch_Name: This field specifies the Switch_Name of the Switch requesting a Domain_ID.

Requested Domain_ID: This field shall contain the requested Domain_ID of the Switch requesting a Domain_ID. This field is set to either the Preferred Domain_ID if it is available, or zero. If more than one Requested Domain_ID is specified, the Domain_IDs shall represent a contiguous Domain_ID space; each Domain_ID shall be separated from another Domain_ID by a value of one.

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the RDI command
Accept (SW_ACC)
Signifies acceptance of the RDI request.

- Accept Payload

Payload: The format of the RDI accept Payload is shown in table 20.

Table 20 – RDI Accept Payload

Item	Size Bytes
hex ‘02’	1
Reserved	1
Payload Length	2
Requesting Switch_Name	8
Reserved	3
Granted Domain_ID #1	1
Reserved	3
Granted Domain_ID #2	1
...	
Reserved	3
Granted Domain_ID #n	1

Payload Length: This field contains a 16-bit unsigned binary integer that specifies the total length of the Payload. The rightmost two bits shall be zero. The value specified shall be equal to the value specified in the Request Payload.

Requesting Switch_Name: This field specifies the Switch_Name of the Switch requesting a Domain_ID.

Granted Domain_ID: This field shall contain the Domain_ID granted by the Domain Address Manager to the requesting Switch. This field is set to either the Preferred Domain_ID specified in the Request if it is available, or zero. If more than one Requested Domain_ID was specified in the Request, the Response shall contain a number of Granted Domain_IDs equal to the number requested, and that represent a contiguous Domain_ID space; each Domain_ID shall be separated from another Domain_ID by a value of one. If the Domain Address Manager is unable to grant the full set of contiguous Domain_IDs, it shall reject the Request.

NOTE – The ability to grant more than one Domain_ID to a single Switch is intended to be used by Switches whose addressing scheme requires the use of more than one Domain_ID. The typical case, however, should be for one Switch to request exactly one Domain_ID.

6.1.7 Build Fabric (BF)

The Build Fabric Switch Fabric Internal Link Service requests a non-disruptive reconfiguration of the entire Fabric. Fabric Configuration is performed as described in clause 7.

> NOTE – Since the RCF causes a complete reconfiguration of the Fabric, and may cause addresses allocated to a Switch to change, the RCF SW_ILS should be used with caution. The BF SW_ILS allows the Fabric to attempt reconfiguration without loss of or change of address. Examples of situations in which BF is appropriate include a loss of a Principal ISL (Link Failure or Offline), or when two Fabrics are joined.

The transmission or reception of BF shall not of itself cause the loss of Class N frames, or cause a busy response to any Class N frames. Active or Open Class F Sequences between the two E_Ports, and any Dedicated Connections, shall not be abnormally terminated.

Use of the BF SW_ILS for Fabric Configuration is described in 7.2 and 7.3. Other uses of BF are not defined by this Standard.

Protocol:

Build Fabric (BF) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Fabric Configuration, the S_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the originating Switch. The D_ID field shall be set to hex’FFFFFD’, indicating the Fabric Controller of the destination Switch.

Payload: The format of the BF request Payload is shown in table 21.

Table 21 – BF Request Payload

Item	Size Bytes
hex ‘17 00 00 00’	4

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the BF command
Accept (SW_ACC)
Signifies acceptance of the BF request.

- Accept Payload

Payload: The format of the BF accept Payload is shown in table 22.

Table 22 – BF Accept Payload

Item	Size Bytes
hex ‘02 00 00 00’	4

6.1.8 Reconfigure Fabric (RCF)

The Reconfigure Fabric Switch Fabric Internal Link Service requests a disruptive reconfiguration of the entire Fabric. Fabric Configuration is performed as described in clause 7.

NOTE – Since the RCF causes a complete reconfiguration of the Fabric, and may cause addresses allocated to a Switch to change, this SW_ILS should be used with caution. Examples of situations in which RCF is appropriate include detection of overlapped Domains, or the failure of a Fabric Reconfiguration initiated by a BF.

When an RCF is transmitted by an E_Port, any Active or Open Class F Sequences between the two E_Ports, and any Dedicated Connections, shall be abnormally terminated. Also, all Class N frames shall be discarded, and all Dedicated Connections shall be abnormally terminated.

When an RCF is received by an E_Port, any Active or Open Class F Sequences between the two E_Ports, and any Dedicated Connections, shall be abnormally terminated prior to transmission of the SW_ACC reply Sequence. Also, all Class N frames shall be discarded, and all Dedicated Connections shall be abnormally terminated prior to transmission of the SW_ACC reply Sequence.

Use of the RCF SW_ILS for Fabric Configuration is described in 7.2 and 7.3. Other uses of RCF are not defined by this Standard.

Protocol:

Reconfigure Fabric (RCF) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: For use in Fabric Configuration, the S_ID field shall be set to hex'FFFFFD', indicating the Fabric Controller of the originating Switch. The D_ID field shall be set to hex'FFFFFD', indicating the Fabric Controller of the destination Switch.

Payload: The format of the RCF request Payload is shown in table 23.

Table 23 – RCF Request Payload

Item	Size Bytes
hex ‘18 00 00 00’	4

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the RCF command
Accept (SW_ACC)
Signifies acceptance of the RCF request.
– Accept Payload

Payload: The format of the RCF accept Payload is shown in table 24.

Table 24 – RCF Accept Payload

Item	Size Bytes
hex ‘02 00 00 00’	4

6.1.9 Disconnect Class 1 Connection (DSCN)

The Disconnect Class 1 Connection Switch Fabric Internal Link Service requests that the receiving E_Port abort an existing Class 1 Connection. This SW_ILS is used only if Link Failure or Link Reset is detected in the connection path. An F_Port that receives this SW_ILS shall perform a Link Reset to abort the connection with the attached N_Port.

NOTE – Normal disconnect should be performed by detecting EOFdt.

Protocol:

Disconnect Class 1 Connection (DSCN) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: The S_ID field shall be set to the address identifier of the sending E_Port. The D_ID field shall be set to the address identifier of the destination E_Port or F_Port.

Payload: The format of the DSCN request Payload is shown in table 25.

Table 25 – DSCN Request Payload

Item	Size Bytes
hex ‘20 00 00 00’	4
Reserved	3
Reason code for disconnect	1

Reason code for disconnect: This field specifies the reason for the disconnect, summarized in table 26.

Table 26 – DSCN Reason Codes

Encoded Value (Bits 7-0)	Description
0000 0001	Link Failure or Link Reset occurred
others	Reserved
1111 1111	Vendor Unique error

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the DSCN command
Accept (SW_ACC)
Signifies acceptance of the DSCN request.

- Accept Payload

Payload: The format of the DSCN accept Payload is shown in table 27.

Table 27 – DSCN Accept Payload

Item	Size Bytes
hex ‘02 00 00 00’	4

6.1.10 Detect Queued Class 1 Connection Request Deadlock (LOOPD)

The Detect Queued Class 1 Connection Request Deadlock Switch Fabric Internal Link Service is used to check for possible deadlocks caused by Connection requests being queued at the destination F_Port (Camp-On). For example, if a connection request from port A is queued at port B, a request from port B is queued at port C, and a request from port C is queued at port A, a deadlock has occurred.

A LOOPD SW_ILS shall be originated by the F_Port that received and queued the Camp-On connection request. The destination of the LOOPD SW_ILS shall be the F_Port with which the originating F_Port has an existing connection. The “Originating F_Port” field in the LOOPD request Payload shall be set equal to the address identifier of the originating F_Port. The LOOPD request and reply Sequences shall be routed by the Fabric from the source F_Port to the destination F_Port in a manner not defined by this Standard.

An F_Port that receives an acceptable LOOPD request shall reply with an SW_ACC reply Sequence and perform one of the following actions:

- If the address identifier of the receiving F_Port is equal to the “Originating F_Port” field in the LOOPD request Payload, and the receiving F_Port has a queued Camp-On connection request from the F_Port that sent the LOOPD request, the receiving F_Port shall reject the queued Camp-On connection request. The sending F_Port shall send an F_BSY to the requesting N_Port with a Reason Code of Fabric_Busy.

- If the address identifier of the receiving F_Port is equal to the “Originating F_Port” field in the LOOPD request Payload, and the receiving F_Port does not have a queued Camp-On connection request from the F_Port that sent the LOOPD request, the receiving F_Port shall perform no action; i.e., not reject any queued Camp-On connection request.

- If the address identifier of the receiving F_Port is not equal to the “Originating F_Port” field in the LOOPD request Payload, and the receiving F_Port has a queued Camp-On connection request pending at another F_Port, the receiving F_Port shall initiate a new LOOPD request to the F_Port with which the receiving F_Port has an existing connection, using without modification the received LOOPD request Payload as the Payload for the new request.

- If the address identifier of the receiving F_Port is not equal to the “Originating F_Port” field in the LOOPD request Payload, and the receiving F_Port does not have a queued Camp-On connection request pending at another F_Port, the receiving F_Port shall perform no action.

Note that if the originating F_Port and the destination F_Port are both part of the same Switch, the deadlock may be detected without use of this SW_ILS.

Protocol:

Detect Queued Class 1 Connection Request Deadlock (LOOPD) request Sequence
Accept (SW_ACC) Reply Sequence

Format: FT–1

Addressing: The S_ID field shall be set to the address identifier of the sending F_Port. The D_ID field shall be set to the address identifier of the destination F_Port.

Payload: The format of the LOOPD request Payload is shown in table 28.

Table 28 – LOOPD Request Payload

Item	Size Bytes
hex ‘21 00 00 00’	4
Reserved	1
Originating F_Port	3

Originating F_Port: This field specifies the address identifier of the F_Port that first originated the LOOPD request. A F_Port that initiates a LOOPD request as a result of receiving a LOOPD request shall use the contents of this field unmodified in the request Payload.

Reply Switch Fabric Internal Link Service Sequence:

Service Reject (SW_RJT)
Signifies the rejection of the LOOPD command
Accept (SW_ACC)
Signifies acceptance of the LOOPD request.

- Accept Payload

Payload: The format of the LOOPD accept Payload is shown in table 29.

Table 29 – LOOPD Accept Payload

Item	Size Bytes
hex ‘02 00 00 00’	4

7 Fabric Configuration

The Fabric Configuration process enables a Switch Port to determine its operating mode, exchange operating parameters, and provides for distribution of addresses. This process is summarized in table 30.

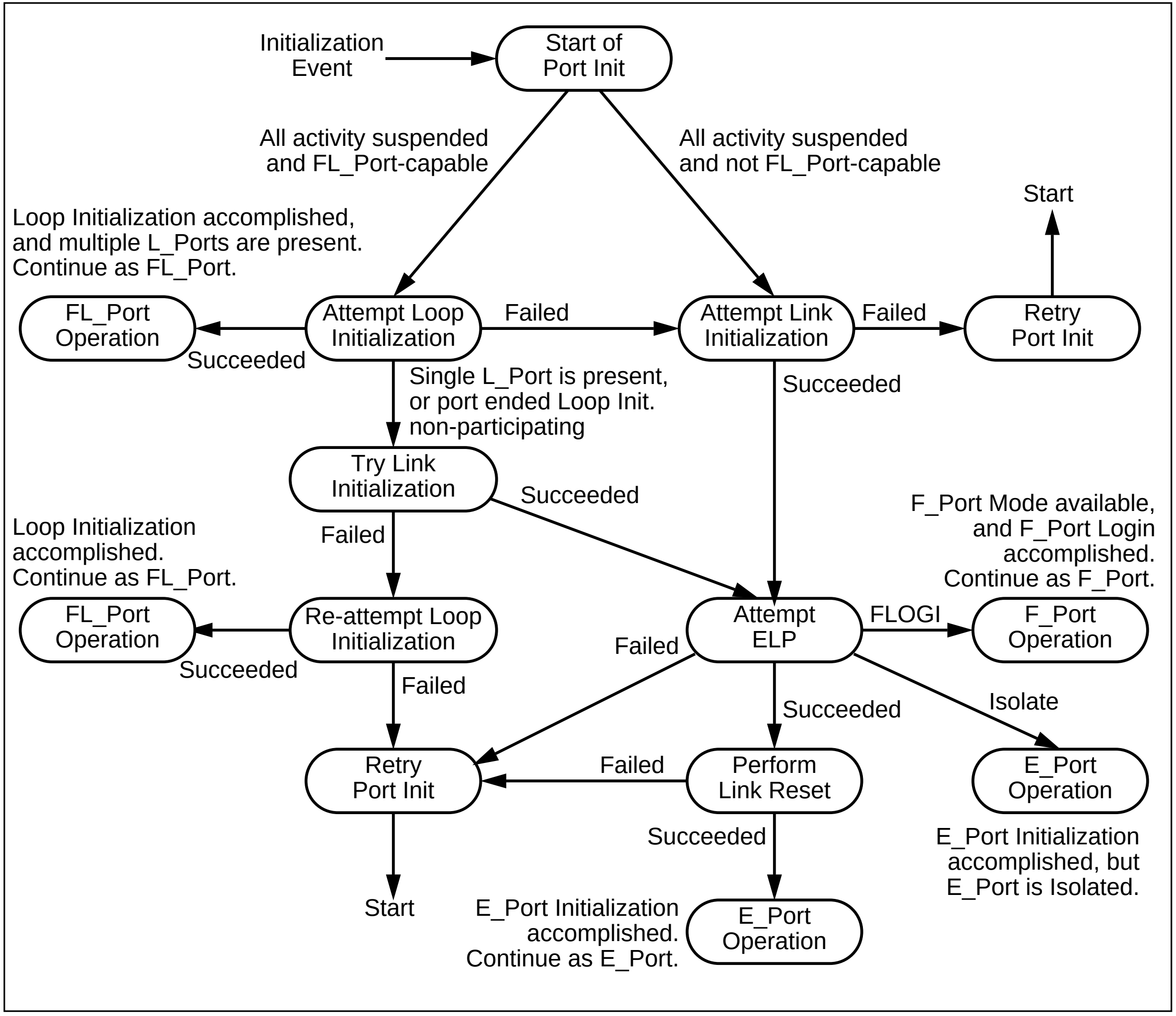
Table 30 – Fabric Configuration Summary

Step	Starting State	Process	Ending State
1. Establish Link Parameters and Switch Port operating mode	Switch Port has achieved word synch.	The Switch Port attempts to discover whether it is an FL_Port, an E_Port or an F_Port.	Switch Port mode is known. If a Port is an E_Port, Link Parameters have been exchanged and Credit has been initialized.
2. Select Principal Switch	BF or RCF SW_ILS transmitted or received.	Switch_Names are exchanged over all ISLs to select a Principal Switch, which becomes the Domain Address Manager.	The Principal Switch is selected.
3. Domain_ID Distribution	Domain Address Manager has been selected.	Switches request a Domain_ID from the Domain Address Manager.	All Switches have a Domain_ID.
4. Path Selection	All Switches have a Domain_ID.	Path Selection is not defined by this Standard.	Fabric is operational.

7.1 Switch Port Initialization

Switch Ports shall initialize as described below. Figure 9 shows a diagram of the process to illustrate the flow. If the figure is different than the text, the text shall apply. Note also that this flow assumes that a Switch Port is capable of at least E_Port operation; either E/F/FL_Port, E/F_Port, E/FL_Port, or E_Port. Initialization of Switch Ports that are F/FL_Port, FL_Port, or F_Port is defined in FC-PH and FC-AL.

