PSF - Q.930/Q.931 (FT/HA)

Functional Specification
1091144 1.2

PSF - Q.930/Q.931 (FT/HA)

Functional Specification
1091144 1.2

Trillium Digital Systems, Inc.
12100 Wilshire Blvd., Suite 1800
Los Angeles, CA 90025-7118
Phone: +1 (310) 442-9222
Fax: +1 (310) 442-1162
Web: http://www.trillium.com

PSF - Q.930/Q.931 (FT/HA)
Functional Specification
1091144 1.2

Preface

Objective

This document provides a functional description of the PSF - Q.930 / Q.931 (FT / HA) software (p / n 1000144) designed by Trillium Digital Systems, Inc. This product is referred to as PSF - Q.930 / Q.931 in the rest of the document.

Audience

Trillium assumes that the readers of this document are familiar with telecommunication protocols, specifically ISDN, Trillium's Fault-Tolerant / High-Availability (FT / HA) concepts, and Trillium's ISDN product.

Document Organization

This document is organized into the following sections:

Section	Description
1 Introduction	Provides an overview of the product, including the product description and features
2 Environment	Describes design assumptions about the operating environment for the PSF - Q.930 / Q.931 software. This section explains the Trillium Advanced Portability Architecture (TAPA) and how PSF - Q.930 / Q.931 fits within the TAPA model.
3 Protocol Characteristics	Specifies the standards to which the software conforms, and the features supported
4 System Characteristics	Describes the features not directly related to the protocol, such as the management interface
5 Memory and Performance Characteristics	Provides the performance characteristics and memory size, which includes total code sizes

Document Set

The suggested reading order for the PSF - Q.930 / Q.931 document set is:

1. *PSF - Q.930/Q.931 (FT/HA) Functional Specification*

   Highlights and describes the protocol and system characteristics of the software, including the memory characteristics and conformance details.

2. *PSF - Q.930/Q.931 (FT/HA) Service Definition*

   Describes the procedures and the layer manager interface used to pass information between the software and other software elements. The Interface Primitives section describes the services of the software. The Interface Procedures section describes and illustrates the flow of primitives and messages across the interfaces.

   Note: *Information on porting the software is contained in the Service Definition.*

3. *PSF - Q.930/Q.931 (FT/HA) Software Test Sample*

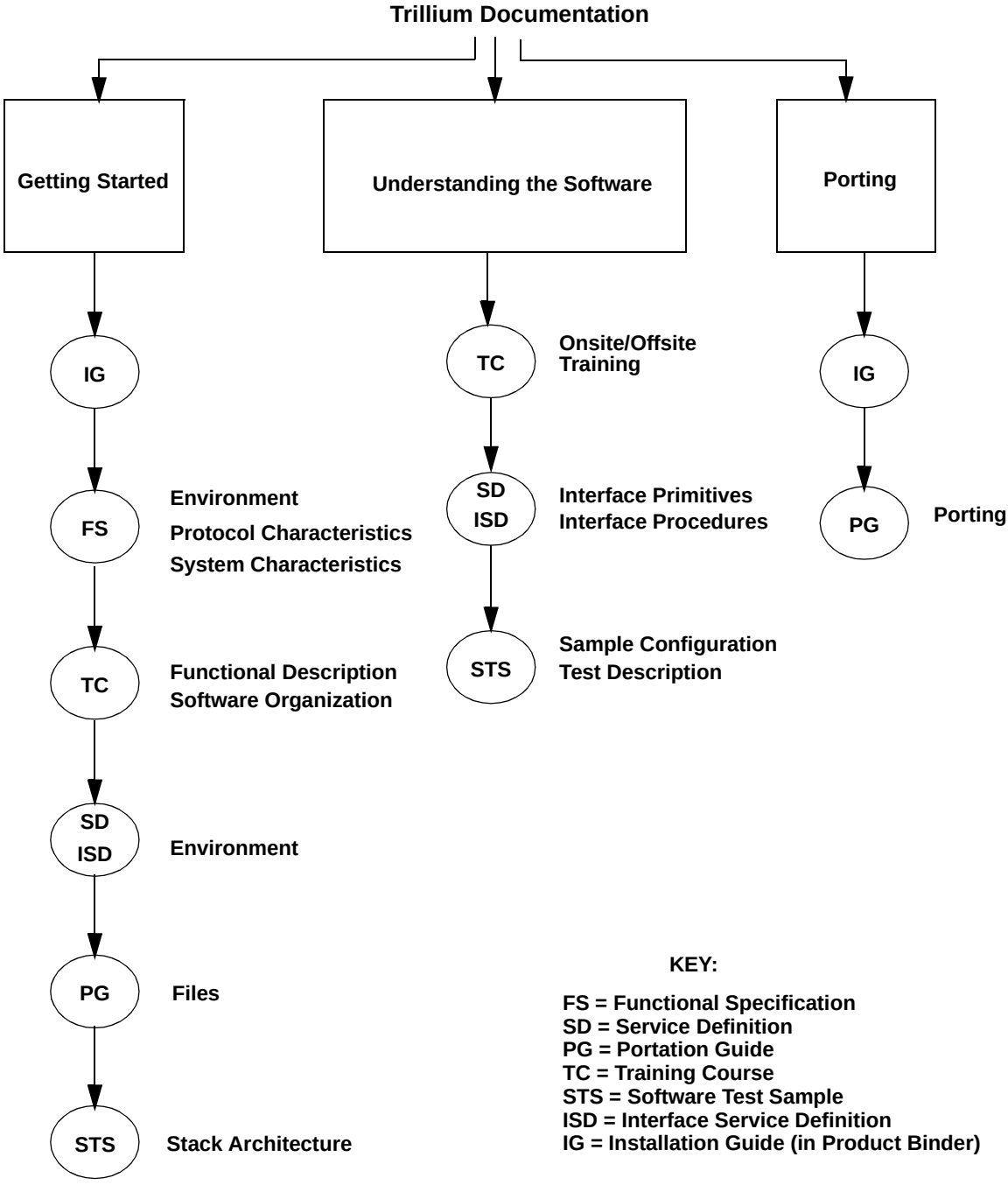
   Describes the sample files delivered with the product and the procedures to build a sample test. This test partially demonstrates the initialization, configuration, and execution of the product. It may contain data flow diagrams illustrating the correct operation of the software.

