

EARTH OBSERVING SYSTEM
DATA AND INFORMATION SYSTEM (EOSDIS)
TEST SYSTEM (ETS)
HIGH-RATE SYSTEM (HRS)
DETAILED DESIGN SPECIFICATION

VOLUME 3

SIGNATURE DRAFT

October 1996



National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland

**Earth Observing System
Data and Information System (EOSDIS)
Test System (ETS)
High-Rate System (HRS)
Detailed Design Specification**

Volume 3

October 1996

Prepared by:

K. Daniel, ETS HRS System Engineer
SGT, Inc.

Concurred by:

W. Fuller, ETS Project Manager
Data Simulation Section
Code 515.2, NASA/GSFC

Reviewed by:

C. Mirchandani, ETS Project Lead
Lockheed Martin Space Mission Systems

Concurred by:

D. Lakins, ESDIS ETS Project Manager
ESDIS Project
Code 505, NASA/GSFC

Edited by:

L. Phillips, Publications Manager
RMS Technologies, Inc.

Approved by:

N. Speciale, Head
Microelectronic Systems Branch
Code 521, NASA/GSFC

Quality Assured by:

C. Peppersack, Sr. Systems Engineer
SGT, Inc.

PREFACE

This document is Volume 3 of a 5-volume set specifying the detailed design for the Earth Observing System Data and Information System (EOSDIS) Test System (ETS). This volume describes detailed design specifications for the ETS High-rate System (HRS) being developed by the Goddard Space Flight Center (GSFC) Microelectronic Systems Branch (MSB).

This document will serve as the single source document for configuration control of the ETS HRS system design, and will serve as the basis for design, implementation, and testing.

This document is under the configuration management of the Microelectronic Systems Branch Configuration Control Board (CCB). Any changes to this document shall be made by Documentation Change Notice (DCN), reflected in text by change bars, or by complete revision.

Requests for copies of this document, along with questions and proposed changes, should be addressed to:

Technical Publications Group,
Microelectronic Systems Branch, Code 520.9
Goddard Space Flight Center
Greenbelt, Maryland 20771

CHANGE INFORMATION PAGE

<i>List of Effective Pages</i>		
Page Number		Issue
Title		Original
Signature Page		Original
iii through xii		Original
1-1 through 1-6		Original
2-1 through 2-5		Original
3-1 through 3-20		Original
4-1 through 4-8		Original
5-1 through 5-28		Original
6-1 through 6-19		Original
7-1 through 7-23		Original
8-1 through 8-7		Original
9-1 through 9-4		Original
10-1 through 10-27		Original
AB-1 through AB-3		Original
<i>Document History</i>		
Issue	Date	DCN No.
Original	October 1996	N/A

TABLE OF CONTENTS

SECTION 1 - INTRODUCTION	1-1
1.1 Purpose.....	1-1
1.2 Overview.....	1-1
1.3 Applicable Documents	1-3
1.3.1 Parent Requirements Documents.....	1-3
1.3.2 Standards.....	1-3
1.3.3 References.....	1-4
1.4 Document Organization.....	1-5
1.5 Terminology.....	1-5
 SECTION 2 - SYSTEM DESIGN OVERVIEW	 2-1
2.1 Design Methodologies.....	2-1
2.2 Design Approaches	2-2
2.2.1 Changing EDOS Baseline and Requirements.....	2-2
2.2.2 Interface Issues.....	2-2
2.2.3 Technical Challenges	2-2
2.2.4 Level Zero Processing Functions.....	2-2
2.2.5 System Tests	2-3
2.2.6 User Interfaces	2-3
2.3 Design Studies and Prototypes	2-3
2.3.1 Disk Array Study and Evaluation	2-3
2.3.2 FDDI Interface Module Evaluation	2-5
2.3.3 SCSI-2 Interface Evaluation	2-5
 SECTION 3 - SYSTEM DESIGN	 3-1
3.1 Introduction	3-1
3.2 VME High-Rate Subsystem	3-1
3.2.1 VHS Functional Overview.....	3-1
3.2.2 VHS System Architecture	3-1
3.2.2.1 VHS System Block Diagram	3-3
3.2.2.2 VHS Data Flow Diagrams	3-5
3.2.3 VHS External Interfaces	3-9
3.2.3.1 ECL Serial Ports.....	3-10
3.2.3.2 Ethernet.....	3-10
3.2.3.3 FDDI Network.....	3-11
3.2.3.4 EBnet Network	3-11
3.3 Tape Recording Subsystem.....	3-12
3.3.1 TRS Functional Overview	3-12
3.3.2 TRS System Architecture.....	3-12
3.3.2.1 System Block Diagram.....	3-14
3.3.2.2 TRS Data Flow Diagrams	3-15
3.3.3 TRS External Interfaces	3-17
3.3.3.1 ECL Serial Ports.....	3-17
3.3.3.2 Ethernet.....	3-17
3.3.3.3 RS-232 Port.....	3-18
3.3.3.4 IEEE 488 Port.....	3-18
3.4 Control and Display Subsystem.....	3-19
 SECTION 4 - USER INTERFACE	 4-1
4.1 Introduction	4-1
4.2 Telemetry Processing and Control Environment.....	4-1

TABLE OF CONTENTS (CONT'D)

4.2.1	Software Description.....	4-1
4.2.2	MEDS Review.....	4-1
4.2.3	Architecture.....	4-2
4.2.3.1	VLSI Server.....	4-3
4.2.3.2	Status Collector.....	4-3
4.2.3.3	Message Collector.....	4-3
4.2.3.4	Process Initiator.....	4-3
4.2.3.5	External Status Server.....	4-4
4.2.3.6	Event Log.....	4-4
4.2.3.7	IPC Server.....	4-4
4.2.3.8	Parameter Server.....	4-4
4.2.3.9	Schedule Reader.....	4-4
4.2.3.10	Activity Schedule.....	4-5
4.2.3.11	Expert System.....	4-5
4.2.3.12	Data Set Distributor.....	4-5
4.2.3.13	Configuration Set Editor.....	4-6
4.2.3.14	Preference Editor.....	4-6
4.2.3.15	Status Page.....	4-6
4.3	Local Operations/OPMAN.....	4-6
4.3.1	Activating a Catalog.....	4-6
4.3.2	Looking at Status.....	4-7
4.3.3	Shutting Off a Catalog.....	4-7
4.3.4	Quicklook Features.....	4-7
4.3.5	Editing a Catalog.....	4-8
SECTION 5 - OPERATIONS SCENARIOS.....		5-1
5.1	Introduction.....	5-1
5.2	Simulate AM-1 Science Return Link Data.....	5-1
5.3	Simulate EDOS Output to Test DAAC Front-End.....	5-3
5.4	Simulate DAAC Front-End to Test EDOS Output.....	5-3
5.5	Generate EDSs and PDSs from Spacecraft-Generated Data.....	5-4
5.6	Read/Transfer from Spacecraft-Generated Data Tapes.....	5-5
5.7	Provide CADU Data Stream for Data Set Generation.....	5-6
5.8	Configuration and Operation.....	5-7
5.8.1	EOS Frame Synchronizer Card.....	5-7
5.8.1.1	Frame Synchronization.....	5-8
5.8.1.2	Modes of Operation.....	5-9
5.8.1.3	Configuration.....	5-10
5.8.2	EOS Reed-Solomon Decoder.....	5-11
5.8.2.1	Reed-Solomon Decoding and Error Correction.....	5-11
5.8.2.2	Modes of Operation.....	5-13
5.8.2.3	Configuration.....	5-14
5.8.3	EOS Service Processor Card.....	5-15
5.8.3.1	Modes of Operation.....	5-16
5.8.3.2	Configuration.....	5-16
5.8.3.3	Operation.....	5-16
5.8.4	EOS Simulator Card.....	5-18
5.8.4.1	Modes of Operation.....	5-18
5.8.4.2	Configuration.....	5-20
5.8.5	Data Set Processor.....	5-20
5.8.5.1	Data Storage Operation.....	5-20

TABLE OF CONTENTS (CONT'D)

	5.8.5.2	Data Output Operation	5-21
	5.8.5.3	Data Capture and Transmission Operation	5-21
	5.8.5.4	Configuration.....	5-22
5.8.6		Annotation Processor.....	5-22
5.8.7		TRS Ampex Tape Drive	5-24
	5.8.7.1	Modes of Operation.....	5-24
	5.8.7.2	General Features.....	5-24
5.8.8		TRS Sony Tape Drive.....	5-25
	5.8.8.1	Modes of Operation.....	5-25
	5.8.8.2	High-Density Recording.....	5-25
	5.8.8.3	Tape Interchange	5-26
	5.8.8.4	High-Speed Recording/Playback.....	5-26
	5.8.8.5	Time Commanding Capability	5-26
	5.8.8.6	Error Correction.....	5-26
	5.8.8.7	Read After Write.....	5-26
	5.8.8.8	Search Function.....	5-26
	5.8.8.9	Remote Control Interface.....	5-26
	5.8.8.10	Self-Diagnostic Function	5-26
5.8.9		TRS Triplex Interface	5-26
	5.8.9.1	Sony DIR-1000 Interface.....	5-27
	5.8.9.2	VMEbus Interface.....	5-27
	5.8.9.3	VSB Interface.....	5-28
	5.8.9.4	External Clock and Sync Pulse Interface.....	5-28
SECTION 6 - DETAILED HARDWARE DESIGN.....			6-1
6.1		VHS Custom Components.....	6-1
	6.1.1	EOS Simulator Card	6-2
	6.1.2	EOS Frame Synchronizer Card	6-4
	6.1.3	EOS Reed-Solomon Card	6-7
	6.1.4	EOS Service Processor Card	6-9
	6.1.5	Data Set Processors 1 and 2.....	6-15
	6.1.6	Annotation Processor.....	6-16
	6.1.7	High-Rate Telemetry Backplane	6-17
6.2		Commercial Components.....	6-18
	6.2.1	System Disk and Interface Module	6-18
	6.2.2	Master Controller Card.....	6-19
	6.2.3	System Memory Cards.....	6-19
	6.2.4	Global Memory Card.....	6-19
SECTION 7 - DETAILED SOFTWARE DESIGN.....			7-1
7.1		VME High-Rate Subsystem	7-1
	7.1.1	VxWorks Environment.....	7-1
	7.1.2	MEDS Environment	7-2
	7.1.3	Telemetry Processing Control Environment.....	7-3
	7.1.4	Master Controller Software	7-4
	7.1.5	Data Set Processor Software.....	7-5
	7.1.6	Annotation Processor Software.....	7-6
	7.1.7	EOS Simulator Card Software	7-7
	7.1.8	EOS Frame Synchronizer Card Software	7-8
	7.1.9	EOS Reed-Solomon Card Software	7-10
	7.1.10	EOS Service Processor Card Software	7-11

TABLE OF CONTENTS (CONT'D)

	7.1.10.1	Header Process.....	7-12
	7.1.10.2	Quality Process.....	7-13
	7.1.10.3	Output Process	7-14
7.2		Tape Recording Subsystem Software.....	7-15
	7.2.1	High-Level Design.....	7-16
	7.2.2	High-Level Design Description	7-17
	7.2.3	Detailed Design	7-19
	7.2.3.1	TRS Generic Interface Handler	7-19
	7.2.3.2	Handle Sony Command and Status Process.....	7-20
	7.2.3.3	Sony Data I/O Handler Process	7-22
	7.2.3.4	Ampex Command and Status Handler.....	7-23
	7.2.3.5	Clock Signal Generator Handler Process	7-23
SECTION 8 - LOCAL AREA NETWORK AND CONFIGURATIONS.....			8-1
8.1		Communication Protocols.....	8-1
	8.1.1	File Transfer Protocol.....	8-1
	8.1.2	Transmission Control Protocol	8-2
	8.1.3	User Datagram Protocol.....	8-3
	8.1.4	Internet Protocol.....	8-3
8.2		Fiber Distributed Data Interface.....	8-4
	8.2.1	FDDI Module.....	8-4
	8.2.2	FDDI Frame	8-4
8.3		Ethernet Interface	8-5
8.4		ETS HRS System Configuration.....	8-6
	8.4.1	Master Controller and CDS Configuration.....	8-6
	8.4.2	EOS Simulator Card and CDS Configuration.....	8-7
	8.4.3	TRS and CDS Configuration.....	8-7
SECTION 9 - SYSTEM TEST APPROACH			9-1
9.1		Introduction	9-1
9.2		System Tests	9-3
9.3		Test Guidelines.....	9-3
9.4		Test Approval	9-3
9.5		Environmental Test Conditions.....	9-3
9.6		Discrepancy Reporting and Retest	9-4
9.7		Failure During Test.....	9-4
9.8		Retest and Regression Testing	9-4
9.9		Test Dependencies.....	9-4
SECTION 10 - REQUIREMENTS TRACEABILITY MATRIX			10-1
10.1		ETS HRS SRS Traced to ETS DDS.....	10-1
10.2		ETS F&P Requirements Traced to ETS SRS.....	10-23
10.3		Build Plan for ETS HRS.....	10-27
ACRONYMS AND ABBREVIATIONS			AB-1

TABLE OF CONTENTS (CONT'D)

TABLES

Table 2-1.	Ciprico SCSI-2, Model 6712, Disk Array Test Results	2-4
Table 2-2.	Data General Clarion Disk Array Test Results	2-4
Table 2-3.	VMEbus FDDI Adapter Performance	2-5
Table 3-1.	VHS External Interfaces and Protocols.....	3-12
Table 3-2.	TRS External Interfaces and Protocols	3-19
Table 10-1.	ETS HRS Projected Build Plan	10-27

FIGURES

Figure 1-1.	ETS HRS Block Diagram.....	1-2
Figure 3-1.	VHS Rack.....	3-2
Figure 3-2.	VHS Functional Block Diagram	3-4
Figure 3-3.	VHS Simulating TGT Output	3-5
Figure 3-4.	VHS Simulating EDOS Output	3-6
Figure 3-5.	VHS Simulating DAAC Front-End.....	3-7
Figure 3-6.	VHS Generating EDOS Data Sets from Spacecraft Test Data Tapes.....	3-8
Figure 3-7.	VHS Generating CADU Data Files from Spacecraft Test Data Tapes.....	3-9
Figure 3-8.	VHS External Interfaces.....	3-10
Figure 3-9.	TRS Rack.....	3-13
Figure 3-10.	TRS Functional Block Diagram.....	3-14
Figure 3-11.	TRS Simulating TGT Output.....	3-16
Figure 3-12.	TRS Transferring and Storing SCITF CADU Data.....	3-16
Figure 3-13.	TRS External Interfaces.....	3-17
Figure 4-1.	TPCE Software Architecture.....	4-2
Figure 5-1.	VHS Simulating AM-1 Science Return Link Data at 150 Mbps.....	5-1
Figure 5-2.	TRS Simulating AM-1 Science Return Link Data at 150 Mbps.....	5-2
Figure 5-3.	VHS Simulating EDOS Output to Test DAAC Front-End.....	5-3
Figure 5-4.	VHS Simulating DAAC Front-End to Test EDOS Output.....	5-4
Figure 5-5.	VHS Generating EDSs and PDSs from Spacecraft-Generated Data.....	5-5
Figure 5-6.	TRS Reading/Transferring from Spacecraft-Generated Data Tapes	5-6
Figure 5-7.	TRS Providing CADU Data Stream for Data Set Generation.....	5-7
Figure 5-8.	EOS Frame Synchronizer Card I/O Diagram.....	5-8
Figure 5-9.	Frame Synchronization	5-9
Figure 5-10.	ERS Card I/O Diagram	5-11
Figure 5-11.	Interleave by 2.....	5-12
Figure 5-12.	Unrouteable/Uncorrectable Determination	5-13
Figure 5-13.	EOS Service Processor Card I/O Diagram.....	5-15
Figure 5-14.	EOS Simulator Card I/O Diagram.....	5-18
Figure 5-15.	Data Set Processor I/O, Data Storage Mode.....	5-21
Figure 5-16.	Data Set Processor I/O, Output Mode.....	5-21
Figure 5-17.	Data Set Processor I/O, Capture and Transmit Mode.....	5-22
Figure 5-18.	Annotation Processor I/O.....	5-23
Figure 6-1.	VHS Chassis.....	6-1
Figure 6-2.	EOS Simulator Card Hardware.....	6-2
Figure 6-3.	EOS Frame Synchronizer Card Hardware.....	6-5
Figure 6-4.	ERS Card Hardware.....	6-7
Figure 6-5.	EOS Service Processor Card Hardware.....	6-10
Figure 6-6.	EOS Service Processor Card Functional Block Diagram.....	6-12
Figure 6-7.	Data Set Processor 1 Data Flow Diagram	6-16

TABLE OF CONTENTS (CONT'D)

Figure 6-8.	Data Set Processor 2 Data Flow Diagram	6-16
Figure 6-9.	Annotation Processor Data Flow.....	6-16
Figure 6-10.	HRTB Hardware	6-17
Figure 7-1.	VHS Software Architecture	7-2
Figure 7-2.	MCC Context Diagram	7-5
Figure 7-3.	Data Set Processor Context Diagram	7-6
Figure 7-4.	Annotation Processor Card Context Diagram.....	7-7
Figure 7-5.	EOS Simulator Card Data Flow Diagram	7-8
Figure 7-6.	EOS Frame Synchronizer Card Data Flow Diagram	7-9
Figure 7-7.	EOS Reed-Solomon Card Data Flow Diagram	7-10
Figure 7-8.	EOS Service Processor Card Task Allocation.....	7-12
Figure 7-9.	Header Process Data Flow Diagram	7-13
Figure 7-10.	Quality Process Data Flow Diagram.....	7-14
Figure 7-11.	Output Process Data Flow Diagram	7-15
Figure 7-12.	TRS Architecture	7-16
Figure 7-13.	TRS High-Level Data Flow.....	7-17
Figure 7-14.	TRS Data Flow	7-18
Figure 7-15.	RSH Option	7-21
Figure 7-16.	Custom Server Option	7-22
Figure 8-1.	TCP Header	8-2
Figure 8-2.	UDP Header	8-3
Figure 8-3.	IP Header Used with TCP.....	8-4
Figure 8-4.	FDDI Frame.....	8-4
Figure 8-5.	Ethernet Application Layers.....	8-5
Figure 8-6.	Ethernet Frame	8-6
Figure 9-1.	ETS HRS System Test Configuration	9-2

SECTION 1 INTRODUCTION

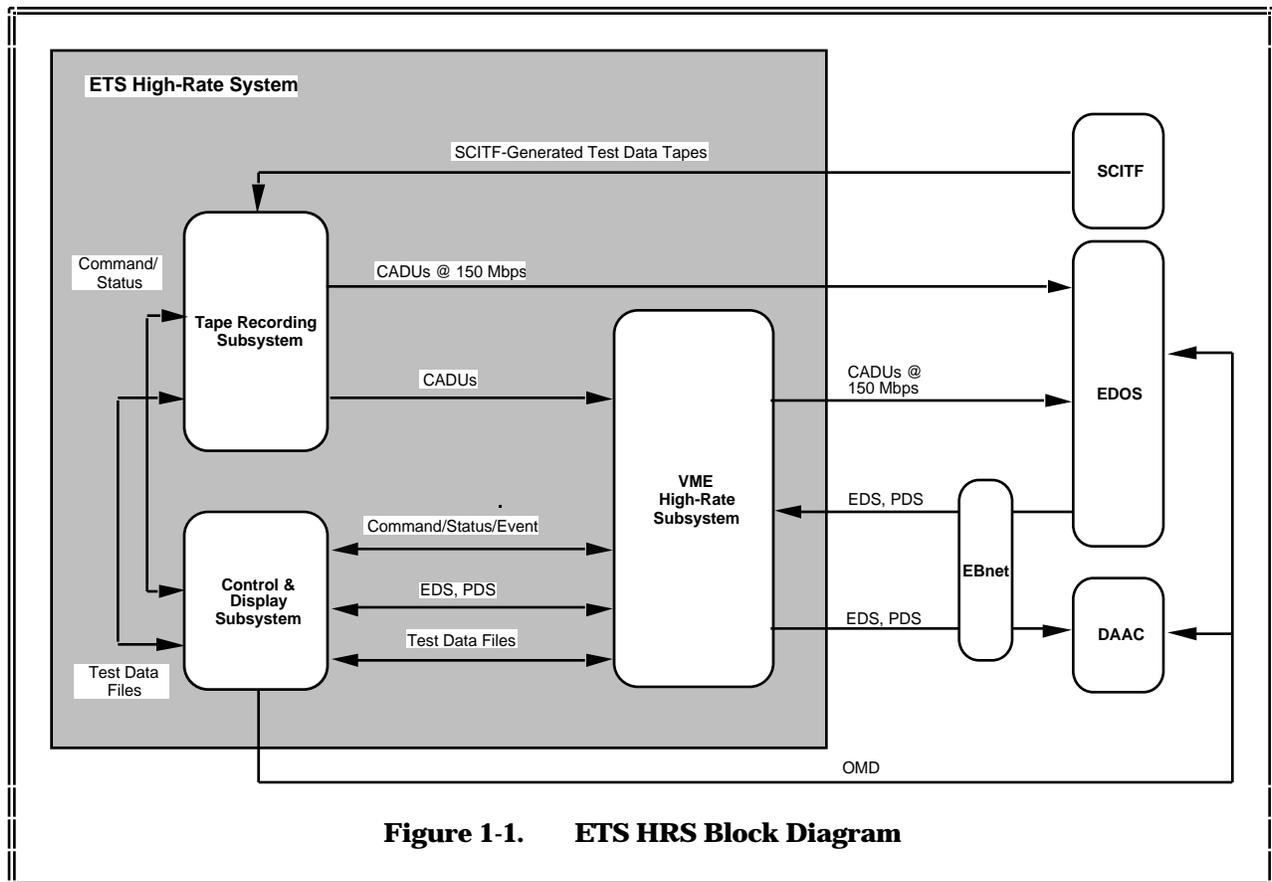
1.1 PURPOSE

This document specifies the system-level design for the Earth Observing System Data and Information System (EOSDIS) Test System (ETS) High-rate System (HRS). The HRS is being developed by the Goddard Space Flight Center (GSFC) Microelectronic Systems Branch (MSB), Code 521, as part of the ETS project to support EOSDIS system integration, testing, verification, and validation.

1.2 OVERVIEW

Figure 1-1 provides a block diagram of the ETS HRS, which consists of the Versa Module Eurocard (VME) High-rate Subsystem (VHS), Tape Recording Subsystem (TRS), and Control and Display Subsystem (CDS). These subsystems will support tests for EOSDIS return link science data processing functions. Specifically, the VHS will provide the following major functions:

- a. Transmit two channels of simulated Tracking and Data Relay Satellite System (TDRSS) Ground Terminal (TGT) return link data and clock using ETS-generated and/or user-provided test data at user-selectable rates up to 150 Mbps each.
- b. Transmit data sets to an external destination through an EOSDIS Backbone Network (EBnet) interface at rates up to 34 Mbps.
- c. Receive data sets from an external source through an EBnet interface at rates up to 34 Mbps.
- d. Accept Spacecraft Integration and Test Facility (SCITF)-generated spacecraft test data on Ampex tape media.
- e. Process SCITF-generated spacecraft test data to generate EOS Data and Operations System (EDOS)-compatible Expedited Data Sets (EDS) and Production Data Sets (PDS).
- f. Provide a local graphics-based user interface.
- g. Provide a control environment that supports automated operations.



The ETS project is required to support EOSDIS test activities in five configurations, as defined in the ETS Operations Concept document. The HRS will be used in Test Configurations 1, 2, and 3. The following paragraphs outline typical use of the HRS in each configuration.

Configuration 1

The HRS will simulate the TGT high-rate interface with EDOS by transmitting up to two channels of serial data and clock at user-selectable rates up to 150 Mbps each. The serial data stream will contain Channel Access Data Units (CADU) with or without errors. These CADUs may be provided by ETS users, or simulated by the ETS data simulation software, SCTGEN, and stored as test data files on disks or tapes. Prior to a test, the user will generate/select test data files. During the test, the HRS will transmit test data and clock at user-selectable data rates up to 150 Mbps. The HRS will update selected fields while repeatedly outputting a 4-Mbyte test pattern over one channel, via the VHS, and output a unique test pattern in a large test data set (up to 25 Gbytes) over the other channel via the TRS.

Configuration 2

The HRS will simulate the EDOS interface with the Distributed Active Archive Center (DAAC) by transmitting EDSs and PDSs through EBnet to a DAAC at rates up to 34 Mbps. These data sets may be provided by ETS users, simulated by SCTGEN, or generated by the HRS based on SCITF test data. Prior to a test, the user will select test data sets and the order in which they will be transferred. During the test, the VHS will transfer these data sets to a DAAC. The size of each data set may vary, up to 26 Gbytes, and may consist of one or more data set files of up to 2 Gbytes each.

Configuration 3

The HRS will simulate the DAAC front-end interface with EDOS by capturing EDSs and PDSs output by the EDOS. The VHS will receive and store these data sets through EBnet during a test. When the test ends, the user will be able to retrieve received data sets for analysis and verification.

Generating Data Sets from Spacecraft Test Data

The HRS will perform EDOS-compatible return link processing on user-provided test data (CADUs) to generate EDSs and PDSs. A typical source of user-provided test data is the EOS-AM SCITF. The spacecraft developer will generate test data during spacecraft testing at the SCITF and record data on Ampex tape media. The TRS will play back these tapes for the VHS to ingest the data and generate EDSs and PDSs. The CDS will retrieve and store the generated EDSs and PDSs for analysis and future testing.

Generating Frame Data Files from Spacecraft Test Data

In other cases, a user may want to insert errors into spacecraft test data stored on a tape in serial data format. The HRS supports this requirement by performing frame synchronization on the serial data and saving synchronized CADUs into a frame-aligned data file. The user will transfer the CADU data file to the CDS and edit the file using SCTGEN.

Self-Test

The HRS can verify a test data file generated by SCTGEN before data is used in operational tests. The HRS VHS will produce frame- and packet-level processing statistics and accounting information and send it to the CDS for verification.

1.3 APPLICABLE DOCUMENTS

The following documents were used as references for development of this system design. They can be used to support, further define, and clarify information in this document.

1.3.1 PARENT REQUIREMENTS DOCUMENTS

- a. Earth Observing System Data and Information System (EOSDIS) Test System (ETS) Operations Concept, May 1995.
- b. Earth Observing System Data and Information System (EOSDIS) Test System (ETS) Functional and Performance Requirements, March 1996.
- c. Earth Observing System Data and Information System (EOSDIS) Test System (ETS) System Design Specification, May 1995.
- d. Earth Observing System Data and Information System (EOSDIS) Test System (ETS) High-rate Subsystem (HRS) System Requirements Specification (SRS), November 1995.

1.3.2 STANDARDS

- a. Consultative Committee for Space Data Systems (CCSDS) Recommendations for Space Data System Standards: Advanced Orbiting Systems, Networks and Data Links, CCSDS 701.0-B-2, November 1992.

- b. Consultative Committee for Space Data Systems (CCSDS) Recommendations for Space Data System Standards: Telemetry Channel Coding, CCSDS 101.0-B-3, May 1992.
- c. Consultative Committee for Space Data Systems (CCSDS) Recommendations for Space Data System Standards: Time Code Formats, CCSDS 301.0-B-2, April 1990.

1.3.3 REFERENCES

- a. Earth Observing System (EOS) Data and Operations Systems (EDOS) and Earth Observing System Data and Information System (EOSDIS) Test System (ETS) Interface Control Document, TRW/GSFC, June 1996.
- b. Earth Observing System (EOS) Data and Operations Systems (EDOS) and TDRSS Ground Terminals (TGT) Interface Control Document, TRW/GSFC, December 1995.
- c. Earth Observing System (EOS) Data and Operations Systems (EDOS) and EOS Ground System (EGS) Elements Interface Control Document, TRW/GSFC, June 1996.
- d. Earth Observing System (EOS) Data and Operations Systems (EDOS) and EOSDIS Backbone Network (EBnet) Interface Control Document, TRW/GSFC, February 1996.
- e. Data Format Control Book for EOS-AM Spacecraft (ICD-106), Martin Marietta Corporation, Astro Space, April 1996.
- f. EDOS Functional and Performance Specification, GSFC, December 18, 1992.
- g. EOS AM HRDTE SFE-FEP System Description Document, 521-SYSM-016, Code 521/GSFC, June 1996.
- h. VLSI LZP Phase II System Description Document, 521-SYSM-018, Code 521/GSFC, August 1995.
- i. EDOS External ICD Data Format Control Document (DFCD), GSFC, Draft, June 1995.
- j. EOS Service Processor Card Hardware Definition Document, 521-H/W-058, Code 521/GSFC, TBS.
- k. EOS Simulator Card Hardware Definition Document, 521-H/W-060, Code 521/GSFC, TBS.
- l. EOS Frame Synchronizer Card Hardware Definition Document, 521-H/W-041, Code 521/GSFC, August 1995.
- m. EOS Reed-Solomon Card, Revision A, Hardware Definition Document, 521-H/W-053, Code 521/GSFC, June 1995.

1.4 **DOCUMENT ORGANIZATION**

Section 2 overviews system design, including methodologies, approaches, and studies that led to the current baseline design.

Section 3 discusses the VHS and TRS system-level designs, including system block diagrams, functional descriptions, system architectures, and external interfaces.

Section 4 provides a high-level description of the interface through which a user controls the system, and sets up and monitors operations.

Section 5 provides scenarios for typical system operations.

Section 6 describes the design of system hardware components.

Section 7 contains a detailed description of software components.

Section 8 discusses issues in system local area network configuration and communications.

Section 9 provides a general description on system test approaches and identifies test tools needed to perform system testing.

Section 10 provides a traceability matrix for mapping requirements from the ETS Functional and Performance Requirements (F&PR)/HRS System Requirements Specification (SRS) to Configuration Items (CI) in this document.

1.5 **TERMINOLOGY**

Configuration Set—set of system parameters necessary to set up the VHS for a processing session.

Data Set—set of source packets or other payload data units sorted by sensors. Packets are ordered in forward time sequence.

Expedited Data Set—set of source packets of the same sensor received from one session. Packets are organized in the same form and order as generated by the spaceborne experiment.

Frame—formatted data unit defined by the Consultative Committee for Space Data Systems (CCSDS) Version 1 Telemetry Transfer Frame (TF), Version 2 Virtual Channel Data Unit (VCDU), or CADU.

Production Data Set—set of source packets of the same sensor received from one or more sessions. Packets are organized in the same form and order as generated by the spaceborne experiment. Redundant packets are deleted from overlap regions. The VHS can verify a test data file generated by SCTGEN before data is used in operational tests. The VHS will produce frame- and packet-level processing statistics and accounting information and send it to the CDS for verification.

Real-time Processing—packets from specified sources are routed electronically to a user as they are received.

Sensor—group of sources defined by system setup; typically corresponds to a spaceborne experiment.

Session—contiguous input data stream delineated by a start of data acquisition and an end of data acquisition.

Source—spaceborne packet generator identified by Spacecraft Identifier (SCID), Virtual Channel Identifier (VCID), and Application Process Identifier (APID). A spaceborne experiment with a single APID may be defined as multiple sources (e.g., real time and playback).

Test Data Base Set File—file containing test data patterns such as a set of CADUs with Multiplexing Protocol Data Units (M_PDU) (packets) in their data fields.

Test Data Update File—file containing update information to a test data base set file. It is typically used to update VCDU and packet sequence count and timecode information in the test data base set while the base set is being output repeatedly in a long duration test. By preserving the integrity of protocol data in the base set, such as sequence counts with real-time update, a realistic long duration test can be achieved with a relatively small amount of data.

SECTION 2

SYSTEM DESIGN OVERVIEW

2.1 DESIGN METHODOLOGIES

During the ETS system-level design process, a comprehensive architectural study was conducted by the ETS HRS design team to identify a system architecture best suited to ETS needs. The result of this study is documented in the ETS System Design Specification (SDS). As its conclusion, Code 521's Very Large Scale Integration (VLSI) system architecture was selected over other candidate architectures to be the basis for the ETS HRS—because of its better performance, lower cost, lower risk, and shorter development cycle. The ETS SDS further allocates the ETS functional and performance requirements to the ETS HRS. The development team analyzed these requirements and further developed detailed system requirements for the VHS and TRS, as documented in the ETS HRS SRS. These system design activities provided the basis for detailed design of the VHS and TRS.

The goal of VHS and TRS design was to identify the most cost-effective and risk-free system configuration and components, which would meet the requirements specified in the ETS HRS SRS. The system design process included the following major phases:

- a. Selection of a system configuration and definition of system external interfaces.
- b. Identification of functional components (Commercial Off-the-Shelf [COTS], Government Off-the-Shelf [GOTS], hardware, software) in the selected system configuration.
- c. Allocation of requirements in the HRS SRS to functional components.
- d. Identification of development tasks and estimation of workload.

Selection of the VHS configuration was based on two existing systems—the EOS AM-1 Integration and Test (I&T) system and the VLSI Level Zero Processor (LZP)-II prototype. EOS AM-1 is an operational system that was developed and delivered to the AM-1 spacecraft developer to support spacecraft checkout. It features 150-Mbps data simulation and return link processing capabilities. The VLSI LZP-II prototype system was developed to demonstrate high-rate CCSDS telemetry level zero processing functions using VLSI technology. The combination of these existing systems formed a solid basis for the VHS. Most of their interfaces directly mapped to ETS requirements. Also, 80 percent of the hardware and software components can be directly reused with minimal modification.

The ETS HRS development team allocated VHS requirements to individual functional components and analyzed the matches and discrepancies. Where possible, the team adjusted system components to better meet ETS needs at a lowest cost. For example, a High-performance Parallel Interface (HiPPI)-based disk array configured in the VLSI LZP-II was replaced with a Small Computer System Interface (SCSI)-2 disk array, which reduced cost by 700 percent and satisfied VHS requirements.

The ETS HRS development team also identified areas where COTS and GOTS components did not meet requirements. These areas were studied to determine a set of additional hardware and software development tasks to cover these areas.

A main feature of Code 521 VLSI technology is its modularity. The ETS HRS team applied this feature to VHS design. The system can be reconfigured to accommodate changing requirements and future mission needs, and can be upgraded to maintain its technology edge.

2.2 DESIGN APPROACHES

2.2.1 CHANGING EDOS BASELINE AND REQUIREMENTS

Most ETS test configurations and interfaces center around the EDOS. In the past year, the EDOS system baseline has gone through a number of major changes. There were also many significant requirement and interface changes. Although the ETS architecture has been selected to be independent of EDOS architecture, these changes have certain impacts on ETS requirements, configurations, and implementations.

To minimize the impact, the ETS HRS team adopted a modular design approach—all functional components, software or hardware, are modularized to the extent possible. Each module can be integrated into a system in a plug-and-play fashion so that the system will be scalable, flexible, upgradable, and extendible. In most cases, if a requirement changes a VHS component may be added, removed, or modified to satisfy the change without affecting overall system design and configuration.

However, if the EDOS architecture, interface, performance, or schedule changes significantly, there may be a ripple effect on ETS. With EOSDIS project support, these developments will be continuously monitored, and adjustments will be made as necessary.

2.2.2 INTERFACE ISSUES

A major risk in ETS development is a lack of well-defined interfaces with EOSDIS operational systems because its development schedule is ahead of those systems. As a result, much of the preliminary design was accomplished based on assumptions and informal user input. To mitigate risk, the ETS HRS team worked closely with the EDOS development team in their interface development efforts. The ETS HRS team reviewed the working draft of the Interface Control Document (ICD) at various stages and fed back over 100 corrections and suggestions, many of which were adopted by the development team. This approach allowed the development team to keep its design closely matched to the evolving operational system.

2.2.3 TECHNICAL CHALLENGES

Although the ETS HRS is built on existing configurations and most of its components are reused from other projects, there are a number of technical challenges that are yet to be met. These include new high-performance mass storage devices, test data updates at 150 Mbps, and high bandwidth requirements on network interfaces. To understand these technical risks and find appropriate solutions, the ETS HRS team initiated a number of prototyping activities. Through these prototyping efforts, the team was able to quantify risk, identify additional development, and select COTS components that could meet requirements at a lowest cost. These prototyping efforts are summarized in Section 2.3.

2.2.4 LEVEL ZERO PROCESSING FUNCTIONS

One of the VHS functional requirements calls for generation of EDOS-like EDSs and PDSs from user-provided spacecraft test data. This requires the VHS to perform LZP functions, including frame synchronization, Reed-Solomon decoding and error correction, packet extraction and reassembly, sorting and grouping, merging, time-ordering, overlap deletion, quality and statistics generation, and data set generation.

The ETS HRS team used the Code 521-developed VLSI LZP-II prototype system as a basis for implementation of LZP functions in the VHS. (The VLSI LZP-II was completed in December 1994.)

It demonstrated full LZP capabilities at 50 Mbps and 13,000 packets per second with simulated EOS AM-1 test data. Although a prototype, the VLSI LZP-II system architecture, most of its software, including core processing algorithms, are very similar to the operational Fast Auroral Snapshot Explorer (FAST) Packet Processing System (PPS) that provides LZP for FAST science data. A part of the FAST ground data systems, the FAST PPS has undergone numerous rigorous mission tests, including anomaly handling. In addition, the FAST spacecraft uses a Solid State Recorder (SSR) as its onboard data storage. Therefore, its data scenario closely reassembles that of EOS AM-1.

In summary, reuse of the VLSI LZP-II system architecture and software to meet VHS LZP requirements represents a low risk, moderate effort approach that best suits VHS development.

2.2.5 SYSTEM TESTS

The ETS HRS is a complex system designed to fulfill complicated functional requirements and stringent performance requirements. It is a challenge to fully exercise and test these requirements. Therefore, a test engineer participated in the design process to verify requirements and test the system. With an intimate knowledge of how the system should work, the test engineer can develop an effective and efficient test plan that exercises all system components and verifies performance with minimum effort. Also, important status information is collected during testing so that problems can be identified.

2.2.6 USER INTERFACES

The Telemetry Processing Control Environment (TPCE) allows the operator to set up the ETS HRS, flow data through the system, and monitor system status from a remote (or local) location that is connected through Ethernet. TPCE runs on an HP-755 workstation using X-Windows. TPCE has a Graphical User Interface (GUI) built on HP-UX with X11R5 and Open Software Foundation (OSF) Motif 1.2.2.

TPCE receives and sends setup, status, and data through the Master Controller of the VHS. Information is passed between the two systems packed in Modular Environment for Data Systems (MEDS) messages, which provide the translation of information in a format that the VHS can process and produce. TPCE translates input from the VHS, and creates displays and updates screens accordingly. It also receives commands and data from the operator and transfers that information to the VHS. TPCE launches the TRS GUI for control and status monitoring of the TRS.

TPCE is a menu-driven window system. Multiple windows can be open, which provides the user with a flexible and efficient method to access information, monitor status, and perform tasks. Windows may be opened/closed as needed, screen positions may be changed, and screens may be closed into icons.

2.3 DESIGN STUDIES AND PROTOTYPES

2.3.1 DISK ARRAY STUDY AND EVALUATION

To support high data rate network communication and LZP functions, the VHS needs a high-performance, low-cost disk array. The disk array used in the VLSI LZP surpasses all performance requirements, but is too costly for the ETS project. To find a COTS disk array suited to ETS needs, the ETS HRS team conducted an extensive market research and identified three candidate vendors for product evaluation: Ciprico, Data General, and Storage Concept. Products from Storage Concept were quickly eliminated because of unstable and confusing product lines.

The team brought in evaluation units from Ciprico and Data General for laboratory benchmark tests. An extensive test suite was designed to determine disk array performance under various conditions simulating potential applications. These conditions included sequential/random accesses, read/write operations, cache on/off, and transfer sizes ranging from 2 Kbytes to 8 Mbytes. Test results are documented in separate test reports.

Tests revealed that although less expensive, Data General's Clarion Disk Array does not meet VHS requirements in typical ETS operation configurations. It would be sufficient for file server types of applications, but is not suitable for handling high-rate data streams. The Ciprico SCSI-2 Disk Array, Model 6712, demonstrated solid performance up to manufacturer's specification and satisfied VHS requirements. Highlights of the test results are provided in Tables 2-1 and 2-2.