Figure 9 – Switch Port Mode Initialization Flow


Initialization Event
Start of Port Init
All activity suspended and FL_Port-capable
All activity suspended and not FL_Port-capable
Start
Loop Initialization accomplished, and multiple L_Ports are present. Continue as FL_Port.
FL_Port Operation
Succeeded
Attempt Loop Initialization
Failed
Attempt Link Initialization
Failed
Retry Port Init
Single L_Port is present, or port ended Loop Init. non-participating
Succeeded
Try Link Initialization
Succeeded
F_Port Mode available, and F_Port Login accomplished. Continue as F_Port.
Loop Initialization accomplished. Continue as FL_Port.
Failed
FL_Port Operation
Re-attempt Loop Initialization
Succeeded
Attempt ELP
FLOGI
F_Port Operation
Failed
Failed
Succeeded
Isolate
Retry Port Init
Failed
Perform Link Reset
E_Port Operation
Succeeded
E_Port Initialization accomplished, but E_Port is Isolated.
Start
E_Port Initialization accomplished. Continue as E_Port.
E_Port Operation

a) **Start of Switch Port Initialization**. Switch Port initialization begins whenever an Initialization Event occurs. An Initialization Event can be:

- a power-on reset condition; or,
- receiving an initialization Primitive Sequence, such as LR or LIP; or,
- outside intervention requesting an initialization; or,
- a transition to Link Offline, as defined in FC-PH; or,
- a loss of word synchronization; or,
- RCF received by an Isolated E_Port (see 7.4); or,
- a failure to successfully complete a prior initialization attempt, and the timeout period has expired.

When an Initialization Event occurs, all activity on the Switch Port is suspended until the Initialization is complete. Go to **step (b)**.

b) **Attempt Loop Initialization**. An FL_Port-capable Switch Port attempts Loop Initialization (as defined in FC-AL clause 10). If the FL_Port transitions from the OPEN_INIT state to the MONITORING state and in participating mode, and the resulting AL_PA bitmap generated during the LISA Loop Initialization Sequence indicates more than one L_Port (other than the Switch Port) is attached, the Switch Port shall go to **step (h)**. If the FL_Port transitions from the OPEN_INIT state to the MONITORING state and in participating mode, and the resulting AL_PA bitmap generated during the LISA Loop Initialization Sequence indicates zero or one L_Port (other than the Switch Port) is attached, the Switch Port shall go to **step (c)**. If the FL_Port transitions from the OPEN_INIT state to the MONITORING state and in non-participating mode, the Switch Port shall go to **step (c)**. Otherwise, go to **step (e)**.

c) **Try Link Initialization**. In this step, the Switch Port is FL_Port-capable, and it has detected zero or one attached L_Port (NL_Port or FL_Port), or has transitioned to the MONITORING state in non-participating mode. There is a possibility that the Switch Port is point-to-point attached to another FL_Port-capable Switch Port. If the Switch Port is in the MONITORING state and in participating mode, it shall attempt Link Initialization as defined in FC-PH. If the Switch Port is in the MONITORING state and in non-participating mode, it shall remain in this state until it detects an attempt at Link Initialization, or it detects a new Initialization Event. If the attempted Link Initialization succeeds, proceed to **step (f)**. If the attempted Link Initialization fails, the Switch Port shall proceed to **step (d)** and retry Loop Initialization.

d) **Re-attempt Loop Initialization**. In this step, the Switch Port has detected that it may be able to operate point-to-point with another loop device, but the attempt to do so failed. In this case, the Switch Port shall then attempt to go back to loop operation by retrying Loop Initialization (as defined in FC-AL clause 10). If the Loop Initialization succeeds (the FL_Port transitions from the OPEN_INIT state to the MONITORING state and participating), the Switch Port shall go to **step (h)**. Otherwise, go to **step (l)**.

e) **Attempt Link Initialization**. The Switch Port shall attempt Link Initialization as defined in FC-PH. If the Link Initialization succeeds, proceed to **step (f)**. Otherwise, the Switch Port shall proceed to **step (l)** and retry the initialization.

f) **Attempt to Exchange Link Parameters**. The Switch Port shall originate an ELP SW_ILS request Sequence (see 6.1.3). Table 31 below defines the responses and actions to an ELP request for the originating E_Port.

Table 31 – Responses to ELP Request for Originating E_Port

Response to ELP	Indication	Originating E_Port Action
1. R_RDY	Request received at destination	Wait E_D_TOV for response frame
2. ACK_1	Request received at destination	Wait E_D_TOV for response frame
3. SW_ACC	Destination E_Port received and processed request	Send ACK_1, continue configuration with **step (g)**
4. F_BSY or P_BSY	Destination is busy	Retry (note 1)
5. F_RJT or P_RJT	The frame is not acceptable	Respond accordingly (note 3)

Table 31 – Responses to ELP Request for Originating E_Port

Response to ELP	Indication	Originating E_Port Action
6. ELP (rcvd E_Port_Name > own E_Port_Name)	Both E_Ports sent ELP at the same time	Send SW_ACC, continue configuration with **step (g)** (see Figure 10 for an example of this response)
7. ELP (rcvd E_Port_Name < own E_Port_Name)	Both E_Ports sent ELP at the same time	Send SW_RJT (note 2) (see Figure 10 for an example of this response)
8. ELP (rcvd E_Port_Name = own E_Port_Name)	E_Port output is looped back to input	Remove loopback condition
9. SW_RJT	Reason code/explanation: - Command already in progress - Logical busy - other	- send SW_ACC (note 4) (see Figure 10 for an example of this response) - retry (note 1) - respond accordingly
10. FLOGI	Destination is an N_Port	Respond accordingly (note 3)
11. any other frame	Could be anything	Discard frame and retry (note 1)
12. E_D_TOV expires	Destination is busy; or, ELP, SW_ACC, ACK_1 frame lost; or, destination is not an E_Port	Retry (note 1)

Notes:

1 The retry is performed following a timeout period, as defined in **step (l)** below.

2 The Reason Code shall be “Unable to perform command request” with an Reason Explanation of “Command already in progress”.

3 Response is defined in FC-PH. A retry may be appropriate.

4 The SW_ACC is sent for the other ELP Exchange in progress, as described in Response #6. See figure 10.

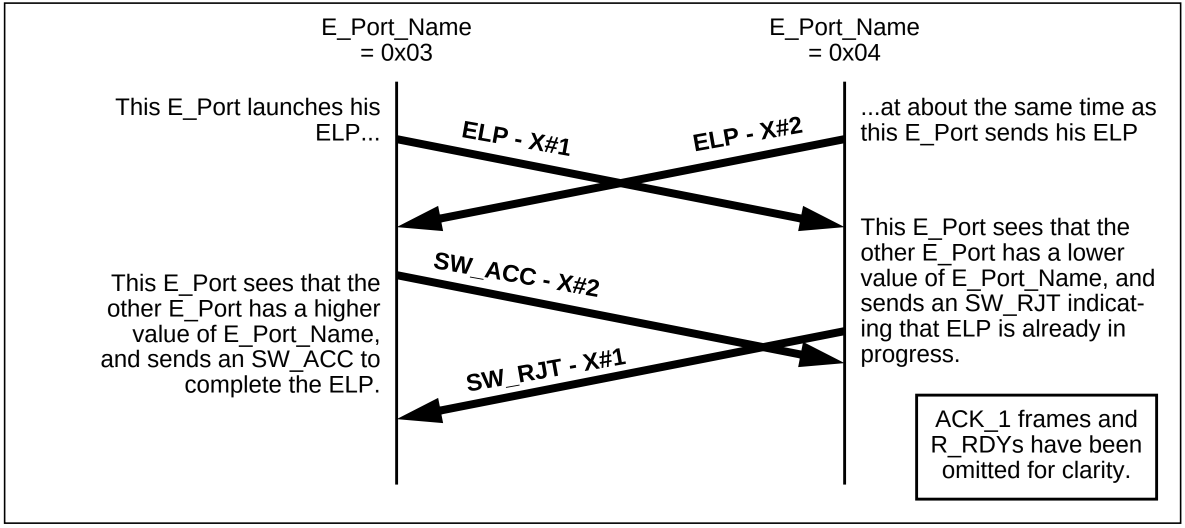
The originating E_Port shall consider the exchange of Link Parameters complete (but not necessarily successful) when it has received the SW_ACC or SW_RJT and has transmitted the ACK_1 for the SW_ACC or SW_RJT reply Sequence. The responding E_Port shall consider the exchange of Link Parameters complete when it has received the ACK_1 for the SW_ACC or SW_RJT. The exchange of Link Parameters shall be considered successful when the exchange of Link Parameters is complete, and the reply to the ELP is an SW_ACC, and both E_Ports agree that the parameters exchanged are acceptable. If the exchange of Link Parameters is successful, the Switch Port shall go to **step (g)**. If the responding E_Port does not agree that the parameters are acceptable, it shall return an SW_RJT reply Sequence indicating the reason for the disagreement, and wait for the originating E_Port to initiate another ELP request Sequence. If the originating E_Port does not agree that the parameters in the SW_ACC are acceptable, or it receives an SW_RJT indicating the parameters in the ELP request were not acceptable to the responding E_Port, it may:

1) originate a new ELP request Sequence with modified parameters; or,

2) go to **step (j)** and operate as an Isolated E_Port (see 7.4); or,

3) perform the Link Offline protocol as defined in FC-PH and go to **step (l)** and retry the initialization.

Figure 10 – Simultaneous ELP Processing


E_Port_Name
= 0x03
E_Port_Name
= 0x04
This E_Port launches his ELP...
ELP - X#1
ELP - X#2
...at about the same time as this E_Port sends his ELP
This E_Port sees that the other E_Port has a lower value of E_Port_Name, and sends an SW_RJT indicating that ELP is already in progress.
SW_ACC - X#2
This E_Port sees that the other E_Port has a higher value of E_Port_Name, and sends an SW_ACC to complete the ELP.
SW_RJT - X#1
ACK_1 frames and R_RDYs have been omitted for clarity.

g) **Perform Link Reset**. Following the successful completion of ELP, the value of buffer-to-buffer and end-to-end Class F Credit are initialized. In order to initialize the Flow Control parameters, the Switch Port that originated the successful ELP SW_ILS shall attempt the Link Reset protocol as defined in FC-PH. If the Link Reset succeeds, go to **step (k)**. Otherwise, go to **step (l)**.

NOTE – The re-initialization of Link credit is necessary since the Flow Control parameters in the ELP Payload are intended to communicate Link credit parameters for a specific credit model. The Link Reset is the common method defined by FC-PH for establishing a known credit state.

h) **Initialize as an FL_Port**. The Switch Port has detected a functional Arbitrated Loop. The Switch Port shall continue to operate as an FL_Port until the next Initialization Event.

i) **Initialize as an F_Port**. The Switch Port has detected an attached N_Port. The Switch Port shall continue to operate as an F_Port until the next Initialization Event.

j) **Initialize as an Isolated E_Port**. The Switch Port has completed the exchange of Link Parameters with another E_Port. If the Link Parameters exchanged were not acceptable, then the E_Port shall become Isolated and not continue with Fabric Configuration, as described in 7.4. The Switch Port shall continue to operate as an E_Port until the next Initialization Event.

k) **Initialize as an E_Port**. The Switch Port has completed the exchange of Link Parameters with another E_Port. If the Link Parameters exchanged were acceptable, then the E_Port shall participate in the next phase of Fabric Configuration, described in 7.2. The Switch Port shall continue to operate as an E_Port until the next Initialization Event.

l) **Retry Switch Port Initialization**. The Switch Port shall wait for R_A_TOV before retrying Switch Port Initialization. If the Switch Port is FL_Port-capable, it shall transmit LIP; otherwise, it shall transmit OLS. If the Switch Port detects an Initialization Event during the timeout period, it shall not wait for the timeout period to expire; the Switch Port shall go to **step (a)**. Otherwise, after the timeout period has expired, the Switch Port shall go to **step (a)**.

7.2 Principal Switch Selection

A Principal Switch shall be selected whenever at least one Inter-Switch Link is established. The selection process chooses a Principal Switch, which is then designated as the Domain Address Manager. The behavior of a Switch during this process is as follows:

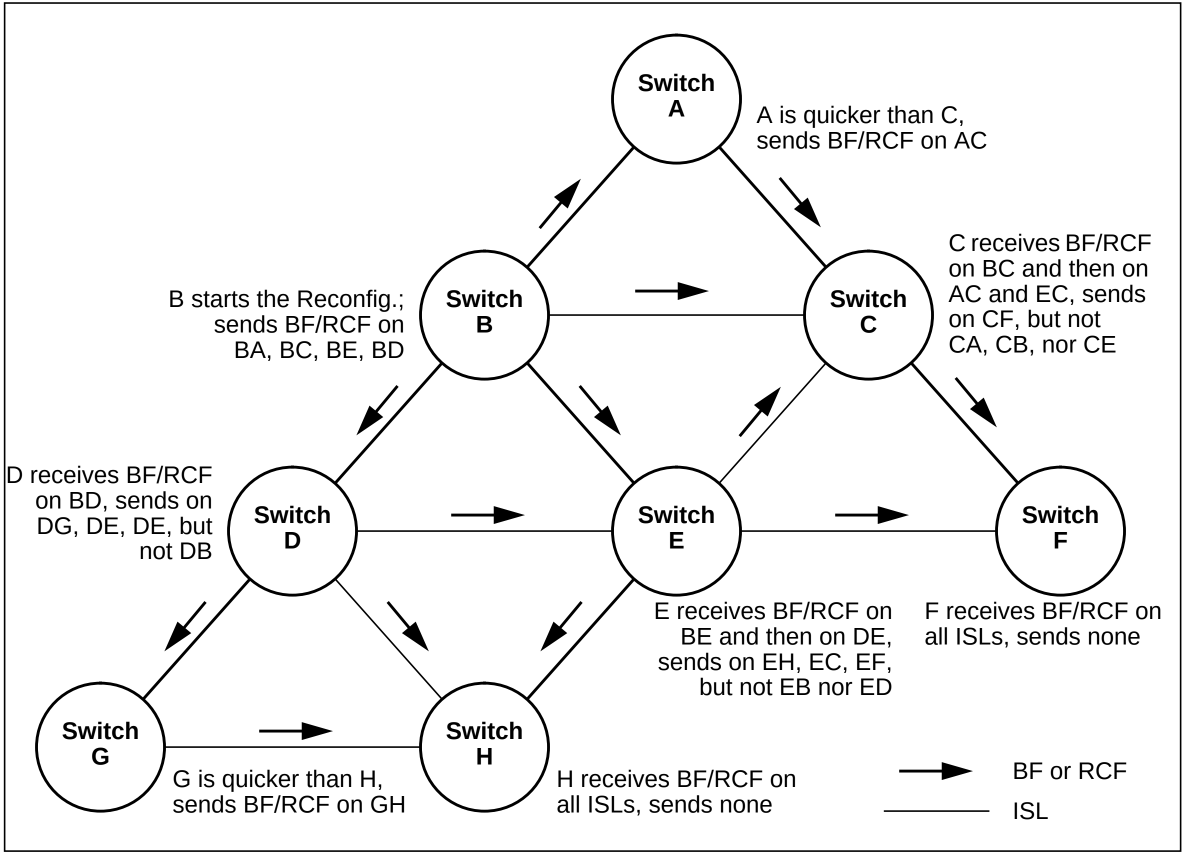
a) A Switch may request a Fabric Reconfiguration at any time by transmitting a BF or an RCF request Sequence. Unless warranted by current conditions, a Switch shall always first attempt a non-disruptive Fabric Reconfiguration by sending BF request Sequence. The recommended uses of BF and RCF are summarized in table 32.

Table 32 – Recommended BF and RCF Usage Summary

Event	BF or RCF Response
A Principal ISL experiences Link Failure or a transition to Offline	BF
A configured Fabric is joined to another configured Fabric, and they do not overlap	BF
An unconfigured Switch or Fabric is joined to a configured Fabric	neither (see below)
A configured Fabric is joined to another configured Fabric, and an overlap is detected	RCF
Reconfiguration caused by BF fails for any reason	RCF

b) If the Switch is attempting a non-disruptive Fabric Reconfiguration, the Switch shall transmit a BF request Sequence on all E_Ports that the Switch has not yet received a BF request. The Switch shall respond appropriately to any BF request Sequence received on any E_Port, and shall not transmit a BF request Sequence on any E_Port from which a BF request Sequence has been received. Figure 11 shows an example diagram of the process to illustrate the flow of the requests. Any Class F frames other than RCF requests and the associated SW_ACC and ACK_1 frames shall receive an F_BSY response, with a Reason Code of “The Fabric is busy”.