In addition to these PSF documents, the following documents should also be read for a better understanding of the fault-tolerant system:

4. *Fault-Tolerant/High-Availability (FT/HA) Core Functional Specification* (p / n 1091133).
5. *Fault-Tolerant/High-Availability (FT/HA) Core Service Definition* (p / n 1092133).

Using Trillium Documentation

The figure below illustrates the various approaches the user can take when utilizing the software documentation. First time users should read the documents under the **Getting Started** column; important sections and subsections are listed to the right of each document. For users familiar with the documentation but who need to look up certain points concerning the use of the software, the **Understanding the Software** column is suggested. The **Porting** column is for those users who are familiar with Trillium software and related telecommunications protocols and who wish to install the software immediately onto their operating systems.



Notations

This table displays the notations used in this document:

Notation	Explanation	Examples
Arial	**Titles**	**1.1 Title**
Palatino	Body text	This is body text.
Bold	**Highlights information**	**Loose coupling, tight coupling**, **upper layer interface**
ALL CAPS	CONDITIONS, MESSAGES	AND, OR CONNECT ACK
Italics	*Document names, emphasis*	*PSF - Q.930/Q.931 (FT/HA) Functional Specification* This adds *emphasis.*
Courier New Bold	**Code Filenames, pathnames**	**PUBLIC S16 ZqMiLzqCfgReq(pst, cfg)** **Pst *pst;** **CmPFthaMngmt *cfg;**

Release History

This table lists the history of changes in successive revisions to this document:

Version	Date	Initials	Description
1.2	12/28/99	sj	Final release. Conforms to software release 1.1.
1.1	09/30/99	pk	Initial release

Contents

Illustrations

1 INTRODUCTION

This document provides a functional description of the Protocol Specific Function (PSF) - Q.930 / Q.931 Fault Tolerant / High Availability (FT / HA) software (p / n 1000144) designed by Trillium Digital Systems, Inc. It also describes the primitives and procedures supported by the PSF - Q.930 / Q.931 software at the layer manager interface.

The PSF - Q.930 / Q.931 software adds Fault-Tolerant / High Availability (FT / HA) functionality to Trillium's Q.930 / Q.931 (ISDN) product. PSF - Q.930 / Q.931 can be viewed as a library of functions that is invoked by the Q.930 / Q.931 layer only in a Fault-Tolerant environment.

The PSF - Q.930 / Q.931 software provides interfaces to perform the following functions:

- Run-time state update of standby
- Warmstart of an Out-Of-Service (OOS) node to make it standby
- Controlled switchover of active and standby nodes
- Forced switchover, via the standby node (PSF), on failure of active node

The PSF - Q.930 / Q.931 software is portable C source code that can be compiled to run on any processor, under any operating system, and with an active / standby system architecture. A modular design and simple interfaces allow the PSF - Q.930 / Q.931 software to be easily ported to almost any environment.