Table 2-1. Ciprico SCSI-2, Model 6712, Disk Array Test Results

Ciprico (Mbps)	Read Sequential	Read Random	Write/ Cache On Sequential	Write/ Cache On Random	Write Cache Off Sequential	Write/ Cache Off Sequential
8 Kbytes	18.1	3.4	11.2	4.8	5.2	3.6
32 Kbytes	53.1	12.4/7.6	35.5	18.3	17.9	12.5
128 Kbytes	99.9	40.2	74.6	62.5	49.4	44.1
512 Kbytes	129.6	87.4	103.3	102.9	87.3	87.8
2 Mbytes	140.0	124.3	114.2	114.1	108.7	108.9

Table 2-2. Data General Clarion Disk Array Test Results

Ciprico (Mbps)	Read Sequential	Read Random	Write/ Cache On Sequential	Write/ Cache On Random	Write/ Cache Off Sequential	Write/ Cache Off Sequential
8 Kbytes	31.3	3.2	12.1	3.2	N/A	N/A
32 Kbytes	55.6	7.6	23.1	9.0	N/A	N/A
128 Kbytes	67.4	11.1	30.0	11.7	N/A	N/A
500 Kbytes	78.1	14.4	43.4	13.5	N/A	N/A
2.5 Mbytes	65.1	17.9	32.6	16.0	N/A	N/A

Based on test results, the Ciprico SCSI-2, Model 6712, disk array was selected as the VHS mass storage device.

2.3.2 FDDI INTERFACE MODULE EVALUATION

To simulate the EDOS output interface and DAAC input interface, the VHS has to implement a network interface compatible with EBnet and capable of transferring data sets at rates up to 34 Mbps. To transmit and receive data files across a network using Transmission Control Protocol/Internet Protocol (TCP/IP) and File Transfer Protocol (FTP) involves significant protocol processing overhead. Few commercial products can meet the performance requirements of sustained 34 Mbps.

A comprehensive market research identified several vendors for evaluation, including Radstone, Rockwell, Vista, and Interphase. After requesting and reviewing vendor-provided benchmarking reports, it became evident that the Interphase Fiber Distributed Data Interface (FDDI) adapter, with a VxWorks driver from AP Lab., meets the requirements. Table 2-3 summarizes performance information for various VME FDDI adapter cards.

Table 2-3. VMEbus FDDI Adapter Performance

<i>Product</i>	Radstone FDDI-1	Rockwell 1258	Vista VC-FDDI-INP II	Interphase 5211
<i>TCP/IP Transfer Rate*</i>	13	32	13.6	48.8

*Based on vendor-provided data.

Currently, the ETS HRS team is negotiating the evaluation units from Interphase for in-house benchmark testing. The final decision will depend on in-house test results.

2.3.3 SCSI-2 INTERFACE EVALUATION

The EOS Simulator Card will use a SCSI-2 mezzanine card to retrieve simulation data from a 4-Gbyte SCSI hard disk. The SCSI-2 mezzanine card uses an NCR53C710 SCSI processor chip to interface with the SCSI hard disk and the EOS Simulator Card. The NCR53C710 is a fast SCSI-2 controller chip with a (vendor-specified) maximum synchronous transfer rate of 10 Mbytes/sec. Though the specifications for the ETS HRS system simulation data transfers and update data transfers are theoretically within the specifications of the SCSI processor chip, further testing will be performed to validate their performance.

SECTION 3 SYSTEM DESIGN

3.1 INTRODUCTION

The ETS HRS consists of three basic elements: VME High-rate Subsystem (VHS), Tape Recording Subsystem (TRS), and Control and Display Subsystem (CDS). The VHS is a high-performance return link telemetry data simulation and processing system. The VHS is controlled through the CDS running TPCE for the VHS. The TRS implements the second high-rate serial data stream, i.e. 150 Mbps, to simulate the TGT interface with EDOS. The TRS will also read spacecraft-generated data from SCITF-generated tapes for data manipulation, processing, and creating data products. The CDS is an HP workstation that provides the control, monitor, and reporting functions for the ETS HRS. TPCE is a UNIX-based control environment that supports manual and automatic operations for the VHS. TPCE features a GUI, scheduling capability, and many standard operational functions. The TRS will be controlled by a GUI also running on the CDS.

3.2 VME HIGH-RATE SUBSYSTEM

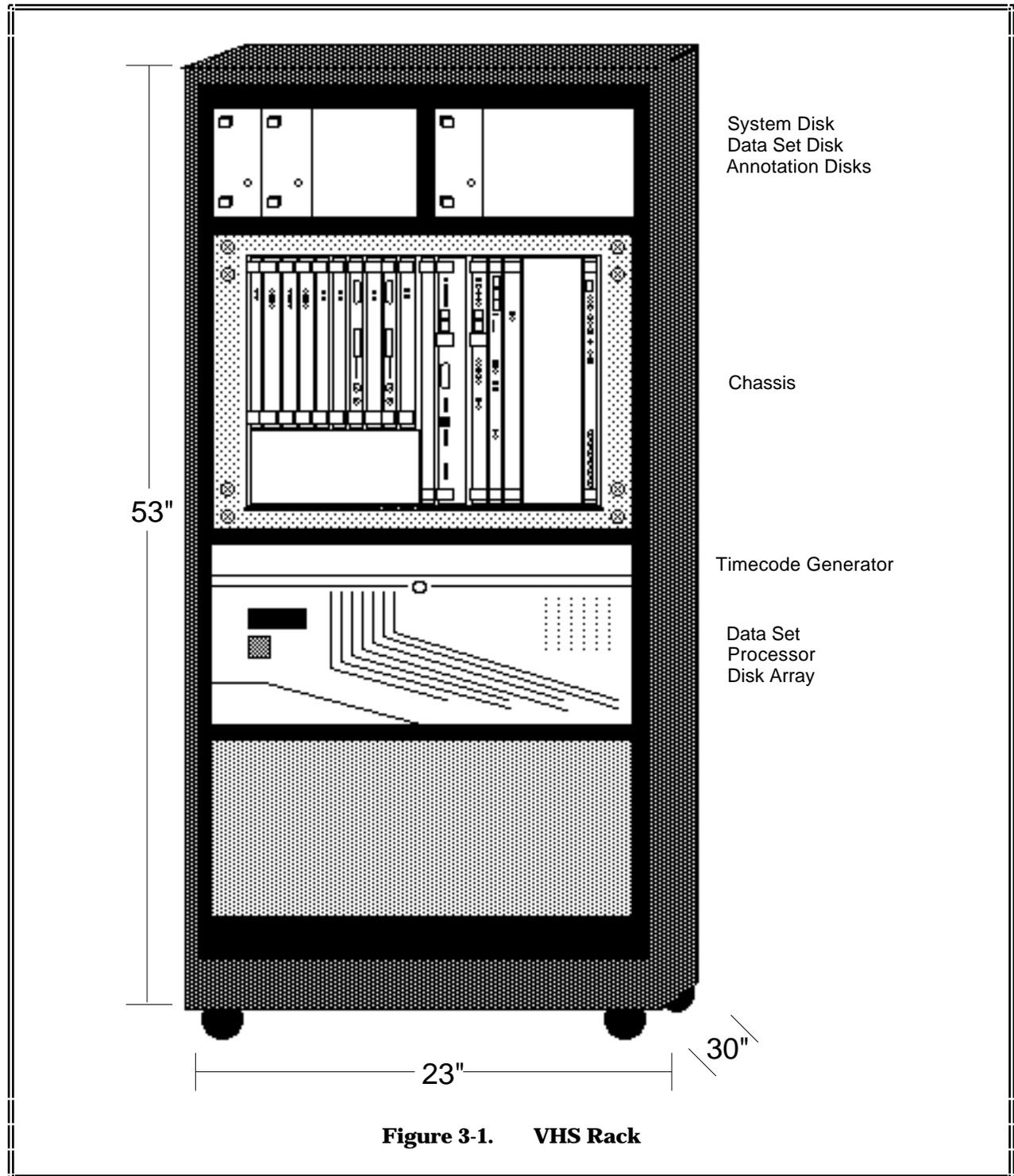
3.2.1 VHS FUNCTIONAL OVERVIEW

The VHS is a high-performance return link telemetry data simulation and processing system. It is based on the operational EOS AM-1 Science Formatting Equipment Front-end Processor (SFE-FEP) and prototype VLSI LZP system architectures. It features a VME open-bus architecture, NASA-developed telemetry processing Application-specific Integrated Circuits (ASIC) and boards, state-of-the-art COTS components, and real-time software. The VHS is controlled through the CDS running TPCE for the VHS. Primary functions of the VHS are summarized as follows:

- a. Simulate TGT high-rate outputs to test EDOS input by transmitting a simulated TGT Ku-band return link data stream at sustained data rates up to 150 Mbps.
- b. Simulate EDOS output to test DAAC input by transferring data sets to a destination via EBnet interface at sustained data rates up to 34 Mbps.
- c. Simulate DAAC front-end functions to test EDOS output by receiving data sets from a source via EBnet interface at sustained data rates up to 34 Mbps.
- d. Support tests with user-provided data by processing SCITF-generated spacecraft Thermal and Vacuum (TV) test data and generating EDOS-compatible EDSs and PDSs.
- e. Process SCITF-generated spacecraft TV data to generate CADU files so that users can manipulate data (i.e., insert errors).

3.2.2 VHS SYSTEM ARCHITECTURE

The VHS is based on the VME standard open-bus architecture. COTS components are integrated with Code 521-developed telemetry processing functional components in a 21-slot VME chassis. The chassis is housed in a standard 19-inch equipment rack, along with a disk array, disk drives, and a power supply (Figure 3-1).



While COTS components will provide system base functions such as system control, network interface, data storage, and memory buffers, Code 521-developed functional components will perform most telemetry simulation and processing functions. These functions include data simulation, frame synchronization, Reed-Solomon decoding and error correction, packet processing, and data set processing.

The VHS uses a disk array to provide the large data storage needed for high-speed data set transfers across EBnet. Another application of the disk array is to support data set processing by buffering sorted source packets.

3.2.2.1 VHS System Block Diagram

Figure 3-2 provides a detailed functional block diagram of the VHS showing 14 functional VME boards integrated in a VMEbus environment, supported by mass storage devices such as disk drives and a disk array. The VHS is comprised of the following VMEbus modules:

- a. Master Controller Card: Motorola MVME-167 Single-board Computer (SBC) (33-MHz, 8 Mbytes) and interface connector paddle card.
- b. FDDI Interface Card for Local Area Network (LAN): Interphase 5211 FDDI.
- c. FDDI Interface Card for Wide Area Network (WAN) (EBnet): Interphase 5211 FDDI.
- d. Three static Random Access Memory (RAM) boards: MicroMemory MMI 6740C/16M.
- e. EOS Simulator Card, Revision A: Code 521/GSFC, #G15427414.
- f. EOS Frame Synchronizer Card: Code 521/GSFC, #G1527415.
- g. EOS Reed-Solomon (ERS) Card, Revision A: Code 521/GSFC, #G1527413.
- h. EOS Service Processor Card: Code 521/GSFC, #G1527421.
- i. 512-Mbyte dual-ported memory buffer: MMI 6390D/512M.
- j. Data Set Processor 2: Nitro60/16M.
- k. Data Set Processor 1: MVME-167 (33 MHz, 8 Mbytes).
- l. Annotation Processor: MVME-167 (33 MHz, 32 Mbytes).
- m. Timecode Processor: Datum BC336VME.

The VHS is also comprised of the following storage devices:

- a. System Disk: 4-Gbyte Seagate Barracuda disk drive assembly.
- b. Simulation Disk: 4-Gbyte, SCSI-2 Seagate Barracuda disk drive assembly.
- c. Two Annotation Disks: 4-Gbyte Seagate Barracuda disk drive assembly.
- d. Data Disk: Ciprico SCSI-2 Disk Array, Model 6712 (32 Gbytes).

The Master Controller serves as a gateway between the VHS and the CDS, its workstation-based remote host. Through an Ethernet connection, the Master Controller will receive commands from the CDS, disseminate commands, and send subcommands to the corresponding functional modules. The Master Controller will also collect command execution and data processing status and report it to the CDS.

The FDDI Interface Controller for LAN provides a high-speed network connection between the VHS and CDS for transferring large test data files. The FDDI Interface Controller for WAN provides a high-speed connection between the VHS and EBnet for transferring test data sets to a

DAAC, or receiving test data sets from a DAAC.

Static memory of 48 Mbytes provides a high-speed global memory buffer for global data structures and other global data storage.

The EOS Simulator Card generates simulated return link CADUs at rates up to 150 Mbps. The test pattern of a size up to 4 Mbytes should be generated offline prior to a test. During testing, the test pattern can be repeatedly output at a user-selected rate. Selected data fields such as sequence count and timecode can be updated during data transmission with pregenerated values.

The EOS Frame Synchronizer and EOS Reed-Solomon Cards perform frame-level processing on an incoming serial CADU stream, including frame synchronization, Reed-Solomon decoding and error correction, and filtering of fill CADUs. They also generate quality annotation information, which is appended to the CADUs.

The EOS Service Processor Card extracts packet pieces from frame-processed CADUs and reassembles them into source packets. These packets are sorted by ID into data records in the 512-Mbyte dual-ported memory, and stored in the data disk through the Data Set Processor. The EOS Service Processor also generates annotation data for each packet containing quality and timecode information that is stored in the Annotation Disk through the Annotation Processor.

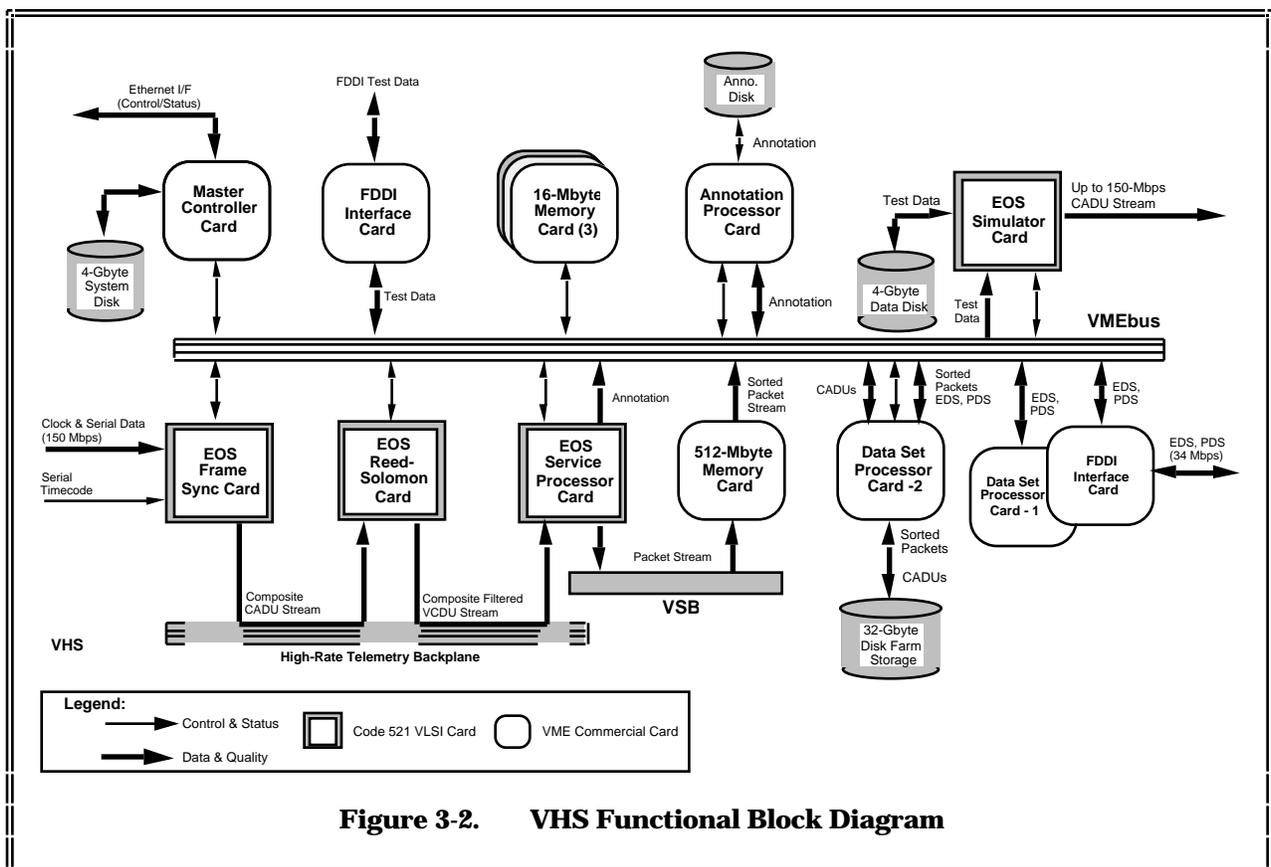


Figure 3-2. VHS Functional Block Diagram

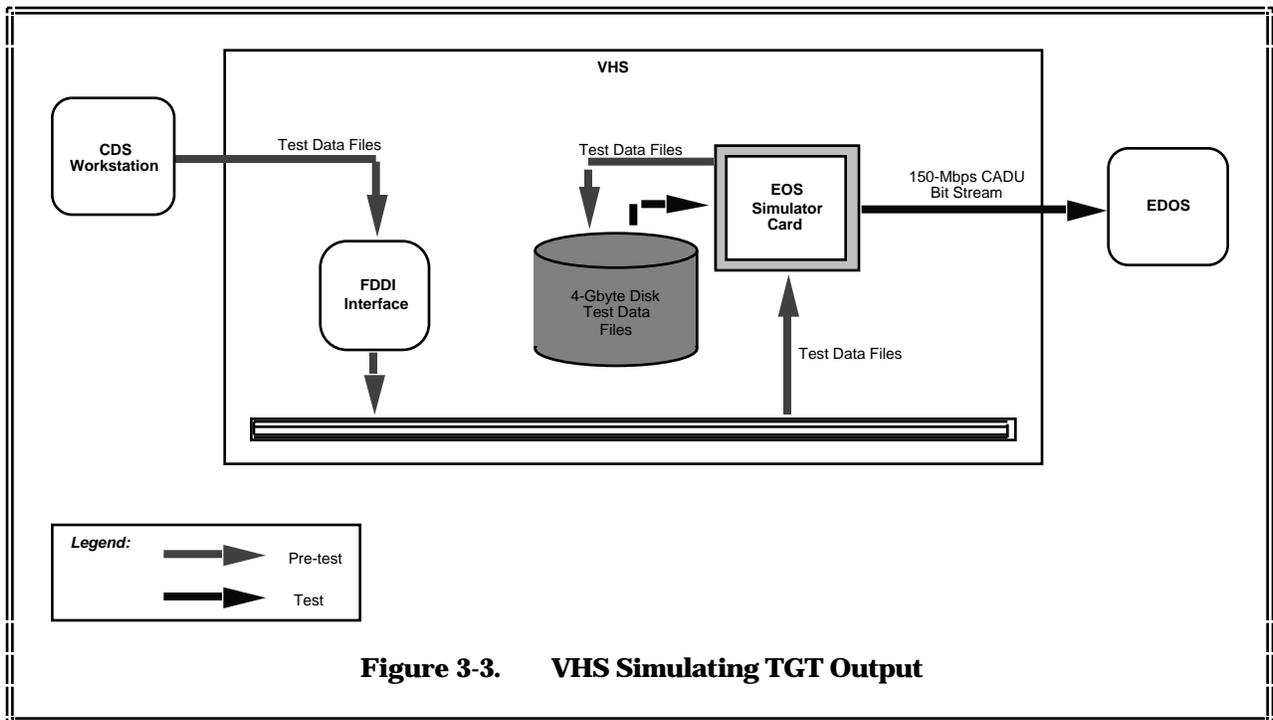
When a session ends, the Annotation Processor processes the annotation data to determine time order and redundancy of data. It produces Data Set Assembly Tables (DSAT), one for each data set. A DSAT consists of a data set construction record and a set of instructions on how to build a data set.

DSP-2 uses a DSAT to construct and output a data set. Packets will be retrieved and output according to instructions in the DSAT (so that packets are time-ordered and redundant packets are deleted). DSP-1 acts as the FDDI Controller to output data sets on the output FDDI link.

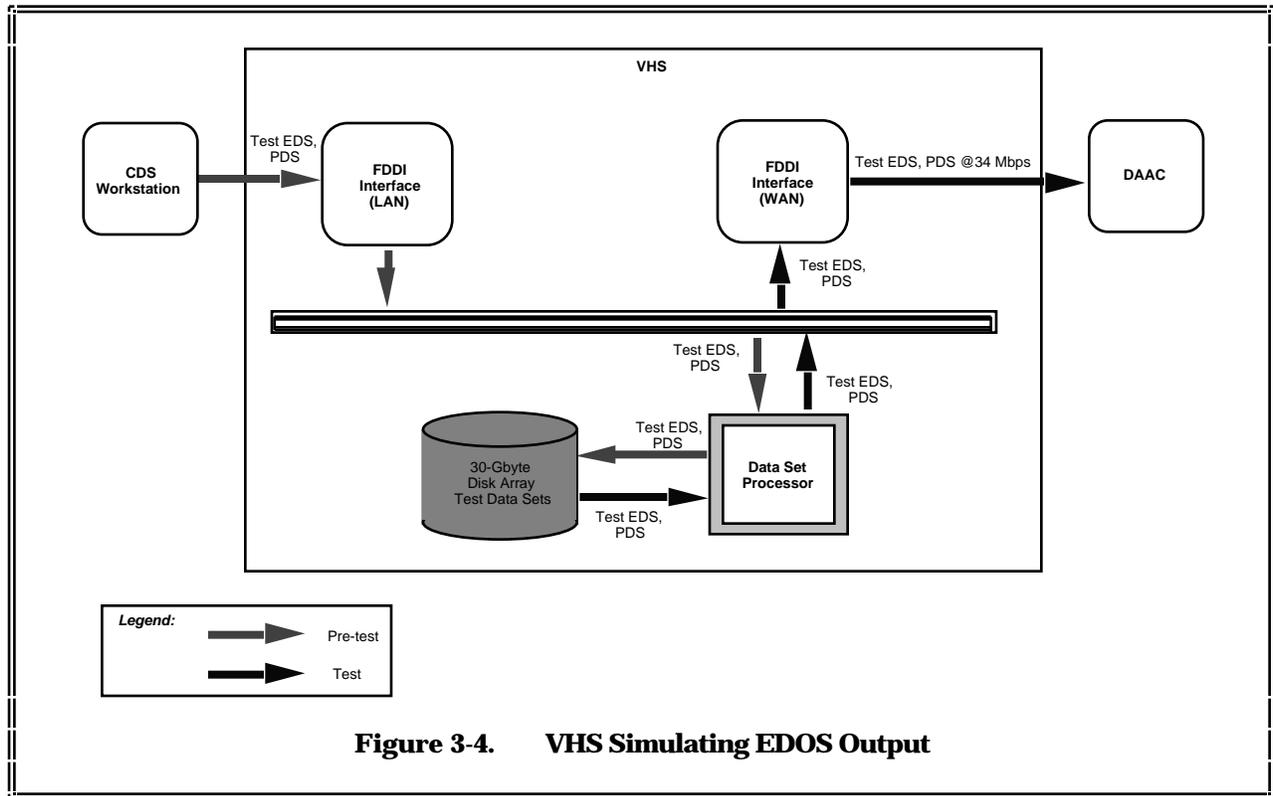
3.2.2.2 VHS Data Flow Diagrams

The VHS will be used to support five major functions for different test configurations (refer to Section 3.2.1). This section describes the data flow of these functions with reference to data flow diagrams. Note that different notations are used to distinguish data flows during the pre-test, test, and post-test phases.

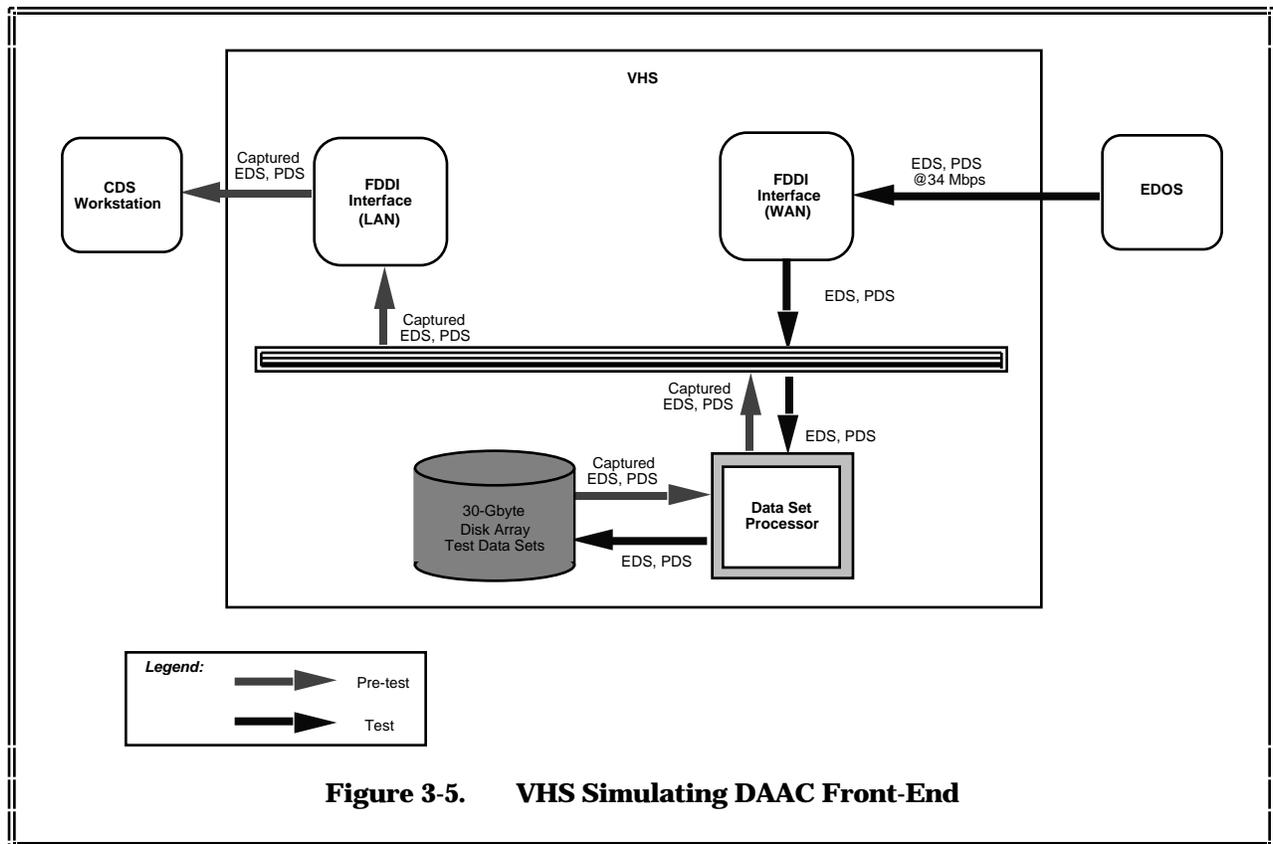
- a. Simulate TGT High-Rate Output: Figure 3-3 illustrates VHS data flow when it simulates TGT high-rate output to EDOS. Prior to a test, test data files are loaded from the CDS workstation to the VHS 4-Gbyte simulation disk through the FDDI interface. During the test, the EOS Simulator Card reads the file and transmits data at a user-specified rate up to 150 Mbps.



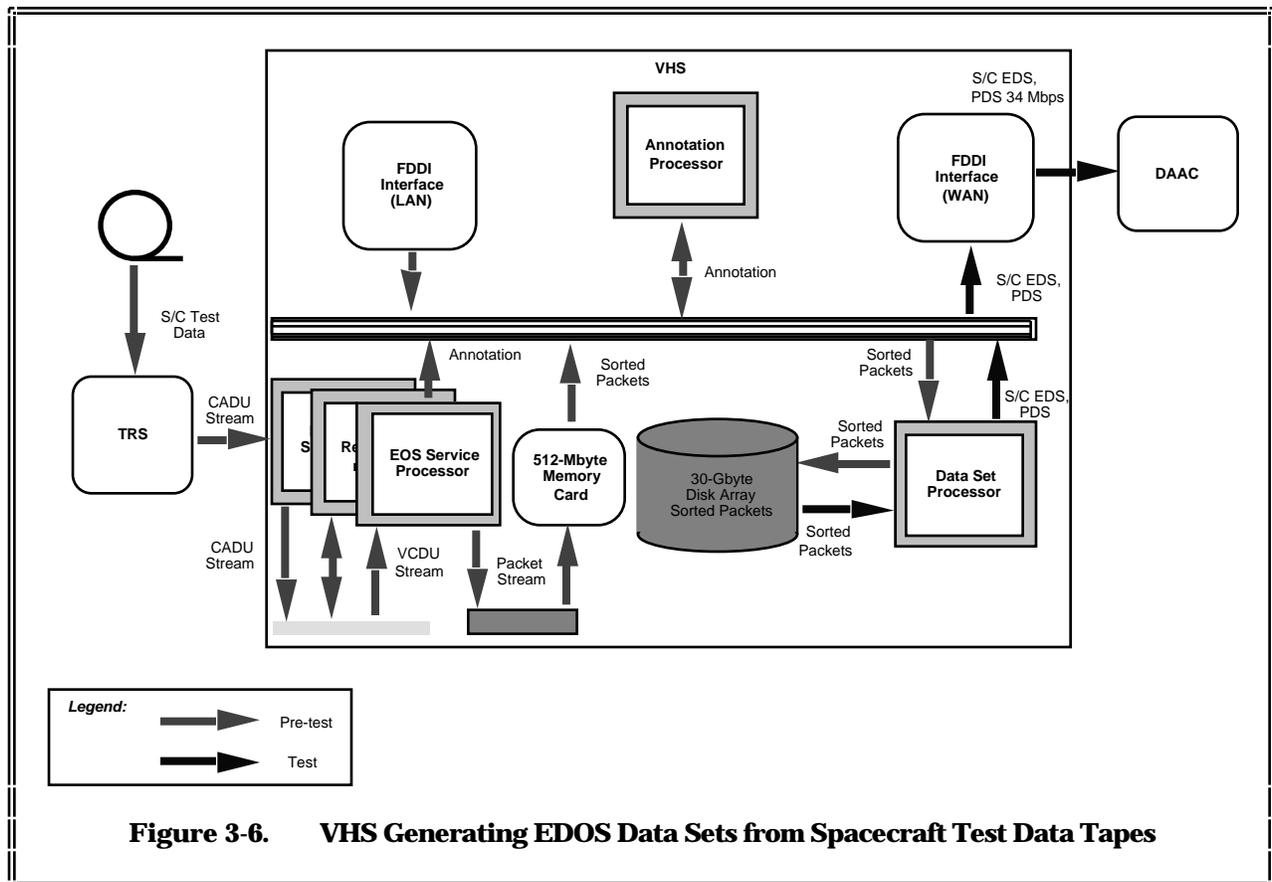
- b. Simulate EDOS Output: Figure 3-4 illustrates VHS data flow when it simulates EDOS output to a DAAC. Prior to a test, test data files are loaded from the CDS workstation to the VHS 30-Gbyte disk array through the FDDI interface and Data Set Processor, which provides an interface between the disk array and the VMEbus. During the test, the Data Set Processor initiates an FTP session and transfers the file to a user-specified remote destination at rates up to 34 Mbps.



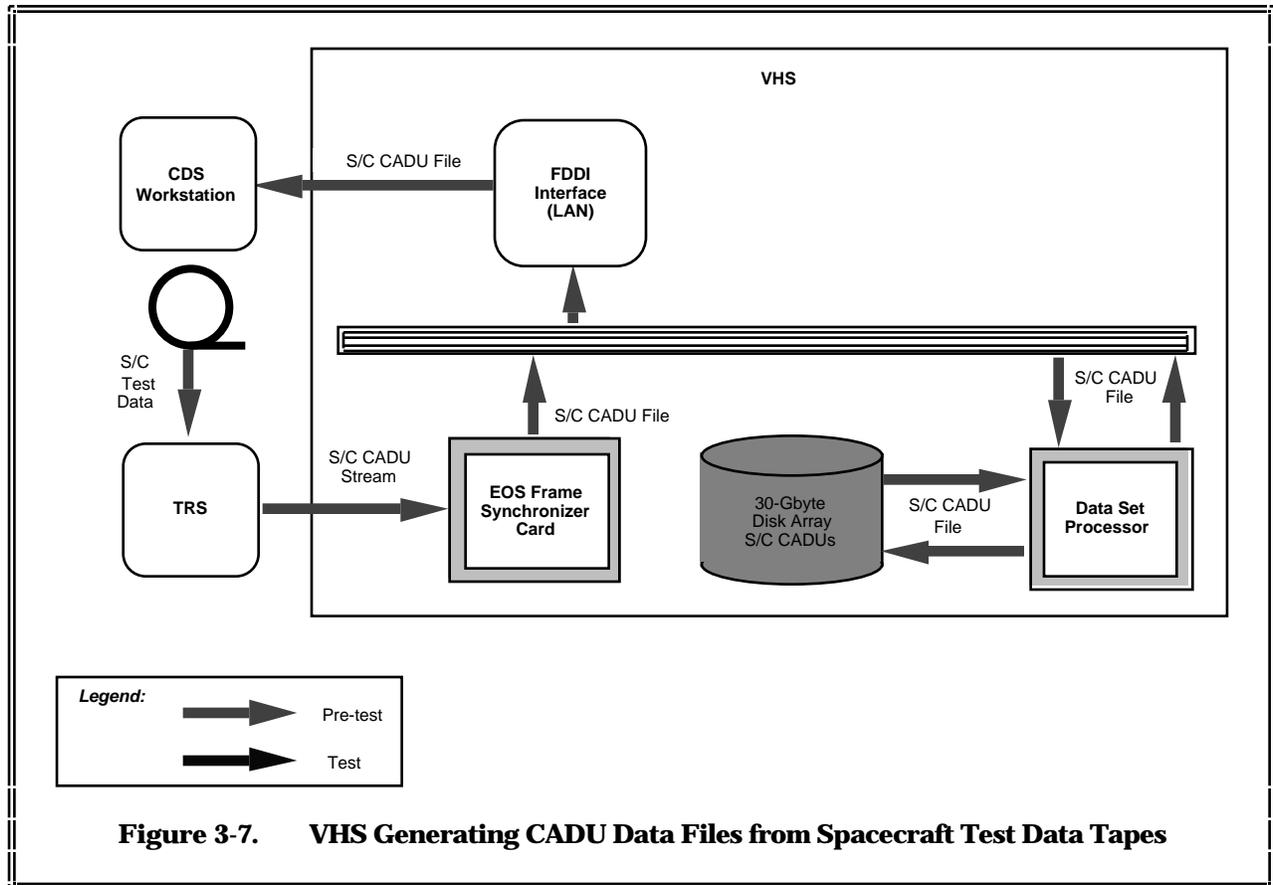
- c. Simulate DAAC Front-End: Figure 3-5 illustrates VHS data flow when it simulates a DAAC front-end to receive data from EDOS. During a test, the VHS receives data sets from EDOS via FDDI (WAN) interface and stores the data sets in the 30-Gbyte disk array. ETS could provide simulated CADUs from which EDOS could generate data sets (EDSs and PDSs) for transmission to the VHS. The VHS can capture data sets at rates up to 34 Mbps. After the test, the Data Set Processor initiates an FTP session and transfers captured data sets to the CDS workstation for data verification and analysis.



- d. **Generate EDOS Data Sets from Spacecraft Test Data Tapes:** Figure 3-6 illustrates VHS data flow when it generates EDOS-compatible data sets such as EDSs and PDSs from a spacecraft test data tape. Prior to the test, the TRS will create a Sony tape of the spacecraft test data. This data, if in packet form, will be encapsulated into CADUs and stored on tape. During the test, this data will be serially fed into the VHS, where it will be level zero processed to generate EDSs and PDSs. The EDSs and PDSs are transferred directly to EDOS via the high-speed FDDI network, or stored on the disk array and sent to the CDS for data verification and analysis.



- e. **Generate CADU Data Files from Spacecraft Test Data Tapes:** Figure 3-7 illustrates VHS data flow when it generates CADU data files from a spacecraft test data tape. Prior to the test, the TRS will create a Sony tape of the spacecraft test data. This data, if in packet form, will first be encapsulated into CADUs and stored on a Sony tape. Prior to the test, this data will be serially fed into the VHS, where it will be frame synchronized and saved as a frame-aligned CADU data file on the 30-Gbyte disk array. This CADU data file is transferred directly to EDOS via the high-speed FDDI network, or stored on the disk array and sent to the CDS for user-specified error insertion and modifications using SCTGEN.



3.2.3 VHS EXTERNAL INTERFACES

The VHS transmits data and communicates with EOSDIS elements and other ETS components through five types of interfaces: Emitter-coupled Logic (ECL) serial port, Ethernet, RS-232, FDDI network, and EBnet (see Figure 3-8). These interfaces are primarily low-level external interfaces. The TCP/IP protocol is used for network access through application layers.

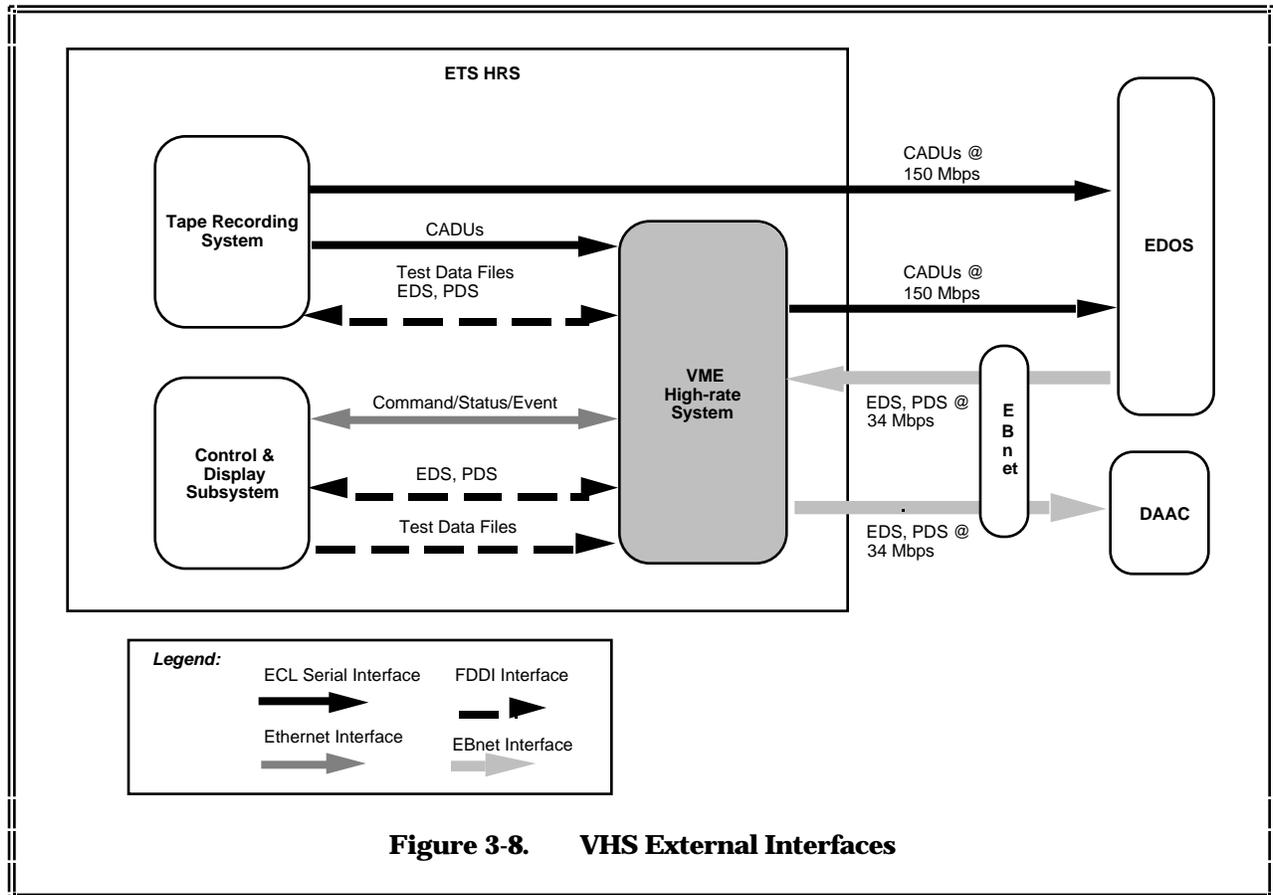


Figure 3-8. VHS External Interfaces

3.2.3.1 ECL Serial Ports

The VHS will provide an input serial port and an output serial port. Both ports will comply with the TGT-EDOS ICD. Ports will comply with the 100K ECL standard with differential data and clock signals. The maximum length for each connection will be 25 feet.