Figure 11 – Propagation of BF and RCF SW_ILS requests


Switch A
A is quicker than C, sends BF/RCF on AC
Switch B
B starts the Reconfig.; sends BF/RCF on BA, BC, BE, BD
Switch C
C receives BF/RCF on BC and then on AC and EC, sends on CF, but not CA, CB, nor CE
Switch D
D receives BF/RCF on BD, sends on DG, DE, DE, but not DB
Switch E
Switch F
E receives BF/RCF on BE and then on DE, sends on EH, EC, EF, but not EB nor ED
F receives BF/RCF on all ISLs, sends none
Switch G
Switch H
G is quicker than H, sends BF/RCF on GH
H receives BF/RCF on all ISLs, sends none
BF or RCF
ISL

c) If the Switch is attempting a disruptive Fabric Reconfiguration, the Switch shall transmit an RCF request Sequence on all E_Ports that the Switch has not yet received an RCF request. The Switch shall respond appropriately to any RCF request Sequence received on any E_Port, and shall not transmit an RCF request Sequence on any E_Port from which an RCF request Sequence has been received.

d) If a Switch receives an RCF request Sequence while it is in the process of attempting a non-disruptive Fabric Reconfiguration, it shall stop the non-disruptive Fabric Reconfiguration and begin processing RCF requests as described above. Any Active or Open BF Sequences shall be abnormally terminated.

e) A Switch that is not yet configured (for example, after initial power-on) is not required to request BF or RCF. It may instead transmit an EFP SW_ILS to all initialized E_Ports to determine if the Switch is attached to a configured Fabric. The Switch shall process any received BF or RCF requests as described above.

f) The Switch shall wait for twice F_S_TOV following the completion of the last BF or RCF Exchange before originating an EFP request Sequence.

g) The Switch shall process all EFP Payloads based on the information available at the time of processing. A Switch may receive an EFP Payload either by receiving an EFP request Sequence at an E_Port, or by receiving at an E_Port an SW_ACC reply Sequence in response to an EFP request Sequence.

h) At the start of a non-disruptive Fabric Reconfiguration (BF), or a a disruptive Fabric Reconfiguration (RCF), the Domain_ID_List shall be empty (“zero Domain_ID_List”).

i) The Switch shall retain a Switch_Priority||Switch_Name value that it believes is the lowest in the Fabric. This value shall be initialized at the start of Fabric Reconfiguration (caused by BF or RCF) to the Switch’s value of Switch_Priority||Switch_Name. After the Switch is configured, it shall retain as the lowest value the Switch_Priority||Switch_Name of the Principal Switch.

j) If the Switch receives in an EFP Payload a non-zero Domain_ID_List (the list contains one or more records) and the Switch has a zero Domain_ID_List, then the Switch shall retain the received Switch_Priority||Switch_Name as the new value. The Switch shall also note from which E_Port it received the new value, for potential use as the upstream Principal ISL during address distribution.

k) If the Switch receives in an EFP Payload a zero Domain_ID_List and the Switch has a non-zero Domain_ID_List (i.e., it has received a Domain_ID), the Switch shall retain its current lowest Switch_Priority||Switch_Name value as the lowest value, without comparing with the received value. The Switch shall send AAI to the Switch from which it received the zero Domain_ID_List as described in 7.3.1.

l) If the Switch receives in an EFP Payload a zero Domain_ID_List and the Switch has a zero Domain_ID_List, and the received Switch_Priority||Switch_Name is lower than its current retained value, it shall discard the old value and retain the new value. The Switch shall also note from which E_Port it received the new value, for potential use as the upstream Principal ISL during address distribution.

m) The Switch shall communicate its retained Switch_Priority||Switch_Name to all E_Ports that it has not yet communicated that value. The Switch shall accomplish this either by originating a new EFP request Sequence, or by an SW_ACC reply Sequence to a received EFP request.

n) If the switch receives a new lower value of Switch_Priority||Switch_Name before it has had a chance to communicate a prior lower value to all other E_Ports, it shall not attempt to communicate the prior value, and shall instead attempt to communicate the new value. The Switch shall not abort or otherwise abnormally terminate an existing EFP Exchange originated by the Switch for the sole reason of the value of Switch_Priority||Switch_Name being adjusted lower prior to the completion of the Exchange.

o) The Switch shall always return the lowest known value of Switch_Priority||Switch_Name in a SW_ACC reply Sequence to an EFP request Sequence.

p) If the Domain_ID_List of the Switch is non-zero, and the Domain_ID_List in a received EFP Payload is non-zero, and if there are no corresponding records in the Domain_ID_Lists set to the same Domain_ID value, then the E_Port shall request a non-disruptive Fabric Configuration, as described above.

q) If the Domain_ID_List of the Switch is non-zero, and the Domain_ID_List in a received EFP Payload is non-zero, and if any corresponding records in the Domain_ID_Lists are set to the same Domain_ID value, then the E_Port shall not continue with Fabric Configuration, and shall become Isolated, as described in 7.4.

r) If the retained value of Switch_Priority||Switch_Name does not change for twice F_S_TOV, and if the retained value of Switch_Priority is equal to 0xFF, then there is no Switch capable of becoming a Principal Switch. The Switch shall cause all E_Ports to become Isolated, as described in 7.4.

s) If the retained value of Switch_Priority||Switch_Name does not change for twice F_S_TOV, and if the retained value of the Switch_Priority||Switch_Name is equal to the value of the Switch, then the Switch has become the Principal Switch.

t) If the Switch receives an AAI request Sequence, then a Principal Switch has been selected. The Switch shall request a Domain_ID as described in 7.3.

u) The Switch shall continue to process and generate EFP requests as appropriate until it either: determines that it has become Isolated from all other Switches; or, it receives or initiates a BF or RCF request (which restarts the selection process).

At the completion of this process, all Switches other than the Principal Switch shall retain knowledge of the E_Port through which was received the lowest value of Switch_Priority||Switch_Name. This E_Port is the start of the first ISL in the path to the Principal Switch for the Switch; this ISL is called the upstream Principal ISL.

7.3 Address Distribution

Once a Principal Switch (Domain Address Manager) has been selected, Switches may request a Domain_ID. The Principal Switch shall assign all Domain_IDs. All other Switches shall request Domain_IDs from the Principal Switch.

7.3.1 Domain_ID Distribution by the Principal Switch

The Principal Switch shall conduct Domain_ID distribution as follows:

a) At the completion of Principal Switch Selection, the Principal Switch shall assume the role of Domain Address Manager. The Principal Switch shall set its Switch_Priority value to hex’02’, if the current value of its Switch_Priority is less than or equal to hex ‘02’. The Principal Switch shall empty its Domain_ID_List. These actions shall not cause an EFP request to be generated as described in item “i“ below.

b) The Principal Switch shall then grant itself one (or more) Domain_ID from the pool of available Domain_IDs. This pool is maintained by the Principal Switch. If the Principal Switch had a specific Domain_ID prior to the Reconfiguration Event, it shall grant itself that Domain_ID. This action shall cause an EFP request to be generated as described in item “i“ below.

c) The Principal Switch shall then transmit an AAI SW_ILS request Sequence via all E_Ports. After receiving the SW_ACC reply, the Principal Switch may receive one or more RDI SW_ILS request Sequences via one or more of the E_Ports.

d) When the Principal Switch receives an RDI SW_ILS request Sequence with a non-zero requested Domain_ID, in the absence of any error condition preventing it, it shall allocate the requested Domain_ID(s) to the requesting Switch, if available. If the requested Domain_ID is not available or is zero, it shall grant an available Domain_ID to the requesting Switch. This Domain_ID is communicated to the Switch by transmitting the SW_ACC reply Sequence via the E_Port on which the corresponding RDI request Sequence was received.

e) The Principal Switch shall not grant the same Domain_ID to more than one requesting Switch.

f) If the Principal Switch receives an RDI request for the same requested Domain_ID as it granted to that Switch in a previous RDI request received after Principal Switch Selection, it shall not be considered an error; the Principal Switch shall grant the Domain_ID to the Switch. If a

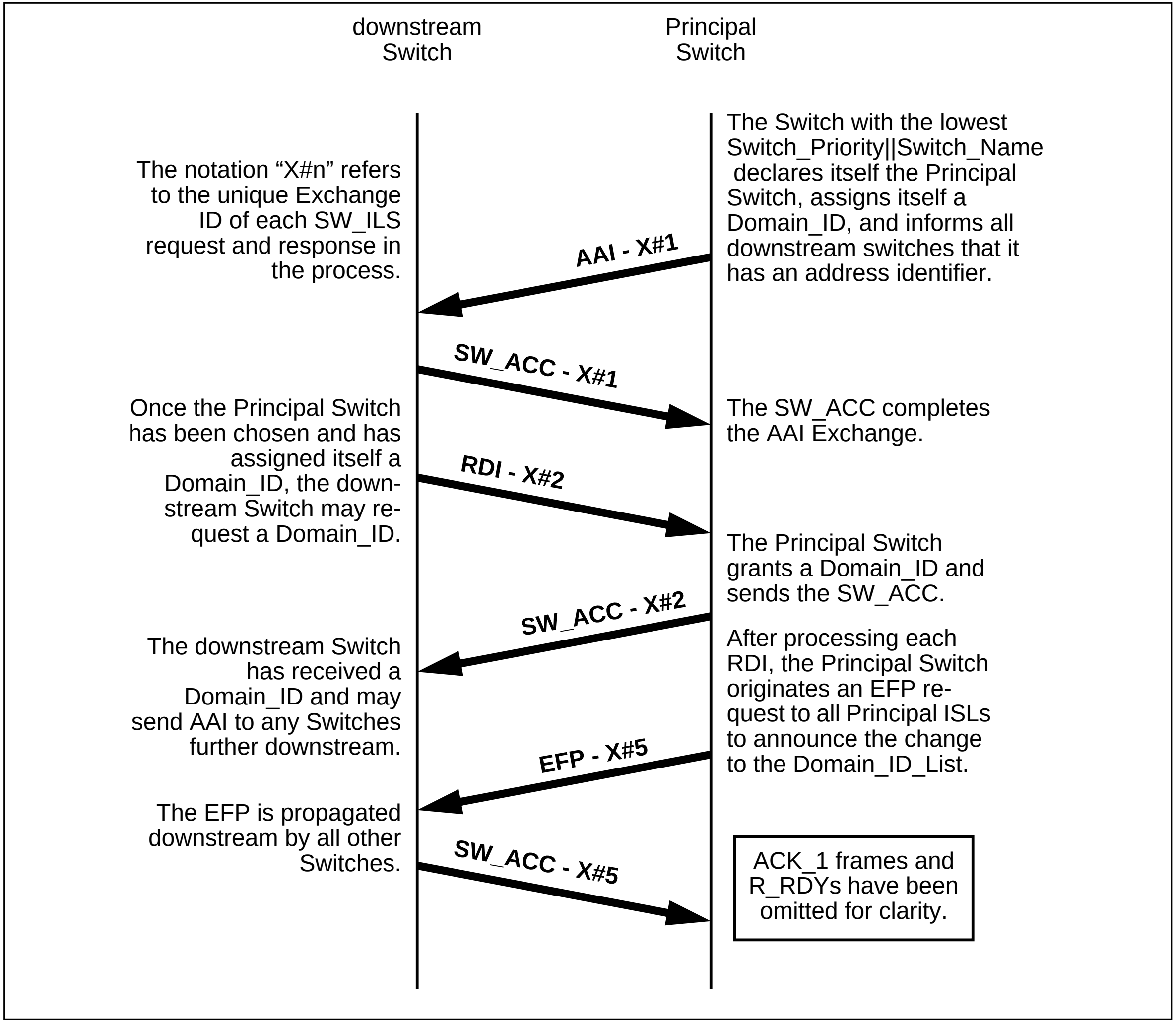
Switch that has already been granted a Domain_ID transmits a request to the Principal Switch for a different Domain_ID, the Principal Switch shall transmit BF or RCF, as appropriate.

g) If the Principal Switch receives an RDI request and no appropriate Domain_IDs are available, the Principal Switch shall return SW_RJT with a reason/explanation of: “Unable to perform command request”, “Domain_ID not available“.

h) All Principal ISLs via which the Principal Switch receives RDI requests shall be downstream Principal ISLs.

i) Each time the Principal Switch grants a Domain_ID to a Switch (including itself), it shall transmit an EFP SW_ILS request Sequence via all E_Ports, with each record in the Domain_ID_List corresponding to a granted Domain_ID set to the Switch_Name granted the Domain_ID. An example of this process is illustrated in Figure 12.

Figure 12 – RDI Request Processing by Principal Switch


downstream Switch
Principal Switch
The Switch with the lowest Switch_Priority||Switch_Name declares itself the Principal Switch, assigns itself a Domain_ID, and informs all downstream switches that it has an address identifier.
The notation “X#n” refers to the unique Exchange ID of each SW_ILS request and response in the process.
AAI - X#1
SW_ACC - X#1
The SW_ACC completes the AAI Exchange.
Once the Principal Switch has been chosen and has assigned itself a Domain_ID, the downstream Switch may request a Domain_ID.
RDI - X#2
The Principal Switch grants a Domain_ID and sends the SW_ACC.
SW_ACC - X#2
The downstream Switch has received a Domain_ID and may send AAI to any Switches further downstream.
After processing each RDI, the Principal Switch originates an EFP request to all Principal ISLs to announce the change to the Domain_ID_List.
EFP - X#5
The EFP is propagated downstream by all other Switches.
SW_ACC - X#5
ACK_1 frames and R_RDYs have been omitted for clarity.

7.3.2 Domain_ID Requests by the Switches

The Switches shall request a Domain_ID as follows:

a) At the completion of Principal Switch Selection, the Switch receives the AAI SW_ILS request Sequence via the upstream Principal ISL, and shall reply to the request with the appropriate SW_ACC or other response. The Switch shall set its Switch_Priority value to a value greater than hex'02'. An AAI request Sequence received on any other ISL shall be replied to with the appropriate SW_ACC or other response, but shall otherwise be ignored. The AAI request received via the upstream Principal ISL is the indication that the Principal Switch has assigned a Domain_ID to all Switches between the Principal Switch and the Switch receiving the AAI request.