1.1 Terms and Definitions

The following terms are used in this document:

Term	Definition
Active node	A node that executes software to provide the necessary protocol functionality. The active node processes the protocol messages and updates the new state information in the standby node.
Controlled switchover	A procedure that makes a standby node active and an active node standby
Fault-tolerant node	A pair of nodes with replicated protocol layers. A fault-tolerant node can be in an active, standby, or Out-Of-Service state.
Forced switchover	A procedure that makes a standby node active when an active node goes OOS
Node	A unit that has a processor(s) with private volatile memory inaccessible to all other nodes, and a private clock governing the execution of instructions on this processor. A node also has a network interface connecting it to a communication network using communication channels. The software governs the sequence of instructions executed on a node.
OOS node	An off-line node that has the ability to become an active or standby node
Run-time state update	The active Q.930/Q.931 node handles protocol events, which can result in internal state changes. The active PSF updates the standby with the state changes to keep the standby synchronized.
Standby node	A node that acts as a backup to an active node
Warmstart	A procedure that makes an OOS active node standby. An active node updates this new standby node with current information using a bulk update procedure.

1.2 Recommended Reading: FT/HA Service Definition

The *FT/HA Service Definition* describes system level fault-tolerant scenarios (for example, switchover and controlled switchover). This document can be referred to for a complete understanding of the functioning of various fault-tolerant layers in a protocol stack residing on different nodes. It describes various system entities which together achieve fault tolerance functionality, as well as the sequence of events that perform fault-tolerant procedures between the protocol layers and other system entities, such as controlled switchover and forced switchover.

Within the *FT/HA Service Definition*, the layer manager functions are split into three parts: system manager, system agent, and stack manager. Trillium provides both the system manager and the system agent components. If the user is not using Trillium's system manager and system agent, all interfaces with the system manager and system agent in the *FT/HA Service Definition* should be viewed as layer manager interfaces. The system manager, system agent, and stack manager can be implemented in a proprietary fashion by the user.

2 ENVIRONMENT

This section describes the environment in which the PSF - Q.930 / Q.931 software is designed to operate.

The Q.930 / Q.931 product conforms to Trillium Advanced Portability Architecture (TAPA). TAPA can be visualized as a box surrounded by four outer boxes — the box in the center represents the Q.930 / Q.931 (ISDN) software (see Figure 2-1 below). The four outer boxes represent other software to which Q.930 / Q.931 can be connected. The separation between the center box and outer boxes defines the interfaces across which Q.930 / Q.931 interacts with the other software.

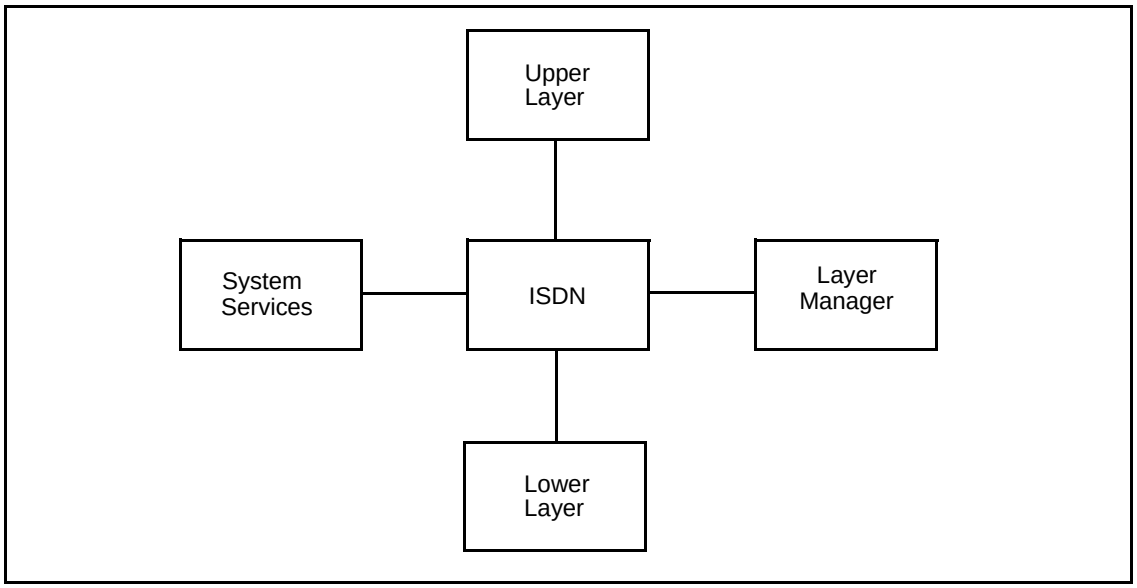


Figure 2-1: Trillium Advanced Portability Architecture (TAPA)

The PSF - Q.930 / Q.931 architecture differs from the architecture of other Trillium products that conform to TAPA, as it does not have an upper or lower layer. PSF - Q.930 / Q.931 does, however, provide the standard Layer Manager (LM) and System Services (SS) interfaces.