Through the output port, the VHS will output test data and clock to the EDOS Data Interface Facility (DIF) as the TGT would in real operations. It will support data rates up to 150 Mbps. Through the input port, the VHS will receive test data and clock from a data source at data rates up to 150 Mbps. Data sources can be the TRS, test data generated by the VHS itself, or other external data sources with compatible interfaces.

3.2.3.2 Ethernet

The VHS will provide an Ethernet interface that will conform to 10Base2 of the Institute of Electrical and Electronic Engineers (IEEE) 802.3 standard, also known as Thin Ethernet. All transmissions over the network are based on TCP (defined in MIL-STD 1778) and IP (defined in MIL-STD 1777).

The VHS will use Ethernet to communicate with other ETS components over the ETS Ethernet LAN. Specifically, the VHS will communicate with the CDS and TRS for the following information:

- a. Commands from the CDS to the VHS.
- b. Status messages from the VHS to the CDS.
- c. Event messages from the VHS to the CDS.
- d. Test data files from the TRS to the VHS.

The Ethernet interface will provide an effective bandwidth of 3 Mbps, excluding protocol overheads.

3.2.3.3 FDDI Network

The VHS will provide a FDDI network interface, which will conform to International Standards Organization (ISO) 9314 FDDI standards for Physical Layer Protocol (PHY), Media Access Control (MAC), and Interface Physical Layer Medium Dependent (PMD). All transmissions over the network are based on Internet protocols, such as FTP, TCP, and IP.

The VHS will use FDDI to exchange data files with the CDS through the ETS FDDI LAN. These data files include:

- a. Test data files from the CDS to the VHS.
- b. Data sets from the VHS to the CDS.

3.2.3.4 EBnet Network

The VHS will provide a high-rate EBnet interface that will comply with the EDOS-EBnet and EDOS-DAAC ICDs. The VHS will use the EBnet interface to exchange data sets (EDSs, PDSs) with EOSDIS elements under test, such as the EDOS and DAACs. Through this interface, the VHS will output data sets to a user-specified DAAC at a maximum data rate of 34 Mbps. Through the same interface, the VHS will receive data sets from the EDOS at rates up to 34 Mbps. Table 3-1 describes each interface and protocol. (Figure 3-8 illustrates the interfaces between the CDS and TRS that are used to control the TRS when used in conjunction with the VHS.)

Table 3-1. VHS External Interfaces and Protocols

Interface	Type	Physical	Protocol	Data	Data Rate (Mbps)	Application
Ethernet	Network	Thin Ethernet (10Base2)	TCP/IP	MEDS message (command, status, event) Data snapshot	3	Communication with CDS, interface with TRS
Ethernet (dedicated)	Network	Thin Ethernet (10Base2)	TCP/IP	MEDS message (event)	3	Annotation Processor communication with CDS
FDDI (LAN)	Network	ISO 9314 PHY, MAC, PMD	TCP/IP	Test data files Captured data sets	10	Communication with CDS
FDDI (WAN)	Network	ISO 9314 PHY, MAC, PMD	TCP/IP	Data sets (EDSs, PDSs)	34	Interface with EDOS, DAAC via EBnet
Serial #1	Serial Output	100K ECL Differential Data and Clock		CADU bit stream	150	Interface with EDOS
Serial #2	Serial Input	100K ECL Differential Data and Clock		CADU bit stream	150	Interface with TRS

3.3 TAPE RECORDING SUBSYSTEM

3.3.1 TRS FUNCTIONAL OVERVIEW

The TRS implements the second high-rate serial data stream (i.e., 150 Mbps) to simulate the TGT interface with EDOS. The TRS will read spacecraft-generated data from SCITF-generated tapes for data manipulation, processing, and creating data products. The TRS is controlled via its own GUI, but is launched through the Menu Controller on the CDS. The TRS is a resource employed to meet the 150-Mbps data rate and large data storage requirements of EOSDIS testing, as well as the requirement for media compatibility with the SCITF.

The TRS allows very large test data files (up to 96 Gbytes on Sony tape) to be created, manipulated, and played back at rates up to 150 Mbps. It also supplies a playback capability for SCITF test data files and data sets that are supplied on 48-Gbyte Ampex tapes.

The TRS is a distributed system comprised of software running on one or more heterogeneous workstations that control a collection of heterogeneous tape recorders and associated interfacing hardware. Primary functions of the TRS are summarized as follows:

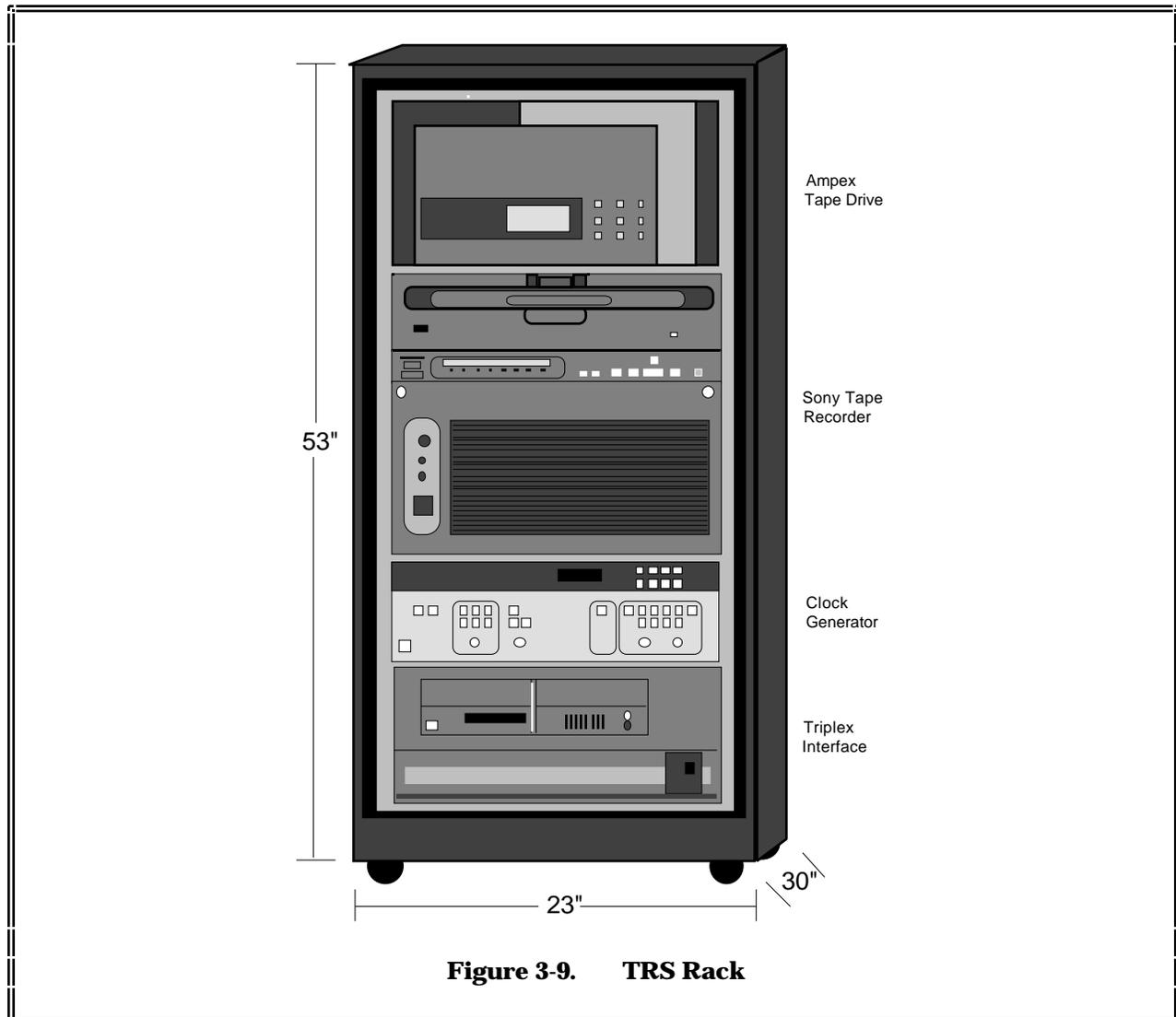
- a. Transfer SCITF-generated test data from Ampex 19mm cartridge tapes to Sony ID-1 cassette tapes at data rates up to 100 Mbps.
- b. Receive SCTGEN-generated or SCTGEN-modified test data to be subsequently output as a serial CADU stream with a 1-kbps resolution.
- c. Simulate TGT high-rate output to test EDOS input by transmitting a simulated TGT Ku-band return link data stream at sustained data rates up to 150 Mbps.

3.3.2 TRS SYSTEM ARCHITECTURE

The TRS architecture is based on constraints that must be dealt with in the TRS design. This includes the TRS hardware and the ETS HRS context in which the TRS exists, including potential

users and data flows between users and the TRS. The TRS consists of COTS hardware that is integrated with Code 521-developed software functional components that will provide system base functions such as system control, network interface, and status monitoring. The TRS consists of a Sony tape drive and an Ampex tape drive. The Sony drive functions as the operational unit in the TRS during EOSDIS testing. The Ampex drive provides media compatibility with the SCITF team, which will supply spacecraft and instrument-generated test data sets on Ampex tapes to ETS. The Ampex drive will be used by ETS for dubbing test data sets to the Sony drive as an offline function.

The two tape drives are mounted in a standard 19-inch equipment rack, along with the Triplex interface and necessary connectors, as shown in Figure 3-9.



3.3.2.1 System Block Diagram

Figure 3-10 provides the detailed functional block diagram of the TRS. It illustrates the COTS hardware interacting with the Code 521-developed software running on the CDS to meet requirements levied on the TRS. The TRS is comprised of the following COTS hardware modules:

- a. Sony high-performance 19mm tape drive (Model SONY-DR1000).
- b. Triplex interface (Model TSCSTX 310).
- c. Ampex tape drive (Model DCRSI 107).
- d. Hewlett-Packard clock generator (Model 8130A, 300 MHz).

The TRS contains one Sony 19mm high-performance tape drive, which uses American National Standards Institute (ANSI) standard data formats and 19mm D-1 tape cartridges that can store data up to 96 Gbytes. The drive is based on field-proven helical scan technology for high data rate and density. The maximum data transfer rate is 256 Mbps over an ECL serial data interface. The data seek time from Beginning-of-Tape (BOT) to End-of-Tape (EOT) is less than 180 seconds.

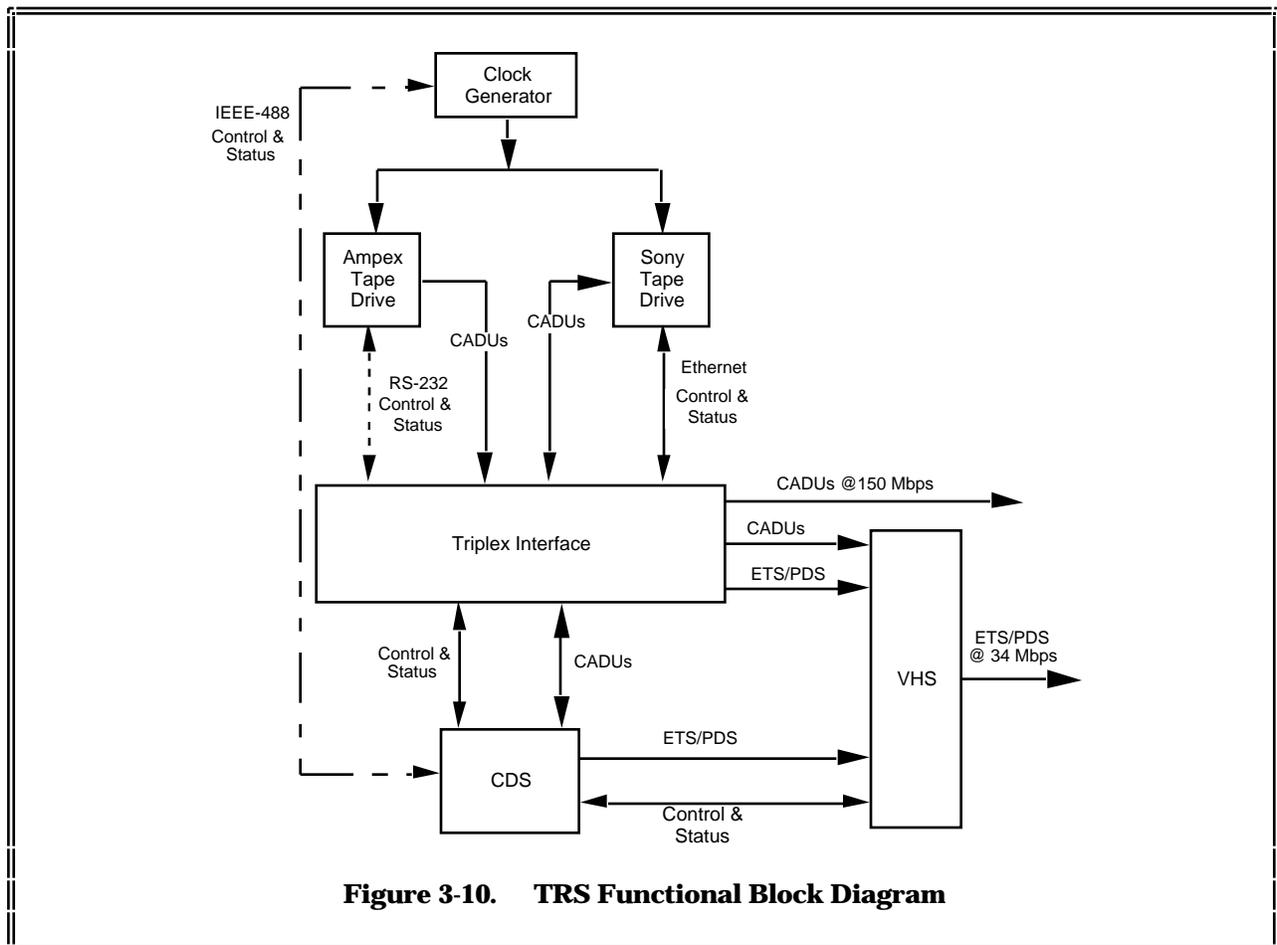


Figure 3-10. TRS Functional Block Diagram

The Sony tape drive is coupled with a Triplex interface that provides a high-level tape drive interface, variable-rate capability, Ethernet interface for operation control, and FDDI network for

data transfer. A Clock Signal Generator (CSG) serves as an external clock source to the Triplex unit for tape playback.

The Triplex control interface uses the classic UNIX device model. It supplies a command set similar to the UNIX *mt* (magnetic tape utility) commands. The Triplex interface also manages tape directory information encoded on the tapes themselves. These features allow tapes and data sets to be manipulated like a disk-based file system.

The TRS control interface will use the functions of the Triplex interface to implement higher-level TRS operations on the Sony drives. Control of the clock signal generator through its IEEE 488 port will be from the CDS. The TRS interface will hide the details of these interfaces behind high-level commands.

The TRS contains a single Ampex tape drive based on 1-inch, transverse scan, rotary digital recording technology. The drive can record and reproduce at any user data rate from 0 to 107 Mbps through an ECL serial data interface. Each tape cartridge can hold up to 48 Gbytes.

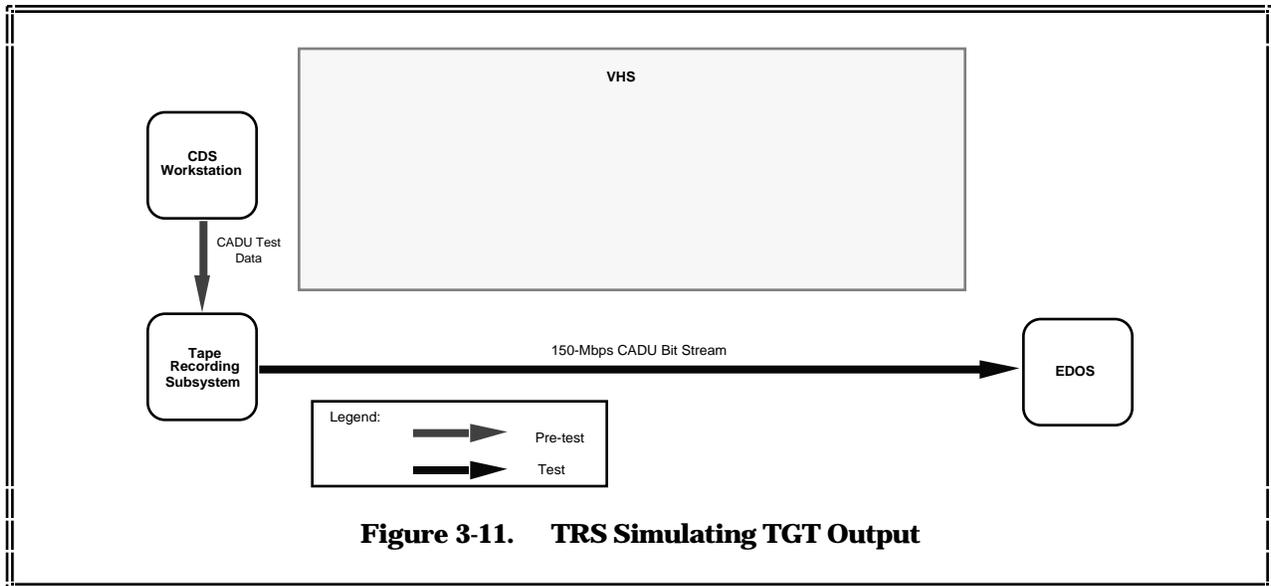
The drive control interface is implemented through an American Standard Code for Information Interchange (ASCII) command set communicated via an RS-232 port. The TRS control interface will use the functions of this interface to implement higher-level TRS operations. It will hide the details of this interface behind high-level commands.

There are no files or directories within the Ampex control interface and tape formatting. However, a file directory could be created at the beginning of each tape using the existing capabilities of the Ampex control interface. A convention must be agreed upon by the SCITF and ETS teams in order to implement this function.

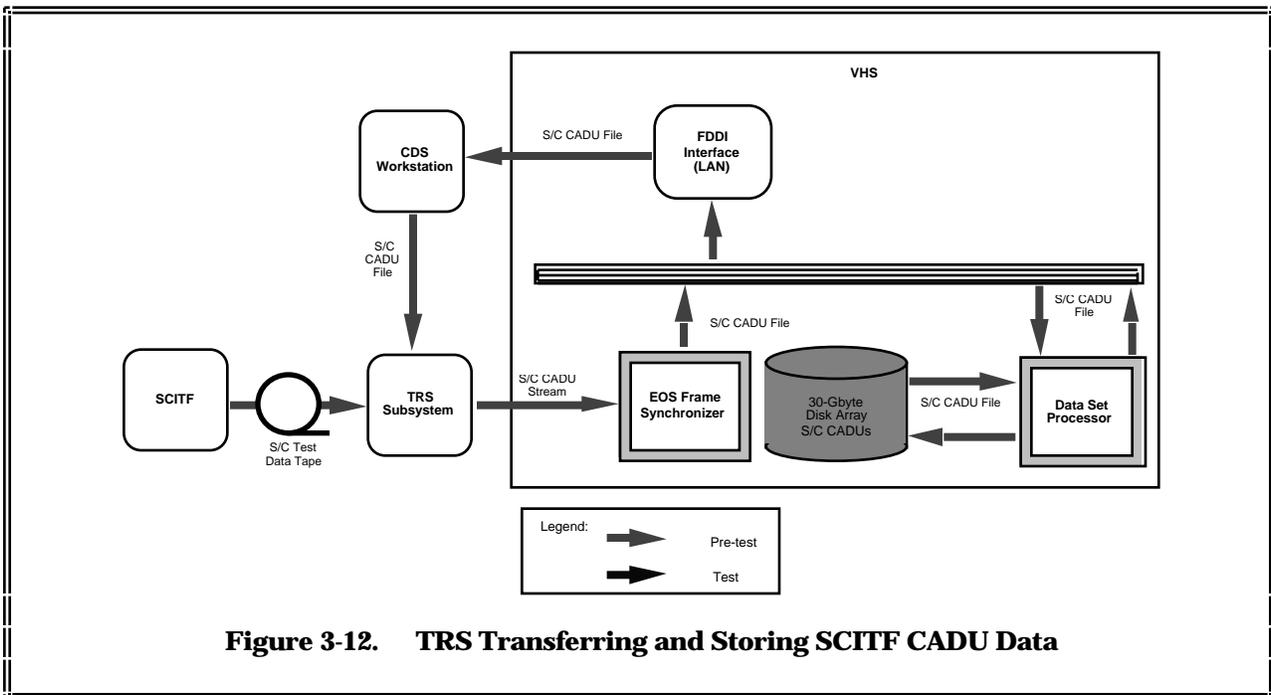
3.3.2.2 TRS Data Flow Diagrams

The TRS will be used to individually support one major function for a test configuration. In addition, the TRS, in conjunction with the VHS, will support two functions as listed in Section 3.2.1. This section describes the data flow of these functions and provides data flow diagrams. Note that different notations are used to distinguish data flows during the pre-test, test, and post-test phases.

- a. Simulate TGT High-Rate Output: Figure 3-11 illustrates TRS data flow when it simulates TGT high-rate output to the EDOS. Prior to the test, test data files are loaded from the CDS workstation to the Sony tape drive through the FDDI interface. During the test, the TRS, via the Sony tape drive and Triplex interface, transmits CADU data to EDOS at a user-specified rate up to 150 Mbps.



- b. Transfer, Modify, Output SCITF-Generated Test Data from Ampex Tapes: Figure 3-12 illustrates the TRS receiving Ampex tapes of SCITF-generated test data. The TRS will create a Sony tape of the spacecraft test data. This data, if in packet form, will be encapsulated into CADUs using SCTGEN. This data will be serially fed into the VHS, where it will be frame synchronized and saved as a frame-aligned CADU data file on the 30-Gbyte disk array. This CADU data file is transferred directly to EDOS via the high-speed FDDI network, or sent to the CDS for user-specified error insertion and modifications using SCTGEN.



3.3.3 TRS EXTERNAL INTERFACES

The TRS transmits data and communicates with EOSDIS elements and other ETS components through five types of interfaces: ECL serial port, Ethernet, RS-232, IEEE 488, and removable storage media Ampex 19mm cartridge tapes (see Figure 3-13).

3.3.3.1 ECL Serial Ports

The TRS will provide an input serial port and an output serial port. The output port will comply with the TGT-EDOS ICD; the input port will communicate with the serial output from the Ampex tape drive. The ports will comply with the 100K ECL standard with differential data and clock signals. The maximum length for each connection will be 25 feet.

Through the output port, the TRS will output test data and clock to the EDOS DIF as the TGT would in real operations. It will support data rates up to 150 Mbps. Through the input port, the TRS will receive test data and clock from a data source at data rates up to 150 Mbps. Data sources can be the TRS, test data generated by the VHS itself, or other external data sources with compatible interfaces.

3.3.3.2 Ethernet

The TRS will provide an Ethernet interface, which will conform to 10Base2 of the IEEE 802.3 standard, also known as Thin Ethernet. All transmissions over the network are based on TCP (defined in MIL-STD 1778) and IP (defined in MIL-STD 1777).

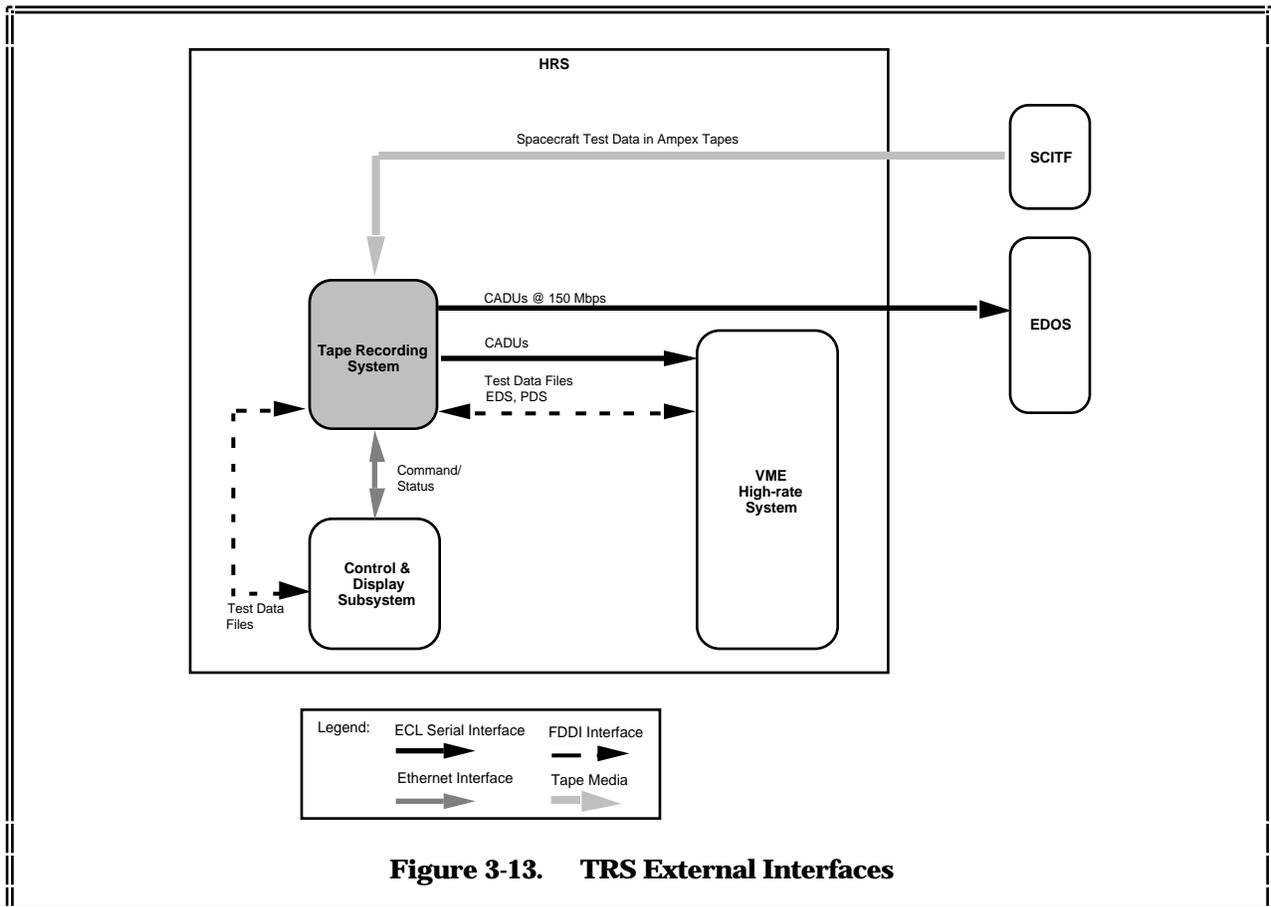


Figure 3-13. TRS External Interfaces

The TRS will use Ethernet to communicate with other ETS components over the ETS Ethernet LAN. Specifically, the TRS Triplex interface will communicate with the CDS for the following information:

- a. Commands from the CDS to the TRS Triplex interface, Sony tape drive, and Ampex tape drive.
- b. Status messages from the TRS Triplex interface, Sony tape drive, and Ampex tape drive to the CDS.
- c. Event messages from the TRS Triplex interface, Sony tape drive, and Ampex tape drive to the CDS.
- d. Test data files, EDSs, and PDSs from the CDS to the TRS.
- e. Test data files, EDSs, and PDSs from the TRS to the VHS.

The Ethernet interface will provide an effective bandwidth of 3 Mbps, excluding protocol overheads.

3.3.3.3 RS-232 Port

The TRS will provide an RS-232 port for the interchange of commands and status between the Triplex interface and the TRS Ampex tape drive. This interface will comply with Ampex control interface documentation.

The TRS control interface will use the functions of this interface to implement higher-level TRS operations; it will hide the details of this interface behind high-level commands. Through this interface, all commands from the CDS via the Ethernet interface will be initiated as commands to the TRS Ampex tape drive to initiate the following functions: play, rewind, fast forward, record, and stop. These basic functions will be hidden behind higher-level functions levied on the Ampex (i.e., dub, which would require the Ampex to play and output data from the tape drive to be recorded by the Sony tape drive). The dub command will output data from the TRS Ampex tape drive at user-specified clock rates, as set by the control interface to the clock generator (maximum of 100 Mbps through the high-speed ECL port to the TRS Sony tape drive via the Triplex interface). Through this interface, the Triplex interface will also receive status on the response to commands, and general status of the Ampex tape drive, which will be made available to the CDS via the Ethernet interface.

3.3.3.4 IEEE 488 Port

The clock generator will provide an IEEE 488 port for the interchange of commands and status between the CDS and the clock generator. The clock generator will supply the clock signal to both the Sony and Ampex tape drives of the TRS. For example, the CDS command to the TRS for dubbing a tape (i.e., transfer data from Ampex tape drive to Sony tape drive) will require the data rate to be set by the clock generator to the user-specified value (maximum of 100 Mbps). Thus, prior to the dub command, or in concurrence with the dub command, the user will have to specify what has to be transferred and at what speed. Table 3-2 describes each interface and protocol.

Table 3-2. TRS External Interfaces and Protocols

Interface	Type	Physical	Protocol	Data	Data Rates (Mbps)	Application
Ethernet	Network	Thin Ethernet (10Base2)	TCP/IP	MEDS message (command, status, event) Data snapshot	3	Communication with CDS
Serial #2	Serial Output	100K ECL Differential Data and Clock		CADU bit stream	150	Interface with VHS
Serial #3	Serial Input	100K ECL Differential Data and Clock		CADU bit stream	150	Interface with EDOS
RS-232	Point-to-point			Commands and status		TRS Triplex Communication with Ampex
IEEE 488	Point-to-point			Commands and status		CDS Communication with clock generator
Delivery	Point-to-point	19mm Tape Cartridge	Ampex	SCITF-generated Test Data		TRS Communication with SCITF

3.4 CONTROL AND DISPLAY SUBSYSTEM

The CDS is a graphic UNIX-based, computer workstation configured to support control, status monitoring, and data distribution operations of the ETS HRS. The CDS is developed on an HP Apollo Series 700, Model 755, workstation running the HP-UX Version A 9.0.5 operating system, with 128-Mbyte RAM, 2 -Gbyte internal hard disk, and a Fast-Wide 4-Gbyte SCSI-II storage system. The system also includes a 4 mm tape drive, 600-Mbyte Compact Disk Read-only Memory (CD-ROM), Digital Audio Tape (DAT), 3.5-inch diskette drive, 19-inch monitor, standard keyboard, and 3-button mouse. The operating system processes on this workstation use the UNIX IPC facilities, including sockets, pipes, files, and shared memory.

Inter-workstation communications are supported via TCP/IP, providing reliable in-sequence delivery of data required for host-to-host communication. Standard FTP is used for the distribution of data notification messages and CODAs to EOC. FTP provides reliable, guaranteed-complete delivery of data files from host to host.

The Menu Controller, OMDSIM, and support utilities developed by Code 515 will be hosted on the CDS. The OMDSIM and support utilities automatically generate reports summarizing data distribution to user sites, and quality and accounting information for completed telemetry processing sessions. The Menu Controller software provides a framework for various components of ETS HRS.

The ETS HRS Menu Controller can be invoked in two ways. If the user is logged in under the HP Visual User Environment (HP VUE), ETS icons are available from the personal toolkit located on the graphical front panel. There are separate icons for the ETS HRS, TRS, and SCTGEN. Double clicking on an ETS icon invokes its execution. Alternatively, the user can start ETS in an xterm window by entering `ets_hrs` as a command line entry. The Menu Controller software presents the list of ETS software component icons appropriate to the selected system. For the HRS, the list includes an HRS TPCE icon, HRS TRS icon, OMDSIM icon, SCTGEN icon, Data Comparison Tool icon, and Data Verification Tool icon.

The HRS TPCE provides mechanisms for controlling and monitoring the system. The HRS TPCE automates operation of the ETS HRS based on a schedule of system telemetry processing activities and user-initiated batch processing sessions. The HRS TPCE running on the CDS provides displays that present the current status of the ETS HRS during processing. These displays present frame synchronization, packet reassembly accounting, and data quality information for ongoing telemetry processing sessions; summary data quality and accounting information for processed sessions; and status of system hardware and software subsystems.

SECTION 4 USER INTERFACE

This section describes the ETS HRS user interface. The Telemetry Processing and Control Environment (TPCE) provides the remote or CDS user interface for overall configuration, control, and status monitoring of the ETS HRS. The Operations Manager (OPMAN) provides a local user interface that allows for overall configuration and status monitoring of each MEDS subsystem (card) in the VHS.

4.1 INTRODUCTION

The TPCE operational user interface, developed by the Software and Automation Systems Branch, Code 522.2, will be used in the ETS HRS to control the VHS and launch the TRS. This section introduces TPCE elements and details major components. The requirements satisfied by TPCE are listed in Section 10, and are mapped to the TPCE component that resolves the requirement. (Components still under development are briefly mentioned to the extent of the changes to be implemented.) This section also describes the Code 521-developed local user interface, OPMAN, which allows for overall configuration monitoring, status monitoring, and testing of each VHS subsystem.

Catalogs are the foundation of system processing. They consist of files that define system configuration, data flow, and data output. The VHS provides a range of capabilities; however, this flexibility demands proper catalog setup to ensure correct data processing.

System processing is initiated by activating a catalog. Catalog pages provide the setup parameters that must be defined for each card. Catalog pages allow the operator to define input source, output destination, and processing mode. Once a catalog is defined and tested, it is ready to be used for data processing.

When data is flowing, each card maintains status and counts, which the Master Controller Card (MCC) gathers. The MCC makes this information available to the user interface for display. Each operator interface has a system status page and individual status pages for each card. These pages allow the operator to monitor system processing and output.

The user interface (both TPCE and OPMAN) provides menu-selectable commands that control the system, and include catalog activation and access to status pages. Refer to the MEDS and TPCE documentation for detailed information on the capabilities and selections available with each operator interface.

4.2 TELEMETRY PROCESSING AND CONTROL ENVIRONMENT

4.2.1 SOFTWARE DESCRIPTION

The TPCE software is a set of reusable C++ classes used to control and monitor a telemetry processing system. The telemetry processing system is a VME-based system developed by Code 521. The VME system consists of a set of hardware components that accomplish telemetry processing. The VME system communicates with TPCE using MEDS.

4.2.2 MEDS REVIEW

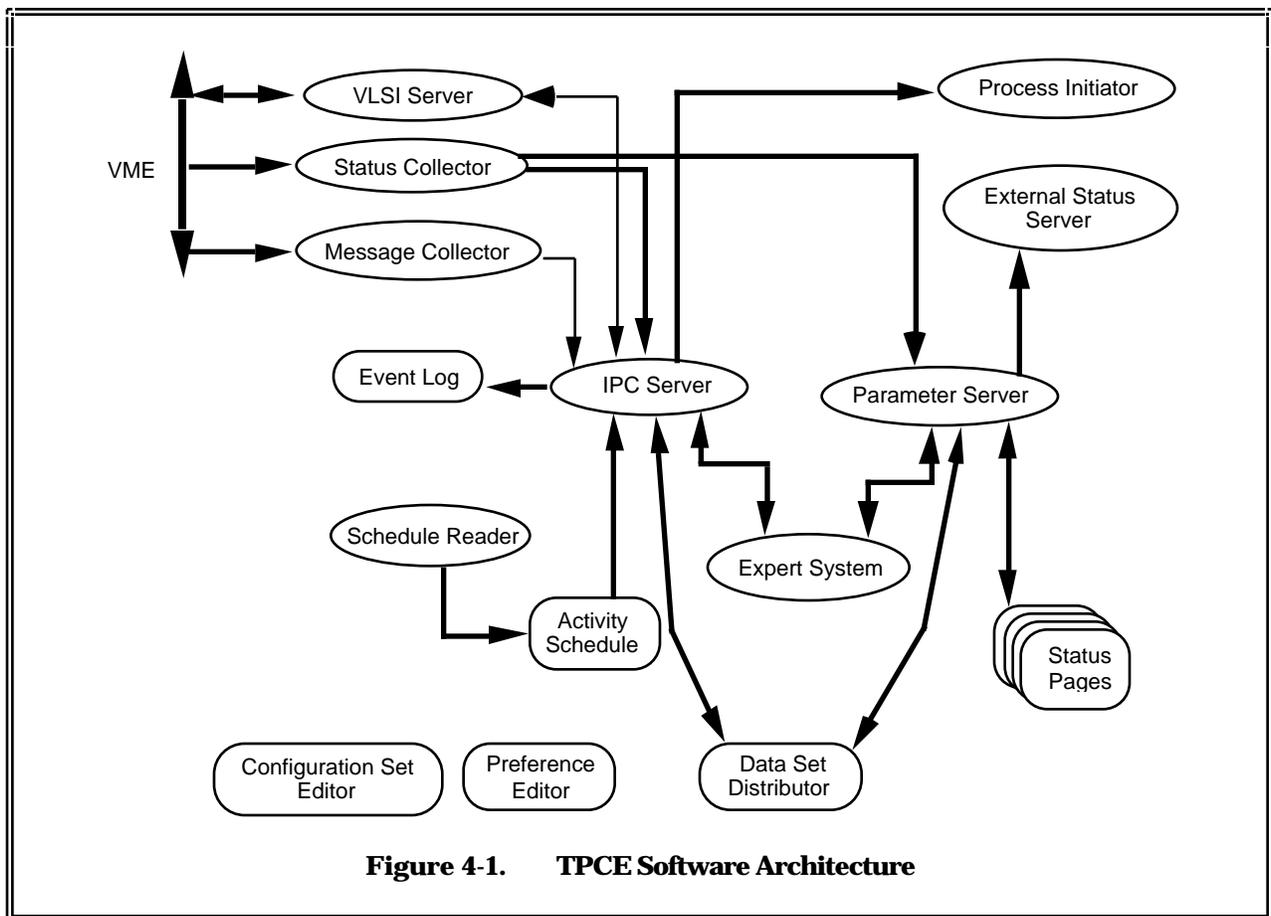
MEDS is a software architecture that brings together each processing component into a complete system. MEDS provides a network protocol in order for external software to monitor and control the VME system. This access protocol is defined by a set of network TCP/IP sockets and data

structures. There are three TCP/IP sockets—one for sending commands and receiving responses, one for receiving status updates, and one for receiving unsolicited event messages.

MEDS commands are organized into two groups: general commands that all MEDS VME systems will respond to, and special commands that an individual processing subsystem element can define. An example general command is Enable. The Enable command is sent to the MEDS Master Controller subsystem; it determines which of its known subsystems will receive the command. A special command is directly routed to the subsystem identified in the MEDS command data structure.

4.2.3 ARCHITECTURE

The TPCE software is organized into 14 major software components (see Figure 4-1). The presented architecture was originally defined for FAST PPS; only minor additions were implemented from that system to increase effectiveness of the TPCE software. The following sections describe each major component.