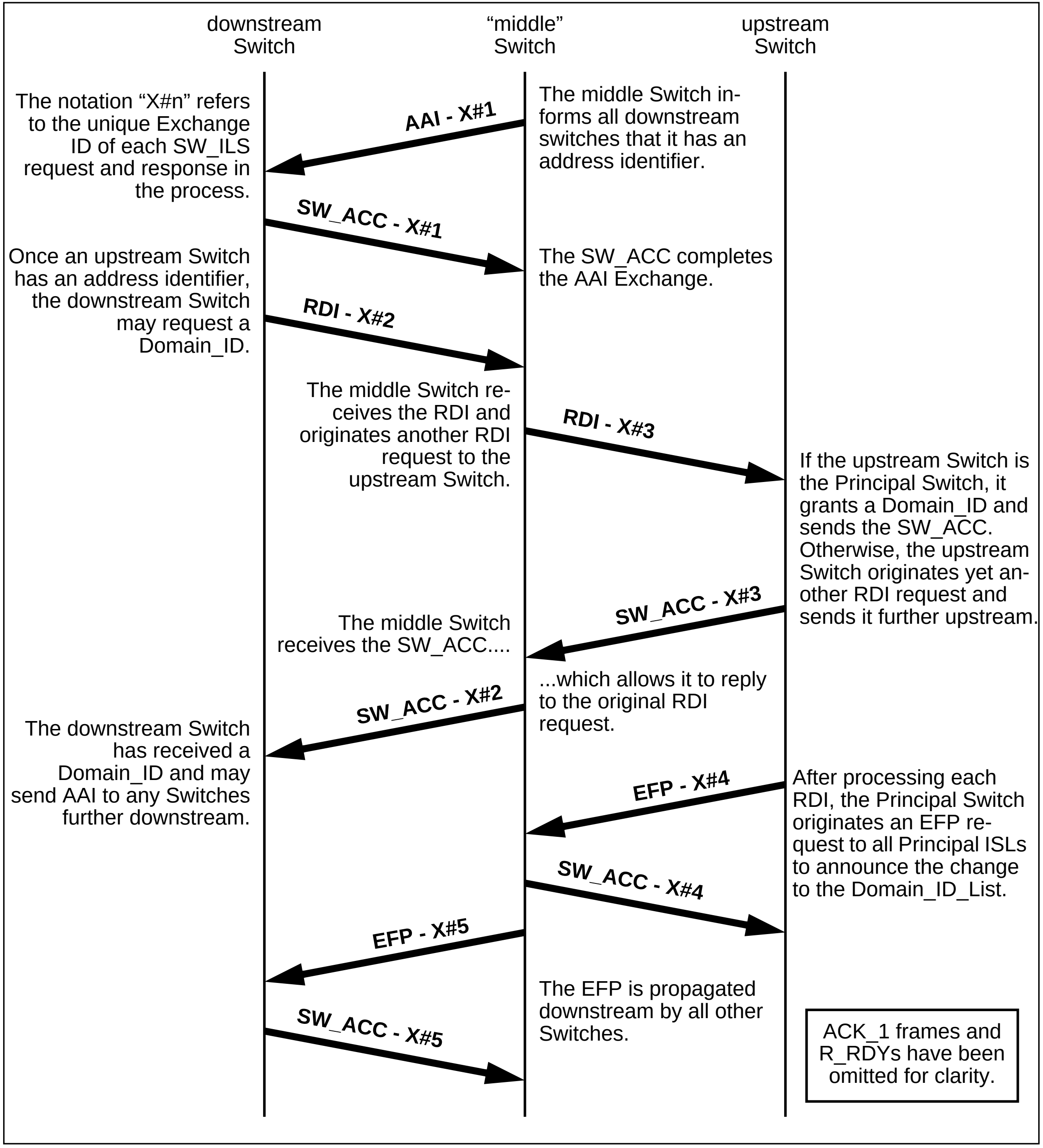
b) After transmitting an SW_ACC reply to the AAI request, the Switch shall transmit an RDI request Sequence via the upstream Principal ISL. When the Switch receives the reply SW_ACC to the RDI request, it shall assign address identifiers to all Ports within its Domain as appropriate.

c) After the Switch is granted a Domain_ID, it shall then transmit an AAI SW_ILS request Sequence via all ISLs other than the Principal ISL. After receiving the SW_ACC reply, the Switch may receive one or more RDI SW_ILS request Sequences from one or more of the E_Ports.

d) All Principal ISLs via which the Switch receives RDI requests shall be downstream Principal ISLs.

e) When the Switch receives an RDI request Sequence from one of its E_Ports via a downstream Principal ISL, it shall originate an RDI request Sequence with the same Payload via its upstream Principal ISL. When the reply SW_ACC is received via the upstream Principal ISL, it shall transmit an SW_ACC reply Sequence via the downstream Principal ISL on which the initial request was received. An example of this process is illustrated in Figure 13.

Figure 13 – RDI Request Processing by non-Principal Switch


downstream Switch
"middle" Switch
upstream Switch
The notation "X#n" refers to the unique Exchange ID of each SW_ILS request and response in the process.
AAI - X#1
The middle Switch informs all downstream switches that it has an address identifier.
SW_ACC - X#1
Once an upstream Switch has an address identifier, the downstream Switch may request a Domain_ID.
The SW_ACC completes the AAI Exchange.
RDI - X#2
The middle Switch receives the RDI and originates another RDI request to the upstream Switch.
RDI - X#3
If the upstream Switch is the Principal Switch, it grants a Domain_ID and sends the SW_ACC. Otherwise, the upstream Switch originates yet another RDI request and sends it further upstream.
SW_ACC - X#3
The middle Switch receives the SW_ACC....
...which allows it to reply to the original RDI request.
SW_ACC - X#2
The downstream Switch has received a Domain_ID and may send AAI to any Switches further downstream.
EFP - X#4
After processing each RDI, the Principal Switch originates an EFP request to all Principal ISLs to announce the change to the Domain_ID_List.
SW_ACC - X#4
EFP - X#5
The EFP is propagated downstream by all other Switches.
SW_ACC - X#5
ACK_1 frames and R_RDYs have been omitted for clarity.

7.4 E_Port and Fabric Isolation

An E_Port connected via an Inter-Switch Link to another E_Port may determine that it cannot communicate with the other E_Port for one of the reasons listed below.

a) The two E_Ports have incompatible Link Parameter requirements. For example, if one Switch has an E_D_TOV setting different than another, Class 2 frames sent by an N_Port on one Switch may not receive timely F_BSY responses from the other Switch. If an Isolated E_Port receives an RCF SW_ILS, it may retry Switch Port Initialization to see if better Link Parameters have become available.

b) Similarly, the two E_Ports have incompatible Fabric Parameter requirements. For example, if an E_Port receives an EFP that contains records it does not support, it shall Isolate.

c) The two E_Ports are a new Link between two existing Fabrics, and the Domain_ID allocations in each Fabric overlap. For example, if each existing Fabric had allocated Domain_ID hex'44' to a Switch, one Switch would have to give up its Preferred Domain_ID to reconfigure; this could cause a major disruption to current traffic.

d) The two E_Ports are a Link between Switches that are not capable of performing the Domain Address Manager function, and are each also not attached via an ISL to any other Switch capable of performing the Domain Address Manager function. Since no Switch can allocate Domain_IDs, no Class N frames can be sent between the Switches.

When any of the above conditions occurs, the E_Port shall Isolate itself from the other E_Port. The following is a list of appropriate Class F frames that may be communicated between Isolated E_Ports.

– An ELP SW_ILS request may be sent by an Isolated E_Port in an attempt to establish a working set of Link Parameters.

– An RCF SW_ILS request may be sent by an Isolated E_Port to begin a disruptive Reconfiguration in an attempt to build a single, non-isolated Fabric.

– An SW_ACC response may be sent in response to any of the above SW_ILS requests.

– An SW_RJT response may be sent in response to any of the above SW_ILS requests, if necessary, and shall be sent as the appropriate response to any other SW_ILS request not listed above.

The buffer-to-buffer credit between the Isolated E_Ports shall be a value of one; no alternate credit shall be in effect. No routing of Class N frames shall occur across the ISL.

If it is still desired to create a single Fabric via Isolated E_Ports, a Switch may override the Isolated condition by originating an RCF SW_ILS request Sequence via the appropriate ISL. The RCF shall force the selection of a single Principal Switch from within the previously Isolated Fabrics.

Annex A
(informative)

Future Projects

A.1 Switch Standards in development

At the time of publication of this standard, FC-SW-2 development had begun. FC-SW-2 is expected to standardize the following additional functions.

- path selection;
- ISL flow control management;
- distributed services;
- multicast.

Contact T11 for the latest status on FC-SW-2.

A.2 Switch Technical Reports in development

At the time of publication of this standard, no Switch Technical Reports were in development. Contact T11 for the latest status.

Annex B
(informative)

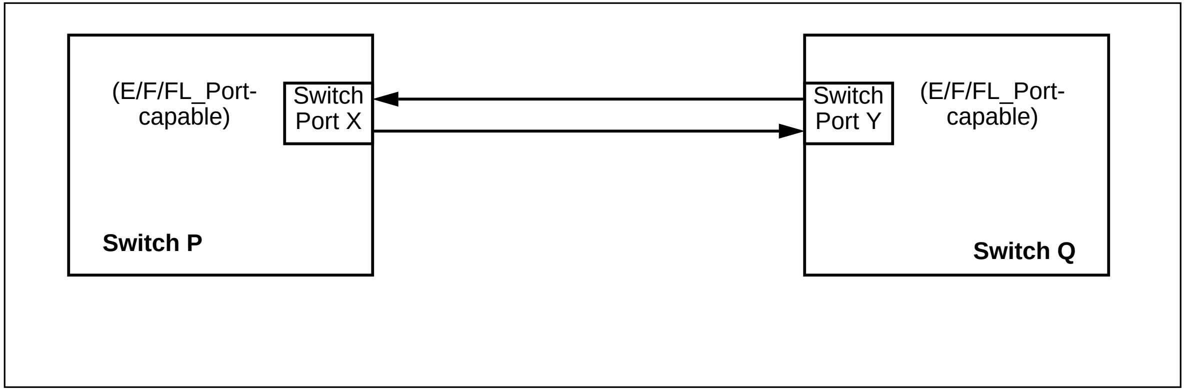
Examples of Switch Port Initialization

This annex presents some example scenarios that may occur during Switch Port Initialization (see 7.1). It is expected that the reader is familiar with Loop Initialization as defined in FC-AL, and with Link Initialization as defined in FC-PH.

B.1 Example 1: two E/F/FL_Port-capable Switch Ports

In this example, two Switch Ports that are E/F/FL_Port-capable are attached to each other. Figure B.1 illustrates this example.

Figure B.1 – Initialization example 1



According to the initialization algorithm, since each Switch Port is E/F/FL_Port-capable, they start the process with Loop Initialization, as defined in FC-AL. LIP Primitive Sequences are sent and received, and each Switch Port starts sending LISM frames. When Switch Port X receives LISM from Switch Port Y, it sees that its Port_Name is lower than the Port_Name in the Payload, and continues sending the same LISM.

On the other hand, when Switch Port Y receives LISM from Switch Port X, it sees that its Port_Name is higher than the Port_Name in the Payload. This causes Switch Port Y to start sending the LISM it received, with the Port_Name belonging to Switch Port X. Switch Port Y also transitions to the MONITORING state in non-participating mode, because only one FL_Port may be participating on a loop.

Switch Port X receives its LISM and assumes the role of Loop Master. Switch Port X then proceeds to send all of the other Loop Initialization Sequences, and by the end of Loop Initialization, discovers that it is the only L_Port on the loop. Because there may be a non-participating Switch Port on the Loop, Switch Port X knows it must attempt Link Initialization.

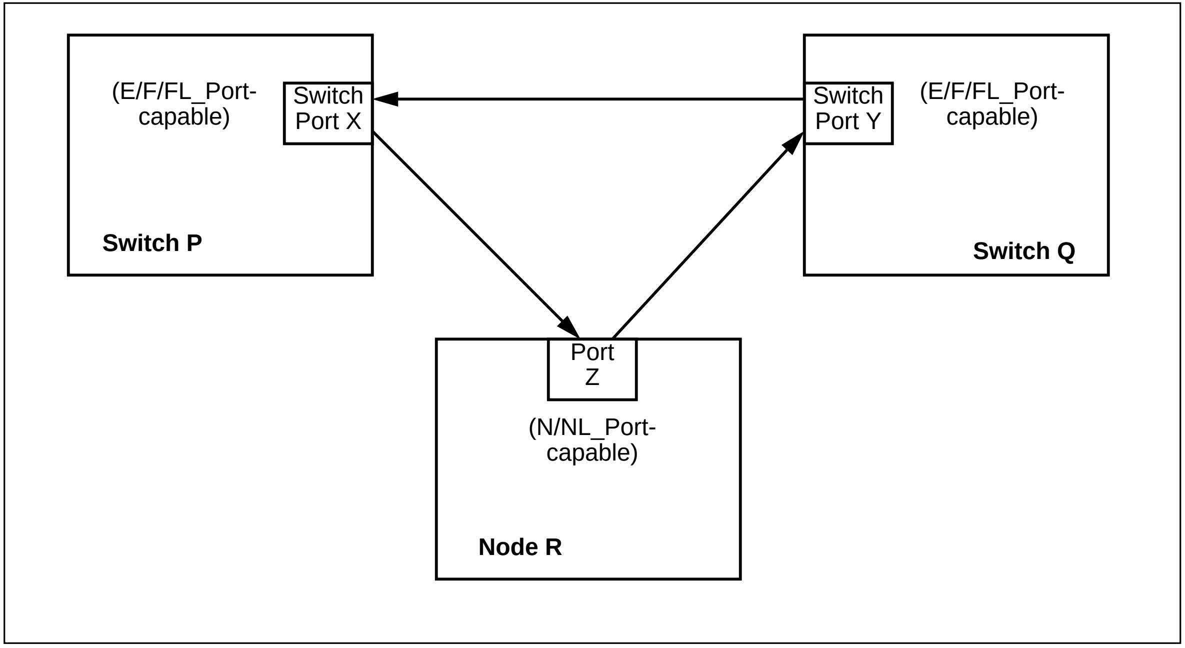
Switch Port X begins Link Initialization by sending the OLS Primitive Sequence. Switch Port Y needs to be looking for FC-PH Primitive Sequences (OLS, NOS, LR, LRR) to transition from the FL_Port operating mode to E/F_Port mode. The Link protocol continues to completion and a point-to-point Link is now Active.

Switch Port X and Switch Port Y may now attempt to Exchange Link Parameters and establish an Inter-Switch Link.

B.2 Example 2: two E/F/FL_Port-capable Switch Ports and one Nx_Port

In this example, two Switch Ports that are E/F/FL_Port-capable are attached to each other as in the first example, but there is also an Nx_Port on the loop. Figure B.2 illustrates this example.

Figure B.2 – Initialization example 2



According to the initialization algorithm, since each Switch Port is E/F/FL_Port-capable and Port Z is N/NL_Port-capable, they start the process with Loop Initialization, as defined in FC-AL. LIP Primitive Sequences are sent and received, and each Switch Port and the Nx_port start sending LISM frames. As in the first example, Switch Port X receives LISM from Switch Port Y, it sees that its Port_Name is lower than the Port_Name in the Payload, and continues sending the same LISM.

When Port Z receives the LISM from Switch Port X, it finds a D_ID of zero, meaning that the originator is an FL_Port. Since an FL_Port always wins Loop Master, the NL_Port continues sending the received LISM from Switch Port X.

When Switch Port Y receives Switch Port X's LISM from Port Z, it sees that its Port_Name is higher than the Port_Name in the Payload. This causes Switch Port Y to start sending the LISM it received, with the Port_Name belonging to Switch Port X. Switch Port Y also transitions to the MONITORING state in non-participating mode, because only one FL_Port may be participating on a loop.

Switch Port X receives its LISM and assumes the role of Loop Master. Switch Port X then proceeds to send all of the other Loop Initialization Sequences, and by the end of Loop Initialization, discovers that there is only one other L_Port on the loop. Because that one other port may be capable of point-to-point operation, Switch Port X knows it must attempt Link Initialization.

Switch Port X begins Link Initialization by sending the OLS Primitive Sequence. If Port Z recognizes the OLS and reacts to it, it will send LR in response. Switch Port Y is looking for FC-PH Primitive Sequences (OLS, NOS, LR, LRR) to transition from the FL_Port operating mode to E/F_Port mode, and therefore sends LRR in response to the received LR. Because receiving LRR in response to OLS is not part of the protocol, Switch Port X will continue to send OLS until it times out. The Link Initialization has failed.