Figure 2-2 illustrates the architecture of the PSF - Q.930 / Q.931 software environment:

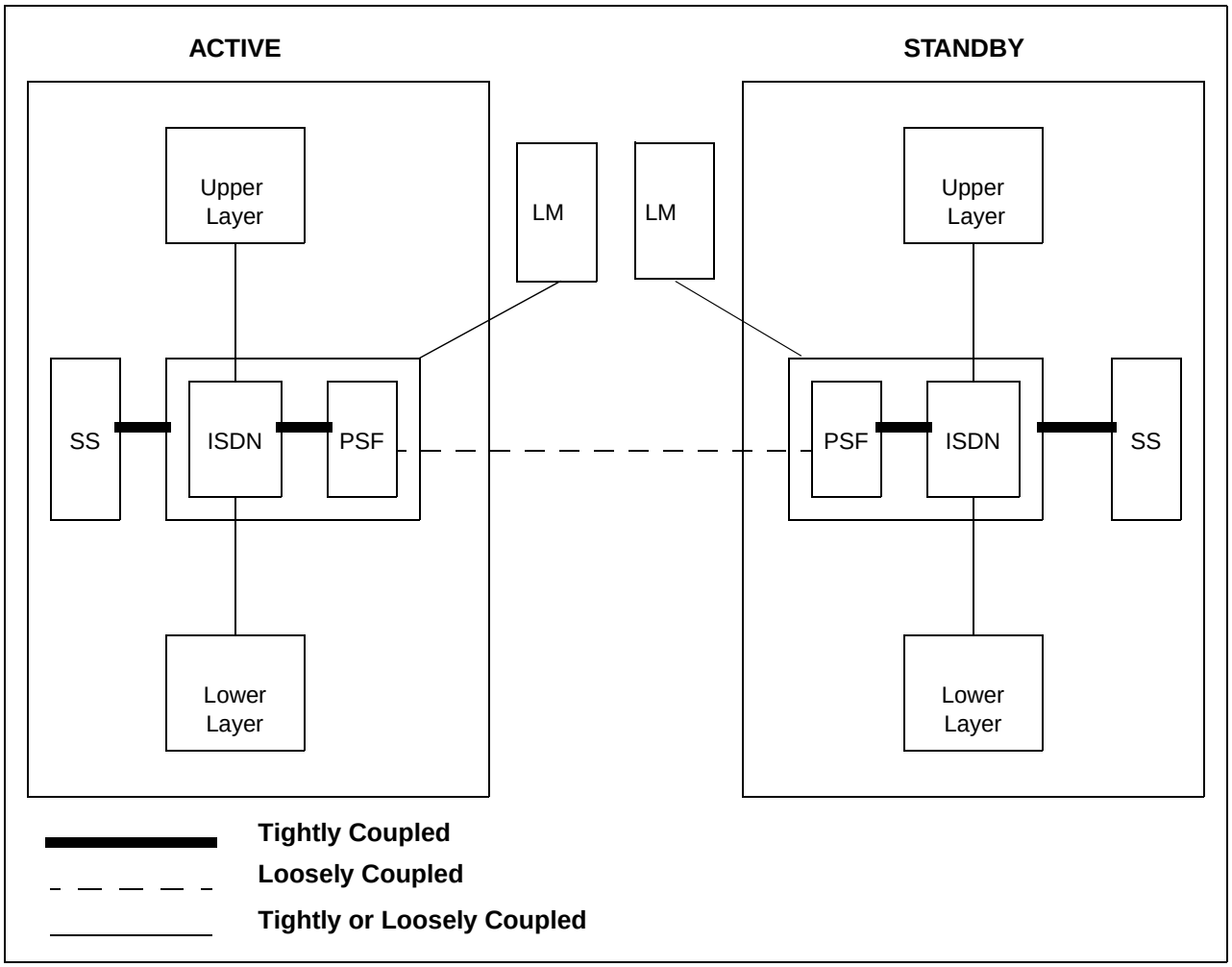


Figure 2-2: PSF - Q.930/Q.931 (FT/HA) software environment

The Q.930 / Q.931 — system services interface provides initialization, timer management, memory management, message, queue management, date and time management, and resource checking. The system services interface is always tightly coupled.

The PSF interface with the Q.930 / 931 protocol layer is an internal interface. The Q.930 / Q.931 invokes PSF functions to carry out run-time updates. This interface is always tightly coupled.

The PSF interface with the peer PSF - Q.930 / Q.931 is always loosely coupled. This interface allows the active PSF - Q.930 / Q.931 to communicate with the standby PSF - Q.930 / Q.931, and vice versa. The functions provided by this interface involve state updates from active to standby during run-time, warmstart, and controlled switchover.

PSF - Q.930 / Q.931 interacts with the layer manager and the peer PSF using a set of primitive functions. These primitives take the form of requests, indications, responses, and confirms, and completely define the interaction between layers.