4.2.3.1 VLSI Server

The VLSI Server acts as a command and response gateway to the VME. This component receives command requests, via the Interprocessor Communications (IPC) Server, from a component attempting to control the VME, and packages the command into MEDS format. Once the MEDS command is created, a unique signature key is placed in the structure header. The command is then written to the VME command port. After the VME handles the MEDS command, a response is returned to the command port. The signature key is used to identify which command the response was for. The command response is unpacked and sent, via the IPC Server, to the requesting component.

The VLSI Server will periodically send a NOOP command to the VME to ensure that the connection to the VME is active. If the command fails to return, the connection is deemed "down." The VLSI Server will broadcast a message to the IPC Server indicating that the connection is inactive. The VLSI Server will attempt to reconnect. When reconnection is established, a message is broadcast to the IPC Server indicating that the connection is active.

4.2.3.2 Status Collector

The Status collector receives all VME status parameters. This component connects to a port on the VME to receive status. After connecting to the port, the Status Collector sends a Return Status command to the VLSI Server via the IPC Server. The Return Status command contains the status update rate the VME will use to send status. Once status is received by the Status Collector, each status item is extracted and written to the Parameter Server.

4.2.3.3 Message Collector

The Message Collector receives all VME unsolicited event messages. This component connects to a port on the VME to receive messages. Messages are sent whenever a predefined event occurs on the VME. The message is received and sent to the Event Log via the IPC Server. Before message processing is completed, the message opcode is examined to determine if the message should be routed elsewhere. If the opcode matches a list of routeable messages, it is broadcast to the IPC Server with a special IPC Server identification. Any component may register to receive the specific messages that it requires to perform its function.

4.2.3.4 Process Initiator

The Process Initiator controls execution of UNIX processes, or sends special messages to other processes. Processes that can be controlled, and how, are defined in a system control file. The control file contains process execution directives or keywords if a special Open message is to be sent to other processes. A process execution directive can describe a process that should be started once the TPCE system starts, or when indicated via an IPC message. For directives that are to be started via IPC message, the Process Initiator receives a message from the IPC Server with the directive identifier. From there, this component reads the definition for the directive and acts on it. An IPC message is sent to open a User Interface (UI) component, spawn a UNIX shell with the specified UNIX command, or access the UNIX system services to execute on another host.

The Process Initiator has the added functionality to route commands to another instance of the initiator software that resides on another host, or to query on the executability on another host. Also, the initiator can query the UNIX display to determine if a specified host can route X-Window displays on its console. This provides the mechanisms to route X-Window displays to an X-Window display on the network.

4.2.3.5 External Status Server

The External Status Server provides status updates to non-TPCE software. This component allows an external status client to connect to a TCP/IP socket on the host computer and receive specified status items. The client will receive only the status items that have been predefined in the TPCE system. This component alleviates the need for a software component to write MEDS translation software, or to compile with TPCE code. The client need only connect to the UNIX port specified in the system services.

4.2.3.6 Event Log

The Event Log component provides event messages that can be displayed on a UI or remote terminal, piped to a file, or sent directly to a printer. Events come from user entry (annotation), VME generation, TPCE software, or external sources.

The user entry mechanism allows users to enter an informational message that is time-tagged and sent to the log. The VME generation mechanism is a connection, via the IPC Server, that logs all messages directed to the Event Log from the Message Collector. TPCE software messages are those events deemed to be significant. These messages are generated by the software and sent to the IPC Server with Event Log direction. External sources can send event messages to the Event Log by sending IPC messages to the IPC Server with Event Log direction. This capability implies that external sources use TPCE software to send messages.

There are four types of Event Log messages: information, warning, error, and debug. By default, debug messages are not displayed on the UI. Debug output can be turned on through a main window View menu item. The tagging of each message depends on message type. Normally, every logged message contains a time-tag followed by the message itself. If the message is an error message, it is prefixed with the string E->. If the message is a warning message, it is prefixed with the string W->. Information and debug messages are not output with a prefix.

4.2.3.7 IPC Server

The IPC Server is the central communication component of the TPCE system. This component provides communications mechanisms for one component to another component. Basically, a message originator registers a connection to the IPC Server for sending messages. As a message is sent to the IPC Server, a message ID is used to uniquely identify the message type. A client of the IPC Server also registers itself with a connection to the IPC Server. The client then registers a request for messages by message type. When the message is received by the IPC Server, a search is performed to determine if any clients have requested to receive this message. If not, the message is stored and not propagated. If a client has requested the message, it is routed and the search continues. A message originator may direct a message to a specific client, or may broadcast the message to any client registered for the message type. A client receiving a message is informed of the originator and type, as well as the contents.

4.2.3.8 Parameter Server

The Parameter Server functions similarly to the IPC Server, with the exception that messages are all status items. Items consist of a group of items, or individual items. Registration is performed by keywords, rather than IDs. When a client receives an item, the item must then be extracted from the received message.

4.2.3.9 Schedule Reader

The Schedule Reader defines the contents of an operational schedule file that may be received from

an entity external to TPCE. The operational file is used to automatically set up activities to be performed at specified times with indicated activity parameters. As each activity is read from the file, a session event is created, which is then passed to the Activity Schedule.

4.2.3.10 Activity Schedule

The Activity Schedule receives a session event from the Schedule Reader and creates an alarm that is initiated at a predefined time prior to the start of the session event. When the alarm signals, the session event is passed to the Expert System to process the session. The Activity Schedule provides a set of UI software to allow a user to create a session event. The UI software also allows a user to modify session events. The actions in each session event are predefined and implemented in the TPCE software.

4.2.3.11 Expert System

The Expert System is a set of C++ software and C Language Integrated Production System (CLIPS) rules used to create actions based on a defined rule base. Each rule is a definition of existing conditions and a set of actions required to change state from the current state to a new state. Together, the rules define state machines that guide the VME system through execution of processing sessions. State machines are organized in a simple hierarchical structure. A master state machine monitors and controls the flow of session events to a set of VLSI (VME) state machines, one for each VME system.

Different VLSI state machines can have different rule bases, which is necessary for the single TPCE system to concurrently control different VME systems, such as the ETS Low-rate System (LRS) and the ETS HRS.

When the master state machine receives a session event, it checks the current state of the intended VLSI state machine and communicates the event to that state machine if it is in a ready state.

When the VLSI state machine receives the event, it initiates actions to sequence the VME system through the states that comprise the session. Commands are sent to the VLSI system, via the VLSI Server. Response to commands, as well as an unsolicited message from the VLSI system, are used as facts asserted to the state machine to synchronize the steps and assure proper setup, execution, and termination of processing.

4.2.3.12 Data Set Distributor

The Data Set Distributor allows TPCE and the user to monitor and control the transmission of data sets produced by the VME system. As data sets are produced, the Data Set Distributor is notified of the progress via unsolicited messages from the VME.

The Data Set Distributor maintains a data base of prescribed data distribution tables indicating which data set types should be transmitted to which users. It also maintains a queue of data sets waiting to be transmitted to users. At any time, it can also query the VME for directories of data sets available for transmission. Based on this information, the Data Set Distributor commands the VME to transmit data sets to intended users.

The Data Set Distribution Editor allows the user to monitor and intervene in the data set transmission operation. The user can also request that data sets be retransmitted when problems occur.

4.2.3.13 Configuration Set Editor

The Configuration Set Editor is a tool that essentially operates offline from the rest of the TPCE software. This component defines the contents of a VME system configuration set. Each configuration set contains a group of VME subsystems, each with a set list of parameters and values. The Configuration Set Editor allows users to create or modify configuration files, and add or remove defined subsystems. Within each subsystem is a set of parameters, each of which has a value with predefined constraints. The UI will allow the user to set each parameter to an acceptable value.

4.2.3.14 Preference Editor

The Preference Editor allows a user to modify the parameters used to configure the TPCE software. These parameters are critical to normal operation of the TPCE system. The form of modifications is similar to the Configuration Set Editor. If preferences are modified, the user must take specific action (copy a file) to ensure that the modified preferences are used. The new preferences will be used during the next execution of the TPCE software.

4.2.3.15 Status Page

A Status Page is a UI component that allows the user to view VME status items. Status Pages are divided into logical organizations, such as VME card or subsystem. Each Status Page registers with the Parameter Server to receive status items. When a status item is received by the Status Page, the item is reflected on the display.

4.3 LOCAL OPERATIONS/OPMAN

OPMAN allows the operator to create, edit, and activate catalogs. OPMAN runs on a VT-100 compatible terminal that is connected to the MCC on the system. It allows the operator to set up the system, flow data, and monitor system status. OPMAN is thoroughly documented in the MEDS User's Guide, which provides a detailed description of each OPMAN screen and the parameters that the operator can define. OPMAN can be used as soon as the system boot sequence is complete.

OPMAN's main menu lists the primary commands available and provides a brief tutorial. From this screen, the operator accesses submenus that allow system command, setup, and monitoring. The following sections briefly review common procedures that are performed when operating the system.

4.3.1 ACTIVATING A CATALOG

The Activate command activates a catalog. This command provides two functions: it loads the catalog, which contains configuration information for the appropriate cards (subsystems), and it enables each card, which allows the system to process data.

When the Activate command is implemented, it lists the current directory and prompts the operator for the name of the catalog to be activated. The directory can be changed by using the FileInfo command.

4.3.2 LOOKING AT STATUS

OPMAN provides various status pages for each card in the system. A system status page overviews the entire system. The system and individual card status pages can be accessed by the Page command.

The system status page lists cards in the system and each card's VME base address, Processor Communications Area (PCA) base address, and status base address. This page also indicates if each card is healthy (OK), enabled (ENA), or disabled (DSB). If a catalog is currently loaded into the subsystem, it is listed along with any comments pertaining to that catalog.

Each card status page is card-dependent. The MEDS User's Guide provides a complete description of each status page. The operator views these pages as data flows through the system to ensure correct processing. For example, the following information about the EOS Frame Synchronizer Card can be gathered from status pages: enabled (is the card activated and processing data), name of loaded catalog, number of frames that entered the card, number of frames that exited the card, operating mode, number of search frames, number of lock frames, number of flywheel frames, number of forward frames, and errors that occurred.

4.3.3 SHUTTING OFF A CATALOG

The Shutdown command stops current catalog processing. It disables all cards currently enabled, and clears the catalog name from system processing.

4.3.4 QUICKLOOK FEATURES

OPMAN provides some quicklook features that the operator can use to ensure that the VHS is correctly processing data. These features allow the operator to view data that has entered the EOS Frame Synchronizer Card, and examine the data as it was stored in the data buffer module.

The operator can access quicklook features through the Commands option of the OPMAN menu. When Commands is implemented, the following quicklook features become available:

- a. Snapshot: allows operator to view raw hex data stored on the data buffer disk. By viewing this data, the operator can verify that valid data was sent out of the VHS.

To snapshot a file, the operator specifies the file name, version, record number, and byte offset into the desired record. In response to a snapshot request, the system returns 2 Kbytes of data, starting from the offset position defined by the operator. (Record numbers start at 1 and byte offsets start at 0.) Data is displayed on a number of pages.

The snapshot feature allows the operator to look past the end of a file, which can be used to verify that all valid data, and nothing extra, left the system. For example, if a file has 200 records and each record is 8 Kbytes, the operator may request a snapshot of record 200 with an offset of 7 Kbytes. This snapshot request would show 1 Kbyte of valid data and 1 Kbyte of data past the End-of-File (EOF).

- b. **Position:** outputs a file, starting from any point within the valid data of the file. The operator can use the snapshot feature to locate the point from which to begin reading a file, and then issue the Position command. To implement a Position command, the operator must supply the file name, version, and record number. Record numbers start at 1 and may not be greater than the total number of records in the file, which can be found in the Total Recs field of the data buffer status page. The file must be open for read-only, and the data buffer must be disabled.

After the Position command is defined, the operator uses the Enable command to output the file.

4.3.5 EDITING A CATALOG

The Edit command provides access to all catalogs in the current directory, or allows the operator to create a new catalog. To correctly define a catalog, it is essential that the user be familiar with the data the system is processing. The data defines many of the entries that are made throughout the catalog pages; however, several catalog entries are dependent on VHS processing mode.

SECTION 5 OPERATIONS SCENARIOS

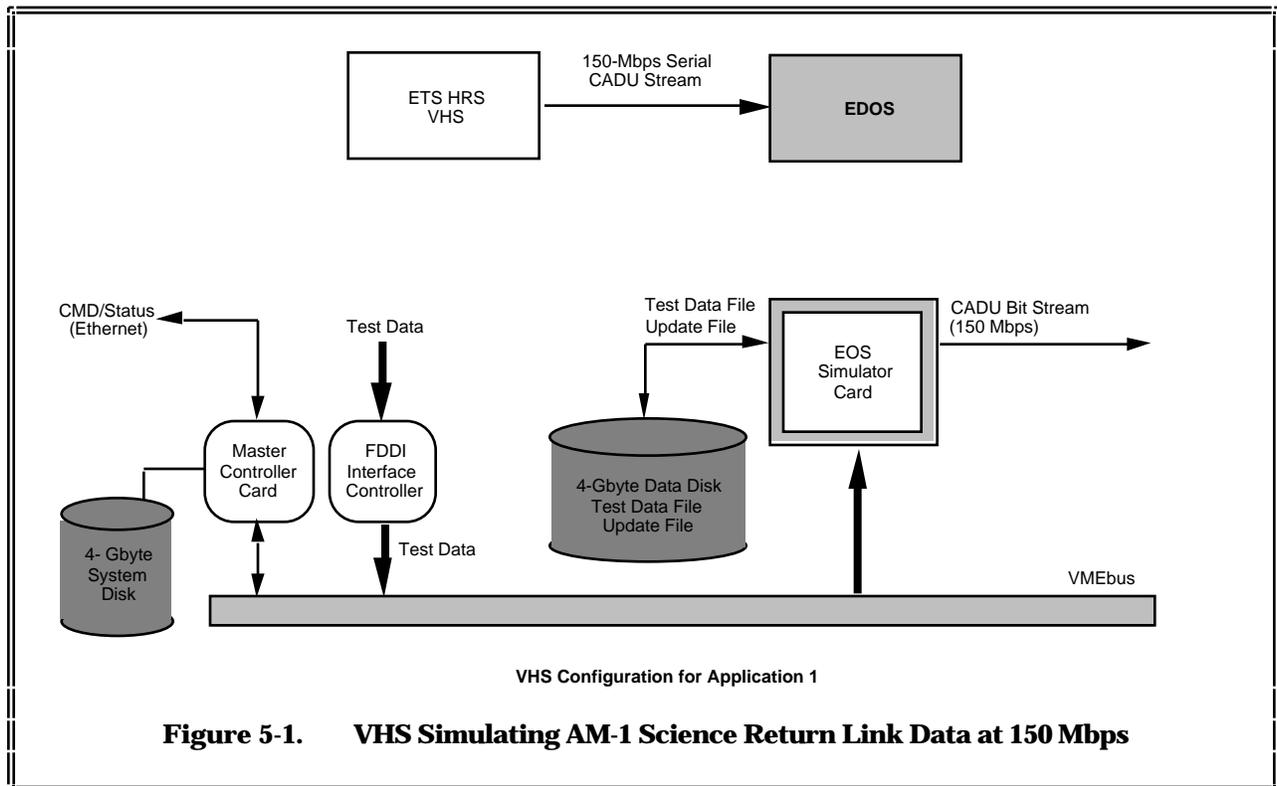
This section describes each ETS application/configuration in terms of system setup, and how it receives input, processes data, and generates output. This section also describes subsystem element setup and the modes of operation that are essential to obtaining the desired product or data flow (i.e., all dependencies are described).

5.1 INTRODUCTION

The ETS HRS test configurations will be used to explain the interaction within each subsystem, (intra-subsystem), and between subsystems (inter-subsystem). Each configuration will show hardware element interaction, and the data flows that emanate between elements in the subsystem and external interfaces.

5.2 SIMULATE AM-1 SCIENCE RETURN LINK DATA

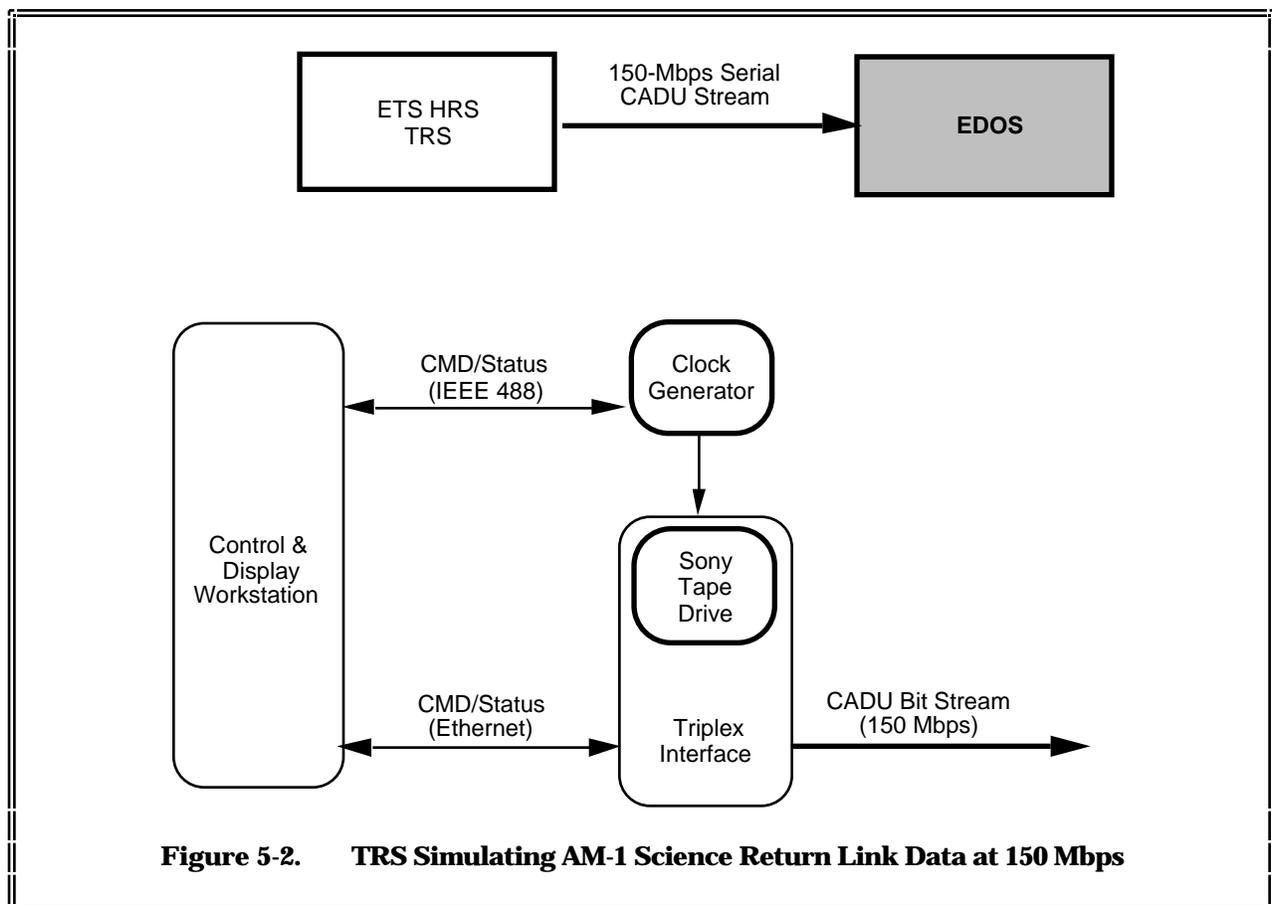
The requirement calls for the capability to generate two CADU data streams, each at 150 Mbps, to simulate the TGT interface to EDOS. The ETS HRS will meet this requirement in the following manner. One data stream will be generated via the VHS; the other will be output from the TRS. Figure 5-1 illustrates the VHS elements used for generation of the VHS 150-Mbps CADU data stream. SCTGEN will create a 4-Mbyte test data file according to user-specified VCIDs, APIDs, data patterns, multiplexing strategy, and frequency of occurrence.



SCTGEN will also create an update file that will enable the update of sequence number, Reed-Solomon check symbols, timecode information, and any updates in user-specified fields. This file is created prior to test, and stored on tapes or disk. Closer to the test session, the file is downloaded

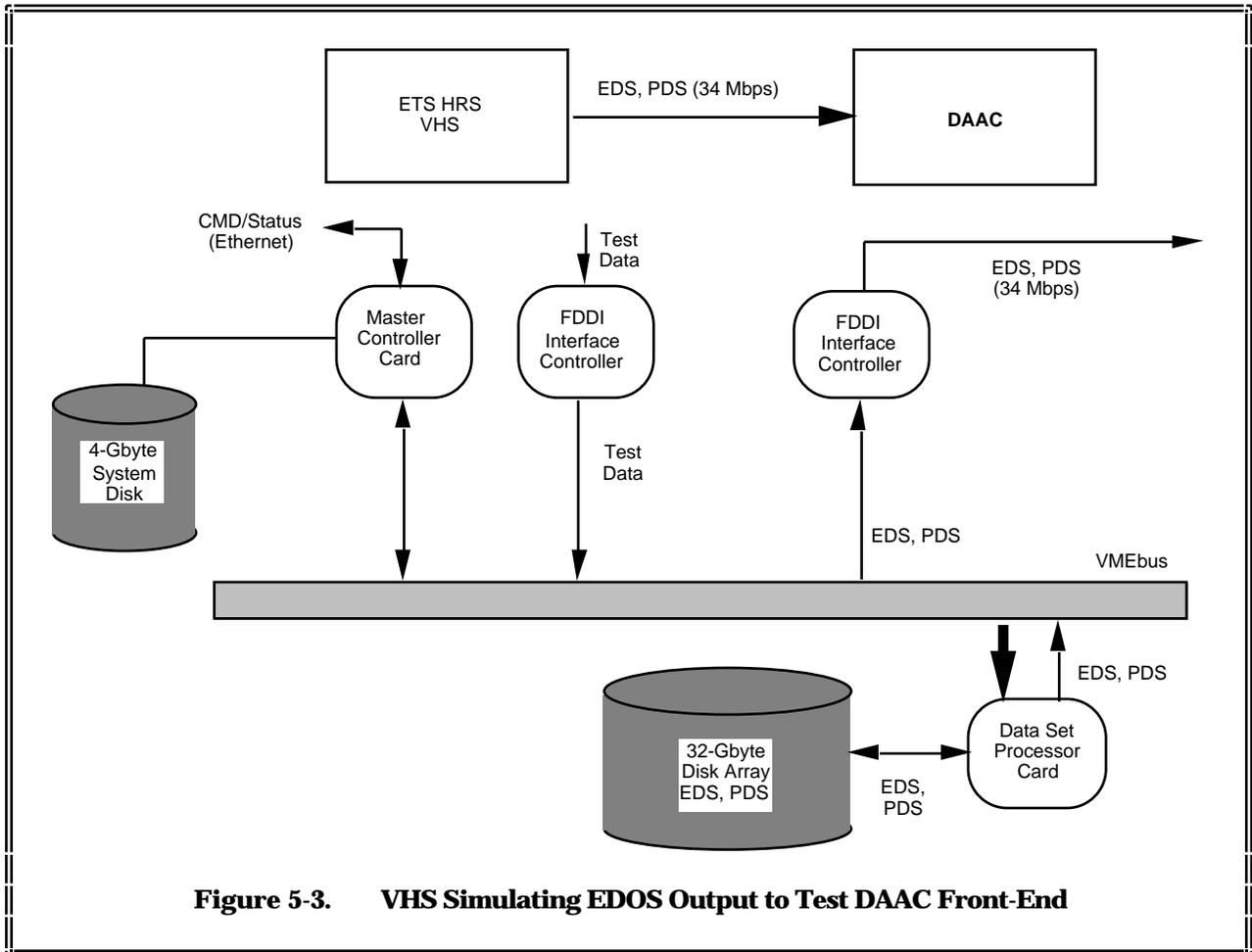
into the 4-Byte data disk via the FDDI interface. When the test is started, the EOS Simulator Card is configured to read the designated test data file and update file, to output data at 150 Mbps, to output a fixed number of CADUs, and to stop transmission after the required number of CADUs have been transmitted. SCTGEN data generation is performed on the CDS using the utility icon designated on the menu controller (discussed in detail in the SCTGEN Detailed Design Document). All setup and configuration of the FDDI interface card, EOS Simulator Card, and Master Controller Card are initiated by TPCE for the VHS. The EOS Simulator Card will load the 4-Mbyte CADU data onto its resident dual-banked RAM, and start transferring data through its output ECL circuitry. As data is output, the update file loaded onto the update First-in, First-out (FIFO) starts updating the values of the next cycle of the same 4-Mbyte CADU data. The update FIFO will update the sequence number, timecode, Reed-Solomon check symbols, and optionally user-specified updates on selected fields.

The second data stream will use the TRS to generate the 150-Mbps output. Once again, the CADU data file or concatenation of files is previously created and stored on Sony tape media. The data source is specified in the requirements. At the time of test, TPCE will spawn the TRS GUI, which will enable the user to specify the file to be transmitted and at what rate. Once the user presses the designated icon to output or playback, data will be output via the ECL serial output on the Triplex interface of the TRS. Figure 5-2 illustrates the configuration for the second AM-1 150-Mbps return link data stream.



5.3 SIMULATE EDOS OUTPUT TO TEST DAAC FRONT-END

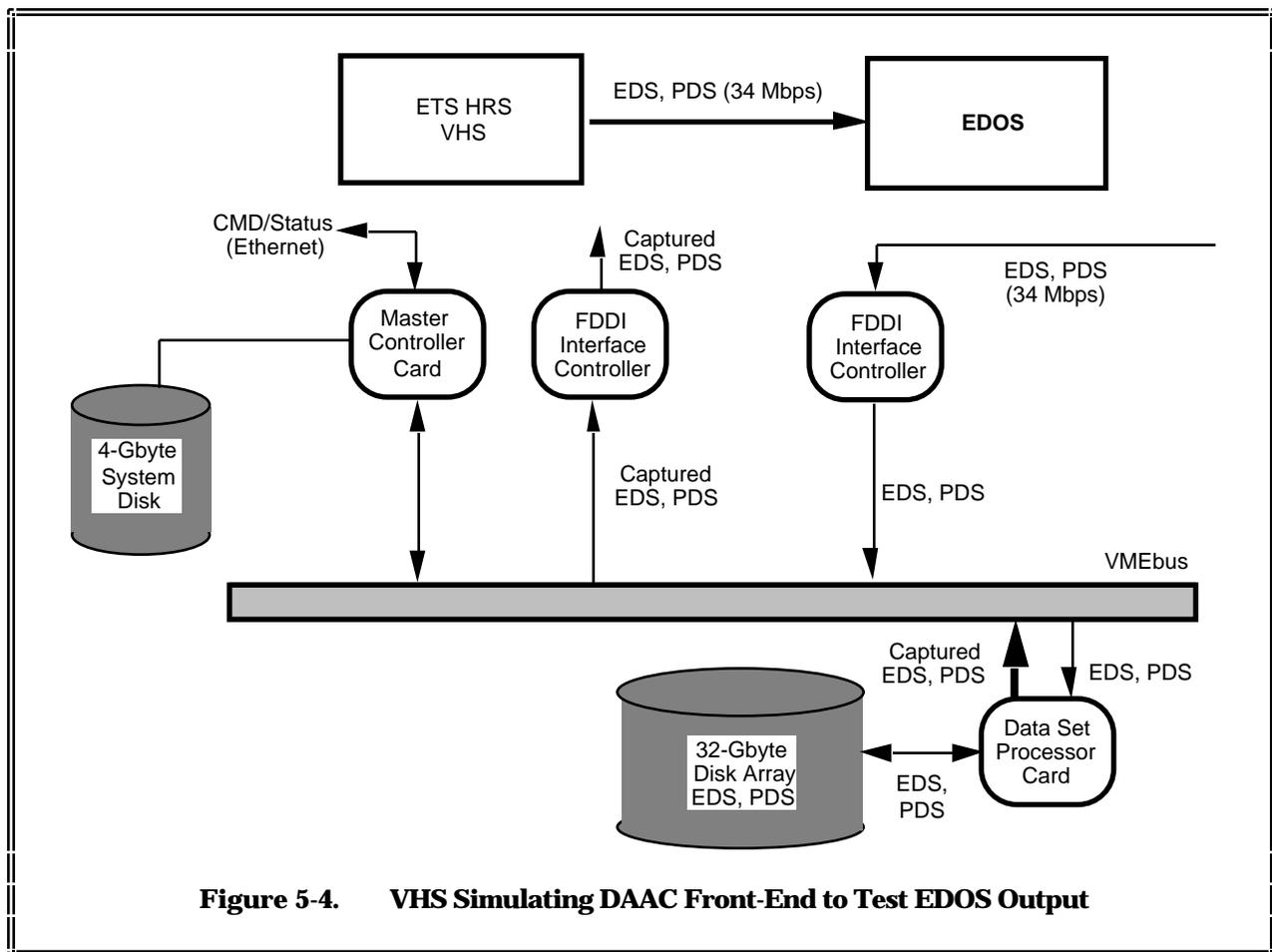
In this configuration, the VHS will simulate EDOS output to test the DAAC front-end. The VHS will output EDSs and PDSs to the DAACs at 34 Mbps via EBnet (i.e., test the interface from EDOS to DAAC). Figure 5-3 illustrates the VHS receiving data sets prior to a test session via the FDDI interface card, and transferring data via VMEbus to the Data Set Processor. The Data Set Processor will transfer data to the 32-Gbyte disk array. Once the test session is initiated, data is transferred over the external FDDI interface card to the DAACs via EBnet. All commands and status to the VHS to set up and transmit data sets via the FDDI external interface are downloaded from the CDS to the Master Controller Card and specific cards on the VHS. File names and file sizes to be transmitted are exchanged via the protocol and delivery notices specified in the ICDs.



5.4 SIMULATE DAAC FRONT-END TO TEST EDOS OUTPUT

In this configuration, the VHS will simulate the DAAC front-end to test EDOS output. The VHS will capture EDSs and PDSs output from EDOS at 34 Mbps and verify receipt of uncorrupted data (i.e., test interface from EDOS to DAAC). Figure 5-4 illustrates the VHS receiving data sets via the FDDI interface card and transferring captured data via VMEbus to the Data Set Processor. The Data Set Processor will transfer data to the 32-Gbyte disk array. Once the test session is over, data is transferred over the internal FDDI interface card to the CDS for analysis, transmission error verification, or header validation in accordance with user-specified criteria. All commands and status to the VHS to set up and receive data sets via the FDDI external interface are downloaded

prior to the test session from the CDS to the Master Controller Card and specific cards on the VHS. File names and files sizes are exchanged via the protocol and delivery notices specified in the ICDs.



5.5 GENERATE EDSS AND PDSS FROM SPACECRAFT-GENERATED DATA

The EDSs and PDSSs transmitted at 34 Mbps from the VHS to the DAACs are either simulated data sets produced by SCTGEN, or VHS-generated data sets. In the former option, data packets contained in the SCTGEN-generated data sets may be SCTGEN-generated data packets, spacecraft-generated data packets received from the SCITF via Ampex tape media, or spacecraft-generated raw sensor data received from the SCITF via Ampex tape media encapsulated into data packets.

If the SCITF data is received as CADUs, this data is processed through the VHS and the sorted packets are stored in the data disk (see Figure 5-5). CADUs are frame synchronized and Reed-Solomon decoded to filter out good data. The extracted Coded Virtual Channel Data Units (CVCDU) with the appended quality information from the EOS Frame Synchronizer and EOS Reed-Solomon Card are transferred via the High-rate Telemetry Backplane (HRTB) to the EOS Service Processor Card. In the EOS Service Processor, the CVCDUs are processed to the user-selected service (i.e., path or VCDU). For path service, the packet annotation is transferred via VMEbus to the 16-Mbyte memory cards. The packet stream is transferred via VME Subsystem Bus (VSB) to the 512-Mbyte memory card, where the sorted packet stream is transferred to the Data Set Processor Card for storage onto the 32-Gbyte disk array. The Annotation Processor passes data set assembly

instructions and data locations to the CDS via VMEbus and the Master Controller Card. When all CADU data has been processed through the EOS Service Processor, the CDS can initiate generation of EDSs and PDSs for immediate transmission, or storage and transmission at a user-selected time.

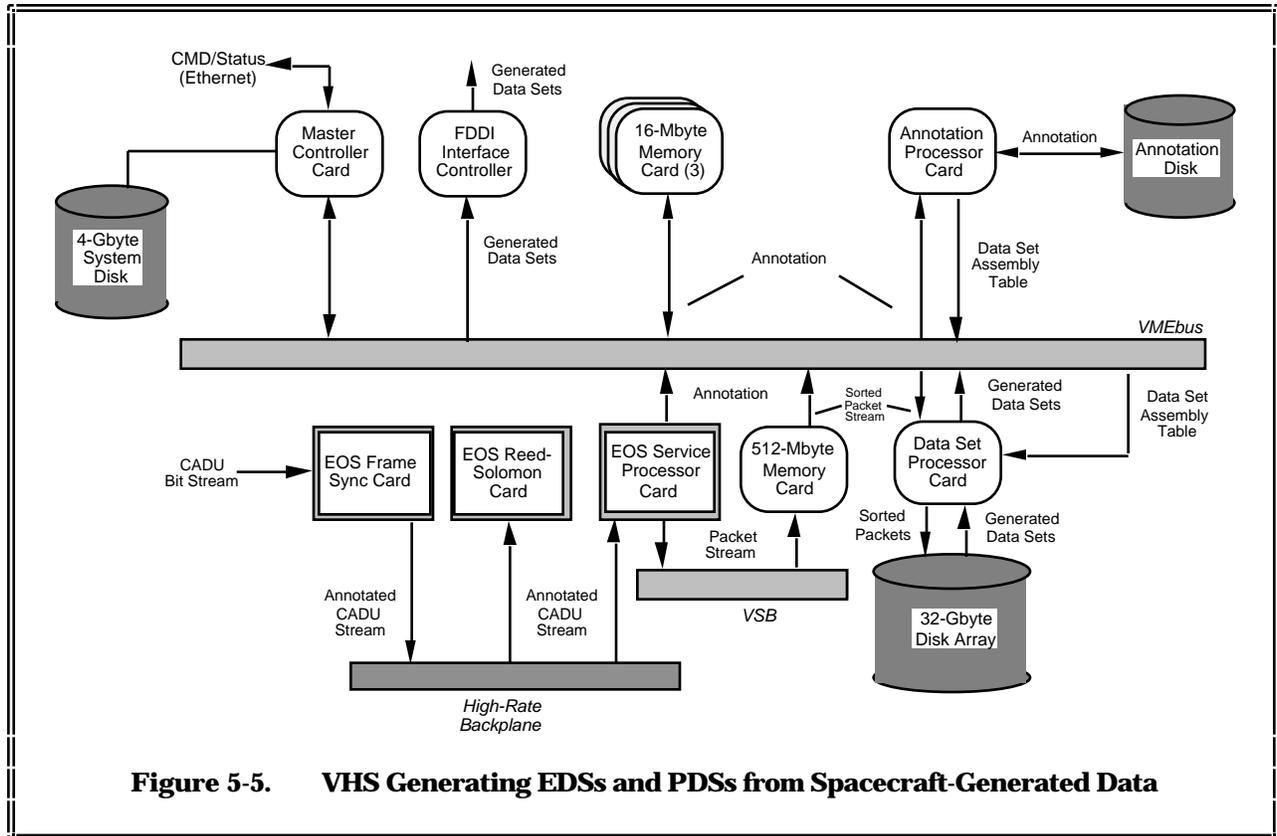


Figure 5-5. VHS Generating EDSs and PDSs from Spacecraft-Generated Data

5.6 READ/TRANSFER FROM SPACECRAFT-GENERATED DATA TAPES

All spacecraft-generated data will be transferred to the ETS HRS via tape media. The SCITF facility will use Ampex DCRSI 107 recorded 19mm cartridge tapes to store spacecraft-generated data. These tapes will be read, as shown in Figure 5-6, using the Ampex tape drive in the TRS and recorded onto Sony ID-1 cassette tapes. The data transfer will be at data rates up to 100 Mbps. The data format on the tape will be raw sensor data, science packet data, or CADUs. Information on data contents will be provided by the SCITF. Once data is captured onto Sony cassette tapes, the user can create EDSs and PDSs, provide CADUs for input to the VHS for the creation of EDSs and PDSs, or simulate the TGT interface to provide CADUs for input to EDOS (as described in Section 5.1). If the data is in raw sensor data format, the user can play back data from the Sony tapes through the Triplex interface via the Ethernet interface to the CDS for data manipulation and encapsulation into packets and/or CADUs. Packet data may be directly used to create EDSs and PDSs using the SCTGEN utility. If packets or data are encapsulated into CADUs, the CADU data stream may be fed in to the VHS to generate EDSs and PDSs (as described in Section 5.4). If the spacecraft-generated data is received in CADU format, the CADU data stream may be directly fed into the VHS for data set generation.

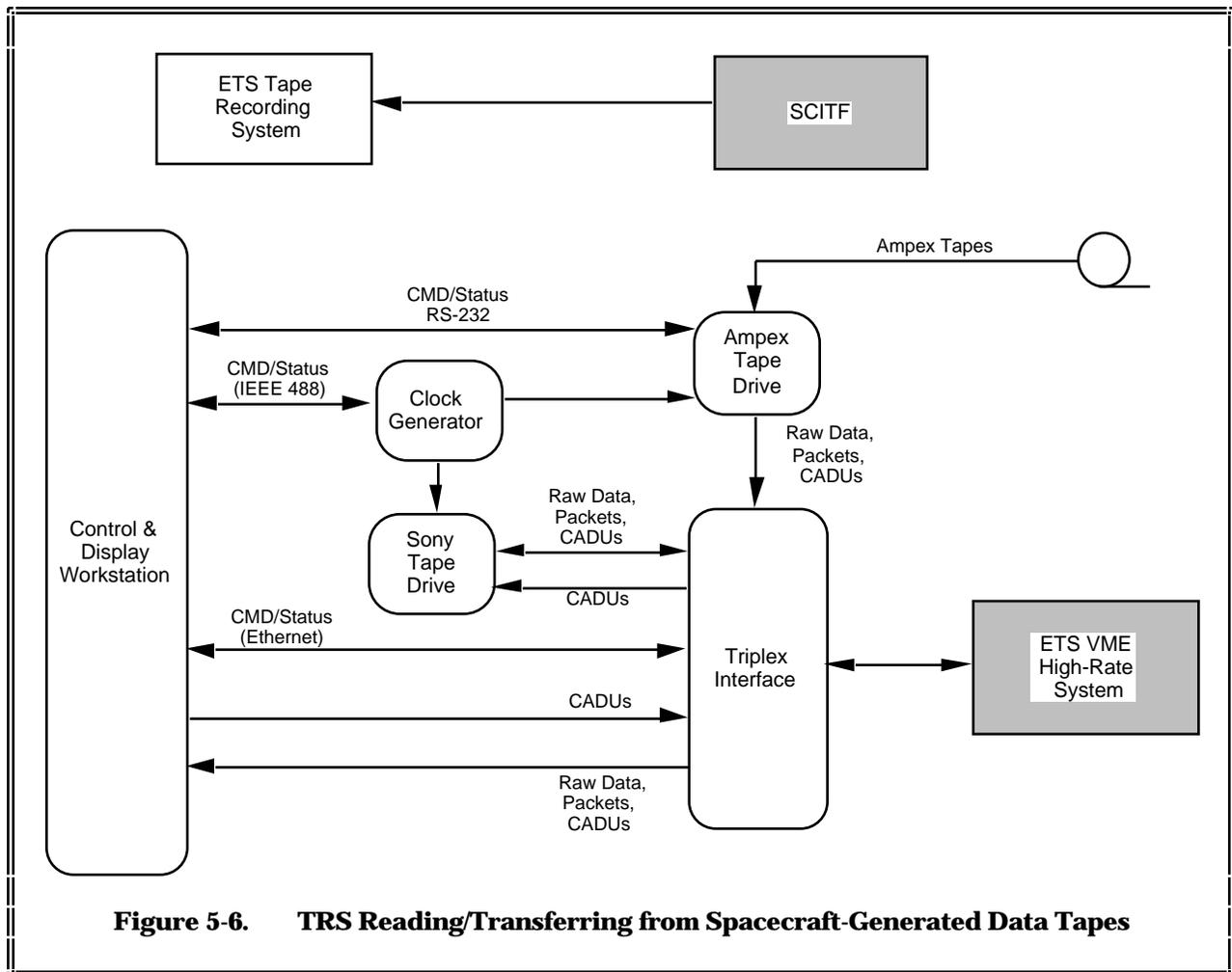


Figure 5-6. TRS Reading/Transferring from Spacecraft-Generated Data Tapes

5.7 PROVIDE CADU DATA STREAM FOR DATA SET GENERATION

Figure 5-7 illustrates the configuration for data set generation using SCITF-generated spacecraft data. As described in Section 5.5, the received data will be input to the VHS as a CADU data stream. Data will be manipulated using SCTGEN to modify fields or insert errors, and subsequently input to the VHS. If data is in raw sensor data format, the user can play back data from the Sony tapes through the Triplex interface to the CDS for data manipulation and encapsulation into packets and CADUs. If data is already in packets, it will be manipulated and encapsulated into CADUs at the CDS. If spacecraft-generated data is received in CADU format, the CADU data stream may be directly fed into the VHS.