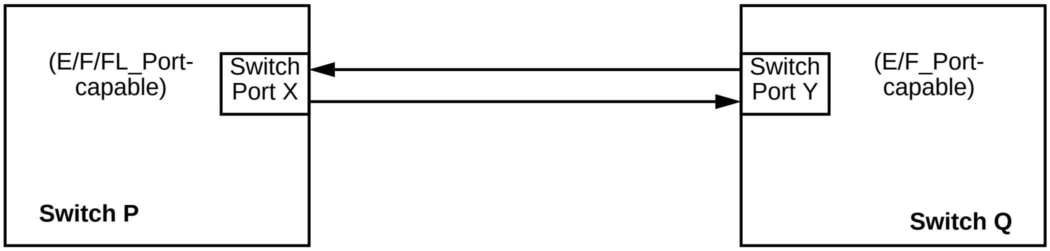
When Link Initialization fails, there is nothing to do other than to go back and re-do the Loop Initialization. The Loop Initialization succeeds, and Switch Port X operates as an FL_Port, and Port Z operates as an NL_Port. Switch Port Y stays non-participating until a system administrator comes to save it from oblivion.

Note that if Port Z had been bypassed, the process would have completed as in Example 1, because the Primitive Sequences would have been ignored by Port Z. At a later time, if Port Z then comes out of the bypass state, an Initialization Event occurs (Port Z starts sending LIP to get an AL_PA), and things sort themselves out as described for Example 2. If Switch Port Y had been bypassed, then Switch Port X would have become an F_Port in a point-to-point Link with N_Port Z.

B.3 Example 3: one E/F/FL_Port-capable Port and one E/F_Port-capable Port

In this example, a Switch Port that is E/F/FL_Port-capable is attached to a Switch Port that is E/F_Port-capable. Figure B.3 illustrates this example.

Figure B.3 – Initialization example 3



According to the initialization algorithm, the Switch Port that is E/F/FL_Port-capable starts the process with Loop Initialization as defined in FC-AL. However, the Switch Port that is E/F_Port-capable starts the process with Link Initialization as defined in FC-PH. Switch Port X sends LIP Primitive Sequences, and Switch Port Y sends OLS Primitive Sequences. This represents a stand-off, in which each Switch Port does not respond in the manner that the other is requesting.

Fortunately, Switch Port X will eventually timeout and attempt Link Initialization. When Switch Port X starts sending OLS, will respond with LR, and the Link protocol continues to completion and a point-to-point Link is now Active.

Switch Port X and Switch Port Y may now attempt to Exchange Link Parameters and establish an Inter-Switch Link.

Annex C
(informative)

Fibre Channel Link Switch Command Set for FC-AE

C.1 Scope

The purpose of this annex is to document a standardized command set for the control of the Fibre Channel “link switch”. This annex was provided by the FC-AE (Avionics Environment) working group of T11.

C.2 Technical Description

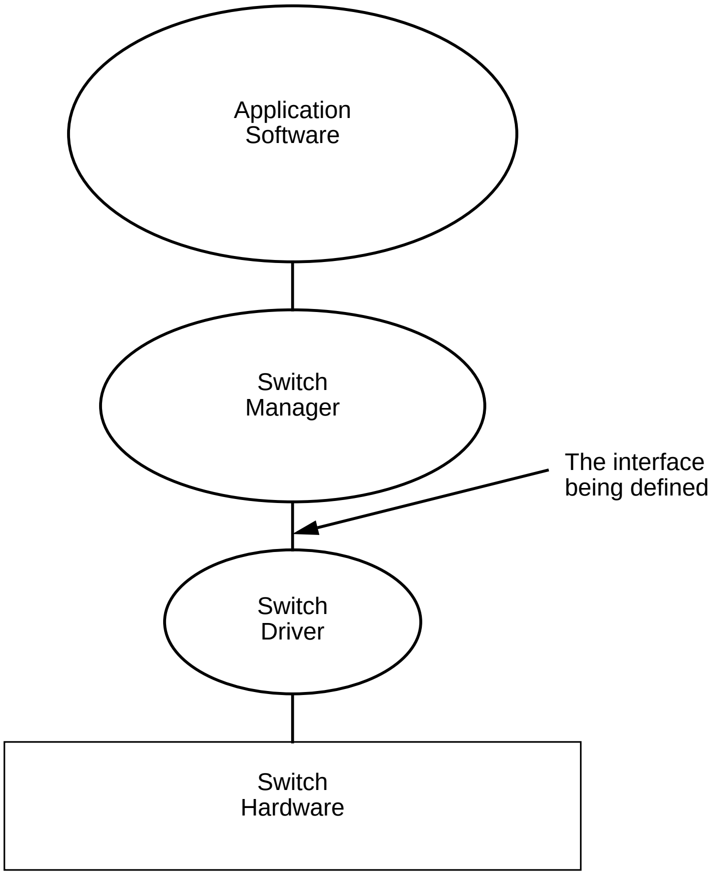
C.2.1 Definition

Link Switch: A unit, externally controlled, capable of rearranging and steering Fibre Channel links.

C.2.2 Context

Figure C.1 shows the software context for the command set described by this document; and Figure C.2 through Figure C.6 show examples of a system context. In these figures, it is intended to not require a great deal of intelligence on the part of the switch hardware, so that this could conceivably be constructed with hardware logic only; i.e., it is intended to allow for implementations that do not include a micro-controller as part of the link switch hardware.

Figure C.1 – Software Context for Link Switch Control



The command set described in C.2.3 is intended to be independent of the transmission medium that is used to transfer the command information to the hardware switch itself. Figure C.2 and Figure C.3 show examples of out-of-band link switch control. Figure C.2 shows an example in which this medium could be a serial (or parallel) point-to-point connection between the link switch control element and the link switch hardware. Figure C.3 shows an example in which this medium could be a parallel (or serial) multi-drop bus connection between the link switch hardware and multiple controlling entities that are parts of the same nodes that contain the serial ports that are to be switched. Figure C.4 shows an example in which the control information is passed over the same physical medium that is used for the transfer of the high-speed serial data that is being switched.

Figure C.2 – Example 1 of System Context With Out-of-Band Control

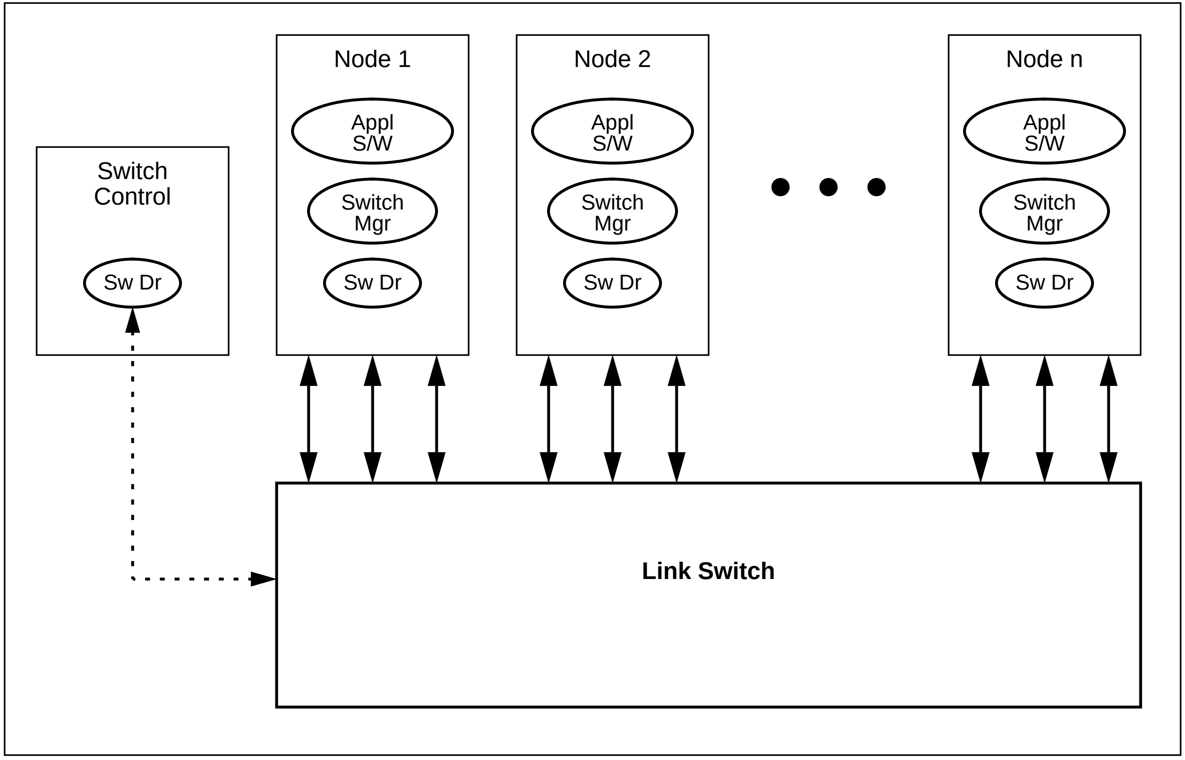


Figure C.3 – Example 2 of System Context With Out-of-Band Control

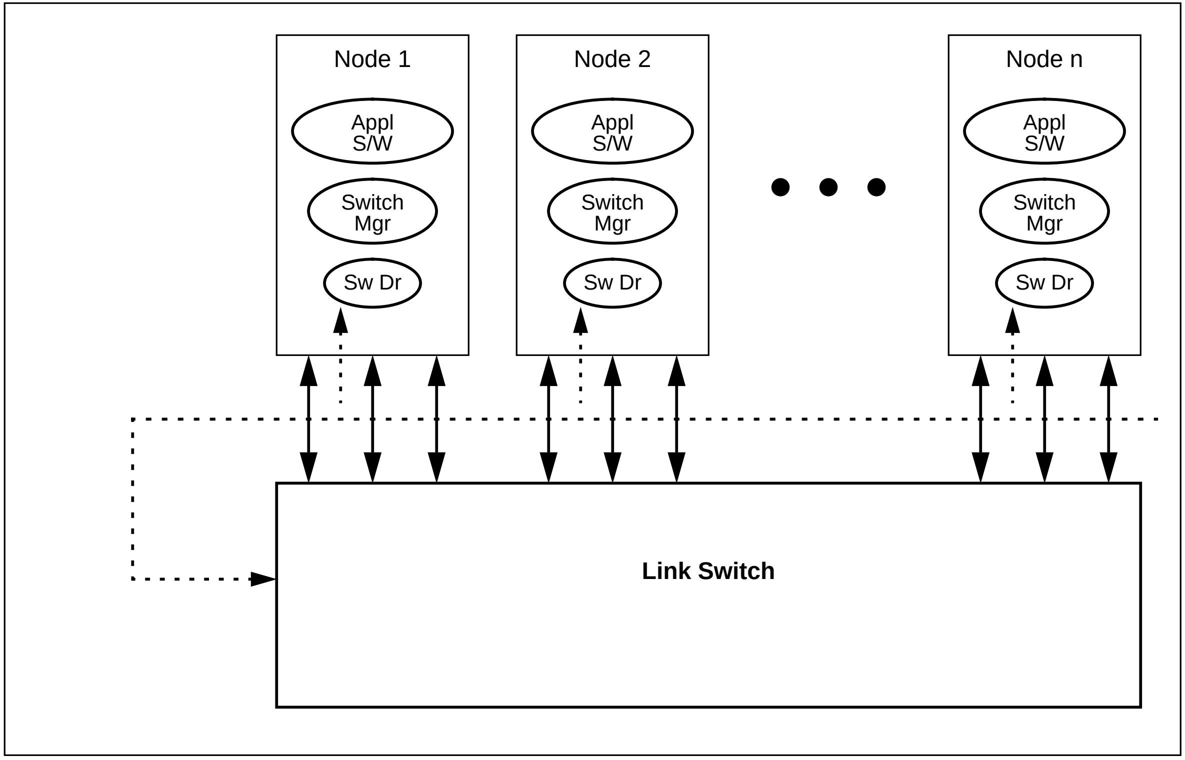
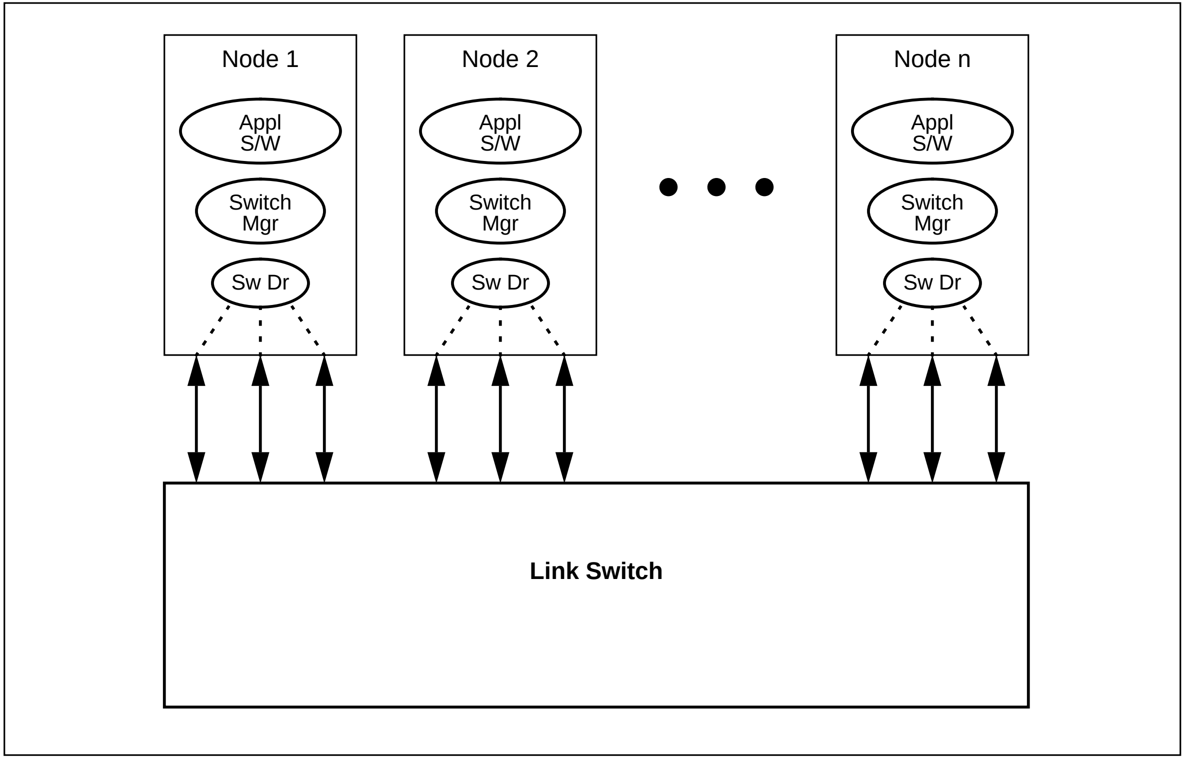


Figure C.4 – Example of System Context With in-Band Control



In all of the examples that show control of the link switch as being accomplished by a distributed switch manager/switch driver, it should be noted that the control issues for this configuration have not been resolved.

The command set of C.2.3 is intended to support system configurations in which multiple link switch elements are used, as illustrated in Figure C.5 and Figure C.6. In these systems, it is assumed that the system-level link switch topology is handled by the Switch Manager, not by the Switch Driver; and is therefore beyond the scope of this document. The Switch Manager (whether centralized or distributed) is responsible for resolving any conflicts over requests for switch connections.