3 PROTOCOL CHARACTERISTICS

This section describes the functions provided and the primitives supported by the PSF - Q.930 / Q.931 software.

PSF - Q.930 / Q.931 implements the functionality required to allow Trillium's Q.930 / Q.931 (ISDN) layer to function as a Fault-Tolerant / High-Availability (FT / HA) system. In a fault-tolerant system, two ISDN nodes are maintained—one as an active node and the other as a standby node. Each Q.930 / Q.931 layer has a corresponding PSF. Only the active copy participates in executing the protocol.

During event handling, the active node can modify its internal state. The active PSF - Q.930 / Q.931 updates these state changes to the standby to keep the standby synchronized with the active node.

To reduce run-time update overhead and complexity, the active PSF - Q.930 / Q.931 updates only the high-level stable states of the active node to the standby. The transient states—that is, the states with a relatively small duration—are updated only at controlled switchover time and not during run time. During a forced switchover, any existing transient state information at that time is lost. Therefore, calls in the transient state are lost. The standby resumes operating with its current, stable states.

3.1 Basic Features

The basic features supported by PSF - Q.930 / Q.931 are:

- Run-time updates
- Warmstart
- Controlled switchover
- Shutdown procedure
- Forced switchover
- Alarm generation

Run-time updates

PSF - Q.930 / Q.931 updates the state changes to the standby during runtime to keep the standby synchronized with the active node. As previously mentioned, only the high-level stable states are updated at run time. Run-time updates are performed when:

- the lower SAP goes to the transfer state,
- the lower SAP goes to the down state,
- the upper SAP goes to the transfer state,
- the upper SAP goes to the down state,
- the Q.931 PSIF (the PSIF is a module in the Q.931 layer, which handles the functionality required for inter-networking) remote SAP goes to the transfer state,
- the Q.931 PSIF remote SAP goes to the down state,
- the Q.931 PSIF upper SAP goes to the transfer state,
- the upper Q.931 PSIF SAP goes to the down state,
- the data link comes up,
- the data link goes down,
- the data link signalling state changes,
- a connection reaches an answered state—that is, receives / sends an alerting message,
- a connection reaches the active state,
- a connection is cleared,
- an interface or channel is restarted,
- the layer manager enables or disables the upper SAP or lower SAP of Q.931, and
- the layer manager enables or disables the upper SAP or remote SAP of Q.931 PSIF.

Warmstart

PSF - Q.930 / Q.931 provides the functionality to warm start a peer Q.930 / Q.931 layer to make it standby. This procedure brings the standby to the most recent stable state of the active node. The active PSF completes the warmstart operation over multiple schedulings. Between these schedulings, the layer can handle normal protocol events. An ongoing warmstart process can be aborted at anytime.

Controlled Switchover

PSF - Q.930 / Q.931 can perform a controlled switchover from the active node to the standby node without losing information or connections.

Shutdown Procedure

PSF - Q.930 / Q.931 provides a shutdown procedure that resets all the states and deallocates the memory allocated by PSF - Q.930 / Q.931, for its operation.

Forced Switchover

PSF - Q.930 / Q.931 provides the functionality for forced switchover when the active node goes Out-of-Service (OOS).

Alarm Generation

PSF - Q.930 / Q.931 generates alarms upon detecting failures.

Note: *PSF - Q.930/Q.931 supports all the variants (supported by the Q.930/Q.931 software), as well as support for the Q.932 service and restart procedures.*

3.2 Interface Primitives

The following section describes the primitives that interface PSF - Q.930/Q.931 with various layers.

3.2.1 Peer PSF Interface Primitives

The following set of primitives are used between the active and standby PSF - Q.930/Q.931:

Name	Description
ZqPiOubDatReq	Outbound data request
ZqPiInbDatReq	Inbound data request
ZqPiOubDatCfm	Outbound data confirm
ZqPiInbDatCfm	Inbound data confirm

3.2.2 Layer Manager Interface Primitives

The following set of primitives are used between the PSF and layer manager:

Configuration

These primitives function to configure the software:

Name	Description
`ZqMiLzqCfgReq`	Configuration request
`ZqMiLzqCfgCfm`	Configuration confirm

Control

These primitives function to control the operation of the software:

Name	Description
`ZqMiLzqCntrlReq`	Control request
`ZqMiLzqCntrlCfm`	Control confirm

Solicited Status

These primitives function to gather information to determine the current state of the software:

Name	Description
`ZqMiLzqStaReq`	Status request
`ZqMiLzqStaCfm`	Status confirm

Unsolicited Status Primitives

This primitive functions to provide information indicating a change in the status of the software:

Name	Description
`ZqMiLzqStaInd`	Configuration request

4 SYSTEM CHARACTERISTICS

This section describes the features of the PSF - Q.930/Q.931 software not directly related to the its state update functionality.