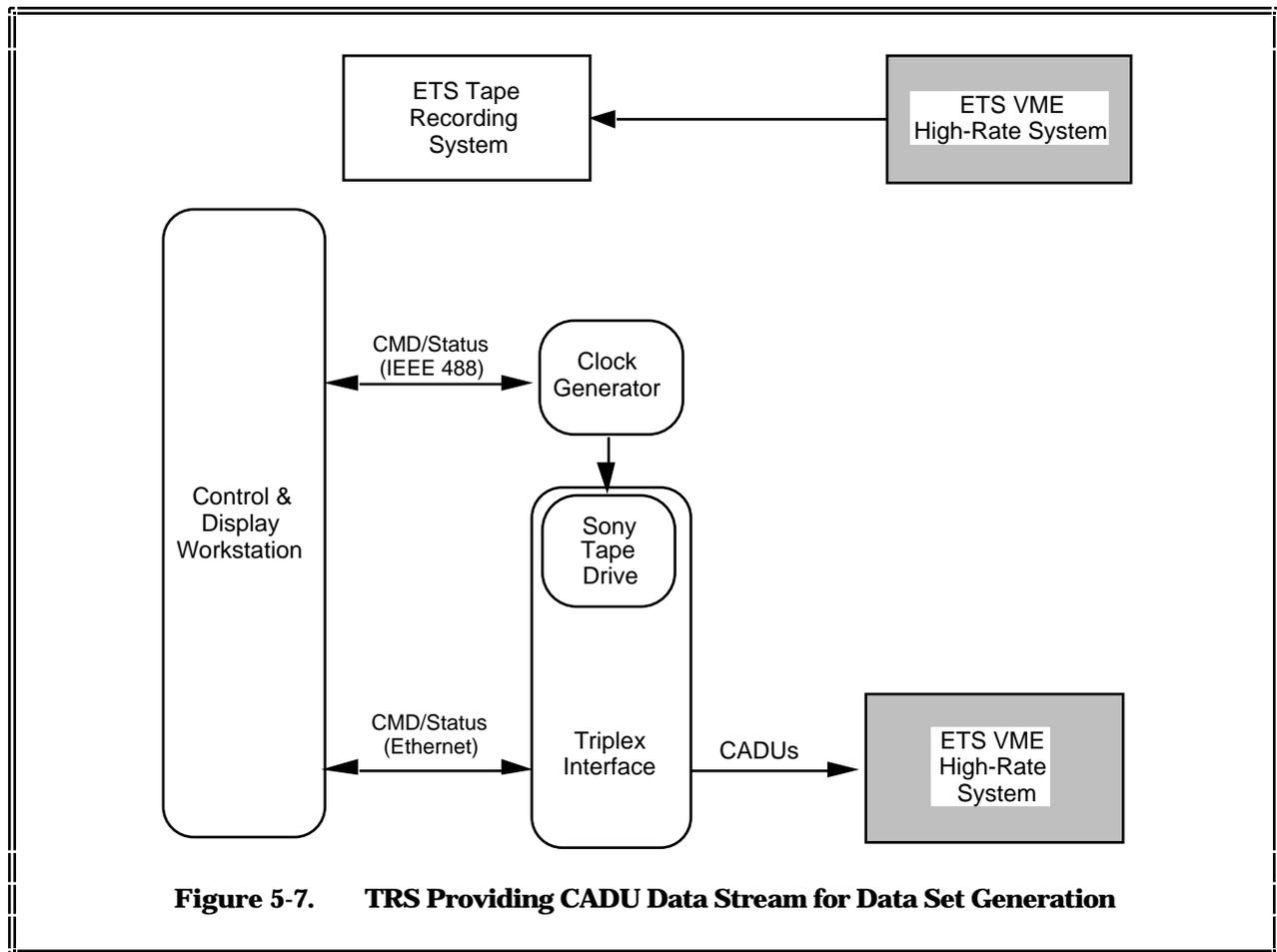


Figure 5-7. TRS Providing CADU Data Stream for Data Set Generation

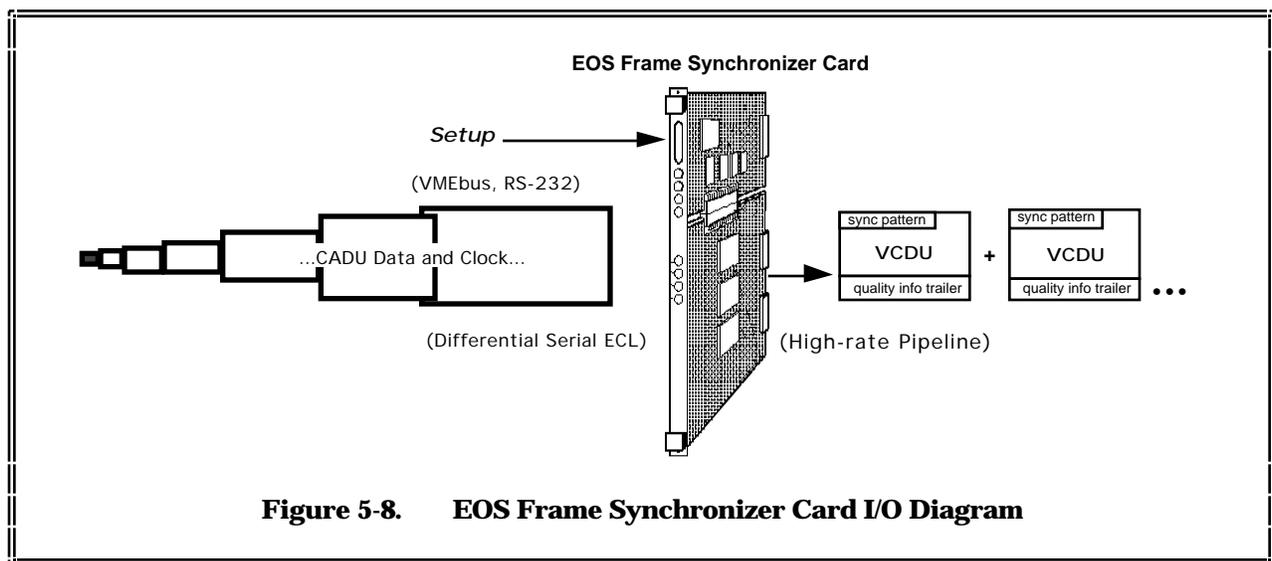
5.8 CONFIGURATION AND OPERATION

The VHS and TRS have general functional, operational, and setup modes. The subsystem elements that will be described are the hardware functions that are essential to the different configurations, namely the custom cards and two tape drives. Commercial cards, other interfaces, and peripheral devices are used for initiating tasks, and providing Input/Output (I/O) and storage media; these are described as software functions.

5.8.1 EOS FRAME SYNCHRONIZER CARD

The EOS Frame Synchronizer Card (Figure 5-8) accepts a serial clock and data stream from one of six different sources, processes and outputs the processed data as word-wide output through its custom HRTB interface. The output data word consists of 17 bits—16 bits of data and 1 mark bit (converted by the HRTB interface to 17 bits and 2 bit flags) delineating the end of the data unit. The mark bit remains low for the entire data unit until the last word of the data unit, when it is set high (1) for the word. The card can also serially output (via front-panel interface) an image of the input data being processed. The ECL I/O interfaces have signal termination options allowing coaxial or twisted pair connections. The ECL Correlator and GaAs Telemetry Frame Synchronizer (GTFS) Chips perform all frame processing on the card, with the exception of Bit Transition Density (BTD) decoding and timecode stamping, which is performed by an Actel Field-programmable Gate Array (FPGA) after frame processing is completed. Frame processing involves a four-mode frame synchronization strategy, data quality monitoring of input data, optional correction of

inverted, reversed, or slipped frames, and annotation trailer appending.



5.8.1.1 Frame Synchronization

A frame is a serial data structure with a fixed, changeable length (in bytes) and a synchronization pattern that signals its start. The synchronization pattern is a user-determined fixed bit pattern; it is the same for every frame of a data flow session. A frame header immediately follows the synchronization pattern, and a frame data field immediately follows the header.

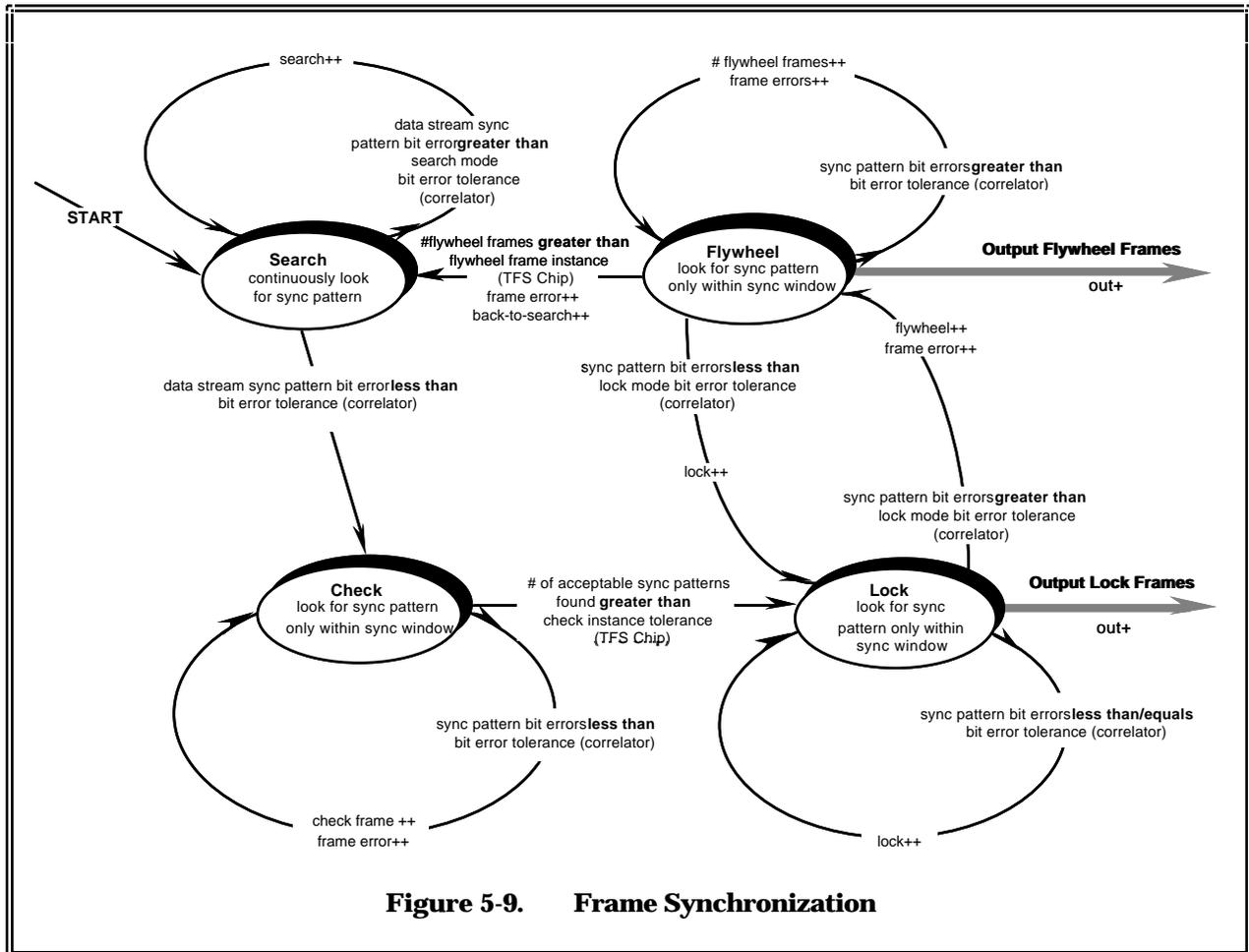
A frame header has a predefined format for information location. Examples of information contained in a header are data source and definition of frame data field contents. A typical frame processed by the EOS Frame Synchronizer Card conforms to CCSDS recommendations. The EOS Frame Synchronizer Card uses a four-mode synchronization strategy. Initially, the synchronization logic is in search mode. It remains in search mode until a valid frame synchronization pattern is received with fewer bit errors than the preprogrammed search mode bit error tolerance. It then enters check mode. Search mode has an optional best-match strategy. If best-match strategy is enabled and the GTFS Chip recognizes another occurrence of a synchronization pattern with fewer bit errors while processing the remainder of the frame, it disregards the previous pattern and restarts the frame synchronization process at the new location.

Once in check mode, the incoming frame synchronization patterns are compared (at the same place in the data stream) to the user-defined synchronization pattern using a check mode bit error tolerance. After a preprogrammed number of valid synchronization patterns are received (check tolerance), the logic enters lock mode. If the check tolerance value is zero, the logic moves directly into lock mode from search mode.

In lock mode, each frame synchronization pattern is compared to the user-defined synchronization pattern in the same relative place in the frame data stream. For any frame, if the number of bit errors exceeds the lock bit error tolerance, synchronization is missed and the logic enters flywheel mode. The GTFS Chip continues to look for the frame synchronization pattern in the same place in the data stream a programmed number of times before it reinitiates the search for the synchronization word. If the frame synchronization pattern is found in the correct place and meets the lock mode bit tolerance, lock mode is re-entered. If the number of consecutive flywheel frames reaches a preprogrammed limit (fly tolerance), the GTFS Chip returns to search

mode. The last frame processed in flywheel mode before the GTFS Chip goes back to search mode is called the back-to-search frame. Flywheel mode can be thought of as a submode of lock mode. For clarity in this discussion, flywheel mode is explained as a separate mode.

Figure 5-9 illustrates the four modes and the sequence in which the logic advances through the modes. Terms followed by ++ are counters that are incremented. If a condition is suffixed with (correlator) or (GTFS Chip), the condition value is set up in that device's register.



5.8.1.2 Modes of Operation

When the EOS Frame Synchronizer Card is processing data, it is operating in one of two possible modes: Normal or Test.

5.8.1.2.1 Normal Mode

In Normal mode, serial data is input via any of five external data input ports, and a 1-kHz timing signal is input via the P2 input interface. The card correlates and frame synchronizes the input data.

Serial correlated data is output to the front-panel Subminiature Assembly (SMA) connector. Parallel byte-wide or word-wide (16-bit) frame synchronized data with an EOF mark is output to the backplane and stored in a quicklook FIFO.

Timecode and/or quality trailers can be optionally appended to output frames. Therefore, HRTB output data can have four formats:

- a. Synchronized frames without annotation.
- b. Synchronized frames with a 2-byte quality trailer generated by the GTFS Chip.
- c. Synchronized frames with a 2-byte quality trailer generated by the GTFS Chip, and an 8-byte timecode trailer generated by the timecode interface logic.
- d. Synchronized frames with an 8-byte timecode trailer generated by the timecode interface logic.

5.8.1.2.2 Test Mode

In Test mode, test data is generated by the card and then processed. Processing and output are the same as in Normal mode; only the input data source differs. Input data is generated by the Test Generator Mezzanine. This mode can be used to perform card self-tests, or to have the card function as a simulated data source for other cards or systems via its front-panel SMA or HRTB output interfaces.

5.8.1.3 Configuration

Before data is flowed through the EOS Frame Synchronizer Card, it must be set up and enabled. The configuration is as follows:

- a. Frame size = 1024-bytes = 8192-bits.
- b. Bit slip detection and correction disabled.
- c. No frame sync pattern errors allowed.
- d. Check mode enabled (programmable: 1-15 frames).
- e. Flywheel mode enabled (programmable: 0-15 frames).
- f. No Cyclical Redundancy Check (CRC) check or Pseudonoise (PN) decoding.
- g. Polarity correction disabled.
- h. Reverse correction disabled.
- i. Append trailer containing quality information to each frame.
- j. Input from front-panel SMA connectors.
- k. True forward synchronization pattern (1ACFFC1D in hexadecimal).

Once the EOS Frame Synchronizer Card synchronizes frames from the incoming data stream using the selected configuration, it outputs them as 16-bit wide parallel synchronized CADUs with annotated trailer to the EOS Reed-Solomon Card over the HRTB via its J3 connector. Trailer annotation includes:

- a. Number of erroneous bits in the synchronization pattern (0-7).
- b. Mode under which CADU/CVCDU was output: search, check, lock, or flywheel.

- c. Direction and polarity: forward true.
- d. Back-to-search occurred (flywheel and check output mode).
- e. Timecode (8 bytes).

The EOS Frame Synchronizer Card is a 9U card; it includes a 3U Controller (68030-based MZ 8130) and a 6U custom card section. The custom section includes a Test Generator Mezzanine, which is used to perform internal lower-rate (<25 Mbps) data flow self-tests that exercise all card functions.

5.8.2 EOS REED-SOLOMON DECODER

The ERS Card (Figure 5-10) accepts byte or word data input (frames), processes data, and selectively outputs data to a variety of locations. Configurable frame processing schemes include Reed-Solomon frame header and data field decoding and error correction. Output formatting is flexible; for example, formatted frame status information may be automatically appended or prepended to output frames. Flexible ERS Card output schemes provide frame routing to internal card buffers and output ports based on user-programmable parameters. Output ports include the HRTB ports (any of six) and an RS-422 front-panel output.

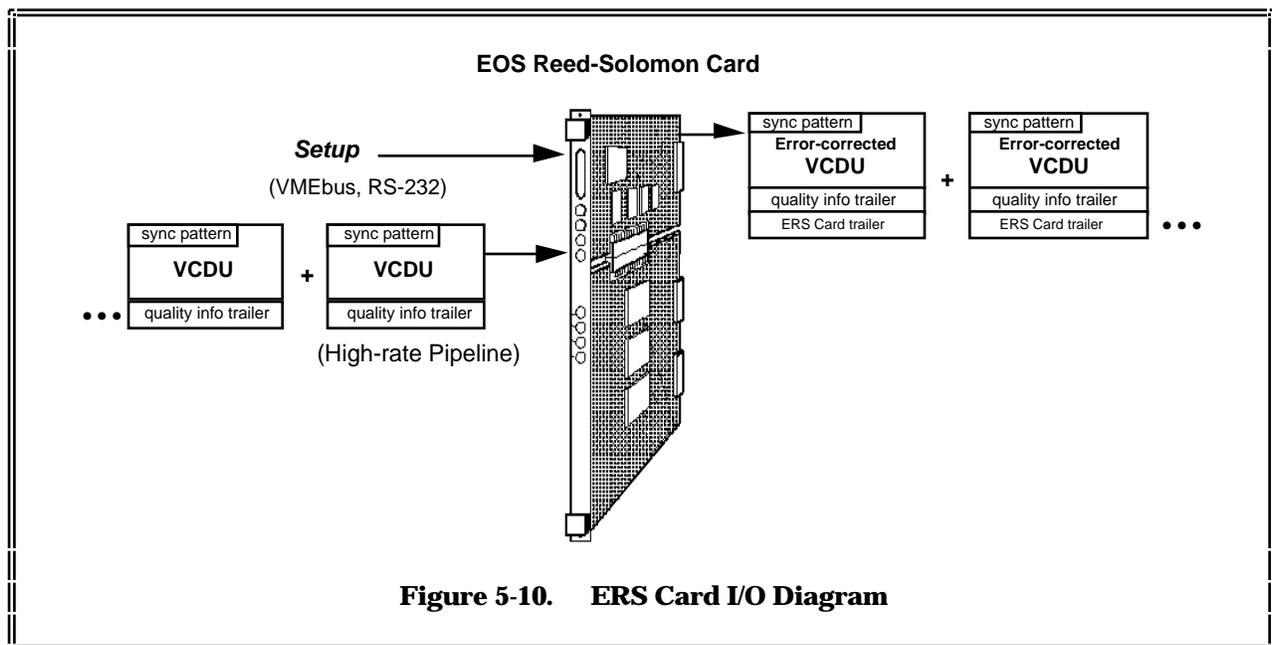


Figure 5-10. ERS Card I/O Diagram

5.8.2.1 Reed-Solomon Decoding and Error Correction

The CCSDS recommends use of Reed-Solomon codes to increase the reliability of transmitted data. Prior to transmission, data may be encoded, which generates parity information that is then appended to data to create a codeword. Following transmission, data may be decoded to locate and correct any errors induced in transmission.

The ERS Card supports decoding and error correction for the two CCSDS-recommended Reed-Solomon codes: (255, 223) Reed-Solomon for frame data, referred to as block code, and (10, 6) Reed-Solomon for frame headers, referred to as header code. The ERS Card can process frames with either type of encoding, both, or neither.

Reed-Solomon codewords have a fixed-length parity field and may have a variable-length data

field (within a range). The ERS Card can process codewords of any valid length. The block code has a parity field length of 32 bytes and a codeword length ranging from 33 to 255 bytes. Thus, valid data field lengths range from 1 to 223 bytes. The header code has a parity field of 2 bytes and a data field of 3 bytes. The block codeword size is variable, but CCSDS standards set a fixed size for header codewords. The block code can correct 16 byte errors; the header code can correct 2 nibble errors. Codewords that contain more errors than the code can correct are termed uncorrectable.

To increase the burst error correction capability of transmitted data, codewords may be interleaved. Interleaved codewords have nonconsecutive bytes. For example, if the interleave size is 2, the bytes of codeword 1 are at positions 1, 3, 5, etc. Figure 5-11 illustrates interleaved data. The ERS Card can process data with interleaves up to 16 codewords.

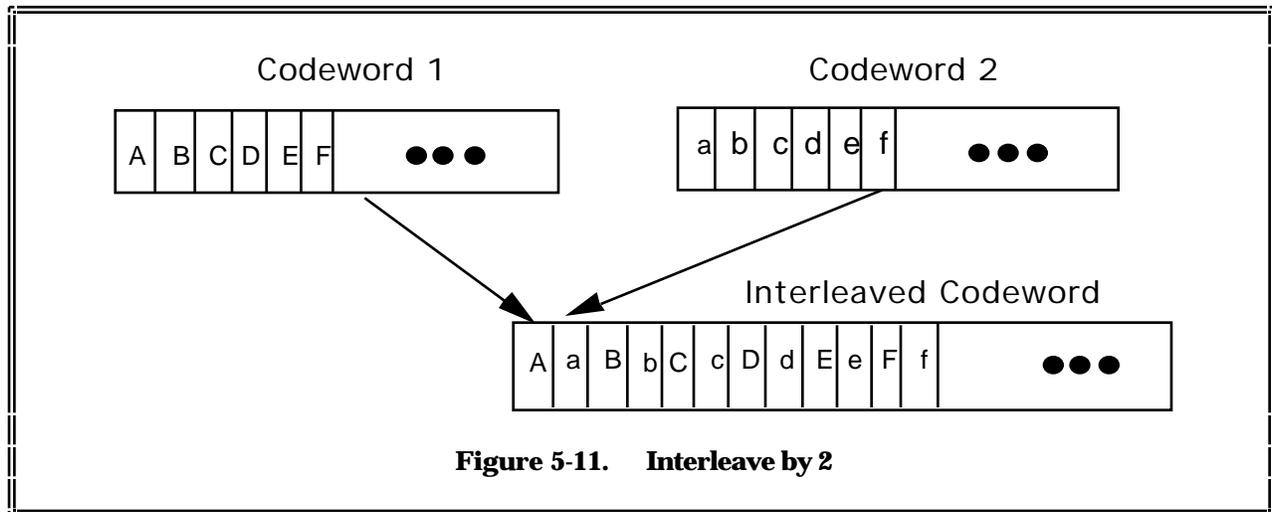
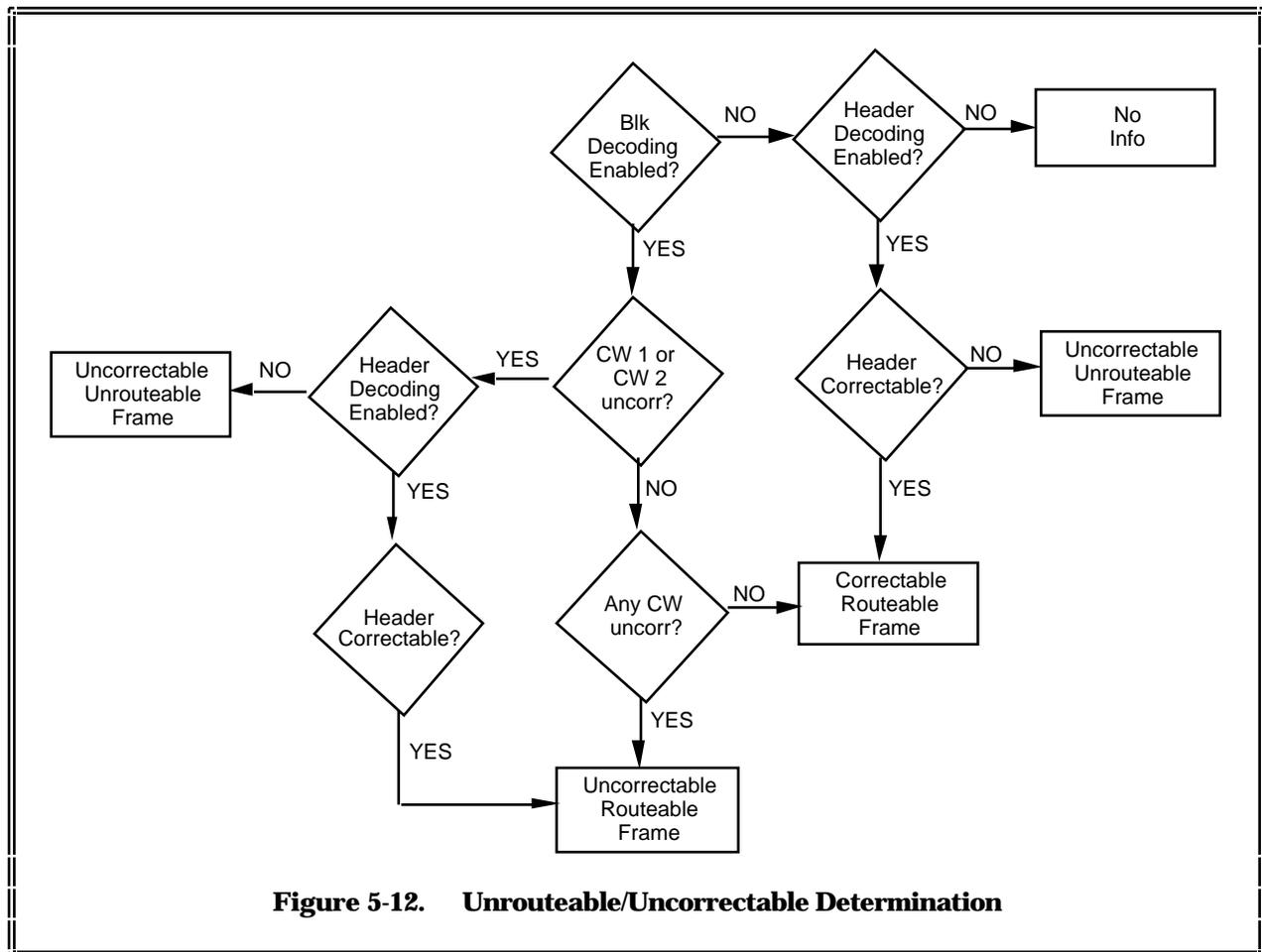


Figure 5-11. Interleave by 2

For a given data flow session, every frame has the same expected length. This expected length is used in conjunction with the EOF marker to verify the length of incoming frames. If the input source becomes unreliable, frames of incorrect length may be received. The card detects and reports frames that are longer or shorter than the expected length. Optionally, the ERS Card may automatically halt processing when a short or long frame is detected.

An uncorrectable frame is one that contains a codeword that has more errors than the Reed-Solomon code can correct. An unrouteable frame is one that has an uncorrectable header. The header field of a frame may be encoded with just the block code, just the header code, or both codes. The header field of block encoded frames is contained in codeword 1 (codeword 2 for interleaves greater than 1). Thus, if a frame is only block encoded and codeword 1 or 2 is uncorrectable, the header is also uncorrectable and the frame is unrouteable. If the frame is header encoded only and the header is uncorrectable, the frame is unrouteable. When the frame is both header and block encoded, the block code is checked first. If codeword 1 or 2 is uncorrectable, the header code is checked. If the header code is also uncorrectable, the frame is unrouteable. Thus, both codes are used to provide information on routeability and correctability (see Figure 5-12).



5.8.2.2 Modes of Operation

The ERS Card custom sidecard has two modes of operation: Normal and Test. Input to the card is frame synchronized data. Output from the card is uncorrected frames, frames with Reed-Solomon error-corrected headers, frames with Reed-Solomon error-corrected codewords, or frames with Reed-Solomon error-corrected headers and codewords. A status trailer can be optionally appended to each output frame.

5.8.2.2.1 Normal Mode

In Normal mode, data is input via a P3 connector. Data is synchronized frames with three control signals (which follow HRTB standards) in parallel bytes or words. Frame error detection and correction are performed based on ERS Card setup parameters. Data is optionally routed based on frame headers to HRTB ports and internal card buffers, and output via a P3 connector in parallel bytes or words. Three HRTB control signals are output as required by HRTB protocol.

5.8.2.2.2 Test Mode

In Test mode, input to the card is from the Test Actel. Data is parallel bytes or words with an EOF marker. Frame error detection and correction are performed based on ERS Card setup parameters. Data is optionally routed based on frame headers to HRTB ports and internal card

buffers, and output via the P3 connector in parallel bytes or words. Three HRTB control signals are output as required by HRTB protocol.

5.8.2.3 Configuration

The ERS Card can perform Reed-Solomon (255, 223) decoding and error correction using from 1 to 16 levels of interleave. It also performs frame routing based on frame SCID/VCID. This function enables the following additional user-configurable capabilities:

- a. Frames may be routed to any one of four HRTB output ports (VHS uses only one output port to the EOS Service Processor Card). This allows the user to select the set of Virtual Channels (VC) (up to eight) that have their data transferred to the EOS Service Processor Card.
- b. Frames may be filtered from the output pipeline data stream (frames filtered include fill VCID frames) and counted (up to 2^{24} frames). Uncorrectable or unrouteable frames also may be filtered.
- c. Invalid VCID frames may be written to a trash buffer FIFO (stores up to eight frames) for offline transfer to the remote user workstation over Ethernet.
- d. Frames with Reed-Solomon errors may be written to a quality reject buffer FIFO (up to eight frames), and the corresponding error locations and magnitudes to a separate buffer. This information will then be accessible to the ERS Card controller CPU for offline transfer to the remote user workstation over Ethernet. Once a CADU is decoded, the ERS Card appends the error status information to the frames. The following information is attached to each frame:
 - (1) Number of errors corrected.
 - (2) Uncorrectable error flag.
 - (3) Frame CADU/CVCDU status: long, short, unrouteable, uncorrectable, block uncorrectable, block code indicated errors, reason for rejection/filtering, number of codeword errors.
 - (4) Number of uncorrectable codewords.
 - (5) Number of codewords with errors.
 - (6) Bit field indicating error status for each codeword.
 - (7) Bit field indicating uncorrectable error status for each codeword.
 - (8) Number of bit errors in frame CADU/CVCDU.
 - (9) Number of errors in each codeword.

The cumulative status maintained by the ERS Card for the system MCC includes counts of:

- (1) Correctable and uncorrectable CADUs.
- (2) Filtered CADUs (including fill CADUs).
- (3) Illegal VCID CADUs.
- (4) Total number of CADUs received.

The ERS Card may be configured to output data in a statistics-only bypass mode. When operating in this mode, no error correction is performed, but routing is still done. Note that in this mode, the

ERS Card may output uncorrected frames to the EOS Service Processor Card, which may cause packet processing errors.

The ERS Card is a 9U card; it includes a 3U Controller (68030-based MZ 8130) and a 6U custom card section. The custom section includes a test mezzanine, which is used to perform internal data flow self-tests that exercise all card functions.

Data flow through the ERS Card is straightforward. In Normal mode, input data from the selected HRTB port is written into the input FIFO. The Reed-Solomon Error Correction (RSEC) Chip reads frames from the input FIFO, processes data as specified in its configuration registers, and outputs periodic status, correction information (if errors were detected), routing information, and the output frame. The routing information is processed in the programmable logic of the ERS Card; the appropriate write enable becomes active, writing the frame into the selected FIFOs (trash, quality reject, HRTB port FIFOs 0-5). Once an HRTB port FIFO becomes nonempty, programmable logic asserts the read enable to the FIFO and data is driven onto the HRTB by ERS Card drivers.

5.8.3 EOS SERVICE PROCESSOR CARD

The EOS Service Processor Card (Figure 5-13) performs high-speed telemetry data processing. It accepts a parallel input data stream that is 16 bits wide, with two 9th bit EOF marks. Typically, the input is CCSDS-defined Version 1 transfer frames or Version 2 CVCDUs. The input stream can contain data from up to 32 different sources, or, for CCSDS-defined frames, data from up to 32 VCs. The card reads the input data header and generates an output reassembly table that identifies the data pieces contained in its data field. The card tracks the data pieces during processing, reassembles the pieces to create a data unit (usually packets), and outputs the data unit. Annotation can be generated and attached to the output data unit as a header and/or trailer. Data is output to the HRTB, VSB, and/or loopback FIFO.

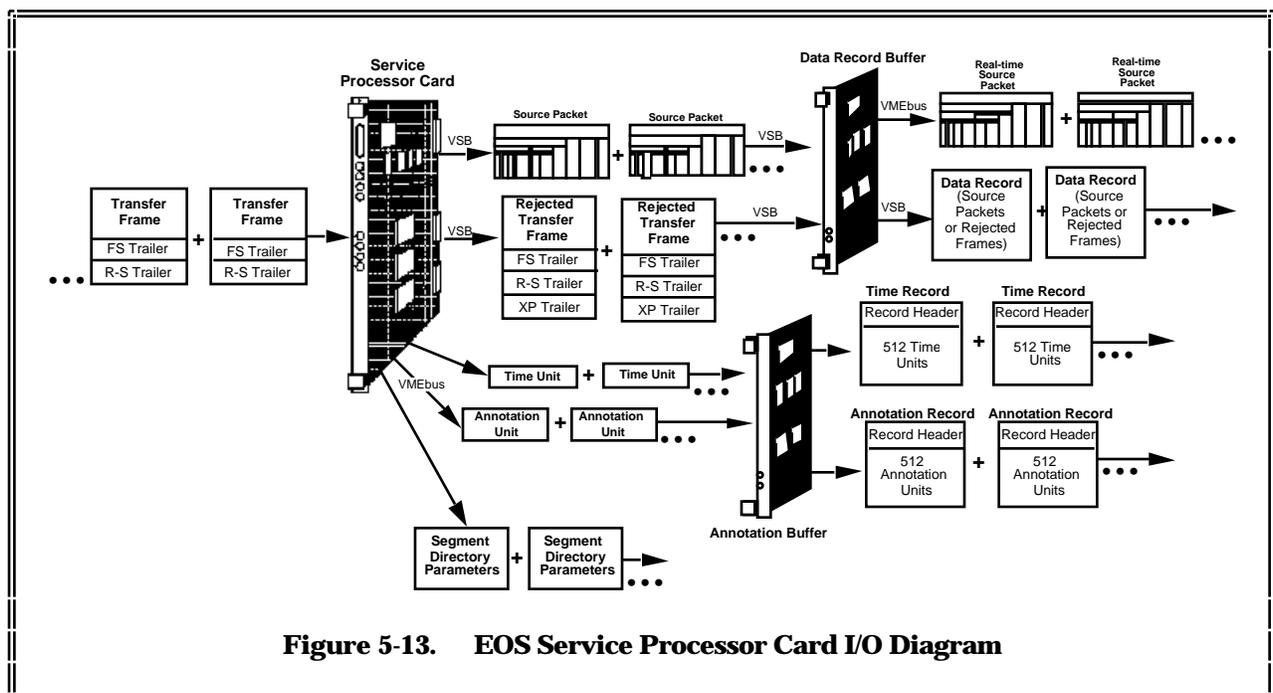


Figure 5-13. EOS Service Processor Card I/O Diagram

The EOS Service Processor Card also provides self-test capabilities. Software-generated test data may be input to the card via the Header Processor, while output data may be inspected by reading

the test FIFO via the Output Processor.

5.8.3.1 Modes of Operation

The EOS Service Processor Card has two modes of operation: Normal and Test. Internal data processing is the same in both modes; however, the input source and output destination differ.

In Normal mode, data is input from the HRTB. Data is output via the HRTB and/or the Direct Memory Access (DMA) circuitry.

In Test mode, data is input from the Header Processor. The data content is software-dependent. Typically, output is sent to the loopback FIFO. Data is examined via the Output Processor to ensure that no corruption occurred.

5.8.3.2 Configuration

All three processors on the EOS Service Processor Card must be set up before the card can flow data. The Quality Processor is the only processor that interfaces with the VMEbus; therefore, it must obtain all setup information and then transfer it to the other two processors. The Quality Processor boots and reads a start file. From this file, it sets up the hardware environment and begins to run the operational software. One of the Quality Processor's setup tasks is to download setup and operational files/parameters to the Header and Output Processors. The Quality Processor then sets a semaphore that tells the other processors to begin reading their start files.

Typical card configuration is as follows:

- a. The card may be configured to extract packets from up to 31 VCIDs, 8 SCIDs, and 512 APIDs. For EOS-AM, the user-selected values are 1 SCID, 8 VCIDs, and up to 64 APIDs.
- b. The card is configured to recognize data based on the combination of SCID, VCID, and APID.
- c. The card may be configured to reject or process bad packets based on user-selected options for bad packets as displayed on the configuration setup page. These selections may include bad APIDs, padded packets, fill packets, or packets from bad frames.
- d. The card may be configured to process or reject bad frames based on user-selected preferences as displayed on the configuration setup page.
- e. The card may be configured to route selected packets as a real-time service.
- f. The card may be configured to receive and pass through frames or CADUs to create a VCDU file.

5.8.3.3 Operation

The EOS Service Processor Card interfaces are shown in Figure 5-13. It accepts a stream of frames annotated with quality information from the ERS Card via the HRTB at rates up to 50 Mbps. The EOS Service Processor Card begins processing when data is input to the card via the HRTB. The HRTB input logic formats and passes data to the Tribuffer Subsystem. The Tribuffer Controller begins to process when the input FIFO goes nonempty.

The EOS Service Processor Card receives setup from, and accumulates quality and accounting status information for transfer to, the system Master Controller via the VMEbus. All card configuration parameters are accessible by the user.

Once data begins to flow, each processor performs its assigned tasks and communicates with the other processors. Communication between processors is via Dual-ported Random Access Memory (DPR) (software) and interrupts (hardware). Each processor includes a memory-mapped address that allows software to generate hardware interrupts to other processors. Reassembled packets may be output to the HRTB and/or VSB. Status and annotation are generated as data is processed. The EOS Service Processor Card implements packet extraction algorithms for CCSDS-recommended packet processing.