Figure C.5 – Example 3 of System Context With Out-of-Band Control

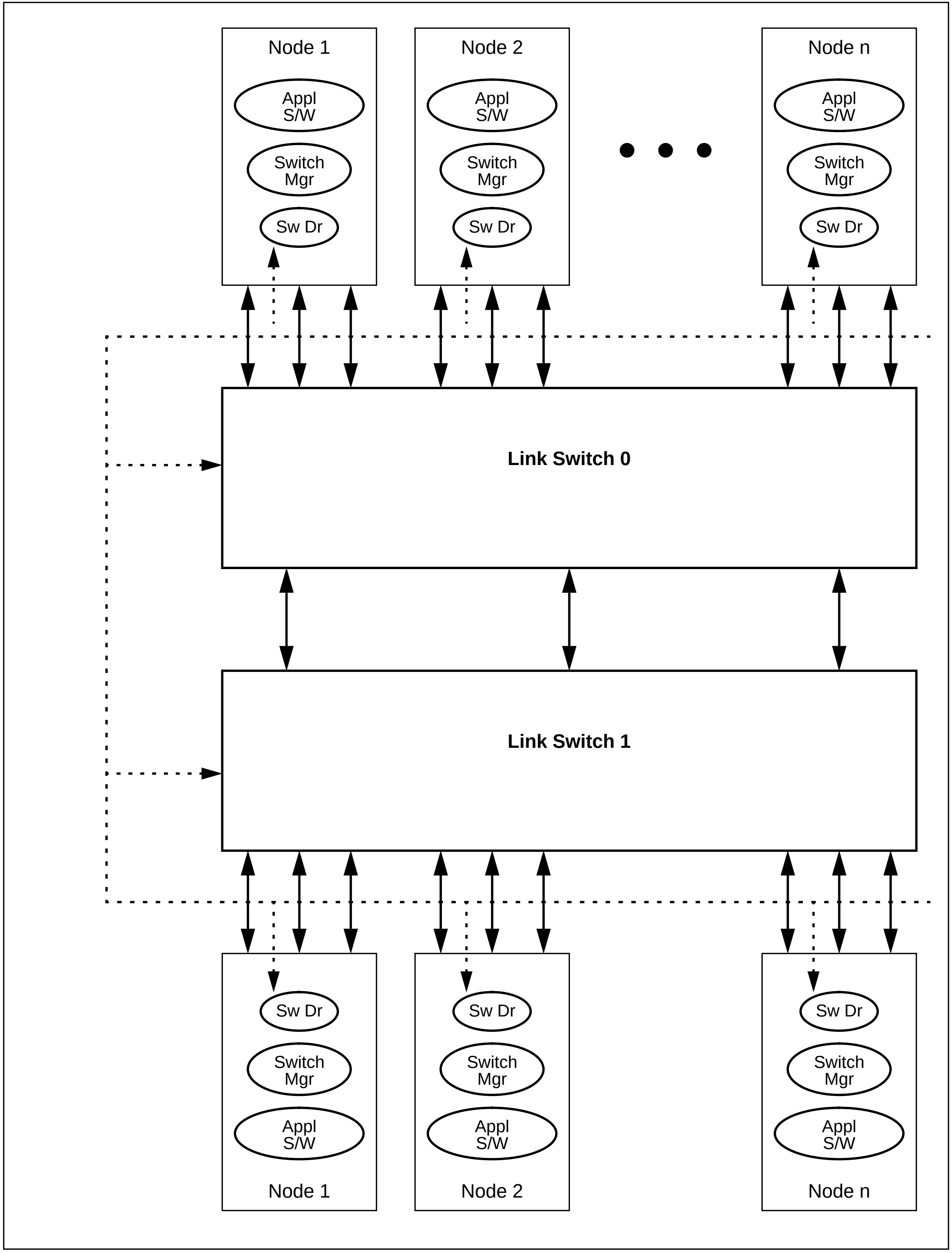
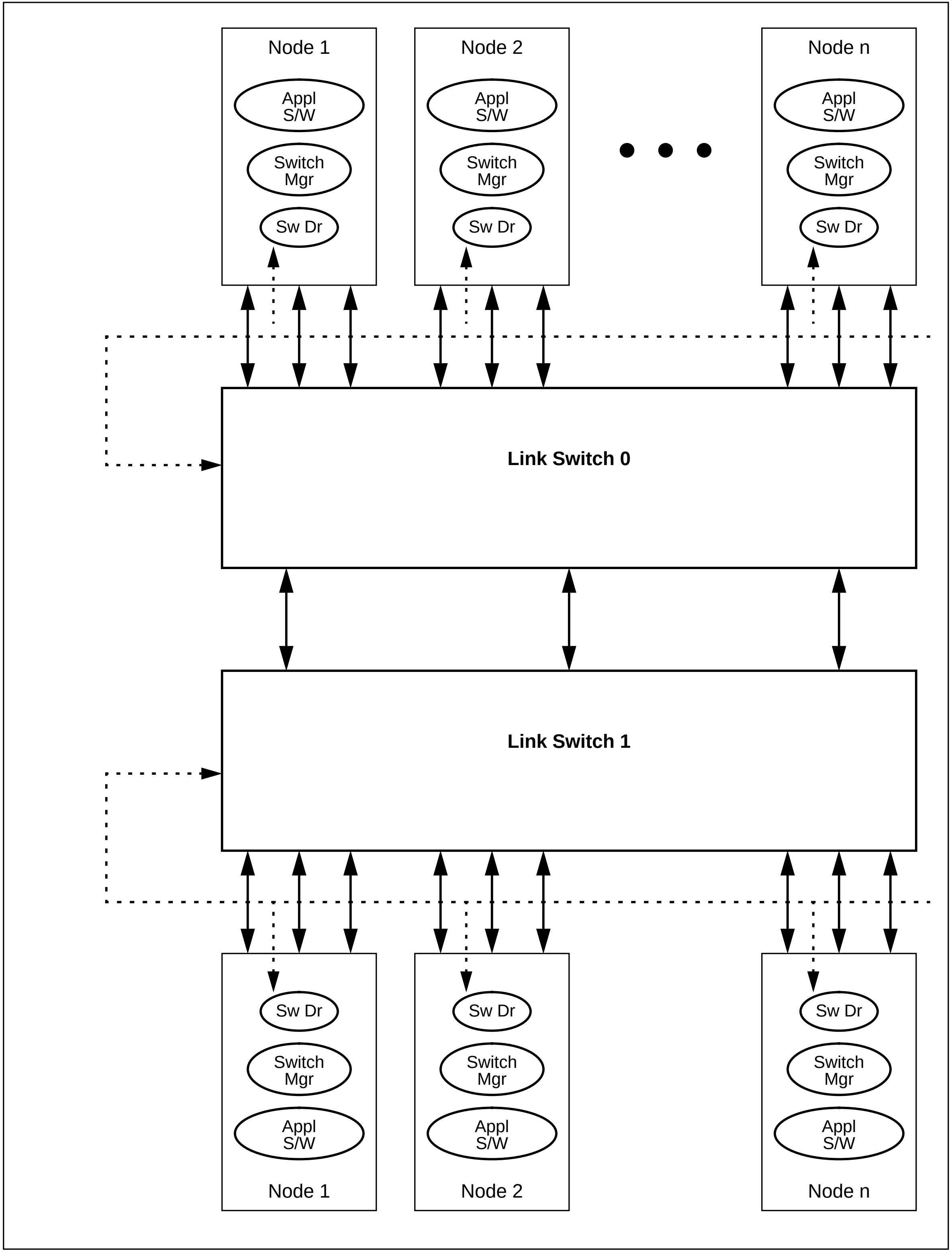


Figure C.6 – Example 4 of System Context With Out-of-Band Control



Conceptually, the link switch itself (or the switch driver, as appropriate) maintains a status indicator for each output port and each path within the link switch. This information is made available to the user as status information.

Depending on the link switch design, there may be more than one path available within the link switch for connecting a given output port to a given input port. No path identification parameters have been provided for in the following subprograms, since it was felt that these would be highly variable from switch design to switch design. Instead, it was assumed that the Switch Driver was free to use any available path to make the connection.

It was envisioned that some very low end link switches might require changing existing path connections in order to be able to establish a requested new connection. Because this would disrupt the existing connections, though, two separate procedures are provided -- one that allows existing paths to be modified, and one that does not.

Because the initial concept of the link switch was of a low-end, inexpensive switch; no built-in diagnostic capability was assumed for the switch itself. Instead, it was assumed that the link switch (hardware or associated Switch Driver) would maintain a marking for each path as to whether that path was usable. But it was assumed that a higher-level entity would indicate to the Switch Driver whether and when a path had been found to be unusable (by calling the appropriate procedure). This path health information would be maintained by the Switch Driver or the link switch hardware.