4.1 Portability

The PSF - Q.930/Q.931 software is written in the C programming language. The software can be compiled using any Kernighan and Ritchie or ANSI compatible compiler.

Trillium software has been compiled under native and cross compilers for many different processors, and has run on various processors, operating systems, and architectures. However, it has not necessarily been compiled under all compilers or run on all combinations of processors, operating systems, and architectures. Adherence to TAPA and its common coding and architectural standards ensures the ability to compile the software under any compiler and to run it under any processor, operating system, and architecture.

4.2 Layer Manager Characteristics

The layer manager provides the functions for configuring, controlling, and monitoring to ensure error-free and efficient operation of the PSF - Q.930/Q.931 software.

4.2.1 Alarms

Alarms are generated to indicate an abnormal condition in PSF - Q.930/Q.931. Alarms are sent to the layer manager as unsolicited status indications whenever a condition that requires attention is detected. The alarm indication can be enabled or disabled dynamically using the control primitives.

The following alarms can be generated:

Name	**Description**
Memory failure alarm	Generated when PSF - Q.930/Q.931 finds a failure while trying to allocate a buffer to send an update message to the standby node
Update message error alarm	Generated when the PSF - Q.930/Q.931 on the standby node detects an error in a state update message received from the active node
Sequence error alarm	Generated when PSF - Q.930/Q.931 on the standby node detects a sequence error in a state update message received from the active node. This implies that an update message sent earlier from the active PSF - Q.930/Q.931 has been lost.

4.2.2 Debug Prints

Debug prints can be generated from PSF - Q.930/Q.931 at different levels. The following debug levels, which can be individually enabled or disabled from the layer manager, are supported by PSF - Q.930/Q.931:

Name	Description
Protocol layer interface debug print	Generated whenever a function is called at the PSF - Q.930/Q.931 and Q.930/Q.931 interface. The print indicates the function name and the parameters passed to the function.
Peer PSF interface debug print	Generated whenever a function is called at the PSF - Q.930/Q.931 and the peer PSF - Q.930/Q.931 interface. The print indicates the function name and the parameters passed to the function.
Layer manager interface debug print	Generated whenever a function is called at the PSF - Q.930/Q.931 and layer manager interface. The print indicates the function name and the parameters passed to the function.
Pack debug print	Generated whenever a pack function is called by the active PSF - Q.930/Q.931 to pack the states in an update message.
Unpack debug print	Generated whenever the standby PSF - Q.930/Q.931 calls an unpack function to unpack the states from an update message received from the active node.

4.2.3 Configuration

Configuration parameters are received from the layer manager. Individual parameters cannot be configured selectively—that is, without specifying the rest of the configuration parameters. The specified ranges can be changed by modifying the appropriate type definitions. The actual values that can be assigned depend on the availability of system resources, such as memory.

Some configuration parameters can be reconfigured in a running system without affecting the operation of the PSF - Q.930 / Q.931 software.

4.2.3.1 General Configuration

The following parameters are configurable for the entire PSF - Q.930 / Q.931 software:

Parameters	Reconfigurable?	Allowable Values
Timer resolution	No	0 to 32767
Virtual processor ID of the Q.930 / Q.931 node	No	0 to 65535
Memory region for allocating message buffers for mailing a message to self	No	0 to 255
Memory pool for allocating message buffers for mailing a message to self	No	0 to 255
Layer manager post structure for reporting alarms	Yes	See note below table

Note: *Refer to the System Services Interface Service Definition for the range of the post structure.*

4.2.3.2 Peer SAP Configuration

This option configures the SAP towards the peer PSF - Q.930 / Q.931:

Parameter	Reconfigurable?	Allowable Values
Memory region for allocating message buffers for mailing a message to the peer PSF - Q.930 / Q.931	Yes	0 to 255
Memory pool for allocating message buffers for mailing a message to the peer PSF - Q.930 / Q.931	Yes	0 to 255
Physical processor ID of the peer PSF - Q.930 / Q.931	Yes	0 to 65535
Entity ID of the peer PSF - Q.930 / Q.931	Yes	0 to 255
Instance ID of the peer PSF - Q.930 / Q.931	Yes	0 to 255
Priority for the post structure of the peer SAP	Yes	Not currently used
Route for the post structure of the peer SAP	Yes	Default
Selector for the post structure of the peer SAP	Yes	Not currently used
Value for the timer started by the active PSF - Q.930 / Q.931 to wait for the confirm from the standby for warmstart or controlled switchover state update	Yes	0 to 65535
Maximum update message size	Yes	170 to (2^{32} - 1) (See note below table)

Note: *The minimum size of the update message must be greater than the maximum size of a table to be packed by PSF - Q.930/Q.931.*

4.2.4 Control

Control requests are issued by the layer manager to change the status of the Q.930/Q.931 (from active to standby or vice versa), and to perform warmstart or controlled switchover procedures. The layer manager can issue a control request at any time.