The EOS Service Processor Card performs packet processing and AOS services by extracting data units from the VCDU data field and checking errors. Each data unit is then packed in a new data unit, called a Service Data Unit (SDU), with quality annotation. For each packet, the EOS Service Processor Card also extracts spacecraft time from its secondary header. SDUs, timecode, and error information are sorted by source and temporarily stored in separate buffers. When a buffer is full, the EOS Service Processor Card notifies the appropriate module that the information is available. Timecode and error information is transferred by the Annotation Processor to the annotation disk for later processing; SDUs are transferred by the Data Set Processor to the data disk.

In addition to reassembling packets and other SDUs, the EOS Service Processor Card outputs rejected VCDUs, which are identified as source zero. Rejected VCDUs can result from frame errors, or errors found in the packets that the VCDU contains. When the EOS Service Processor Card outputs a rejected VCDU, it adds a 32-bit quality trailer to identify the location and nature of the error. For EOS-AM, the EOS Service Processor Card creates a file of VCDUs from frames with invalid VCs to simulate a “trash” buffer file.

Time and annotation records are briefly defined as follows:

- a. Time Record: EOS Service Processor Card extracts the sequence count and spacecraft time from each packet primary and secondary header, and builds an 8-byte timecode *unit* for every packet. A time *record* is complete when 512 timecode units from the same source are accumulated. A 4-byte record header is appended to every time record.
- b. Annotation Record: during packet reassembly, each packet is checked against specified criteria for errors. The EOS Service Processor Card generates an 8-byte annotation *unit* for any packet with detected errors. An annotation *record* is complete when 512 annotation units from the same source are accumulated. A 4-byte record header is appended to every annotation record.

Finally, the EOS Service Processor Card updates several parameters in the segment directory, which is a data base that tracks received telemetry data. The segment directory is shared by all processing subsystems.

The header of each SDU contains the following information:

- a. Type of data (e.g., packet, VCDU, or rejected VCDU).
- b. Size of SDU header and data field.
- c. SCID and VCID.
- d. Location of fill bytes in the SDU (note that all SDUs are filled to longword boundaries).
- e. Error flags.
- f. Size of VC gaps, if any.

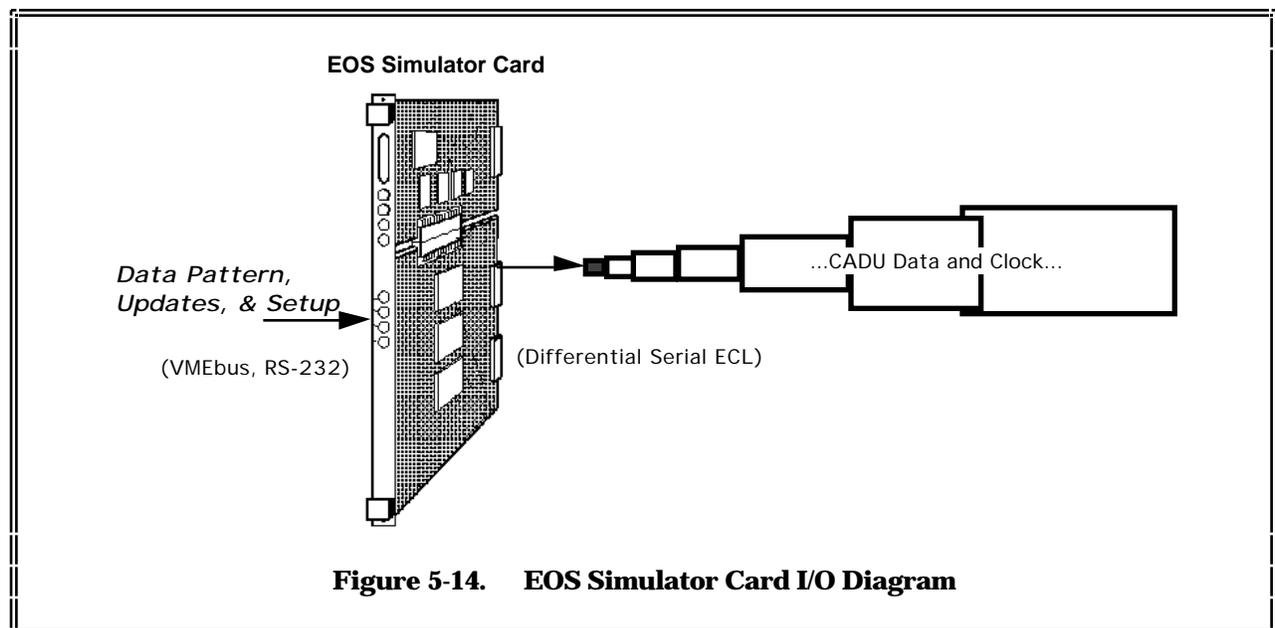
The cumulative status maintained by the EOS Service Processor Card for the system Master Controller includes:

- a. VCDUs received for each VC.
- b. SDUs received for each source.
- c. Incomplete or unrouteable SDUs.
- d. SDU size errors.
- e. VC sequence count errors.
- f. Packet sequence count errors.

The EOS Service Processor Card continues to process data as long as data is available. To finalize processing, the card requires a Flush command. This command cycles the Tribuffer Controller (transfers the last frame out), which allows the RAM Controller to process the remaining data without additional data being input.

5.8.4 EOS SIMULATOR CARD

Operationally, the EOS Simulator Card loads 4 Mbytes of dual-banked pattern memory with data generated by SCTGEN. The GTDG Chip then begins alternately reading the memory banks while an FPGA updates these banks with update information also supplied by SCTGEN. The GTDG Chip outputs data in byte-serial fashion, where it is optionally Reed-Solomon, CRC, and/or BTB encoded. The encoded data is transmitted byte serially across the HRTB, or serialized to create CADUs and transmitted from the front panel. EOS Simulator Card I/O is shown in Figure 5-14.



5.8.4.1 Modes of Operation

The EOS Simulator Card provides four areas of configuration: formatting, sequencing, updating, and encoding. Many combinations of configuration are possible, providing for an extremely flexible CADU generator.

5.8.4.1.1 Formatting

The GTDG Chip can be configured to operate in one of four possible formatting configurations, as follows:

- a. Forward True (FT): formats GTDG Chip output data in forward order with true polarity. This configuration is the most common; all available encoding configurations may be implemented in this mode.
- b. Forward Inverted (FI): formats GTDG Chip output data in forward order with inverted polarity. This configuration can only be implemented when no encoding is performed.
- c. Reverse True (RT): formats GTDG Chip output data in reverse order with true polarity. This configuration can only be implemented when no encoding is performed.
- d. Reverse Inverted (RI): formats GTDG Chip output data in reverse order with inverted polarity. This configuration can only be implemented when no encoding is performed.

5.8.4.1.2 Sequencing

The GTDG Chip is designed to access the dual-banked pattern memory as a programmable sequence of memory reads. This type of memory allocation allows the user to execute a sequence of memory accesses, with the option of switching between banks and/or repeating n times until a specific number of patterns are output. Pattern size can be configured as any number of bytes up to a maximum of 64 Kbytes. The GTDG Chip provides an EOF mark and an end-of-sequence mark. The GTDG Chip can be configured to operate in one of three possible sequencing configurations, as follows:

- a. Single Pattern (SP) Mode: configures GTDG Chip to transfer only one pattern of data and then stop.
- b. Multiple Pattern (MP) Mode: configures GTDG Chip to transfer a programmable number of patterns of data and then stop. The maximum size of patterns to be transferred in this mode is 4 Mbytes.
- c. Continuous Pattern (CP) Mode: configures GTDG Chip to continuously transfer patterns of data. This will continue until the GTDG Chip is configured to stop.

5.8.4.1.3 Updating

The EOS Simulator Card can be configured to provide dynamic updating of the dual-banked pattern memory. This mode allows the user to load VCDUs into memory and to then dynamically update certain fields to sequence the various counters within the VCDUs. These updates may be either pregenerated, or generated by the CPU.

5.8.4.1.4 Encoding

The EOS Simulator Card is designed to optionally encode VCDUs with a Reed-Solomon code, a CRC16 code, and/or a BTD code. With these available options, the card can be configured to operate in one of the following possible encoding schemes:

- a. Reed-Solomon Configuration: provides Reed-Solomon encoding of VCDUs to generate CVCDUs. This mode may operate with interleave values from one to four.
- b. CRC16 Configuration: provides CRC16 encoding of VCDUs.

- c. BTD Configuration: provides BTD to VCDUs by XOR'ing a PN255 code to the data stream. This encoding configuration may be combined with the Reed-Solomon or CRC16 configuration.

5.8.4.2 Configuration

The EOS Simulator Card is typically configured to output CADUs as follows:

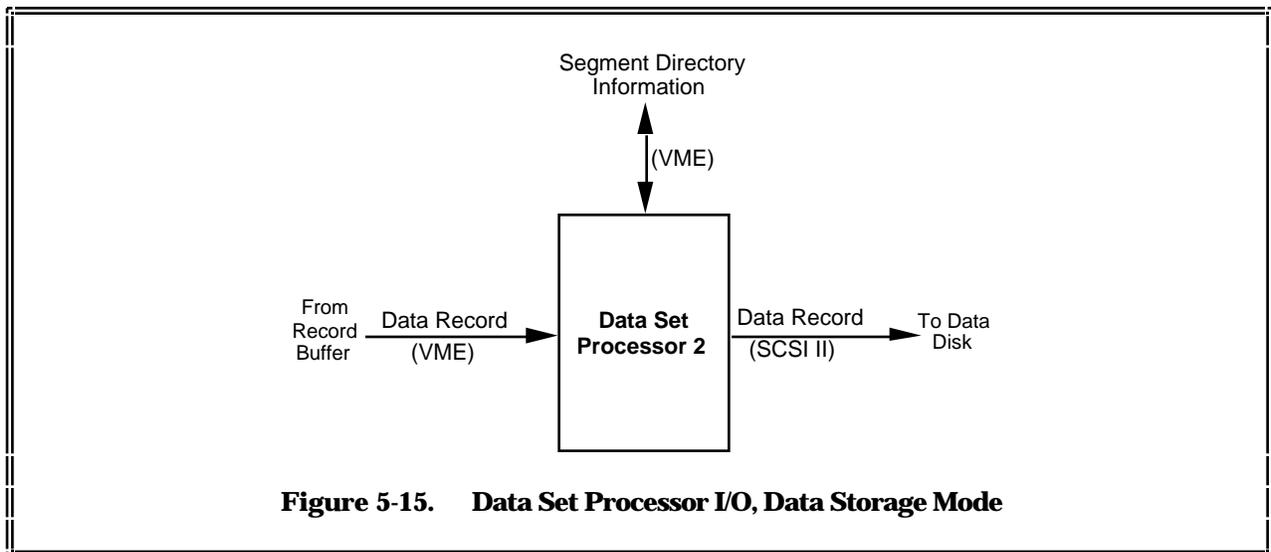
- a. When configured in continuous mode, the card will continuously output the contents of its 4-Mbyte data pattern memory.
- b. The card can be configured to output an exact number of CADUs, which may be a subset of the CADUs in the memory space, or a multiple of the CADUs in the memory space.
- c. The card can be set to output data at a clock rate from 1 kHz to 150 MHz, in 1-kHz steps.
- d. The CADU size is set in number of bytes.
- e. The CADU data file name (i.e., the CADU patterns) has to be entered for the correct pattern to be loaded onto the memory.
- f. The option to implement a bit slip of ± 3 bits.
- g. The option to Reed-Solomon encode a VCDU output data stream.

5.8.5 **DATA SET PROCESSOR**

There are two Data Set Processor Cards to perform the primary functions of storing data records, capturing data sets, and outputting data sets. They are largely independent functions (see Figures 5-15, 5-16, 5-17).

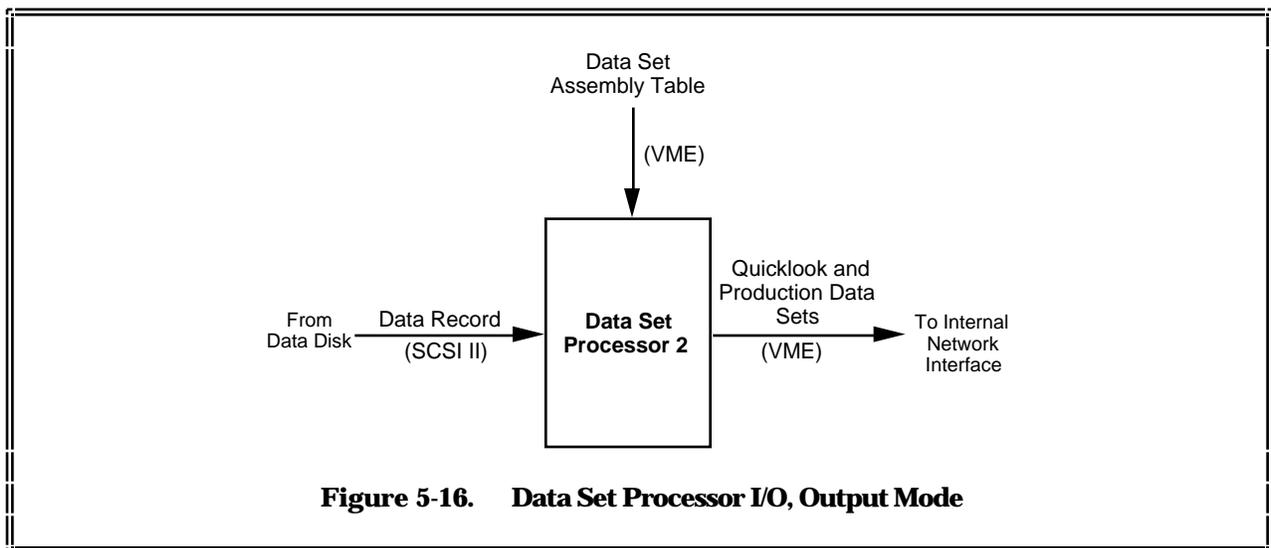
5.8.5.1 Data Storage Operation

Figure 5-15 illustrates Data Set Processor 2 I/O in data storage mode. When storing data records, the Data Set Processor waits for the EOS Service Processor Card to indicate that a data record on the data record buffer is ready for transfer. Each data record contains a set of SDUs, such as VCDUs and packets from a single source. The Data Set Processor then retrieves the record using VME64 DMA and transfers the record to the data disk through the SCSI interface. Data records are stored onto the data disk in sequential order; information about data record location is maintained in the segment directory.



5.8.5.2 Data Output Operation

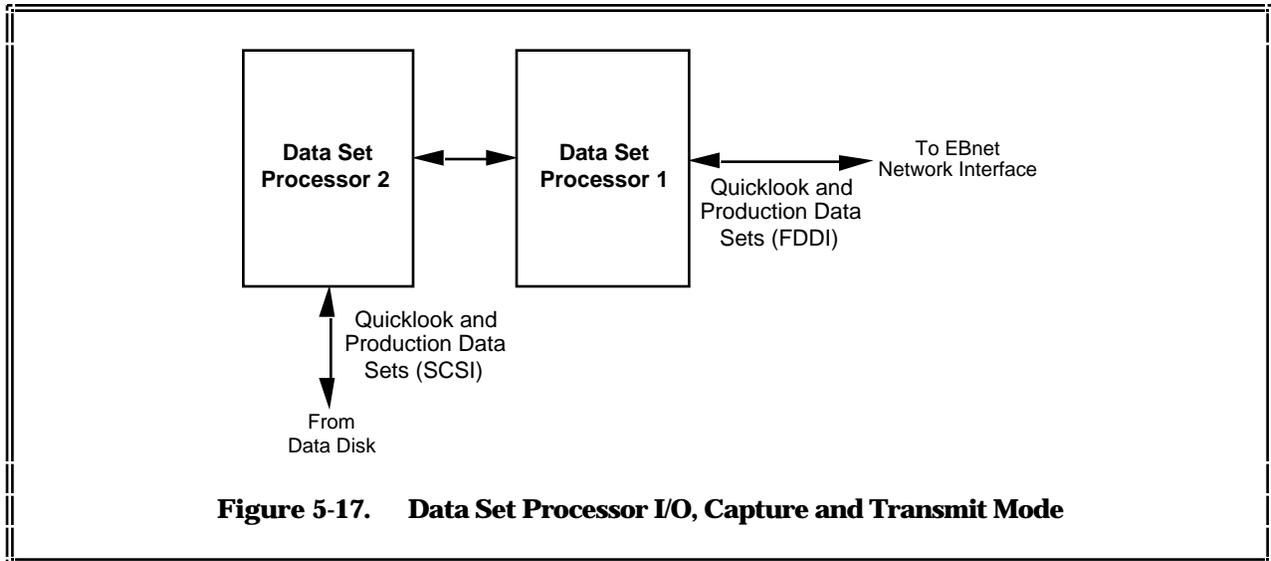
Figure 5-16 illustrates Data Set Processor 2 I/O in output mode. After receiving a data set request from the Control and Display Workstation, the Data Set Processor uses the segment directory and Packet Assembly Table (PAT) file to generate a data set. The PAT file is created by the Annotation Processor. It contains an instruction set on how to retrieve SDUs in the data disk to assemble a data set. Following these instructions, the Data Set Processor 2 reads SDUs from the data disk in specified order to form the data set, together with a data set header. The data sets that are created by the VHS are stored on tape media and transferred to the Data Disk prior to transmission to the External Network.



5.8.5.3 Data Capture and Transmission Operation

Figure 5-17 illustrates the data capture and transmit modes. When the data set is created either by the VHS or the SCTGEN, it is stored on the disk array just prior to transmission to the EBnet network interface. On initiating the FTP protocol, the data set is read from the disk array by Data

Set Processor 2 and transferred to the Data Set Processor 1 I/O for transmission. Data Set Processor 1, which acts as the FDDI Controller, delivers the data sets to the DAACs via the VMEbus; to the FDDI External Interface Card using FTP and TCP/IP. In data capture mode, data is received via FTP by Data Set Processor 1 I/O and transferred via Data Set Processor 2 to the disk array.



5.8.5.4 Configuration

The Data Set Processor has no specific setup procedures. Data storage mode is initiated when the EOS Service Processor indicates that a data record on the data record buffer is ready for transfer. Each data record, a set of SDUs in this case, is transferred from the 512-Mbyte memory card to the data disk via SCSI interface.

5.8.6 ANNOTATION PROCESSOR

The Annotation Processor interfaces are illustrated in Figure 5-18. It accepts time and annotation records from the EOS Service Processor Card via VMEbus, and transfers records to the annotation disk through the SCSI bus. Based on information in the segment directory, and information stored on the annotation disk, the Annotation Processor generates Data Set Assembly Table (DSAT) files, which the Data Set Processor uses to generate data sets. The Annotation Processor also generates Segment Order List (SOL) files that can be used later for generating custom-ordered data sets.

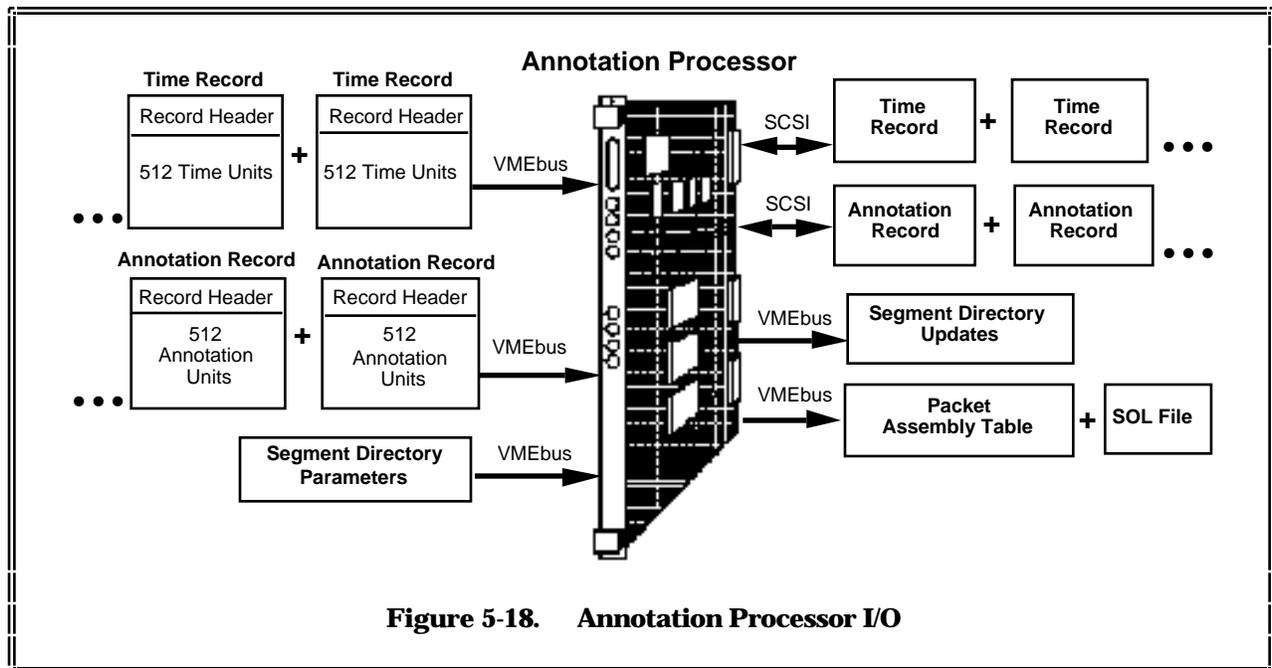


Figure 5-18. Annotation Processor I/O

At the end of each session, the Annotation Processor begins data set processing with newly received data based on production requests. DSAT files are generated for EDSs and PDSs, and stored on the system disk until all sensors are served. For each DSAT file generated, the Annotation Processor sends an event to the data distribution manager on the Control and Display Workstation to inform the availability of the data set. The Annotation Processor also periodically checks a timeline to see if any time-based data set request is valid. If yes, the Annotation Processor generates a DSAT file for the data set and sends an event to the data distribution manager.

The DSAT file provides the location on the data disk of the pieces that are to be output as a data set. The table includes a header, session list, error list, and gap list, and is generated using the following procedures:

- a. **Defining Data Segment:** for each source, the Annotation Processor checks to see if all packets received are in the same time order. The current segment ends and a segment begins when there is a time-order reversal in packet timecodes. Each segment makes an entry in the SOL file, characterized by the spacecraft time of the first and last packet of that segment.
- b. **Defining Data Set:** to generate an EDS, all segments received from a source are put in the DSAT file in the order received. To generate a PDS, time boundaries of all segments received from a sensor are compared and time-ordered before they are put in a DSAT file. If the time boundaries of two segments overlap, redundant packets are identified and deleted according to the quality of each individual packet.

As in the case of the Data Set Processor, the Annotation Processor has no specific setup. It is data-driven by the EOS Service Processor Card. While data is processed by the EOS Service Processor, the Annotation Processor receives annotation records and timecode information, which is stored on the annotation disk.

5.8.7 TRS AMPEX TAPE DRIVE

This system will be used primarily for transferring data from SCITF-generated Ampex tape cartridges to Sony cassettes. The only operational mode of interest is playback mode. The Ampex system will be provided with an external clock signal to output data from the tapes. General features of the system and operational modes are briefly described in the following sections.

5.8.7.1 Modes of Operation

- a. Record: data is recorded onto tape. The following conditions must be met before data is transferred to the tape:
 - (1) Record enable.
 - (2) Record command.
 - (3) Data ready (data and clock present).
 - (4) User data ready asserted.
- b. Playback: data is transferred from the tape media to the high-speed I/O channel, which in the TRS, is a serial ECL I/O port. The following condition must be met before data can be reproduced: data ready in asserted.
- c. Search: in Record or Playback modes, this feature allows the user to locate a given address, footage, or timecode on the tape from which to begin record or playback. When not in either of these modes, it allows the user to locate a given timecode, footage, or address.
- d. Shuttle: rapid transit of the tape in forward or reverse direction to locate an address, footage, or timecode value.
- e. Loading: when a cartridge is inserted into the drive, the cartridge is not yet locked into place. The tape within the cartridge has not yet been fully engaged within the drive path of the transport mechanism and the reel locks are still applied; the system is in the "unloaded" condition. In Loading mode, the load command engages the tape in the drive path.
- f. Unloading: similarly, the unload command puts the drive in unload mode and disengages the tape from the drive path; reel locks are applied.

5.8.7.2 General Features

The DCRsi 107 is capable of recording digital (or digitized analog) data at rates up to 107 Mbps for up to 1 hour. The system architecture combines high data rates, and high packing density and cartridge to provide long play time with ease of operation. The tape drive uses a rotary head/transport with transverse scanning, and supports error correction capabilities.

The DCRsi 107 is designed for remote control operation (only) of all primary modes (such as record, playback, and shuttle). There are no switches on the front panel that an operator can use to perform these primary functions. Remote control operation of the DCRsi 107 is performed via an RS-232 port through the CDS user interface. The DCRsi 107 includes the following capabilities:

- a. Reed-Solomon error correction with a Bit Error Rate (BER) of 1 in 10^9 .
- b. Variable data rates (continuous, in bursts, or changing).

- c. Timecode/block addressing to identify data blocks for marking events, search, etc.
- d. User data logging for storage of housekeeping information.
- e. One-hour maximum storage time per cartridge at 107 Mbps.
- f. "Instant on" record/playback.

5.8.8 TRS SONY TAPE DRIVE

The Sony DIR-1000 recorder provides large storage capacity and very high data transfer rates. General features of the Sony DIR-1000 and its many operational modes are briefly described in the following sections.

5.8.8.1 Modes of Operation

- a. Tape Out: a cassette is not in the unit. The drum does not rotate in this mode.
- b. Loading: from TAPE OUT mode, a cassette is inserted and the system goes to Loading mode, where the tape is wrapped around the drum. The drum rotates only while the tape is being wrapped.
- c. Standby: tape is wrapped around the drum and the drum stops rotating. The tape can start moving quickly from Standby mode. However, the tape might become damaged if it stays in this mode for a long time.
- d. Unloading: from Stop or Standby mode, the tape wrapped around the drum goes into the cassette and the system goes into Unloading mode, where the cassette is ejected from of the DIR-1000.
- e. Fwd: in this mode, data is reproduced. The signal on the control track and reference selected in the REF select mode are synchronized during reproduction.
- f. Rec: in this mode, data is recorded. It is necessary that CLOCK and SYNC signals are input from DATA INPUT because they are used as reference.
- g. Fast Fwd: tape moves in forward direction at the given speed. Speed is automatically reduced at the beginning and ending of tape. Maximum/minimum speeds are 10,000 mm/sec and 10 mm/sec, respectively.
- h. Fast Rev: same as Fast Fwd, except for direction and minimum speed, which is 100 mm/sec.
- i. Search: tape moves to search point. The DIR-1000 detects the specified point and automatically adjusts the speed. This function can only be used from remote control.
- j. Preroll: similar to Search mode, except that tape stops at a preroll ID count before the specified point.
- k. Sync: this mode controls tape phase in order to reach tape position (ID [SYNC point] specified when defining timeline value [SYNC time]).

5.8.8.2 High-Density Recording

The system uses helical scan recording to record wideband and high data rate digital signals. Using L-size cassettes, the DIR-1000 provides up to 770 Gbits of data storage capacity, which is

equivalent to 500 reels of conventional computer magnetic tape.

5.8.8.3 Tape Interchange

The system complies with ANSI X3.175-1990 ID-1 standards, maintaining compatibility with other data recorders using the same format.

5.8.8.4 High-Speed Recording/Playback

The system is capable of very high-speed recording and playback with a maximum data rate of 256 Mbps—approximately 10 times that of typical computer Magnetic Tape (MT) drives.

5.8.8.5 Time Commanding Capability

Recording and playback are possible at six different data rates: 256, 128, 64, 32, 16, and 10.7 Mbps, making the system suitable for different applications. Data can be recorded at an extremely high speed and played back at a slower speed, which is suitable for computer processing.

5.8.8.6 Error Correction

The system performs Reed-Solomon error correction using customized encoder and decoder chips to permit powerful error correction. After error correction, a bit error rate of better than 1×10^{-10} is achieved.

5.8.8.7 Read After Write

The playback heads of the system are placed such that recorded data is immediately played back during recording. This read after write facility makes it possible to monitor error conditions of recording in real time.

5.8.8.8 Search Function

The track set ID numbers recorded on the control track can be read at any tape speed during fast forward or rewind. When the search function is executed via remote control interface, a specified point can be found quickly and automatically by using the track set ID.

5.8.8.9 Remote Control Interface

The system is equipped with RS-422, RS-485, RS-232C, and GP-IB communication capabilities on the remote control interface, namely the Triplex Interface. The CDS will interface with the Sony tape drive via this remote control interface.

5.8.8.10 Self-Diagnostic Function

The system employs a built-in diagnostic system, which is designed to detect an operation error or hardware fault. An error message or warning information is fed to the CDS via the remote control interface, as well as to the front-panel display.

5.8.9 TRS TRIPLEX INTERFACE

The Triplex interface has a twofold function. First, it implements rate-buffering and transmission of the output data stream from the Sony tape drive at the required output data rate. Second, it provides the remote control interface to the Sony tape drive. The following sections briefly discuss operation and modes of the Triplex hardware.

The Triplex interface board is constructed with an industry-standard 6U VME form factor. The board sets used in the Triplex system in the TRS provide a direct cable connection to the Sony DIR-1000 tape drive. Multiple VSB/VME DPR boards are dedicated as buffer memory.

5.8.9.1 Sony DIR-1000 Interface

The Triplex Control Systems-Input/Output Control System (TCS-IOCS) Card implements all signals that comprise the Sony DIS-1000's parallel interface. The remote control serial port is fully supported using RS-422 differential drivers and receivers located on the card. The byte-wide parallel ECL interface to the Sony uses DB-25 connectors located on the IOCS front panel. The RS-422 serial command/control interface to the Sony uses a DB-9F connector located on the TSC-IOCS front panel.

5.8.9.1.1 Input from the Sony

This has three ports, namely ECL Parallel Data Out, ECL Data Clock, and ECL Sync. For ETS applications, the first two are used. ECL Parallel Data Out is byte-wide parallel data output from the Sony tape drive synchronous with data clock. ECL Data Clock is output clock from the tape drive synchronous with parallel data, with a maximum rate of 32 Mbytes per second.

5.8.9.1.2 Output from the Sony

Similarly, this has three interface ports: ECL Parallel Data In, ECL Data Clock In, and ECL Sync. For ETS applications, the first two are used. ECL Parallel Data In is byte-wide parallel data input to the Sony tape drive synchronous with data clock. ECL Data Clock In is input clock from the tape drive synchronous with parallel data, with a maximum rate of 32 Mbytes per second.

5.8.9.1.3 Control

A status and control port provides a 38.4 k baud, asynchronous communication interface to the Sony tape drive. An RS-422 electrical interface is provided to improve interface reliability.

5.8.9.2 VMEbus Interface

The TCS-IOCS is an intelligent peripheral device that implements both slave and master VME functions.

5.8.9.2.1 Master Mode

A32 addressing mode with D32 data format block transfers is used for TSC-IOCS VME master mode data transfers to/from VME memory.

5.8.9.2.2 Slave Mode

A24 and A16 addressing modes and D08 and D32 data formats are used.

5.8.9.3 VSB Interface

The VSB provides an alternative data path supporting transfer rates in excess of 32 Mbytes per second. DPR boards may be connected to the VSB to provide a high-speed channel between the TSC-IOCS data ports and VMEbus-accessible memory. Each TSC-IOCS in the system is configured via commands received from the host workstation across the VME interface. The TSC-IOCS can be configured as a data sink, data source, or passive.

TSC-IOCS VSB master mode transfers data over the VSB using 32-bit longword-aligned block mode data transfers. Transfer count and size registers are available, and generate VSB end-of-block interrupts to the onboard microprocessor or to the host system VMEbus.

5.8.9.4 External Clock and Sync Pulse Interface

The Triplex interface card, TSC-IOCS, can derive the Sony data clock signal from an external Transistor-to-Transistor (TTL) source. The Hewlett-Packard clock generator will be used to generate the clock signal when the Sony is recording data off the Ampex tape drive.

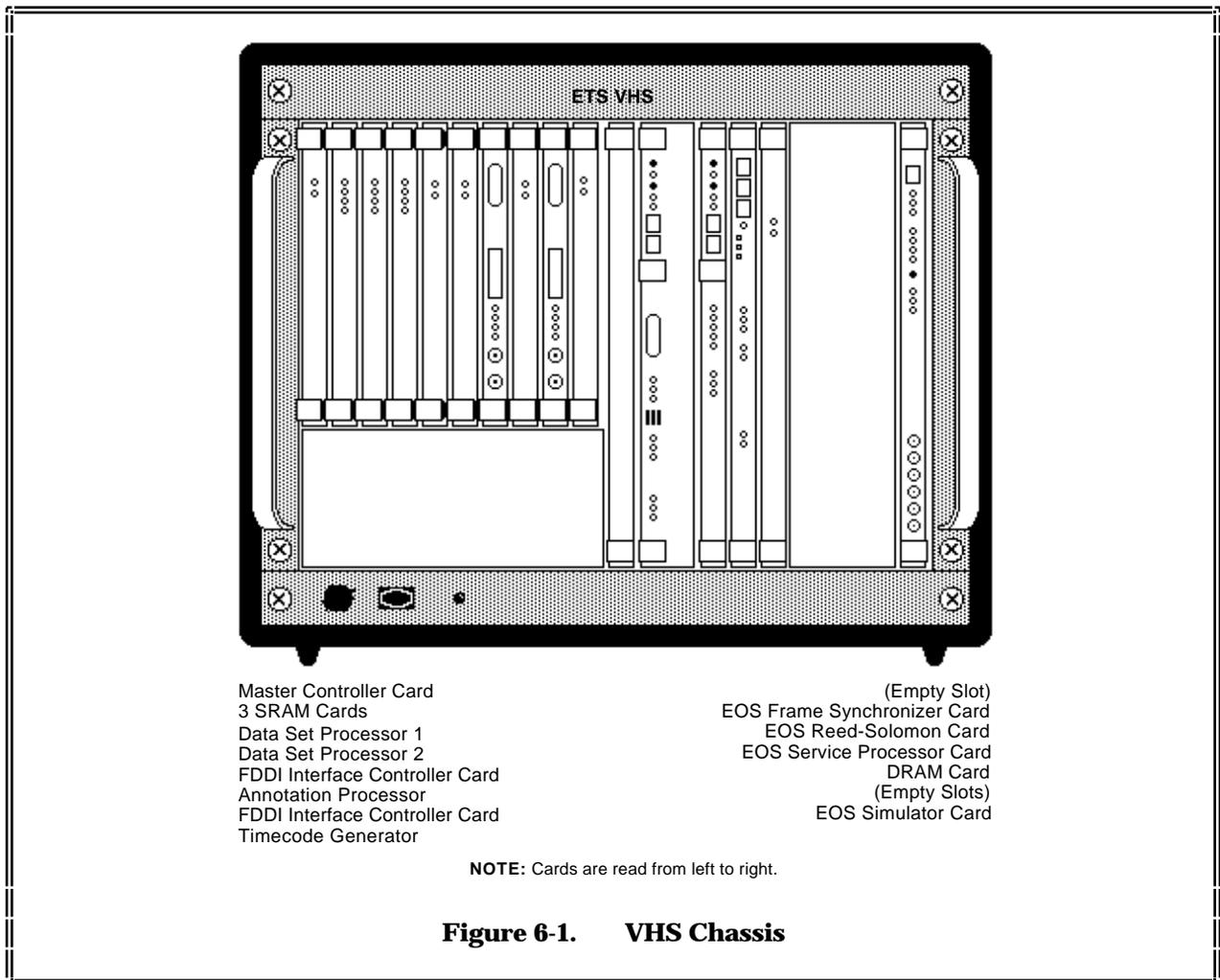
SECTION 6 DETAILED HARDWARE DESIGN

This section provides a board-level description of VHS hardware design. (The TRS hardware description and configurations are detailed in the Triplex, Sony DR-1000, and Ampex DCRSI107 system documentation.)

6.1 VHS CUSTOM COMPONENTS

The VHS includes five Code 521-developed custom components: EOS Frame Synchronizer Card, ERS Card, EOS Simulator Card, EOS Service Processor Card, and HRTB. Figure 6-1 illustrates the VHS chassis with the board and device configuration. In addition to custom cards, three COTS cards are software-configured to perform back-end data output generation and transmission. These three cards, Annotation Processor, Data Set Processor 1, and Data Set Processor 2, are data-driven by output data from the EOS Service Processor Card (and are described in this context only).

These cards are reused from previous projects; detailed information concerning front panels, Light-emitting Diode (LED) displays, jumper configurations, and card setups is provided in the respective Hardware Definition Documents (HDD) as listed in the Section 1.3.3.



6.1.1 EOS SIMULATOR CARD

The EOS Simulator Card is a 9U VMEbus A32:D32 master/slave card using a 68040 CPU mezzanine controller module mounted on the card. The custom logic card provides the specialized hardware functions for data simulation. The controller provides setup, hardware diagnostics, and control functions for the custom logic card. Refer to Section 7 for a description of EOS Simulator Card software.

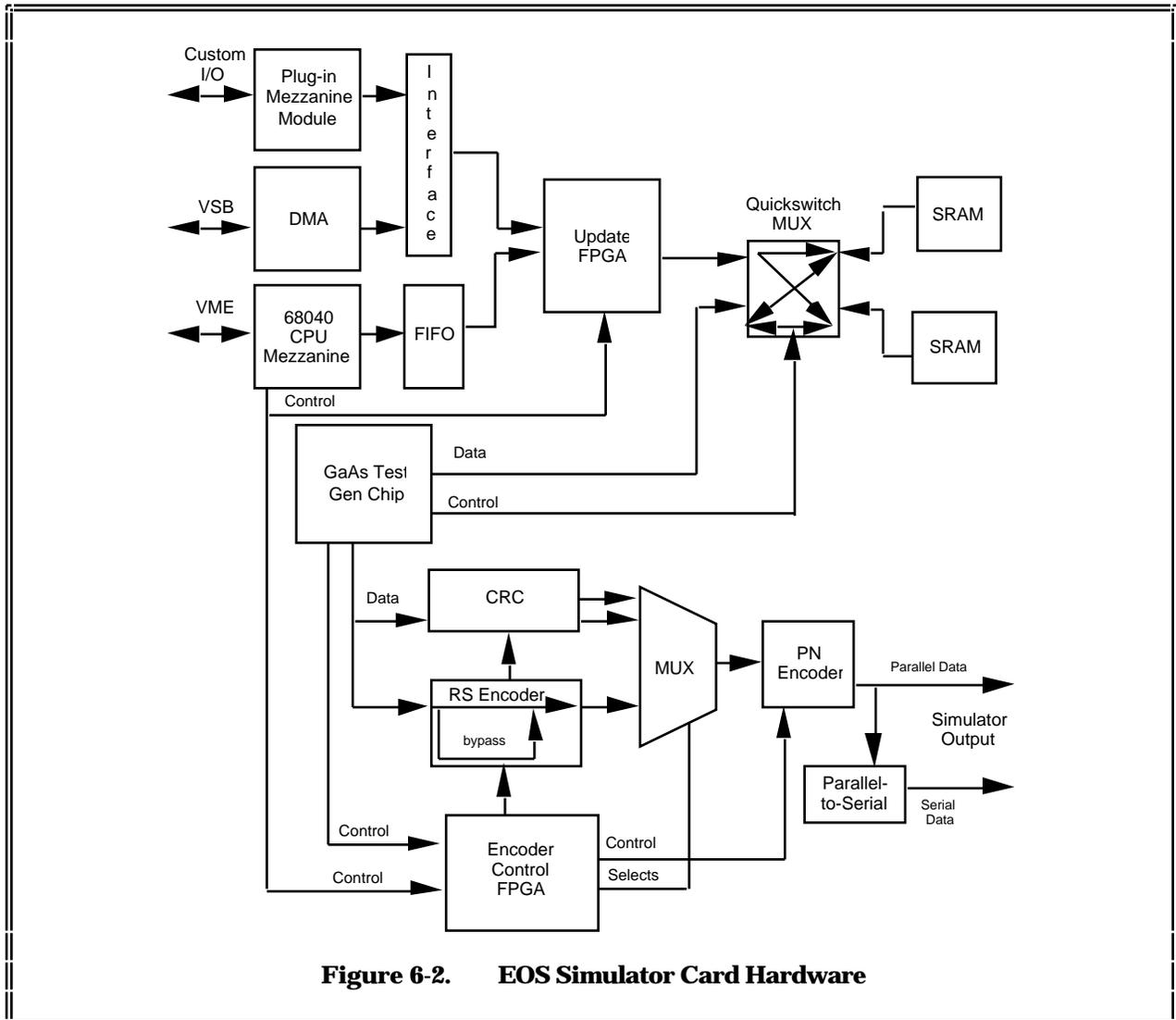


Figure 6-2. EOS Simulator Card Hardware

Figure 6-2 illustrates the EOS Simulator Card hardware. Functional elements of the custom logic card hardware are briefly defined as follows:

- a. 68040 microprocessor mezzanine controller provides control circuitry to address and configure the different card elements. The CPU mezzanine controller transfers setup and control information from/to the MCC via VMEbus, and writes CADU pattern data to the 16-Mbyte memory, as well as updated sequence numbers to the update FIFO and FPGA to support output of simulated CADU streams at rates up to 150 Mbps.
- b. Dual-banked 4-Mbyte static RAM pattern memory holds the SCTGEN-generated VCDU

data pattern. It is arranged as two memory banks that allow access from the Code 521 GTDG chip set, update FIFO, or CPU mezzanine controller. Each bank is implemented as two 256-Kbyte x 32 Static Random Access Memory (SRAM) modules. Two stages of crosspoint switches allow bidirectional multiplexing of buses from all three sources. Multiplexing is controlled by the controller or the GTDG. During memory loading, the controller operates the switches. Once output has started, the GTDG controls the switches, allowing ping-pong memory access between banks to provide updating on one bank while it outputs from the other bank.