C.2.3 Command Set

This section contains a definition of the standardized driver services. The description uses Ada as the design language.

```
package LINK_SWITCH_DRIVER is

  --------------------------------------------------------------
  -- Definitions
  --------------------------------------------------------------

  -- INPUT : There are two possible points of view for the term
  --   “input”.  One is from the view of the link switch, and the
  --   from the view of an attached Fibre Channel N_PORT.  The
  --   default use of the term “input” in this package will be
  --   from the view of the link switch; ie, “input” refers to the
  --   “input” of the link switch.  In those cases where the view is
  --   intended to be from the N_PORT, the term “N_PORT” will be
  --   explicitly included, eg. N_PORT_INPUT.

  -- OUTPUT : Similarly to the term “input”, the default use of the
  --   term “output” will be from the view of the link switch; ie, “output”
  --   refers to the “output” of the link switch.  In those cases where
  --   the view is intended to be from the N_PORT, the term “N_PORT”
  --   will be explicitly included, eg. N_PORT_OUTPUT.

  -- PORT : The term “port” in this package is used to refer to
  --   an individual serial signal point of connection to a link switch.
  --   This is distinguished from the term “N_PORT”.

  -- INPUT PORT : Consistent with the preceding definitions, an
  --   input port is an individual serial signal point of connection
  --   to a link switch that carries information into the switch.  It does
  --   NOT refer to the signal line that carries information into
  --   an N_PORT.
-- OUTPUT PORT : Consistent with the preceding definitions, an
--   output port is an individual serial signal point of connection
--   to a link switch that carries information into the switch.  It does
--   NOT refer to the signal line that carries information out of
--   an N_PORT.

-- N_PORT : The term “N_PORT” in this package is used in the
--   Fibre Channel sense to refer to a pair of oppositely-directed
--   serial signal points of connection at an end node that
--   generates and receives Fibre Channel-compliant signals.



    ---------------------------------------------------------
    --                                                     --
    --           LINK SWITCH CONFIGURATION PARAMETERS      --
    --                                                     --
    ---------------------------------------------------------



--
-- The MAX_SWITCHES is to be set to the highest-valued address
-- accepted by the switch hardware.  It is assumed that the switch
-- addresses will range from 0 to (MAX_SWITCHES -1).
--
MAX_SWITCHES     : constant INTEGER := (configuration dependent);

--
-- The NUMBER_OF_INPUT_PORTS and NUMBER_OF_OUTPUT_PORTS are
-- configuration parameters for the system that contains the
-- link switch.  The NUMBER_OF_PORTS is also a configuration
-- parameter.  Its value is to be derived as follows:
--
--   NUMBER_OF_PORTS := max(NUMBER_OF_INPUT_PORTS, NUMBER_OF
--                          OUTPUT_PORTS)
--
-- It is assumed that the port numbers range from 0 to
-- (NUMBER_OF_INPUT_PORTS - 1), and 0 to (NUMBER_OF_OUTPUT_PORTS - 1),
-- respectively.
--
NUMBER_OF_INPUT_PORTS : constant INTEGER := (configuration dependent);
NUMBER_OF_OUTPUT_PORTS: constant INTEGER := (configuration dependent);
NUMBER_OF_PORTS       : constant INTEGER := (configuration dependent);

--
-- The MAX_N_PORTS parameter is to be set to the highest number of
-- N_PORTs that the user intends to define.
--
MAX_N_PORTS : constant INTEGER := (configuration dependent);
```

```
------------------------------------------------------------------
--                  Types and Constants                         --
------------------------------------------------------------------

--
-- In the PORT_TYPE, it is intended that the values
-- (0 .. (NUMBER_OF_PORTS - 1)) be used as identifiers of the ports
-- that are being switched; and the value (-1) be used when none of
-- the other values are appropriate (Eg. when a STATUS_OF function
-- wants to return a value that indicates that the port is
-- not currently connected to anything.)  (See also the NULL_PORT
-- constant.)
--
type PORT_TYPE is range -1 .. (NUMBER_OF_PORTS - 1);

--
-- It is intended that the NULL_PORT constant be used rather than the
-- literal value when referring to the PORT_TYPE value of (-1).
--
NULL_PORT : constant PORT_TYPE := -1;

subtype INPUT_PORT_TYPE  is PORT_TYPE
  range NULL_PORT .. (PORT_TYPE(NUMBER_OF_INPUT_PORTS)  - 1);
subtype OUTPUT_PORT_TYPE is PORT_TYPE
  range NULL_PORT .. (PORT_TYPE(NUMBER_OF_OUTPUT_PORTS) - 1);

--
-- In the N_PORT_TYPE, it is intended that the values
-- (0 .. (MAX_N_PORTS - 1)) be used as identifiers of the N_PORTS
-- that are being switched; and the value (-1) be used when none of
-- the other values are appropriate (Eg. when a STATUS_OF function
-- wants to return a value that indicates that the port is
-- not currently connected to anything.)  (See also the NULL_N_PORT
-- constant.)
--
type N_PORT_TYPE is range -1 .. (MAX_N_PORTS - 1);

--
-- It is intended that the NULL_N_PORT constant be used rather
-- than the literal value when referring to the N_PORT_TYPE value
-- of (-1).
--
NULL_N_PORT : constant N_PORT_TYPE := -1;

--
-- The SWITCH_ADDRESS_TYPE is used to distinguish among multiple
-- link switches in a system that are controlled via a common hardware
-- interface.  It is intended that the values
-- (0 .. (MAX_SWITCHES - 1)) be used as identifiers of the switches;
-- and the value (-1) be used when none of the other values are
-- appropriate.  (See also the NULL_SWITCH constant.)
--
type SWITCH_ADDRESS_TYPE is range -1 .. (MAX_SWITCHES - 1);
```

```
--
-- It is intended that the NULL_SWITCH constant be used rather
-- than the literal value when referring to the SWITCH_ADDRESS
-- _TYPE value of (-1).
--
NULL_SWITCH : constant SWITCH_ADDRESS_TYPE := -1;

--
-- The PORT_STATE_TYPE values refer to a designated (input or
-- output) port on the link switch.
--
type PORT_STATE_TYPE is

  (UNCONNECTED,  -- The designated link switch port is not currently
                 --  connected to any other port.  It is reachable
                 --  from at least some other port, but not
                 --  necessarily from all ports.
                 --
   CONNECTED,    -- The designated link switch port is currently
                 --  connected to another port.
                 --
   REACHABLE,    -- This status only applies in cases in which a
                 --  port of the opposite (input/output) type has
                 --  also been designated.  The two ports can be
                 --  connected without changing any existing
                 --  connections.
                 --
   REARRANGEABLE,-- This status only applies in cases in which a
                 --  port of the opposite (input/output) type has
                 --  also been designated.  The two ports can be
                 --  connected; but such a connection will require
                 --  changing some existing connections.
                 --
   UNREACHABLE,  -- This status only applies in cases in which a
                 --  port of the opposite (input/output) type has
                 --  also been designated.  The two ports cannot be
                 --  connected; but the subject port is still
                 --  REACHABLE from some port of the opposite type.
                 --
   ISOLATED);    -- The designated link switch port is not currently
                 --  connected to any other port; and cannot be
                 --  connected by any path in the link switch.  (This
                 --  is an indication that at least one failure
                 --  has occurred in the link switch.)


--
-- The N_PORT_STATE_TYPE values refer to a designated N_PORT
-- connected to the link switch.
--
type N_PORT_STATE_TYPE is

  (UNCONNECTED,  -- The designated N_PORT is not currently
                 --  connected to any other N_PORT.  It is reachable
```

```
              --  from at least some other N_PORT, but not
              --  necessarily from all N_PORTs.
              --
              --  Strictly speaking, this status implies that
              --  both the N_PORT_OUTPUT N_PORT_INPUT that have
              --  been DEFINEd to constitute the N_PORT are
              --  UNCONNECTED (in the PORT_STATE_TYPE
              --  sense).
              --
CONNECTED_PP, -- The designated N_PORT is currently
              --  connected to one other N_PORT in a point-to
              --  point configuration.
              --
              --  To be CONNECTED implies that both the
              --  N_PORT_OUTPUT AND_N_PORT_INPUT that have
              --  been DEFINEd to constitute the N_PORT
              --  are CONNECTED (in the PORT_STATE_TYPE
              --  sense) to the OUTPUT_PORT and INPUT_PORT,
              --  respectively, of another defined N_PORT.
              --
CONNECTED_LOOP,-- The designated N_PORT is currently
              --  connected to two other N_PORTs in a
              -- (potential) loop configuration.
              --
              --  To be CONNECTED implies that both the
              --  N_PORT_OUTPUT AND_N_PORT_INPUT that have
              --  been DEFINEd to constitute the N_PORT
              --  are CONNECTED (in the PORT_STATE_TYPE
              --  sense) to the OUTPUT_PORT and INPUT_PORT,
              --  respectively, of the other defined N_PORTs.
              --
              --  Even though the name is CONNECTED_LOOP, there
              --  is no guarantee that the loop is closed.  It
              --  is left up to the user to determine this.
              --
REACHABLE,    -- This status only applies in cases in which one
              --  or two other N_PORT have also been designated.
              --  If one other N_PORT is designated, this value
              --  indicates that the two N_PORTs can be connected
              --  to each other in a point-to-point configuration
              --  without affecting any existing connections.
              --  If two other N_PORTs are designated, this value
              --  indicates that the N_PORTs can be connected to
              --  each other in a loop configuration.
              --
REARRANGEABLE,-- This status only applies in cases in which one
              --  or two other N_PORT have also been designated.
              --  The meanings are similar to those for the
              --  REACHABLE status value; except that one or more
              --  existing connections will have to be changed in
              --  order to establish the indicated configuration.
              --
ISOLATED,     -- The designated N_PORT is not currently
              --  connected to any other N_PORT; and cannot be
```

```
                  --  connected by any path in the link switch.  (This
                  --  is an indication that at least one failure
                  --  has occurred in the link switch.)
                  --
   UNDEFINED,     -- One of the N_PORTs has not been DEFINEd.
                  --
   OTHER);        -- This value is used to cover any situation not
                  --  specifically included in any of the other
                  --  status values.  This would include such
                  --  situations as where the N_PORT_INPUT of the
                  --  subject N_PORT is connected to the N_PORT_OUTPUT
                  --  of another N_PORT, but its N_PORT_OUTPUT is
                  --  unconnected.

--
-- The N_PORT_INFO_TYPE is used to carry information about the
-- setting of the link switch connected to a designated N_PORT.  The
-- SWITCH component indicates to which link switch the N_PORT is
-- connected; and the N_PORT_OUTPUT and N_PORT_INPUT components
-- indicate to which input port and output port on that link switch
-- the N_PORT is connected.
--
-- If the designated N_PORT is CONNECTED_PP, both the
-- UPSTREAM and the DOWNSTREAM components are set to the
-- value of the connected N_PORT.
--
-- If the designated N_PORT is CONNECTED_LOOP, the UPSTREAM
-- component indicates the N_PORT whose output is connected to
-- the designated N_PORT’s input port; and the DOWNSTREAM
-- component indicates the N_PORT whose input is connected to
-- the designated N_PORT’s output port.
--
type N_PORT_INFO_TYPE is record
  N_PORT_OUTPUT : INPUT_PORT_TYPE     := NULL_PORT;
  N_PORT_INPUT  : OUTPUT_PORT_TYPE    := NULL_PORT;
  SWITCH        : SWITCH_ADDRESS_TYPE := NULL_SWITCH;
  STATE         : N_PORT_STATE_TYPE   := UNDEFINED;
  UPSTREAM      : N_PORT_TYPE         := NULL_N_PORT;
  DOWNSTREAM    : N_PORT_TYPE         := NULL_N_PORT;
end record;

type N_PORT_SWITCH_STATUS_TYPE is array (0 .. (MAX_N_PORTS - 1))
  of N_PORT_INFO_TYPE;

type OUTPUT_STATUS_ARRAY_TYPE
  is array (0 .. (NUMBER_OF_OUTPUT_PORTS - 1))
    of PORT_STATE_TYPE;

--
-- The INPUT_PORT_STATUS_TYPE record contains information about
-- a designated input port on a designated link switch.
--
-- The STATE array contains the status of each output port on the
-- designated link switch relative to the designated input port.  The
-- index to the array is the output port number.
--
-- If the designated port has been defined to be part of an N_PORT,
-- that N_PORT is indicated in the PARENT component.  Otherwise
-- that component contains the NULL_N_PORT value.
--
type INPUT_PORT_STATUS_TYPE
  is record
    STATE  : OUTPUT_STATUS_ARRAY_TYPE;
    PARENT : N_PORT_TYPE      := NULL_N_PORT;
  end record;

--
-- The OUTPUT_PORT_STATUS_TYPE record contains information about
-- a designated output port on a designated link switch.
--
-- If the designated port is CONNECTED to an input port on the
-- link switch, that port is indicated in the OUTPUT_PORT component of
-- the record.  Otherwise that component contains the NULL_PORT
-- value.
--
-- If the designated port has been defined to be part of an N_PORT,
-- that N_PORT is indicated in the PARENT component.  Otherwise
-- that component contains the NULL_N_PORT value.
--
type OUTPUT_PORT_STATUS_TYPE
  is record
    STATE      : PORT_STATE_TYPE;
    INPUT_PORT : INPUT_PORT_TYPE;
    PARENT     : N_PORT_TYPE      := NULL_N_PORT;
  end record;

--
-- The SUCCESS_TYPE is used to indicate the status of a call to
-- the various subprograms in this package.
--
type SUCCESS_TYPE is (UNSUCCESSFUL, SUCCESSFUL);


--
-- The SWITCH_STATUS_TYPE contains information regarding the status
-- of a specific output port on the link switch.  (The identification of
-- which output port is involved is implied by the context -- see
-- the definition of specific subprograms for details.)
--
-- When the OUTPUT_STATUS component of the record indicates that
-- the output port is CONNECTED, REACHABLE, REARRANGEABLE, or
-- UNREACHABLE, the INPUT_PORT component contains the number of
-- the associated input port.  When the OUTPUT_STATUS component
-- indicates that the output port is UNCONNECTED or ISOLATED,
-- the INPUT_PORT component is not meaningful and may contain
-- an arbitrary value.
--
type SWITCH_STATUS_TYPE
```

```
    is record
      OUTPUT_STATUS : OUTPUT_PORT_STATUS_TYPE;
      INPUT_PORT    : INPUT_PORT_TYPE;
    end record;

  type INPUT_ARRAY_TYPE is array (0 .. (NUMBER_OF_INPUT_PORTS - 1))
    of INPUT_PORT_STATUS_TYPE;

  type OUTPUT_ARRAY_TYPE is array (0 .. (NUMBER_OF_OUTPUT_PORTS - 1))
    of OUTPUT_PORT_STATUS_TYPE;

  --
  -- The BASIC_SWITCH_STATUS_TYPE record is used to carry information
  -- about the overall settings of a designated link switch.  The view of
  -- the link switch is that of individual input ports and output ports,
  -- not of N_PORTs.
  --
  -- The INPUT component contains information about the state of each
  -- of the input ports on the link switch; and the OUTPUT component
  -- contains information about the state of each of the output ports
  -- on the link switch.
  --
  type BASIC_SWITCH_STATUS_TYPE is record
    INPUT  : INPUT_ARRAY_TYPE;
    OUTPUT : OUTPUT_ARRAY_TYPE;
  end record;



  ------------------------------------------------------------------
  --                      Subprograms                             --
  ------------------------------------------------------------------

    --------------------------------------------------------------
    --          Note Regarding Use of Name Overloading          --
    --                                                          --
    -- The Ada programming language allows for the overloading  --
    -- of subprogram names (ie, having more than one subprogram --
    -- with the same name).  It is only required that the       --
    -- compiler be able to unambiguously determine which sub-   --
    -- program call is intended by looking at the name, plus    --
    -- such things as the number, types, and names of the       --
    -- parameters associated with the call.                     --
    --                                                          --
    -- Overloading is used in the following subprogram          --
    -- definitions.  Comments will be used to help the reader   --
    -- understand the differences among the overloaded names.   --
    --                                                          --
    --------------------------------------------------------------

  --
  -- The INITIALIZE procedure prepares the SWITCH for operation,
  -- starting from any link switch condition, including initial power on.
  -- If default connections are part of the link switch design, these
```

```
-- default connections will be in place following the execution
-- of this procedure.  If the link switch design includes redundant
-- paths, all paths are marked as being usable.
--
procedure INITIALIZE (SWITCH : in SWITCH_ADDRESS_TYPE := 0);



--
-- The RESET procedure performs a subset of the actions taken by
-- the INITIALIZE procedure.  Specifically, the RESET procedure
-- only establishes any default connections that are part of the
-- link switch design.  This procedure does not modify any path usability
-- information that may be stored with the SWITCH.
--
procedure RESET (SWITCH : in SWITCH_ADDRESS_TYPE := 0);

--
-- The DEFINE procedure is used to create an association between
-- an input/output pair of ports on the link switch and a variable of
-- the type N_PORT_TYPE.  It is intended that this pair of ports
-- be connected in the system to a common Fibre Channel N_PORT.
-- Once it is defined, the N_PORT_TYPE variable can be used in
-- the appropriate procedure calls to more easily manipulate
-- connections to Fibre Channel N_PORTs.
--
-- CAUTION : Because the focus in this procedure is on the N_PORT,
-- the terms “input” and “output” are being used in two different
-- senses in the procedure definition.  Note that the N_PORT_OUTPUT
-- is actually an INPUT_PORT_TYPE (since the N_PORT output is
-- connected to an input of the link switch).  Similarly, the
-- N_PORT_INPUT is actually an OUTPUT_PORT_TYPE.
--
-- If neither the N_PORT_OUTPUT nor the AND_N_PORT_INPUT are
-- currently included in the definition of a different N_PORT,
-- the requested association is made and the SUCCESSFUL value
-- is returned as the STATUS.  Otherwise the requested
-- association is not made, and the UNSUCCESSFUL value is
-- returned as the STATUS.  Further information about the
-- reason for the lack of success may be obtained by means of
-- one of the STATUS_OF functions.
--
procedure DEFINE (N_PORT           : in     N_PORT_TYPE;
                  N_PORT_OUTPUT    : in     INPUT_PORT_TYPE;
                  AND_N_PORT_INPUT : in     OUTPUT_PORT_TYPE;
                  ON_SWITCH        : in     SWITCH_ADDRESS_TYPE;
                  STATUS           :    out SUCCESS_TYPE);

--
-- For the link switch indicated by the ON_SWITCH parameter, procedure
-- CONNECT causes the port indicated by the OUTPUT parameter to
-- be connected to the port indicated by the TO_INPUT
-- parameter.  If the requested connection is successfully made,
-- the STATUS parameter returns the SUCCESSFUL value.
--
```

```
-- If the requested connection is not made, it returns the
-- UNSUCCESSFUL value in the STATUS parameter.  Further information
-- regarding the reason for the lack of success can be obtained
-- by use of one of the STATUS_OF functions.
--
-- This procedure will not modify any existing connections in an
-- attempt to make the requested connection.  Any link switch outputs
-- that are currently connected to the designated TO_INPUT will
-- remain connected.
--
--       CONNECT SWITCH OUTPUT (POSSIBLY MULTIPLE) TO INPUT
--
procedure CONNECT (OUTPUT    : in     OUTPUT_PORT_TYPE;
                   TO_INPUT  : in     INPUT_PORT_TYPE;
                   ON_SWITCH : in     SWITCH_ADDRESS_TYPE := 0;
                   STATUS    :    out SUCCESS_TYPE);

--
-- The CONNECT_SINGLE procedure is identical to the preceding
-- CONNECT procedure (CONNECT SWITCH OUTPUT (POSSIBLY MULTIPLE)
-- TO INPUT), except that the CONNECT_SINGLE procedure will return
-- an UNSUCCESSFUL STATUS if the designated TO_INPUT on the link switch
-- is already connected to some other output.  When the UNSUCCESSFUL
-- status is returned, the link switch remains unchanged.
--
--       CONNECT A SINGLE SWITCH OUTPUT TO INPUT
--
procedure CONNECT_SINGLE (OUTPUT    : in     OUTPUT_PORT_TYPE;
                          TO_INPUT  : in     INPUT_PORT_TYPE;
                          ON_SWITCH : in     SWITCH_ADDRESS_TYPE := 0;
                          STATUS    :    out SUCCESS_TYPE);

--
-- The following CONNECT procedure will attempt to accomplish the
-- link switch settings necessary to connect the N_PORT and the
-- TO_N_PORT in a point-to-point configuration.  It is intended
-- only for connections on a single link switch.  (Hint:  for purposes
-- of this package, inter-switch connections could be defined as
-- pseudo-N_PORTs.)
--
-- If the connection is made, the SUCCESSFUL value is returned in
-- the STATUS parameter.  Otherwise the UNSUCCESSFUL value is
-- returned.  Further information regarding the reason for the
-- lack of success can be obtained by use of one of the STATUS_OF
-- functions.

--
-- This procedure will not modify any existing connections in an
-- attempt to make the requested connection.
--
--       CONNECT N_PORTS POINT-TO-POINT
--
procedure CONNECT (N_PORT    : in     N_PORT_TYPE;
                   TO_N_PORT : in     N_PORT_TYPE;
                   STATUS    :    out SUCCESS_TYPE);

--
-- The following CONNECT procedure will attempt to accomplish
-- the link switch settings necessary to connect the N_PORT input
-- to the N_PORT output of the TO_UPSTREAM_PORT; and the
-- N_PORT output to the N_PORT input of the AND_DOWNSTREAM_PORT.
--
-- If the connection is made, the SUCCESSFUL value is returned in
-- the STATUS parameter.  Otherwise the UNSUCCESSFUL value is
-- returned.  Further information regarding the reason for the
-- lack of success can be obtained by use of one of the STATUS_OF
-- functions.
--
-- This procedure will not modify any existing connections in an
-- attempt to make the requested connection.
--
--        CONNECT N_PORTS IN LOOP
--
procedure CONNECT (N_PORT              : in     N_PORT_TYPE;
                   TO_UPSTREAM_PORT    : in     N_PORT_TYPE;
                   AND_DOWNSTREAM_PORT : in     N_PORT_TYPE;
                   STATUS              :    out SUCCESS_TYPE);



--
-- The REARRANGE_TO_CONNECT procedures are the same as the
-- corresponding CONNECT procedures except that the
-- REARRANGE_TO_CONNECT procedures will modify existing
-- connections if necessary in order to successfully make
-- the requested connection.
--
-- Note that some link switch designs might not be able to make the
-- requested connection even with rearrangement of existing
-- connections.
--
--        REARRANGE TO CONNECT SWITCH OUTPUT TO INPUT
--
procedure REARRANGE_TO_CONNECT
  (OUTPUT    : in     OUTPUT_PORT_TYPE;
   TO_INPUT  : in     INPUT_PORT_TYPE;
   STATUS    :    out SWITCH_STATUS_TYPE;
   ON_SWITCH : in     SWITCH_ADDRESS_TYPE := 0);

--
--        REARRANGE TO CONNECT N_PORTS IN LOOP
--
procedure REARRANGE_TO_CONNECT
  (N_PORT              : in     N_PORT_TYPE;
   TO_UPSTREAM_PORT    : in     N_PORT_TYPE;
   AND_DOWNSTREAM_PORT : in     N_PORT_TYPE;
   STATUS              :    out SUCCESS_TYPE);

--
```

```
--        REARRANGE TO CONNECT N_PORTS POINT-TO-POINT
--
procedure REARRANGE_TO_CONNECT
  (N_PORT    : in     N_PORT_TYPE;
   TO_N_PORT : in     N_PORT_TYPE;
   STATUS    :    out SUCCESS_TYPE);

--
-- If the current state of the link switch (indicated by the ON_SWITCH
-- parameter) shows that the TO_OUTPUT port is, in fact, marked
-- as being in use and connected to the FROM_INPUT port, the
-- following RELEASE_CONNECTION procedure causes the output port to
-- be marked as unconnected.  In this case, the SUCCESSFUL value
-- is returned in the STATUS parameter.
--
-- Otherwise, the state of the link switch is left unchanged, and the
-- UNSUCCESSFUL value is returned in the STATUS parameter.  Further
-- information regarding the reason for the lack of success can be
-- obtained by calling one of the STATUS_OF functions.
--
--        RELEASE CONNECTION TO SWITCH OUTPUT, CONDITIONALLY
--
procedure RELEASE_CONNECTION
  (TO_OUTPUT  : in     OUTPUT_PORT_TYPE;
   FROM_INPUT : in     INPUT_PORT_TYPE;
   ON_SWITCH  : in     SWITCH_ADDRESS_TYPE := 0;
   STATUS     :    out SUCCESS_TYPE);



--
-- If the current state of the TO_OUTPUT port is CONNECTED, it is
-- marked as UNCONNECTED.  Otherwise the state of the link switch is
-- left unchanged.
--
--        RELEASE CONNECTION TO SWITCH OUTPUT, UNCONDITIONALLY
--
procedure RELEASE_CONNECTION
  (TO_OUTPUT  : in     OUTPUT_PORT_TYPE;
   ON_SWITCH  : in     SWITCH_ADDRESS_TYPE := 0);

--
-- If the current state of the link switch indicates that the FROM_N_PORT
-- is CONNECTED_PP to the TO_N_PORT, both N_PORTs are marked as
-- UNCONNECTED; and the SUCCESSFUL status is returned.
--
-- Otherwise, the state of the link switch is left unchanged; and the
-- UNSUCCESSFUL status is returned.
--
--        RELEASE POINT-TO-POINT CONNECTION, CONDITIONALLY
--
procedure RELEASE_CONNECTION
  (FROM_N_PORT : in     N_PORT_TYPE;
   TO_N_PORT   : in     N_PORT_TYPE;
   STATUS      :    out SUCCESS_TYPE);
```

```
--
-- If the current state of the link switch indicates that the FROM_N_PORT
-- is CONNECTED_LOOP between the TO_UPSTREAM_PORT AND_DOWNSTREAM_PORT,
-- the following RELEASE_CONNECTION procedure causes the FROM_N_PORT
-- to be marked as UNCONNECTED; and both the TO_UPSTREAM
-- _PORT AND_DOWNSTREAM_PORT to be marked as appropriate.  (They
-- may be marked as UNCONNECTED if their other constituent port is
-- also UNCONNECTED; or they may be marked as OTHER if their other
-- constituent port is still connected to another N_PORT.)
--
--        RELEASE LOOP CONNECTION, CONDITIONALLY
--
procedure RELEASE_CONNECTION
  (FROM_N_PORT         : in     N_PORT_TYPE;
   TO_UPSTREAM_PORT    : in     N_PORT_TYPE;
   AND_DOWNSTREAM_PORT : in     N_PORT_TYPE;
   STATUS              :    out SUCCESS_TYPE);

--
-- The RELEASE_CONNECTIONS procedure ensures that both the
-- N_PORT_INPUT and the N_PORT_OUTPUT of the TO_N_PORT are marked
-- as UNCONNECTED to any other port of the link switch.  This results
-- in the TO_N_PORT as being marked as UNCONNECTED.  This may
-- result in the modification of the status of other INPUT_PORTs,
-- OUTPUT_PORTs, and/or N_PORTs.
--
--
--        RELEASE N_PORT CONNECTIONS UNCONDITIONALLY
--
procedure RELEASE_CONNECTIONS
  (TO_N_PORT : in     N_PORT_TYPE);

--
-- The following STATUS_OF function returns information about the
-- designated individual link switch OUTPUT_PORT.
--
--       STATUS OF SWITCH OUTPUT
--
function STATUS_OF (OUTPUT_PORT : in     OUTPUT_PORT_TYPE;
                    ON_SWITCH   : in     SWITCH_ADDRESS_TYPE := 0)
         return OUTPUT_PORT_STATUS_TYPE;

--
-- The following STATUS_OF function returns information about the
-- designated individual link switch INPUT_PORT.
--
--       STATUS OF SWITCH INPUT
--
function STATUS_OF (INPUT_PORT  : in     INPUT_PORT_TYPE;
                    ON_SWITCH   : in     SWITCH_ADDRESS_TYPE := 0)
         return INPUT_PORT_STATUS_TYPE;

--
```