Control functions are shown in the following table:

Control Function	Description
Go active	Makes Q.930/Q.931 node active with peer node enabled or disabled, depending on whether the peer is standby or OOS. This action can be used in the following scenarios: • When both nodes are OOS and one of them is to be made active • When an OOS Q.930/Q.931 node must be made active • When a standby Q.930/Q.931 must be made active for switchover • When a current active node must remain active (see note below table)
Go standby	Makes a Q.930/Q.931 node standby with the peer SAP enabled. This action can be used in the following scenarios: • When an OOS Q.930/Q.931 node must be made standby • When an active Q.930/Q.931 must be made standby for controlled switchover • When a current standby node must remain standby (see note below table)
Warmstart	Enables and warmstarts the SAP towards the peer (i.e., sends all stable state information to the standby). This action is used by the layer manager when an OOS Q.930/Q.931 node must be made standby. This action is valid only for the active PSF - Q.930/Q.931 node.
Synchronize	Updates the standby with the transient state information. This action is used by the layer manager during a controlled switchover of an Q.930/Q.931 node. The active PSF - Q.930/Q.931 sends all transient states (which are not updated during run time) to the standby. This action is valid only for the active PSF - Q.930/Q.931 node.
Disable peer SAP	Disables the peer SAP. This action is used by the layer manager when a standby Q.930/Q.931 node goes OOS. The active node stops sending run-time state update messages after the peer SAP is disabled. This action is valid only for the active Q.930/Q.931 node.
Shutdown	Shuts down PSF - Q.930/Q.931 operation. The PSF - Q.930/Q.931 deallocates all allocated memory and goes into the state that it maintains after initialization. PSF - Q.930/Q.931 can then be configured again.
Abort	Aborts the ongoing warmstart or controlled switchover procedure. The layer manager can use this action to abort an ongoing procedure when a warmstart or synchronization request has been sent to the active. This action is valid only for the active PSF - Q.930/Q.931 node.

Control Function	Description
Enable alarm indication	Enables the generation of unsolicited status indications that indicate an abnormal situation encountered by the software. Alarm indication is disabled by default.
Disable alarm indication	Disables the generation of unsolicited status indications that indicate an abnormal situation encountered by the software. Alarm indication is disabled by default.
Enable debug print generation	Enables the generation of debug prints. Debug prints are disabled by default.
Disable debug print generation	Disables the generation of debug prints. Debug prints are disabled by default.

Note: *The layer manager can request the active PSF - Q.930/Q.931 node to synchronize the standby node to perform a controlled switchover. When the active node completes synchronization, the layer manager sends a control request to the active node to become standby, and another control request to the standby node to become active. After synchronization, the layer manager may decide not to continue with the controlled switchover because of synchronization failure or because the user wants to abort the switchover. In such a case, the layer manager needs to send a control request to the active node to remain active and a control request to the standby node to remain standby, in order to resume protocol operation.*

4.2.5 Status

Status information indicates the current state of the software. This information can be used in software debugging. Status information can be gathered at any time by the layer manager. Collection of status information does not change any of the information examined.

4.2.5.1 Solicited Status

The layer manager can issue a status request to determine the Q.930/Q.931 state. The layer can be in the following states.

- Active
- Standby
- OOS

The layer manager can issue a status request to determine the bind state and the update state of the peer SAP. The peer SAP can be either bound or unbound. The update state can be the following:

- Idle
- Warmstarting
- Synchronizing
- Waiting for confirm from standby (on completion of warmstart)
- Waiting for confirm from standby (on completion of synchronization)

5 MEMORY REQUIREMENTS

This section describes the code size, data size, and performance characteristics of the PSF - Q.930 / Q.931 software.

5.1 Code Size

The code size is the number of bytes of memory needed for the executable code. It includes all function calls to the layer manager and the upper layer, but does not include the actual code provided within these functions.

The code size depends on the options delivered, compiler, linker, locator, memory model, and whether all delivered features (for example, error checking, management capabilities, and protocol capabilities) are retained. The code size is determined from the software link map.