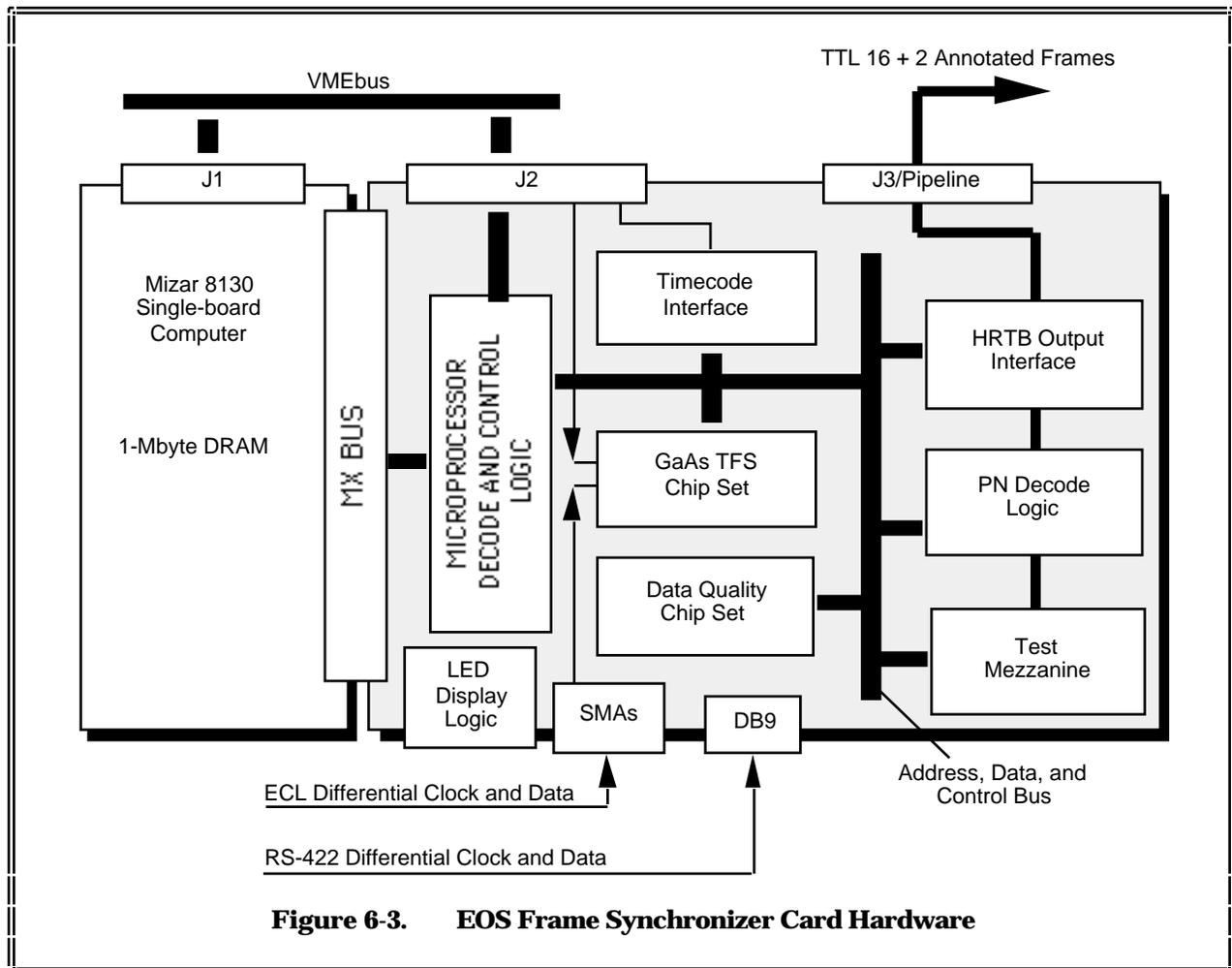
- c. Update circuitry, under SCSI configuration, updates one of the memory banks with information previously loaded by the SCSI interface while the GTDG outputs from the alternate bank. A 64-Kbyte update FIFO is used to decouple the SCSI interface from the dual-banked pattern memory during update mode. The update FPGA reads and decodes the update address from the update FIFO to determine which module and bank is to be accessed. It then determines which memory bank the GTDG is currently accessing and whether the update can be performed. If not, the FPGA waits for the GTDG to toggle to the other bank. Additional circuitry is included to interrupt the controller in case the GTDG toggles into the bank currently being updated by the update FPGA. This would indicate that the GTDG is operating at a faster rate than the updates can be loaded into the update FIFO.
- d. Numerically Controlled Oscillator (NCO) provides programmable clock rates for the GTDG chip set and the output serial-to-parallel conversion subsystem at rates from 1 kHz up to 150 MHz in 1-kHz increments.
- e. GTDG chip set, under controller configuration, alternates output of VCDU data from the two static RAM memory banks previously loaded by the controller using SCTGEN-generated data files. Patterns may be output continuously, or the operator may control the number of CADUs to be transmitted (up to 2^{24} CADUs) by switching between the two memory banks and repeating a programmable number of times until a specific number of patterns are output. The GTDG may optionally output data in forward, reverse, true, or inverted formats. The VHS only uses true forward data. Output streams are sent to the encoder section in byte-wide serial order. A built-in parallel-to-serial converter is provided on the chip to allow output as serial ECL clock and data at rates up to 150 Mbps. Once the GTDG has started a sequence, it assumes total control over data access and transfers.
- f. Encoder control FPGA controls the Reed-Solomon, CRC16, and BTD encoders. It generates the timing signals required for the transfer of data from the GTDG Chip to the encoders. A single control register is programmed to determine the type of encoding to be performed, whether the frame synchronization pattern word is to be attached to the data, the Reed-Solomon interleave depth, and the data frame size to be encoded. In the VHS, only Reed-Solomon encoding and frame synchronization word generation are used. The resulting data stream undergoes parallel-to-serial (clock and data) conversion at rates up to 150 Mbps.
- g. ERS Card encoder consists of a byte-wide ASIC developed by the University of New Mexico, which implements the CCSDS-recommended (255, 223) code. It receives VCDU data from the GTDG Chip in byte-serial fashion under control of the FPGA. The ASIC can support interleave depths (I) of one to eight. The Reed-Solomon code is a cyclic symbol error correcting code that is useful in correctly transferring data through a channel characterized by burst errors. The encoder operates on the 892-byte ($223 * I = 223 * 4 = 892$, for $I = 4$) long VCDU data space, and generates a 128-byte ($32 * I = 32 * 4 = 128$, for $I = 4$) long field of parity that is appended to the VCDU data to form a CVCDU (as

recommended for Grade-2 service in the CCSDS AOS Blue Book).

- h. CRC16 encoder consists of a byte-wide programmable logic device implementation of the CCSDS-recommended CRC16 cyclic redundancy code. It receives VCDU data from the GTDG Chip in byte-serial fashion under control of the FPGA. The cyclic code provides the capability of detecting errors that may have been introduced into transmitted VCDUs. The encoder operates on the VCDU data space to generate a 2-byte long field of cyclic redundancy code parity that is appended to the VCDU data as a VCDU data trailer field (as recommended for Grade-3 service, or optionally for Grade-2 service, in the CCSDS AOS Blue Book). The encoder is initialized to an all-ones condition between successive frames during the time allocated to frame synchronization mark insertion. The VHS does not use the CRC16 encoder.
- i. BTD encoder consists of programmable logic devices implementing the CCSDS-recommended pseudorandom sequence. It receives VCDU data from the error encoders in byte-serial fashion under control of the FPGA. The pseudorandom sequence is bit-wise exclusive-OR'ed with the data received in order to guarantee a minimum density of bit transitions in the data before it is modulated onto the physical space link channel. The encoder operates on the complete VCDU data space (as recommended in the CCSDS AOS Blue Book), but not on the sync marker. The encoder is initialized to an all-ones condition between successive frames during the time allocated to frame synchronization mark insertion. The VHS does not use the BTD encoder.
- j. Test FIFO provides card self-test ability. This 16-Kbyte x 8 FIFO receives VCDU data from the error encoders in byte-serial fashion under control of the FPGA. During each session, the test FIFO is written to by the FPGA and GTDG Chip until full. The FIFO has a byte-wide interface to the controller for data verification.
- k. Front-panel SMA output interface includes a parallel-to-serial converter that serializes and formats data from the encoder section as a differential 100K ECL output made available to the external world through front-panel SMA connectors.
- l. Generic plug-in mezzanine module input interface allows use of interface plug-in modules for access from external devices. Currently defined interfaces are: High-speed Interface (HSI) and SCSI. New interfaces can be accommodated through the creation of new plug-in modules. The SCSI mezzanine and interface are used in the VHS .
- m. LED control and display logic provides a visible representation of card status.

6.1.2 EOS FRAME SYNCHRONIZER CARD

The EOS Frame Synchronizer Card is a 9U card with a 3U MZ 8130 controller and a 6U custom logic card. The custom logic card provides the specialized hardware functions for CADU stream frame synchronization. The controller provides setup, hardware diagnostics, and control functions for the custom logic card. Refer to Section 7 for a description of EOS Frame Synchronizer Card software. Figure 6-3 illustrates the EOS Frame Synchronizer Card hardware.



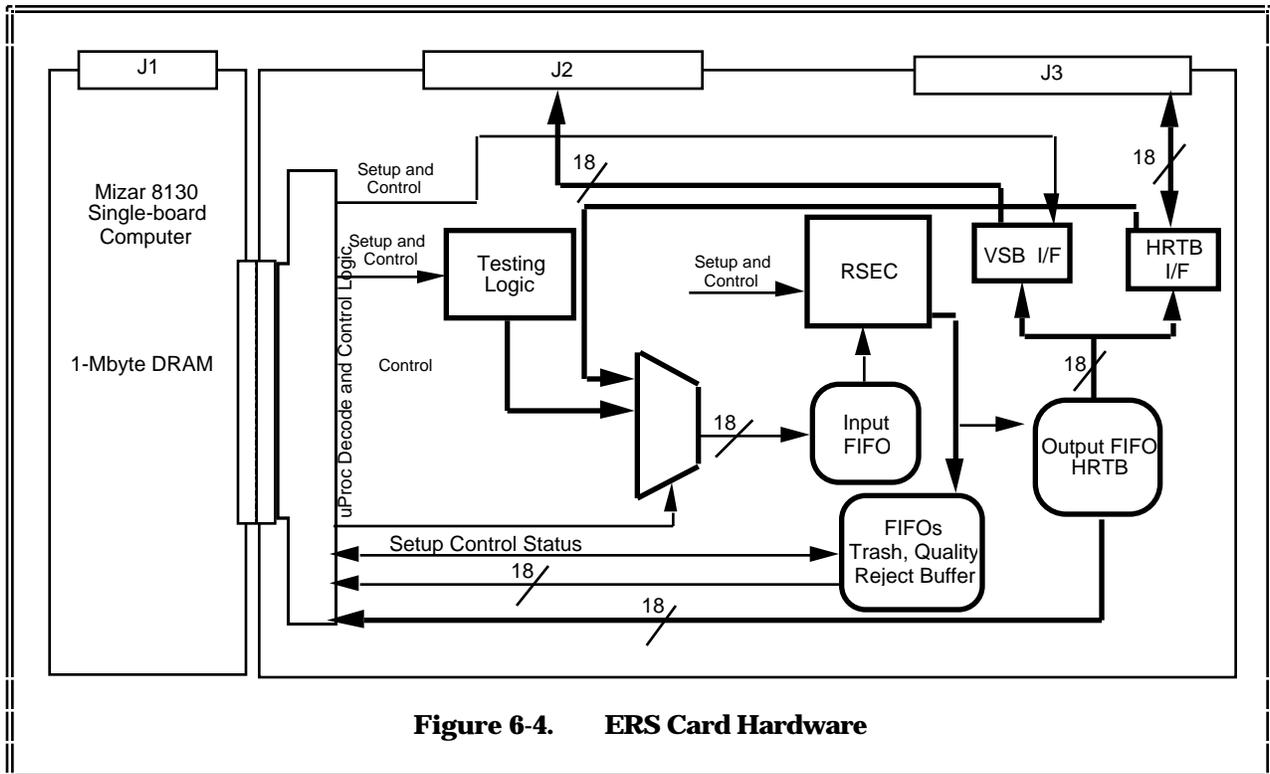
Functional elements of the custom logic card hardware are briefly defined as follows:

- a. Microprocessor decode and control logic interfaces with the MZ 8130 controller's MX bus side connector and supports the transfer of setup and control information from the controller to other custom logic card sections. It provides address and data bus buffering, and decodes the sidecard address space and the VME address selection.
- b. Test generator mezzanine, under controller configuration, generates test serial data for card self-test. It receives setup and test data from the controller via the motherboard. The test data pattern is stored in an onboard 64-Kbyte DPR. Once enabled by the controller, it can output the data pattern once, a fixed number of times, or continuously. The pattern memory is divided into two spaces, allowing update to one space while outputting from the other. It can accept external clocks from the motherboard, or clocks generated by an onboard NCO at rates up to 25 Mbps. Output data is transferred to the multiplexing logic as serial clock, data, and an EOF marker signal.
- c. Multiplexing logic, under controller configuration, selects the input to be passed to the ECL correlators: P2 connector RS-422 clock and data, front-panel DB-9 RS-422 clock and data, front-panel SMA ECL input clock and data, or two P2 connector ECL clock and data. The VHS only uses the front-panel SMA ECL input clock and data input. On powerup, data output to the ECL correlators is disabled until enabled by the controller.

- d. ECL correlators perform digital correlation to patterns configured by the controller. Two correlators are used to allow simultaneous correlation to reverse and forward patterns. In each correlator, an 8-bit mask register allows the synchronization pattern to be set to any length up to 32 bits, in increments of 4 bits. It receives the serial data stream selected by the multiplexing logic configuration and outputs it to the GTFS Chip and the front-panel SMA output interface connectors. Each correlator performs a bit-by-bit comparison of the incoming serial data stream, reports to the GTFS Chip when successful correlations within a programmable threshold occur, and indicates the occurrence of correlation errors.
- e. Frame synchronization logic is implemented using the GTFS chip set. The GTFS receives serial input data, and successful correlation and sync error magnitude signals from the ECL correlators. The chip set frame synchronizes data, processes synchronization error counts from the ECL correlators, optionally corrects inverted or reversed frames, converts serial data to parallel, computes and maintains data quality information, and outputs the frame with an optional 2-byte annotation trailer reflecting data quality. The GTFS uses two RAMs to simultaneously input a new frame and output the last stored frame. Frames of incorrect length are adjusted to the proper length. It also performs CRC16 error detection when enabled.
- f. Output FIFO receives parallel data from the frame synchronization logic and temporarily stores it before it is sent to the HRTB output interface.
- g. Quicklook FIFO receives synchronized frame data from the frame synchronization logic before annotation or bit transition decoding. It may be accessed by the card controller to examine the data and ensure that the synchronization circuitry is operating properly.
- h. HRTB output interface, under controller configuration, transfers frame data from the output FIFO to the HRTB via the J3 connector as 16 bits of data and an EOF marker. It is implemented in an FPGA. It receives synchronized frames including the 2-byte trailer from the output FIFO and the value stored in a BTD dual-ported memory. It optionally time-stamps the frame with an 8-byte time trailer from the timecode interface. It transfers the resulting data synchronously to the HRTB via the output port configured by the controller.
- i. Timecode interface is implemented as part of the HRTB output interface FPGA. It accepts a 1-kHz timing signal input from the P2 connector, computes Greenwich Mean Time (GMT), and provides an 8-byte PB-5 timecode, which may optionally be used to time-stamp output frames. At card initialization, timecode registers are loaded from a valid time source, and the time used as a reference for all subsequent frame time-stamping. Timecode counters are incremented by the input timing signal. The VHS will use this function.
- j. BTD dual-ported memory contains a static pseudonoise bit pattern that can optionally be exclusive OR'ed with frame data to perform bit transition decoding. The pattern is loaded by the controller before data flow is initiated.
- k. Cumulative data quality FPGA accumulates counts for hardware event status. It contains sixteen 32-bit ripple counters available as hardware or software counters that are accessible to the controller via a 16-bit interface. These counts are displayed on the Frame Synchronizer Card status page. Bit Error Rate (BER) statistics can be ascertained by monitoring these counts.
- l. LED display logic displays frame synchronization status.

6.1.3 EOS REED-SOLOMON CARD

The ERS Card is a 9U card with a 3U MZ 8130 controller and a 6U custom logic card. The custom logic card provides the specialized hardware functions for CADU stream Reed-Solomon error decoding, detection, and correction. The ERS Card receives CADU frames from the ERS Card, detects and corrects errors in data, and outputs data to the EOS Service Processor Card. The functionality of this card can be optionally bypassed. The controller section provides setup, hardware diagnostics, and control functions for the custom logic card. Refer to Section 7 for a description of ERS Card software. Figure 6-4 illustrates ERS Card hardware.



Functional elements of the custom logic card hardware are briefly defined as follows:

- a. Microprocessor decode and control logic interfaces with the MZ 8130 controller's MX bus side connector and supports the transfer of setup and control information from the controller to other custom logic card sections. It provides address and data bus buffering, and decodes the sidecard address space and the VME address selection.
- b. Test logic and data generator mezzanine, under controller configuration, generates test data for card self-testing. It receives setup and test data from the controller via the motherboard. The test data pattern is stored in onboard RAM. Once enabled by the controller, it can output the data pattern once, a fixed number of times, or continuously. Output data is transferred to the input FIFO by a multiplexer that selects whether the card is processing test data, or data received from the pipeline interface.
- c. Input FIFO holds input CADUs received from the EOS Frame Synchronizer Card via the pipeline interface, or from the test logic mezzanine before processing by the RSEC Chip.
- d. RSEC chip and routing logic, under controller configuration, reads data frames from the input FIFO, deinterleaves the frame (1 to 16 levels), detects errors in codewords

and/or headers (several codewords comprise one frame or data unit), corrects errors, annotates frames with quality data, and performs table lookup routing/filtering based on spacecraft/VCDU. The RSEC Chip performs most functions of the ERS Card at rates up to 150 Mbps.

The Reed-Solomon correction mechanism is described as follows:

- (1) A frame enters the card via the input FIFO and proceeds to the RSEC Chip, which conducts four concurrent processes that form a processing pipeline.
 - The first process buffers the current frame.
 - The second process deinterleaves and reads the header of the frame that came in one frame time before the current frame, and passes it to the correction and annotation logic.
 - The third process annotates correction and quality status onto the frame that came in two frame times before the current frame, and passes it to the output control logic.
 - The fourth process passes the frame that came in three frame times before the current frame to the output FIFO.
- (2) Each frame header field is error-corrected using the CCSDS (255, 223) Reed-Solomon block and/or (10, 6) Reed-Solomon header code. Use of either code is optional. The VHS does not use the (10, 6) header code.
- (3) Each frame data field is deinterleaved (programmable from 1 to 16 levels) into component codewords, which serve as input to the ERS Card decoder/encoder chip set. The chip detects and attempts to correct any errors; a maximum of 16 errors per frame can be detected and corrected. It then annotates the frame with error status, including the number of corrected errors in the frame and a flag indicating whether there were any uncorrectable errors in the frame. The RSEC Chip may attach additional data flow information. This is not needed in the VHS.
- (4) For the VHS, if there are no errors or all errors are correctable, decoding is considered successful and the data field-corrected frame is output. In addition to frame data, the RSEC Chip also outputs routing data (obtained from the routing table memory over seven dedicated output pins). The routing logic Programmable Logic Device (PLD) uses this information to generate the signals required to write output frame data to the destination. Possible destinations are: one or two of six HRTB output ports, VSB interface, trash FIFO, quality reject FIFO, loopback FIFO, and RS-422 output interface. The routing table is also configured so that any frame corresponding to an invalid SCID/VCID combination will be output and routed to the trash FIFO, and any frame found to have uncorrectable errors, or found to be unrouteable, will be output and routed to the quality reject FIFO. Reed-Solomon correction information for frames stored in the quality reject buffer is sent to a separate buffer. All other frames (including frames from the fill SCID/VCID combination) are filtered (not output).

An independent count of frames output to each output port or FIFO is maintained. The RSEC Chip also provides signals and/or counts to the FIFO to maintain cumulative status of card data flow: correctable frames, uncorrectable frames, filtered frames, illegal VCID frames, etc.

The RSEC Chip and routing table may also be configured in a bypass mode that will output all received frames to one or more RSEC Chip output ports, without regard to error status. All cumulative statistics will still be available.

- e. Routing table memory, implemented as a 64-Kbyte x 8 array and configured by the card's controller, is used by the RSEC Chip and routing logic to determine which output ports to

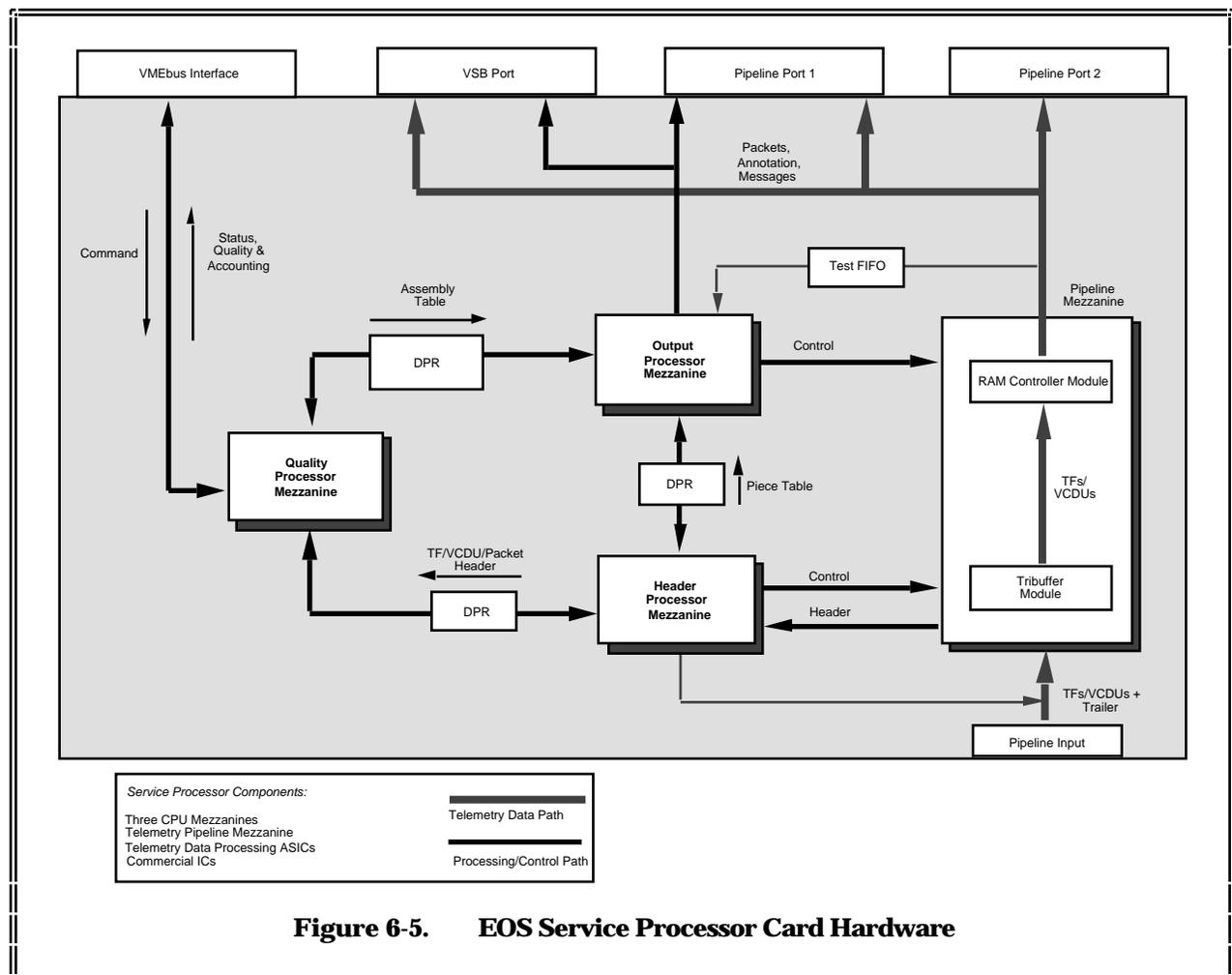
route and output each frame's data to based on SCID/VCID field combination. The RSEC Chip outputs this routing information over seven dedicated output pins.

- f. Trash FIFO, implemented as a 4-Kbyte x 16 array, is used to hold up to seven invalid VCID CADUs before transfer to the user upon request via the ERS Card and system MCCs.
- g. Quality reject FIFO, implemented as a 4-Kbyte x 16 array, is used to hold up to seven erroneous CADUs before transfer to the user upon request via the ERS Card and system MCCs. Reed-Solomon correction information for the eight frames stored in the quality reject buffer is maintained in a separate buffer that is also available to the controller for transfer to the user.
- h. OutputFIFO is used to hold corrected and annotated CADUs before transfer to the EOS Service Processor Card via the high-rate pipeline interface.
- i. High-rate pipeline interface logic is used for configurable input of frame data from the EOS Frame Synchronizer Card via one of the six ports in the high-rate telemetry pipeline, or the output of frame data to the EOS Service Processor Card via the high-rate telemetry pipeline. Selecting the same port for input and output will configure the port as output only.
- j. VSB interface logic is used for configurable I/O of frame data to the system's VSB. The VHS does not use this function.
- k. Quality spectrum accumulator FPGA/FIFO includes a set of registers that maintain hardware counts and status derived from signals and counts received from the RSEC Chip and other logic.
- l. LED display logic displays Reed-Solomon decoding and I/O status.

6.1.4 EOS SERVICE PROCESSOR CARD

The EOS Service Processor Card is a 9U VMEbus A32:D32 master/slave card using four card-mounted mezzanines: three 68040 CPU mezzanine controllers and a telemetry data pipeline mezzanine. The 68040 mezzanines are designated as the Header Processor, Quality Processor, and Output Processor. These processors are configured to be independently programmed and operated. The pipeline mezzanine stores frames and assembles packets. The main board provides interfaces for the VMEbus, VSB, backplane, and the 68040 mezzanines. Each of the three processors is linked to the others by DPR, has a separate bus, and executes independently. Each processor contains an onboard VxWorks System/Environment (VxWorks) operating system.

In conjunction with the four mezzanines, the custom logic card provides the specialized hardware functions for service processing, including packet extraction from incoming frames. One of the CPU mezzanines (Quality Processor) acts as a card controller, and provides setup, hardware diagnostics, and control functions for the custom logic card. Refer to Section 7 for a description of EOS Service Processor Card software. Figure 6-5 illustrates EOS Service Processor Card hardware.



Functional elements of the custom logic card hardware are briefly defined as follows:

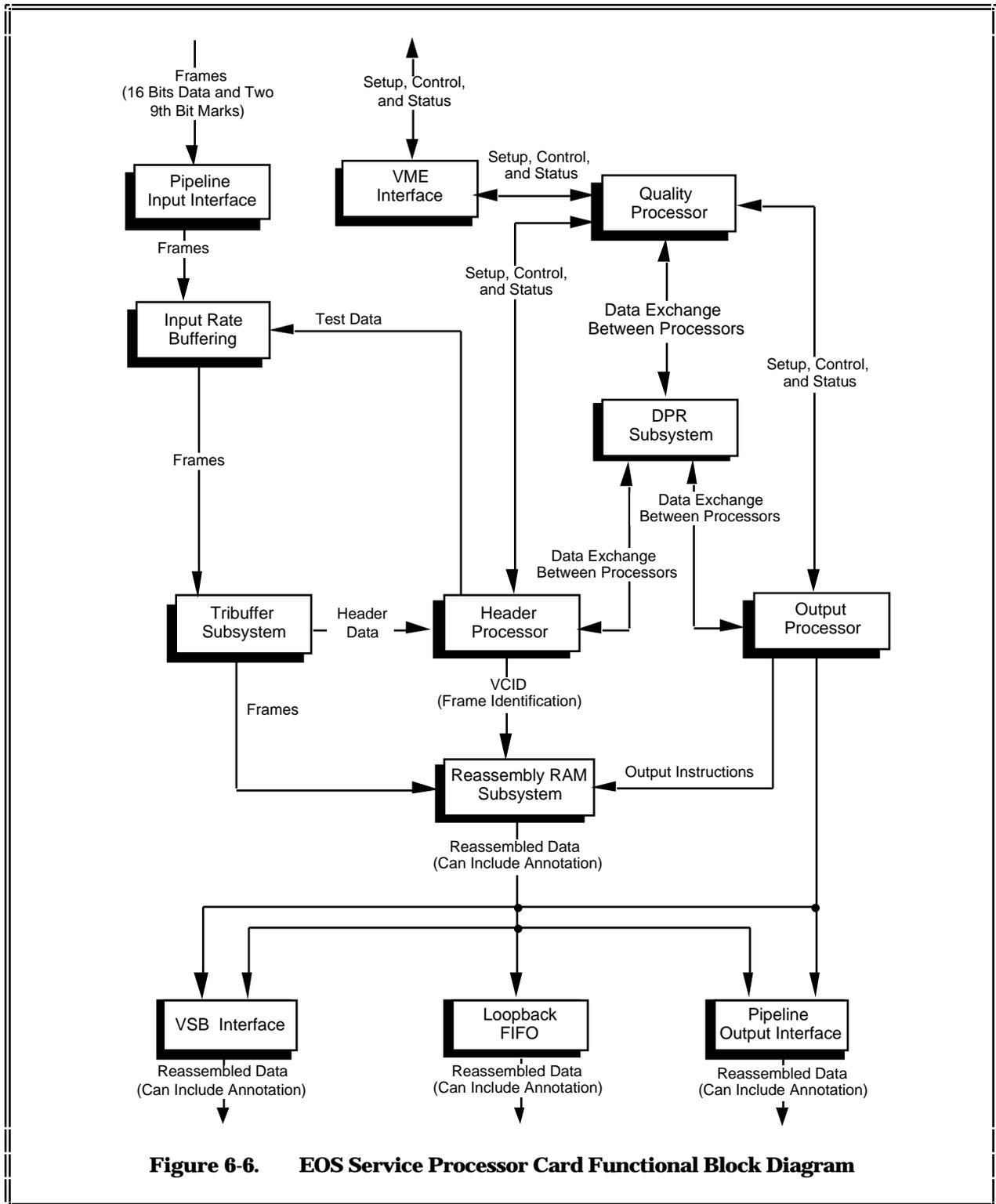
- a. **Pipeline input interface** receives a parallel input data stream from the HRTB, which is the third bus connector on the VMEbus. The maximum sustained input rate for this portion of the card is over 500 Mbps. To achieve higher transfer rates, the HRTB transfers 2 bytes (and 9th bits) simultaneously. The HRTB input logic destacks these words into a 9-bit stream to the Tribuffer subsystem. The 9 bits in an input data unit are defined as follows: 8 bits data and 1-bit EOF mark (9th bit). On the last byte of a frame, the 9th bit is set to 1; otherwise, it is 0. The expected data type is CCSDS-defined Version 1 transfer frames or Version 2 CVCDUs with a 9th bit EOF mark; however, any data unit with a 9th bit EOF mark can be processed. The pipeline logic, which includes the input interface, can be reset in one of two ways: manual card reset via front-panel reset switch, or software-generated pipeline reset via Output Processor.
- b. **Input rate buffering subsystem**, a 4K x 18-bit input synchronous FIFO in series with a 16K x 9-bit asynchronous FIFO, provides rate-buffering of data input to the EOS Service Processor Card. The asynchronous FIFO is connected to the HRTB input bus (via external P3/J3 connection), or the Header Processor data bus (via internal mezzanine-to-motherboard connection). Bit 3 in the Header Processor status/control register determines the input source (0 = external, 1 = internal). Header Processor input is only

used for testing purposes.

- c. Tribuffer subsystem provides temporary storage of data frames; it includes a Code 521-designed VLSI Enhanced Tribuffer Controller ASIC and three 8-Kbyte RAM buffers. The subsystem reads data frames from the input FIFO, stores each frame separately, and creates each frame and its associated packet primary header(s). After the Header Processor reads and processes a frame, it outputs that frame to the reassembly RAM subsystem. The Enhanced Tribuffer Controller is a generic, three-stage pipelined data buffer controller with several built-in functions that are useful in telemetry data processing. These pipeline controllers are normally executed simultaneously to allow data processing at very high data rates. Pipeline stages are defined as follows:
- (1) Input Controller (first stage): moves data units from a 9-bit FIFO interface to one of three temporary storage RAMs. Special features include an automatic header data capture mechanism and an input timeout generator.
 - (2) Header Controller (second stage): extracts and examines certain fields within a data unit from one of the temporary storage RAMs, and moves these fields to a 9-bit FIFO interface. There are three Header Controller modes: normal extraction, automatic deinterleaving extraction, and automatic CCSDS packet header extraction.
 - (3) Output Controller (third stage): moves data units from one of three temporary storage RAMS to another 9-bit FIFO interface. The Output Controller can also be used to transfer data units from the output FIFO interface to the Enhanced RAM Controller Chip.

The Tribuffer Controller waits for the Header Processor to issue a Cycle command, indicating that the Header Processor has finished processing data, and tells the Tribuffer Controller to start the next cycle. The Tribuffer Controller ASIC then issues a cycle interrupt to all three CPUs.

- d. Reassembly RAM subsystem accepts data from the Tribuffer subsystem, stores it, and outputs it from the EOS Service Processor Card. Data output, controlled by the Output Processor, can include fill and/or annotation. The output data destination can be the HRTB, a loopback FIFO, and/or VSB DMA circuitry. This subsystem consists of a 4-Mbyte RAM buffer and an Enhanced RAM Controller. The 4-Mbyte buffer is logically partitioned into thirty-two 128-Kbyte partitions that allow separate storage of frames from 32 different VCs. The Output Processor controls data output from the reassembly RAM subsystem, which is the EOS Service Processor Card output. To provide control, the Output Processor writes instructions to an 8-deep instruction FIFO. The RAM Controller automatically reads from the instruction FIFO whenever the FIFO contains an instruction.



The RAM Controller subsystem has three output modes: data, 8-bit fill, and 16-bit fill. Output mode is determined by instructions from the Output Processor. Each output mode is defined as follows:

- (1) Data mode reads and outputs data streams, which are identified by the start and end RAM addresses. This provides the ability to output reassembled packets from frame data fields.
 - (2) Fill modes provide the capability to insert fill annotation patterns of any length in the output data stream.
 - (3) The order of data segments read and fill annotation location are determined by the order of instructions passed by the Output Processor.
- e. VSB interface subsystem provides a VSB interface for the Output Processor and a VSB DMA interface for reassembly RAM subsystem output. The subsystem uses a Motorola MVSB2400 to provide a master A32/D32 interface to the VSB. For DMA of reassembly RAM subsystem output data, a 4K x 32-bit FIFO buffers data and converts it to 32-bit data. The FIFO is logically partitioned into four sections. Each 8-bit data output is stored in a partition; when all partitions are nonempty, they are simultaneously transferred to the MVSB2400 chip for DMA. A state machine Programmable Array Logic (PAL) provides control logic for FIFO I/O and MVSB2400 transfers. An address counter maintains the 32-bit address of data that is DMA-transferred. When the 9th bit mark indicates End-of-Data (EOD), the MVSB2400 chip is informed that DMA for that data unit is complete. DMA transfers that cross a 64-Kbyte/8-Mbyte longword boundary must be split. This boundary selection is specified by a control bit from the Output Processor. In 64-Kbyte boundary mode, address bits <31..18> are not incremented by the counter; only bits <17..2>. In 8-Mbyte boundary mode, address bits <31..26> are held constant while address bits <25..2> increment. Address bits <1..0> are tied low in either mode because all transfers are on a quad-byte boundary.
- f. Loopback FIFO, 16K x 9-bit, is connected to RAM Controller output; or, in test mode, to the Output Processor. This FIFO is typically used for card self-testing. To check a selected output, the Output Processor resets the loopback FIFO to clear previous data, loads an instruction set into the RAM Controller instruction FIFO, and repetitively reads captured data from the loopback FIFO. Only the first 16 Kbytes are captured; all subsequent bytes are dropped. Alternatively, the Output Processor can source data directly into this FIFO and read it back to verify correct operation.
- g. Pipeline output interface outputs parallel data to the HRTB via the third (J3) bus connector. The typical data output is CCSDS-defined source packets with annotation. Output from the RAM Controller subsystem is 9 bits with a write strobe. The 9 bits are defined as follows: 8 bits data and 1-bit EOF mark (9th bit). The 9th bit is used to signal the end of the data unit. On the last byte of data and annotation, the 9th bit is set to 1; otherwise, it is always 0. The HRTB output logic stacks the asynchronous 9-bit output from the RAM Controller into synchronous 18-bit data for transfer across the HRTB. The HRTB output logic, including the output interface, can be reset in one of three ways: manual card reset via front-panel reset switch, system reset from the VME master, or software-generated pipeline reset via Header Processor.
- h. VMEbus interface transfers setup data and commands to the Quality Processor, which then transfers information to the Header and Output Processors. The Quality Processor subsystem communicates with the VMEbus as a master or a slave via the VME interface. A Motorola MVME6000 provides the master/slave interface to the VMEbus. PLDs allow the logic to arbitrate the Quality Processor's local bus, which allows the VMEbus master to

access the local SRAM of the Quality Processor. When a master accesses the address defined by the EOS Service Processor Card rotary switches, the card responds as a VMEbus slave. The only available memory on the EOS Service Processor Card that a master may address is the Quality Processor memory. The VMEbus interface can be summarized as follows:

- (1) Master A32/A24/A16/D32/D16/D8 interface.
- (2) Slave A32/A24/D32/D16/D8 interface.
- (3) Includes all VMEbus requests; levels 0 through 3.
- (4) Can serve as VMEbus controller (i.e., a slot one arbiter).