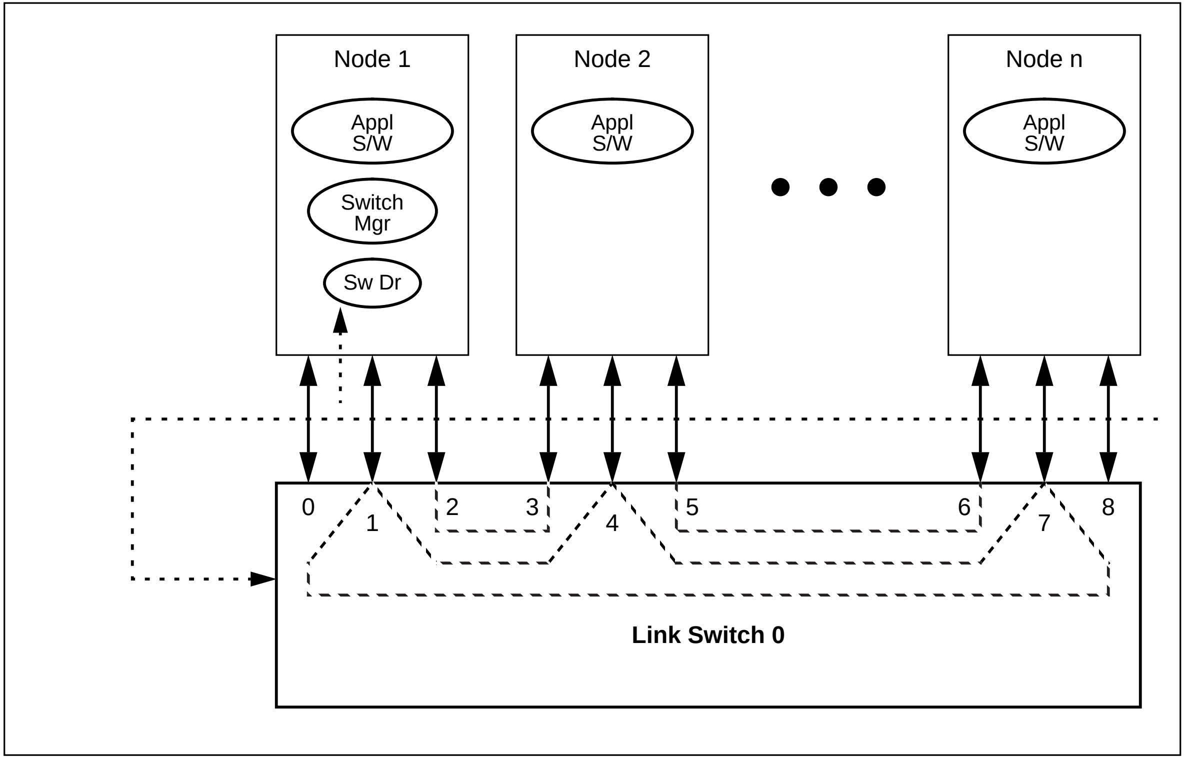
```
-- The following STATUS_OF function returns information about the
-- link switch connections to the designated N_PORT.
--
--       STATUS OF N_PORT
--
function STATUS_OF (N_PORT    : in     N_PORT_TYPE)
  return N_PORT_INFO_TYPE;

--
-- The following STATUS_OF function returns information about the
-- (possibility of) link switch connections between the designated
-- N_PORT and the RE_N_PORT in a point-to-point configuration.
--
--       STATUS OF POINT-TO-POINT-CONNECTED N_PORT
--
function STATUS_OF (N_PORT    : in     N_PORT_TYPE;
                    RE_N_PORT : in     N_PORT_TYPE)
  return N_PORT_INFO_TYPE;

--
-- The following STATUS_OF function returns information about the
-- (possibility of) link switch connections between the designated
-- N_PORT and the RE_UPSTREAM_PORT AND_DOWNSTREAM_PORT in a loop
-- configuration.
--
--       STATUS OF LOOP-CONNECTED N_PORT
--
function STATUS_OF (N_PORT              : in     N_PORT_TYPE;
                    RE_UPSTREAM_PORT    : in     N_PORT_TYPE;
                    AND_DOWNSTREAM_PORT : in     N_PORT_TYPE)
  return N_PORT_INFO_TYPE;

--
-- The following STATUS_OF function returns information about
-- all of the individual input and output ports on the designated
-- SWITCH.  The view is that of the individual input and output
-- ports of the link switch, not of the attached N_PORTs.
--
--       STATUS OF SWITCH
--
function STATUS_OF (SWITCH : in     SWITCH_ADDRESS_TYPE)
         return BASIC_SWITCH_STATUS_TYPE;

--
-- The following STATUS_OF function returns information about
-- all of the N_PORTs in the system.  The view is that of N_PORTs,
-- not of individual input and output ports on a link switch.
--
--       STATUS OF ALL N_PORTS
--
function STATUS_OF_N_PORTS
         return N_PORT_SWITCH_STATUS_TYPE;

--
```

```
  -- If the output indicated by the TO_OUTPUT parameter is marked as
  -- being CONNECTED to some input port, the MARK_PATH_UNUSABLE
  -- procedure marks the path from the current input port to the
  -- TO_OUTPUT port as being unusable; and will return the SUCCESSFUL
  -- value in the STATUS parameter.  Once this is done, the link switch
  -- will not utilize that path when requested to make a connection
  -- between the TO_OUTPUT port and the current input port.
  --
  -- If the output indicated by the TO_OUTPUT parameter is marked as
  -- not currently in use, this procedure returns the UNSUCCESSFUL
  -- value in the STATUS parameter.
  --
  procedure MARK_PATH_UNUSABLE
    (TO_OUTPUT  : in     OUTPUT_PORT_TYPE;
     ON_SWITCH  : in     SWITCH_ADDRESS_TYPE := 0;
     STATUS     :    out SUCCESS_TYPE);



  --
  -- The MARK_ALL_PATHS_USABLE procedure marks all paths in the link switch
  -- as being available for use.  This effectively inhibits the return
  -- of the ISOLATED value for any PORT_STATE_TYPE parameter in
  -- any subsequent subprogram invocation.
  --
  procedure MARK_ALL_PATHS_USABLE
    (ON_SWITCH  : in     SWITCH_ADDRESS_TYPE := 0);


end LINK_SWITCH_DRIVER;
```

C.2.4 Example Usage of Link Switch Control Interface

Figure C.7 shows the system that is to be connected by the link switch. For simplicity, a centralized link switch manager/switch driver is assumed.

Figure C.7 – Example System Interconnection



The following are some code fragments that illustrate the use of some of the facilities of the standard interface.

C.2.4.1 Example Using Only Individual Link Switch Port Commands.

```
with LINK_SWITCH_DRIVER;
use  LINK_SWITCH_DRIVER;

                       .
                       .
                       .

  --
  -- Initialize the link switch
  --
  INITIALIZE (SWITCH => 0);


  --
  -- Determine system interconnect requirements.
  --

                       .
                       .
                       .

  --
  -- Make the point-to-point connections.
```

```
--
CONNECT (OUTPUT    => 3,
         TO_INPUT  => 2,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (OUTPUT    => 2,
         TO_INPUT  => 3,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (OUTPUT    => 5,
         TO_INPUT  => 6,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (OUTPUT    => 6,
         TO_INPUT  => 5,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;
--
-- Make the Arbitrated Loop connections.
--
CONNECT (OUTPUT    => 4,
         TO_INPUT  => 1,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;
CONNECT (OUTPUT    => 7,
         TO_INPUT  => 4,
         ON_SWITCH => 0,
         STATUS    => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (OUTPUT    => 1,
         TO_INPUT  => 7,
         ON_SWITCH => 0,
         STATUS    => STATUS);
  if (STATUS = UNSUCCESSFUL) then
    raise CONNECTION_FAILED;
  end if;

  --
  -- At this point, the link switch has been configured and is ready for
  -- use.
  --
```

C.2.4.2 Example Using N_Port-Level Commands.

```
with LINK_SWITCH_DRIVER;
use  LINK_SWITCH_DRIVER;

                       .
                       .
                       .

  --
  -- Initialize the link switch
  --
  INITIALIZE (SWITCH => 0);


  --
  -- Define the N_Ports for the nodes.
  --

  DEFINE (N_PORT           => 0;
          N_PORT_OUTPUT    => 0;
          AND_N_PORT_INPUT => 0;
          ON_SWITCH        => 0;
          STATUS           => STATUS);
  if (STATUS = UNSUCCESSFUL) then
    raise N_PORT_DEFINITION_FAILED;
  end if;

  DEFINE (N_PORT           => 1;
          N_PORT_OUTPUT    => 1;
          AND_N_PORT_INPUT => 1;
          ON_SWITCH        => 0;
          STATUS           => STATUS);
  if (STATUS = UNSUCCESSFUL) then
    raise N_PORT_DEFINITION_FAILED;
  end if;

  DEFINE (N_PORT           => 2;
          N_PORT_OUTPUT    => 2;
          AND_N_PORT_INPUT => 2;
          ON_SWITCH        => 0;
          STATUS           => STATUS);
  if (STATUS = UNSUCCESSFUL) then
    raise N_PORT_DEFINITION_FAILED;
```

```
end if;

DEFINE (N_PORT           => 3;
        N_PORT_OUTPUT    => 3;
        AND_N_PORT_INPUT => 3;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
end if;

DEFINE (N_PORT           => 4;
        N_PORT_OUTPUT    => 4;
        AND_N_PORT_INPUT => 4;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
end if;

DEFINE (N_PORT           => 5;
        N_PORT_OUTPUT    => 5;
        AND_N_PORT_INPUT => 5;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
end if;

DEFINE (N_PORT           => 6;
        N_PORT_OUTPUT    => 6;
        AND_N_PORT_INPUT => 6;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
end if;

DEFINE (N_PORT           => 7;
        N_PORT_OUTPUT    => 7;
        AND_N_PORT_INPUT => 7;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
end if;

DEFINE (N_PORT           => 8;
        N_PORT_OUTPUT    => 8;
        AND_N_PORT_INPUT => 8;
        ON_SWITCH        => 0;
        STATUS           => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise N_PORT_DEFINITION_FAILED;
```

```
end if;


--
-- Determine system interconnect requirements.
--
                     .
                     .
                     .
--
-- Make the point-to-point connections.
--

CONNECT (N_PORT    => 2;
         TO_N_PORT => 3;
         STATUS    STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (N_PORT    => 5;
         TO_N_PORT => 6;
         STATUS    STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

--
-- Make the Arbitrated Loop connections.
--
CONNECT (N_PORT              => 4;
         TO_UPSTREAM_PORT    => 1;
         AND_DOWNSTREAM_PORT => 7;
         STATUS              => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;

CONNECT (N_PORT              => 1;
         TO_UPSTREAM_PORT    => 7;
         AND_DOWNSTREAM_PORT => 4;
         STATUS              => STATUS);
if (STATUS = UNSUCCESSFUL) then
  raise CONNECTION_FAILED;
end if;


--
-- At this point, the link switch has been configured and is ready for
-- use.
--
```

Index