5.1.1 Tightly Coupled Interfaces

A sample compile under the following conditions:

Type	**Condition**
Product	PSF - Q.930 / Q.931, version 1.1
Product option	None
Compiler	Microtec C 68k Compiler, version: 4.4
Compiler options	**`-p68040 -H -A -v -Fsm -O -Ob -Oe -Ot -nKc -nKm -Mcp -Mdp -Ml -Gf`**
Processor	Motorola 68040
Error checking	Disabled
Layer coupling	Tightly coupled across all interfaces
Total code size	**35244 bytes**

5.1.2 Loosely Coupled Interface

A sample compile under the following conditions:

Type	Condition
Product	PSF - Q.930 / Q.931, version 1.1
Product option	None
Compiler	Microtec C 68k Compiler, version: 4.4
Compiler options	`-p68040 -H -A -v -Fsm -O -Ob -Oe -Ot -nKc -nKm -Mcp -Mdp -Ml -Gf`
Processor	Motorola 68040
Error checking	Disabled
Layer coupling	Loosely coupled across all interfaces
Total code size	**35524 bytes**

5.2 Static Data Sizes

Static data size is the number of bytes of memory needed for initialized variables and structures (for example, state matrices and strings). It is allocated at compile time and represents the global variables and structures used by the software. Static data does not include any structures allocated at run time.

The static data size depends on the options delivered, compiler, linker, and memory model, and is determined from the software link map.

5.2.1 Tightly Coupled

A sample compile under the following conditions:

Type	**Condition**
Product	PSF - Q.930 / Q.931 (FT / HA), version 1.1
Product option	None
Compiler	Microtec C 68k Compiler, version: 4.4
Compiler options	**`-p68040 -H -A -v -Fsm -O -Ob -Oe -Ot -nKc -nKm -Mcp -Mdp -Ml -Gf`**
Processor	Motorola 68040
Error checking	Disabled

Yielded the following static data sizes:

Type	**Size**
Strings	6144
Constants	17996
Initialized variables	604
Uninitialized variables	20372
Total code size	**45116 bytes**

5.2.2 Loosely Coupled

A sample compile under the following conditions:

Type	Condition
Product	PSF - Q.930 / Q.931 (FT / HA), version 1.1
Product option	None
Compiler	Microtec C 68k Compiler, version: 4.4
Compiler options	`-p68040 -H -A -v -Fsm -O -Ob -Oe -Ot -nKc -nKm -Mcp -Mdp -Ml -Gf`
Processor	Motorola 68040
Error checking	Disabled

Yielded the following static data sizes:

Type	Sizes
Strings	6208
Constants	17996
Initialized variables	604
Uninitialized variables	20372
Total code size	**45180 bytes**

5.3 Dynamic Data Size

Dynamic data size is the number of bytes of memory needed for:

- Structures (for example, control points and SAPs) used to manage the interface and protocols
- Buffers used to store messages

Dynamic data is allocated at run time and represents memory managed by the operating system. Allocation of dynamic data depends on the compile-time configuration, run-time configuration, and net flow of traffic through the software.

The dynamic data size is dependent upon the maximum allowable configuration (A), the dynamically allocated structure size (B), the maximum number of messages to be stored (C), and the message buffer size (D). This is computed by the following formula: (A x B) + (C x D).

Abbreviations

The following abbreviations are used in this document:

Abbreviation	Definition
FT / HA	Fault-Tolerant / High-Availability
ISDN	Integrated Services Digital Network
LM	Layer Manager
OOS	Out-Of-Service
PSF	Protocol-Specific Function
SAP	Service Access Point
SS	System Services
TAPA	Trillium Advanced Portability Architecture

References

Refer to the following documents for more information:

Fault-Tolerant/High-Availability (FT/HA) Core Functional Specification, Trillium Digital Systems, Inc. (p / n 1091133).

Fault-Tolerant/High-Availability (FT/HA) Core Service Definition, Trillium Digital Systems, Inc. (p / n 1092133).

INT Interface Service Definition, Trillium Digital Systems, Inc. (p / n 1100018).

PSF - Q.930/Q.931 (FT/HA) Service Definition, Trillium Digital Systems, Inc. (p / n 1092144).

PSF - Q.930/Q.931 (FT/HA) Software Test Sample, Trillium Digital Systems, Inc. (p / n 1094144).

Q.930/Q.931 Functional Specification, Trillium Digital Systems, Inc. (p / n 1091009).

Q.930/Q.931 Portation Guide, Trillium Digital Systems, Inc. (p / n 1093009).

Q.930/Q.931 Service Definition, Trillium Digital Systems, Inc. (p / n 1092009).

Q.930/Q.931 Software Test Sample, Trillium Digital Systems, Inc. (p / n 1094009).

Q.930/Q.931 Training Course, Trillium Digital Systems, Inc. (p / n 1095009).

Q.930 (I.450) - ISDN User-Network Interface Layer 3, ITU.

Q.931 (I.451) - ISDN User-Network Interface Layer 3 Specification for Basic Call Control, ITU.

Q.932 (I.452) - ISDN User-Network Interface Layer 3 Specification - Generic Procedures for the Control of ISDN Supplementary Services, ITU.

System Services Interface Service Definition, Trillium Digital Systems, Inc. (p / n 1111001).