- i. Quality Processor subsystem processes quality information and communicates with the VMEbus interface. It also includes a serial Universal Asynchronous Receiver-Transmitter (UART) interface that provides direct terminal interaction for software development and debugging. The Quality Processor sets up the Header and Output Processors by accessing their local memory. It also communicates with the Header and Output Processors through DPR, and may send interrupts to each. This processor is dedicated to the analysis of extracted data for quality evaluation; it performs several typical functions:

- (1) Checks quality.
- (2) Generates a piece assembly list.
- (3) Generates quality/production accounting statistics.

The Quality Processor receives data from the Header Processor. It can be programmed to recognize the availability of new data either on the Tribuffer Controller cycle interrupt, or by monitoring an index defined in the Header/Quality-shared DPR. The Quality Processor derives output assembly information from the data it receives via the Header Processor, and passes the information to the Output Processor. On reset, the Quality Processor boots up from its Erasable Programmable Read-only Memory (EPROM), which is configured to user requirements. Bootup procedures include initialization of the VMEbus interface system. An external master card can access all RAM of the Quality Processor and take control of its address and data buses via the VMEbus. Using this capability, an external master card can directly load application software into the general purpose RAM of the Quality Processor. In addition, the Quality Processor can directly load application software into its RAM without intervention from an external master card.

- j. Header Processor subsystem controls the Tribuffer RAM subsystem. It also communicates with the Quality and Output Processors through DPR and interrupts. For testing purposes, the Header Processor writes data to the input FIFO, where data is processed and checked in the loopback FIFO. The Header Processor subsystem includes a serial UART interface, which provides direct terminal interaction for software development and debugging. The Header Processor reads and processes data stored in the Tribuffer subsystem RAM. Typically, it reads and processes frame or CVCDU headers and packet headers. It is the only processor with direct access to input data, and it must provide any data-dependent information required by the Quality and Output Processor functions:

- (1) The Output Processor requires location information for pieces to be reassembled for output. An indexed listing is passed from the Header Processor to the Output Processor via DPR. For example, the Header Processor may provide start/end locations of packet pieces within each frame.

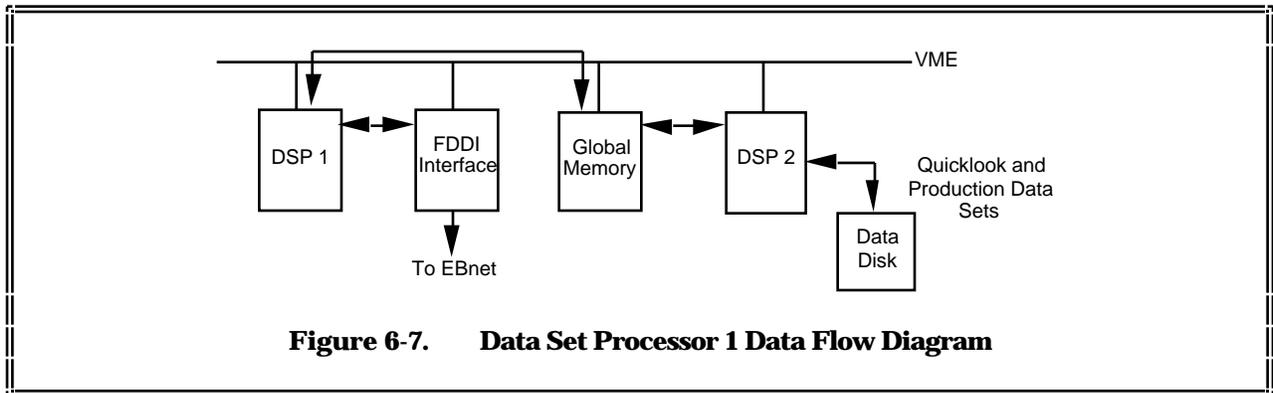
- (2) The Quality Processor also receives an indexed listing of information from the Header Processor via DPR. Typically, the Quality Processor requires header-derived information that describes frames and packet pieces.

The Header Processor software should include a loop initiated by the Tribuffer Controller cycle interrupt, and all processing of data extracted from a frame should be complete before signaling the Enhanced Tribuffer Controller that header reading is complete. On reset, the Header Processor boots up from its EPROM, which is configured to user requirements.

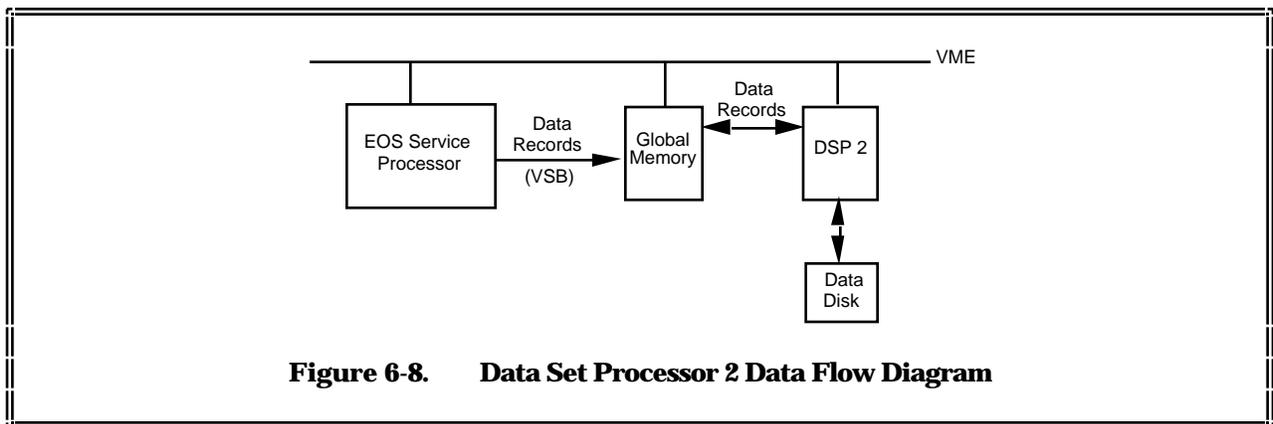
- k. Output Processor subsystem controls the reassembly RAM subsystem and tracks the location of frames and pieces of frames stored in the reassembly RAM. It also communicates with the Quality and Header Processors through DPR and interrupts. It includes a serial UART interface that provides direct terminal interaction for software development and debugging. The Output Processor receives an indexed list of data field piece locations within each frame from the Header Processor via shared DPR. The availability of new data can be signaled by the Tribuffer Controller cycle interrupt, or by a message field defined in DPR. Based on the Header Processor data list, the Output Processor maintains a frame and piece directory. Using this and the assembly list from the Quality Processor, it generates instructions and defines the format for data output. Instructions are written to the instruction FIFO. A set of output instructions is generated whenever assembly data is available from the Quality Processor. The Output Processor can write directly to the HRTB or the VSB. The Output Processor controls the HRTB setup registers for I/O. When reset, the Output Processor boots up from its EPROM, which is configured to user requirements.
- l. DPR subsystem: the EOS Service Processor's three microprocessors are configured to be programmed and operated independently. Asynchronous communication between processors is provided by direct interrupts, or DPR. The Header/Quality DPR is an 8K x 32-bit DPR (32 Kbytes total). It provides the primary communication between the Header and Quality Processor subsystems while the EOS Service Processor Card is running. The Header Processor passes Tribuffer subsystem RAM information to the Quality Processor via this DPR. The Quality Processor passes data, status, and commands to the Header Processor via the same DPR. The Header/Output DPR is an 8K x 32-bit DPR for (32 Kbytes total). It provides the primary communication between the Header and Output Processor subsystems while the EOS Service Processor Card is running. The Header Processor passes Tribuffer subsystem RAM information to the Output Processor via this DPR. The Output Processor sends and receives data to/from the Header Processor via the same DPR. The Quality/Output DPR is an 8K x 32-bit DPR (32 Kbytes total). It provides the primary communication between the Quality and Output Processor subsystems while the EOS Service Processor Card is running. The Quality Processor passes data, status, and commands to the Output Processor via the same DPR; the Output Processor sends and receives data to/from the Quality Processor via the same DPR.

6.1.5 DATA SET PROCESSORS 1 AND 2

Data Set Processor 1 is a 6U MVME167 VMEbus A32 card with a 68040 CPU. Essentially, this card is used to initiate the FTP transfer of data sets to and from the data disk via the FDDI Interface and EBnet. The processor initiates one block transfer of data from the global memory to local Data Set Processor 1 memory from where the data transferred via the FDDI Interface Card to EBnet. Figure 6-7 illustrates the data flow to and from Data Set Processor 1.

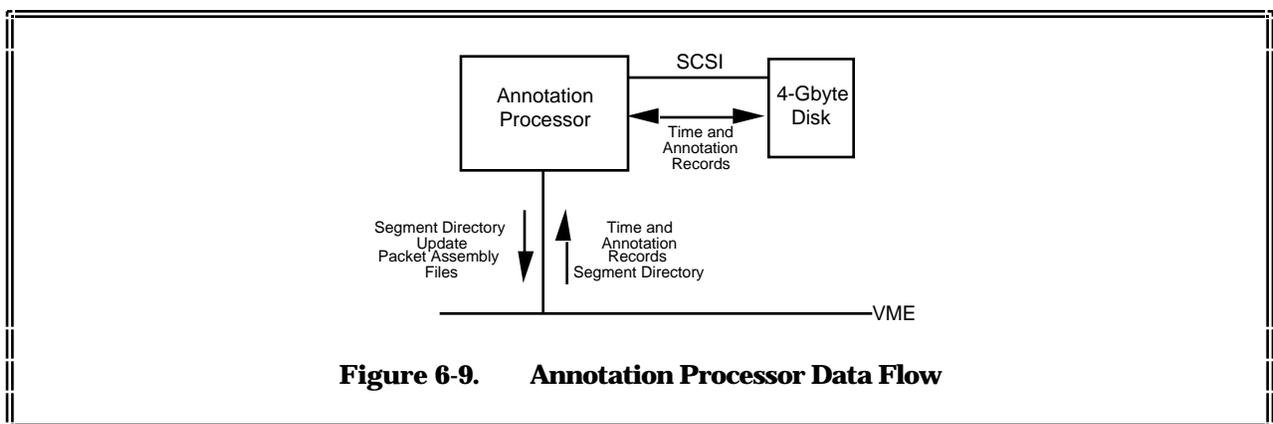


Data Set Processor 2 is a 6U Nitro60 card that acts as the interface between the data disk and VME. Data Set Processor 2 retrieves data records from the Global Memory Card and stores them on the data disk and vice-versa. Figure 6-8 illustrates the data flow to and from Data Set Processor 2.



6.1.6 ANNOTATION PROCESSOR

The Annotation Processor is a 6U MVME167 VMEbus A32 card with a 68040 CPU. Essentially, the Annotation Processor CPU performs the memory intensive operations to generate that DSAT and PAT files needed to create all the data sets. Figure 6-9 illustrates Annotation Processor data flow.



6.1.7 HIGH-RATE TELEMETRY BACKPLANE

The HRTB (Figure 6-10) supports the following functions in the VHS:

- Telemetry transfer of data between custom cards in the system at rates up to 150 Mbps. It can support transfers at rates up to 300 Mbps.
- Ability to dynamically reconfigure connections to meet the needs of different system interconnection topologies.
- Ability to support “upstream” flow control. The receiving card can signal the sender to hold transfers until ready.

The HRTB implements the following interconnection features:

- Six to twenty data paths per system.
- Nine-link, high-speed serial daisy chain (not used in VHS).
- Ten-card Futurebus-style connectors.
- Eighteen repeater chips to reduce path physical lengths.
- Thirty-seven interconnection configuration switches.
- Onboard three-chip transfer clock generation.
- Allows input for external transfer clock.
- Includes ten-chip “serpentine” test logic.
- Supports up to twelve 6-Amp power taps.

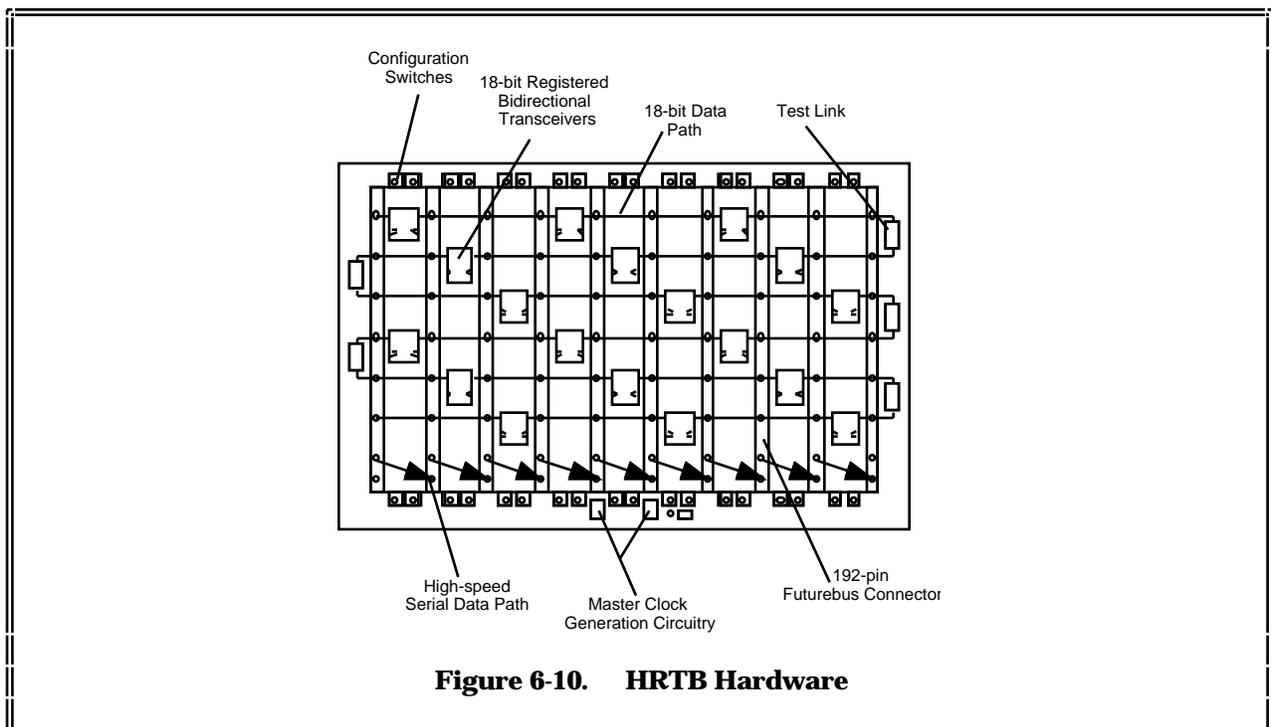


Figure 6-10. HRTB Hardware

HRTB design is divided into the following sections:

- a. Six parallel data channels, each of which can be dynamically subdivided into up to four segments per channel. This provides 6 to 20 data paths per system. Configuration of connections is software-controllable from each card.

A data channel consists of 18 bits per channel: 16 bits of data and 2-bit encoding of an EOF bit and a data valid bit to handle odd-byte length transfers. Each channel can support a 20+ MHz transfer and clocking rate. When transferring 16-bit words, this amounts to 300+ Mbps. Data transfers are performed synchronously using a “backplane” clock. A 20-MHz onboard oscillator is provided, but an external clock may be used. The clock is distributed using two clock distribution Integrated Circuits (IC). There is one net routed to each card, and one net to each pair of transceivers. Each line is independently terminated using a resistor/capacitor termination.

The interconnection topology of the six parallel channels may be configured using the 37 configuration switches on the backplane. These switches provide a “default” configuration during system powerup. All switches are readable by all cards connected to the HRTB. If a connected card has the appropriate driver hardware, its software may override its switch configuration dynamically. There are 18 repeater chips (FCT162501) spaced at a maximum of three slots to reduce path physical lengths. Each of these chips has two surface-mount control switches visible and changeable from the front of the backplane.

Each card connected to the HRTB has a 4-bit slot ID value (1 - 10) that tells each card its position in the HRTB.

- b. Nine-link, high-speed serial daisy chain allows a differential serial input from the “upstream” card and an output to the “downstream” card. This function is not used in the VHS.
- c. Power pins: four VCCs and 12 GNDs. Each pin can handle 1 Amp. This is in addition to the VMEbus J1 connector that has three VCCs and eight GNDs, and the VMEbus J2 that has three VCCs and thirteen GNDs. There are also four groups of “other” voltage pins allocated. Each group consists of six pins (6 Amps). Each card is connected to two groups: cards 1-5 are connected to groups one and two; cards 6-10 are connected to groups three and four. The four groups can be mixed to provide one to four other voltages. Two of these groups are used in the VHS to carry 100-Kbyte ECL power supplies (-2 V, -5 V) to the EOS Simulator and EOS Frame Synchronizer Cards. Unused groups may be redefined for other use.

6.2 **COMMERCIAL COMPONENTS**

Commercial components refer to those functional elements, modules, and subsystems that are supplied by vendors external to NASA/GSFC, Code 521. In the VHS, commercial cards provide for the exchange and storage of data, control, status, and quality information. They provide processing, storage, and I/O functions required by the overall VHS application.

6.2.1 **SYSTEM DISK AND INTERFACE MODULE**

The System Disk and Interface Module (4-Gbyte hard disk) provides storage for system configuration files and the software modules to be downloaded to each functional element.

6.2.2 MASTER CONTROLLER CARD

The MCC is a 33-MHz, 68040-based Motorola MVME-167SA-2 CPU board. The MVME-167-032 is a 6U card with 32-bit access through both VME connectors (P1 and P2). The board has several built-in functions including 8 Mbytes of onboard shared parity-protected Dynamic Random Access Memory (DRAM), four serial ports, parallel port, SCSI bus controller, tick timer, watchdog timer, time-of-day clock/calendar, and A32:D32 VMEbus interface with system controller functions. The serial ports provide a user interface specific to this card. Onboard functions include: 68882 floating point coprocessor, Ethernet interface controller, and SCSI bus interface with DMA channel. The card requires power supplies of +5 V @ 7 Amps maximum, +12 V @ 1 Amp maximum (100 mA maximum with no Ethernet LAN), and -12 V @ 100 mA maximum. A detailed description of the MVME-167 is provided in Motorola's MVME167S MPU VME Module User's Manual.

The Ethernet interface controls and moves data and command traffic over Ethernet via TCP/IP protocols; it interfaces the data flow between the VHS and workstation(s). The Ethernet interface also controls and accounts for data flowing through the network to/from the VHS. It includes a 10-MHz Motorola 68010 microprocessor that autonomously handles the transmission and receipt of TCP/IP packets from the network.

6.2.3 SYSTEM MEMORY CARDS

Three system memory cards are included in the VHS; each card is a 6U MicroMemory MM6740CN high-speed, 16-Mbyte SRAM memory board. The MM6740CN transfers 32-, 16-, or 8-bit data over the VME, and may be addressed using 32- or 24-bit VME address paths. It can support VME block transfers. The board generates and stores a parity bit for each byte memory location during write cycles and checks it during read cycles. A status bit is set to indicate parity errors. It provides for the temporary logical buffering of data. The card requires a power supply of +5 V @ 7.75 Amps maximum. A detailed description of the MM6740CN is provided in Micro Memory's MM6740CN User's Guide.

6.2.4 GLOBAL MEMORY CARD

The Global Memory Card is a 6U MicroMemory MM6390D high-speed, 512-Mbyte DRAM memory board. The MM6390D transfers 32-, 16-, or 8-bit data over the VME and may be addressed using 32- or 24-bit VME address paths. It can support VME block transfers. The board generates and stores a parity bit for each byte memory location during write cycles and checks it during read cycles. A status bit is set to indicate parity errors. It provides for the temporary logical buffering of data. The card requires a power supply of +5 V @ 7.75 Amps maximum. A detailed description of the MM6390D is provided in MicroMemory's MM6390D User's Guide.

SECTION 7

DETAILED SOFTWARE DESIGN

This section overviews the ETS HRS application-specific software. The application-specific code for the VHS is developed in a workstation environment using the Wind River VxWorks and Code 521-developed Modular Environment for Data Systems (MEDS) platform. The application-specific code for the TRS is developed on a UNIX platform.

7.1 VME HIGH-RATE SUBSYSTEM

The VHS software architecture will be based on VxWorks and MEDS (see Figure 7-1). VxWorks is a COTS high-performance, real-time operating system and a powerful software development environment. The MEDS software package is built on top of VxWorks and standard UNIX communication utilities. It provides two system functions fundamental to multiprocessing telemetry data systems: configuration control and status reporting.

7.1.1 VXWORKS ENVIRONMENT

VxWorks includes a fast run-time system, testing and debugging facilities, and a UNIX cross-development package with extensive UNIX-compatible networking facilities. The networking facilities allow VxWorks and UNIX to combine to form a complete, integrated development and operational environment. The UNIX system is used for software development and the nonreal-time components of applications. VxWorks is used for testing, debugging, and running real-time applications.

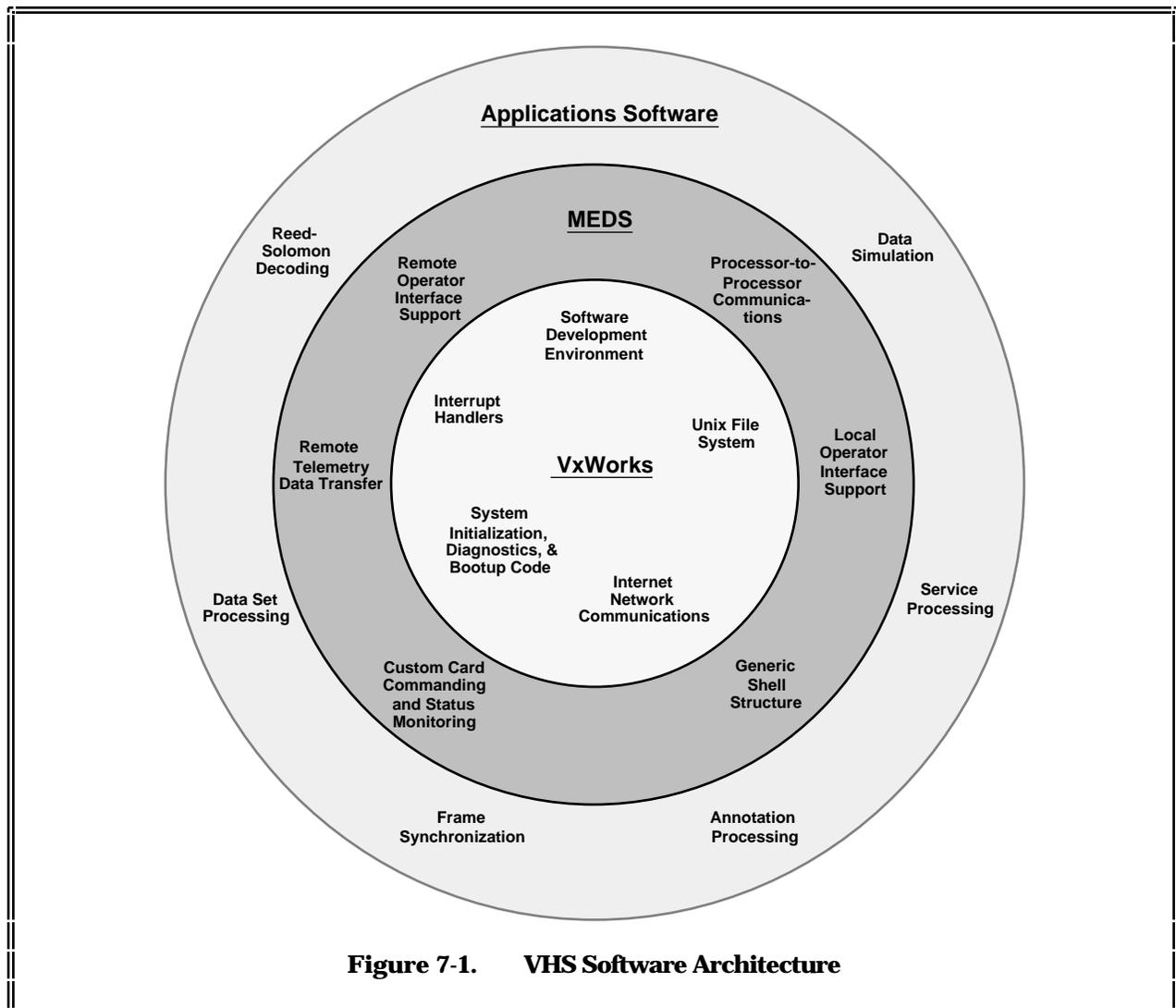
The ETS HRS uses VxWorks, which implements high-performance, real-time kernel software development is based on UNIX workstations using Wind River, GNU and Greenhills development tools. VxWorks provides extensive UNIX-compatible networking utilities that allow seamless integration of VxWorks and UNIX.

VxWorks consists of three integrated components: a set of powerful cross-development tools that run on a host development system; a high-performance, scalable, real-time operating system that executes on a target processor; and a variety of communications options to connect the target and the host. Across these components, VxWorks adheres to industry standards, including POSIX 1003.1b real-time extensions, ANSI C, and TCP/IP networking.

Host tools consist of the Green Hills Multisystem and standard GNU tools. Multisystem provides a complete cross-compiler and debug environment in a single tool. In addition, the standard host development tools included with VxWorks consist of a GNU cross-compiler toolkit and the VxGDB source debugger.

Target tools include extensive system diagnostics and performance monitoring, application prototyping facilities, comprehensive networking facilities, POSIX 1003.1b compliance, and dynamic linking and loading for rapid edit-test-debug cycles or dynamic run-time flexibility.

Also, the use of additional operating system accessories and productivity-enhancing tools can result in dramatically shorter time-to-market for embedded applications. In addition to the supported targets, VxWorks provides easy porting to custom hardware. Unique to VxWorks is a powerful interactive shell interface that allows users to interact with all VxWorks facilities. Unlike other shells, the VxWorks shell provides one simple, powerful capability: it can interpret and execute almost all C-language expressions, including calls to functions and references to variables whose names are found in the system symbol table.



7.1.2 MEDS ENVIRONMENT

In the VHS approximately 30 application tasks will run from 13 microprocessors. They will communicate with a UNIX host and among themselves through standard socket connections. Higher-level communication protocols such as FTP and Network File Server (NFS) are also supported. All application files are stored on the UNIX host, and are NFS-mounted on the subsystems for access and execution.

A MEDS system consists of one Master Controller subsystem and multiple slave subsystems, such as the High-rate Frame Synchronizer, Reed-Solomon Decoder, etc. The Master Controller subsystem, which runs on the Master Controller Card, is the primary pipeline through which all commands and status information pass. It receives operator commands from the CDS, parses commands, derives subcommands, and sends subcommands to appropriate subsystems. The Master Controller subsystem also periodically collects status from each subsystem, builds status messages, and sends back messages to the CDS. For development and contingencies, a local operator interface is also provided through an RS-232 port.

The Master Controller subsystem downloads tables of common information to each subsystem.

This information allows subsystems to communicate with each other without knowledge of their physical attributes.

The primary function of each slave subsystem is to process and/or transfer data. Each subsystem has a command handler task to manage MEDS bootup, initialize the subsystem after bootup, and process commands. Most subsystems also run other application tasks to control telemetry processing hardware and perform telemetry processing functions.

7.1.3 TELEMETRY PROCESSING CONTROL ENVIRONMENT

The CDS provides mechanisms for controlling and monitoring operation of the ETS HRS. The CDS automates operation of the ETS HRS based on a schedule of telemetry processing activities and user-initiated batch processing sessions. The operational schedule is a week-long schedule that identifies spacecraft pass events, including groundstation for downlink, data rate for downlink, and start/stop times for each event. The CDS displays this activity schedule to the user, and allows the user to add/delete session events and initiate telemetry processing of data contained on tapes received from groundstations.

The CDS is designed as a distributed, multi-process system: a number of simultaneously executing processes cooperate to control and monitor ETS HRS operation. The subsystem may operate in a multi-host mode, wherein a specific process may be allocated to one of a number of host workstations, or on a single host workstation.

Processes are classified into two categories based on the temporal nature of their operations: static or transient. Static processes form the core of the CDS, and are required to execute continuously, throughout operation of the system. Termination of any static process is considered a failure of the CDS. Transient processes are initiated and terminated upon user request. For example, most processes that drive status and monitoring displays are transient.

The internal structure of almost every CDS process is virtually the same: a series of C++ class objects that are initialized by the process and then invoked through system event function callbacks. When a system event occurs—for example, user input, clock timeout, IPC communication, or file input—an event is generated, causing a specific, predefined member function of some class to be called. This event-based callback structure is supported generically by a set of C++ classes that provide callback capabilities for specific types of system events. Examples include:

- a. WrEvent: provides callbacks based on user input to CDS displays.
- b. Alarm Clock: provides callbacks based on wall clock timers.
- c. IPC Server: provides inter-process communication callbacks.
- d. Data Server: provides status and monitoring data to registering objects on a regular basis through callbacks.

The CDS automatically sets up the VHS for telemetry processing based on scheduled processing sessions of captured telemetry. The CDS displays the current status of the ETS HRS during processing. These displays present frame synchronization and packet reassembly data quality/accounting information for ongoing telemetry processing sessions; summary data quality/accounting information for processed sessions; and status of ETS HRS hardware and software subsystems. The CDS also automatically generates reports summarizing data distribution to user sites and quality/accounting information for completed telemetry processing sessions.

A graphic UNIX-based computer workstation is configured to support the control, status monitoring, and data distribution operations of ETS HRS. Operating system processes on this workstation use the inter-process communications facilities of UNIX, including sockets, pipes, files, and shared memory.

Inter-workstation communications are supported via TCP/IP, providing reliable in-sequence delivery of data required for host-to-host communication. The standard FTP is used for distribution of telemetry data sets and Customer Operations Data Accounting (CODA) to the EOS Operations Center (EOC). FTP provides reliable, guaranteed-complete delivery of data files from host-to-host.

All CDS displays use the X-Windows system and the OSF Motif tool kit.

7.1.4 MASTER CONTROLLER SOFTWARE

The MCC software supports the following functions:

- a. Maintenance, formatting, and transfer of telemetry processing and system status information to the user workstation via Ethernet.
- b. Translation of user commands for the VHS system into lower-level card commands.
- c. Transfer of error tag records to the user workstation for Error Tag Record mode of operation.
- d. Transfer of buffers to the user workstation for Erroneous Frame Capture mode of operation.

The Master Controller software consists of the following tasks:

- a. MCBOOT: performs system boot sequence during system reset. This process is executed once and then terminated. This task also identifies all application cards based on the system setup file and bus test.
- b. MCLOADER: monitors health of all application cards and responds to a single-card reboot.
- c. MCRCMD: receives commands from the user workstation and translates them into lower-level card commands. This task also returns card responses to the workstation.
- d. MCRSTS: gathers application card status and sends it to the user workstation.
- e. MCEVENT: delivers all unsolicited event messages to the workstation. It reads messages from a globally known mailbox and writes them to a TCP/IP socket.
- f. tmbcEvt: broadcasts event messages to any connected workstations. For Example:

```

MCEVENT >-----> tmbcEvt >..< / |----> Workstation #1
                                     \ |---->.....
                                     \ |----> Workstation #M

```

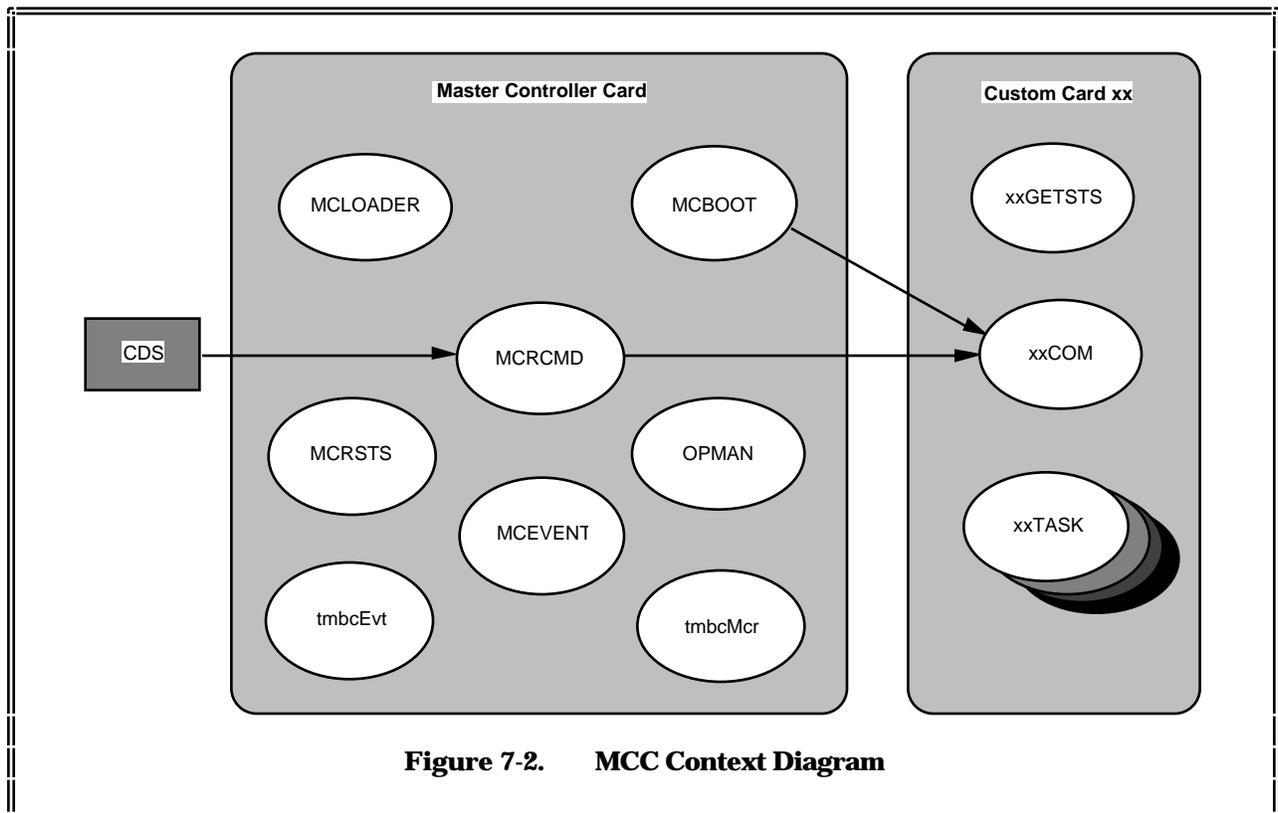
- g. **tmbcMcr**: allows multiple workstations to send commands to, and receive responses from, the MEDS command processor. For example:

```

MCR CMD >-----> tmbcMcr / |----> Workstation #1
                                >...< |---->.....
                                \ |----> Workstation #M
    
```

- h. **OPMAN**: operator interface that allows the user to control and monitor the system using a local terminal. It is also used to create and download catalogs to application cards.

Figure 7-2 provides the MCC context diagram.



7.1.5 DATA SET PROCESSOR SOFTWARE

Data Set Processor software is shared between two CPUs, and resides on Data Set Processor 1 (MVME-167) and Data Set Processor 2 (Nitro 60). DSP 2 is used for high-speed transfers via the SCSI interface to/from the disk farm (used interchangeably with the disk array). DSP 1 controls the FDDI interface to provide the high-speed transfer of data sets, EDSs, and PDSs to/from the disk farm (used interchangeably with the disk array). Figure 7-3 provides the Data Set Processor context diagram. Data Set Processor tasks are described as follows:

- a. **DSPCOM**: initializes Data Set Processor and starts other processes on the Data Set Processor subsystem. This task also receives commands from OPMAN and passes them to other processes.
- b. **DSPINPUT**: constructs data set instructions for DSPOUTPUT. It interacts with the Annotation Processor for data set instructions.

- c. **DSPOUTPUT**: constructs data sets from instructions received from DSPINPUT. This task delivers data sets to users over the FDDI WAN interface to EBnet. This task is used for data retrieval and storage from the disk farm.
- d. **DSPTRANSCIEVER**: receives incoming data sets for storage on the disk farm. This task also constructs a catalog of files stored on the disk farm, and sends out requested files from the disk farm.
- e. **RECOM**: initializes Data Set Processor subsystem and starts other processes.
- f. **REMAIN**: interacts with EOS Service Processor to receive and store packet data on the disk farm while cataloging the same.

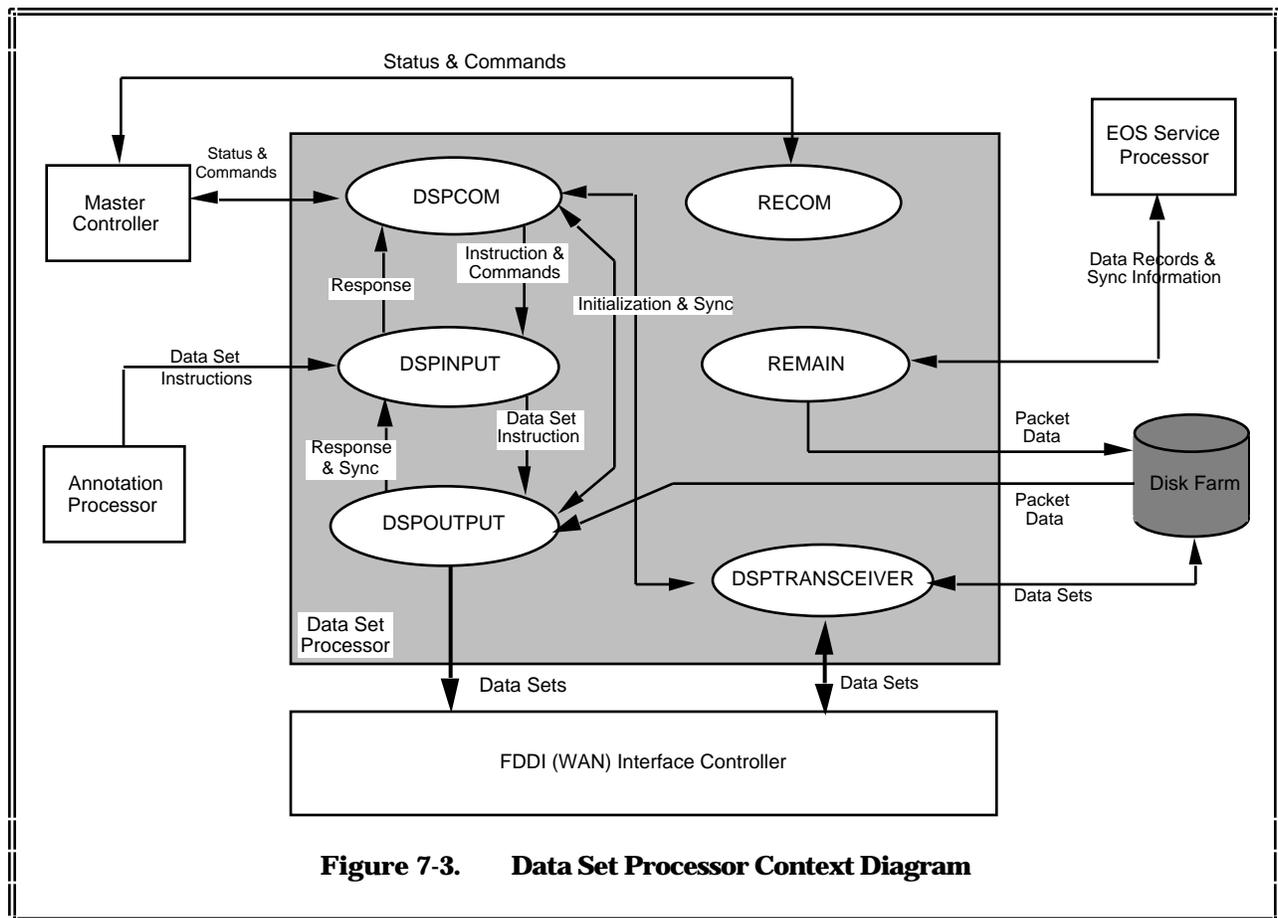


Figure 7-3. Data Set Processor Context Diagram

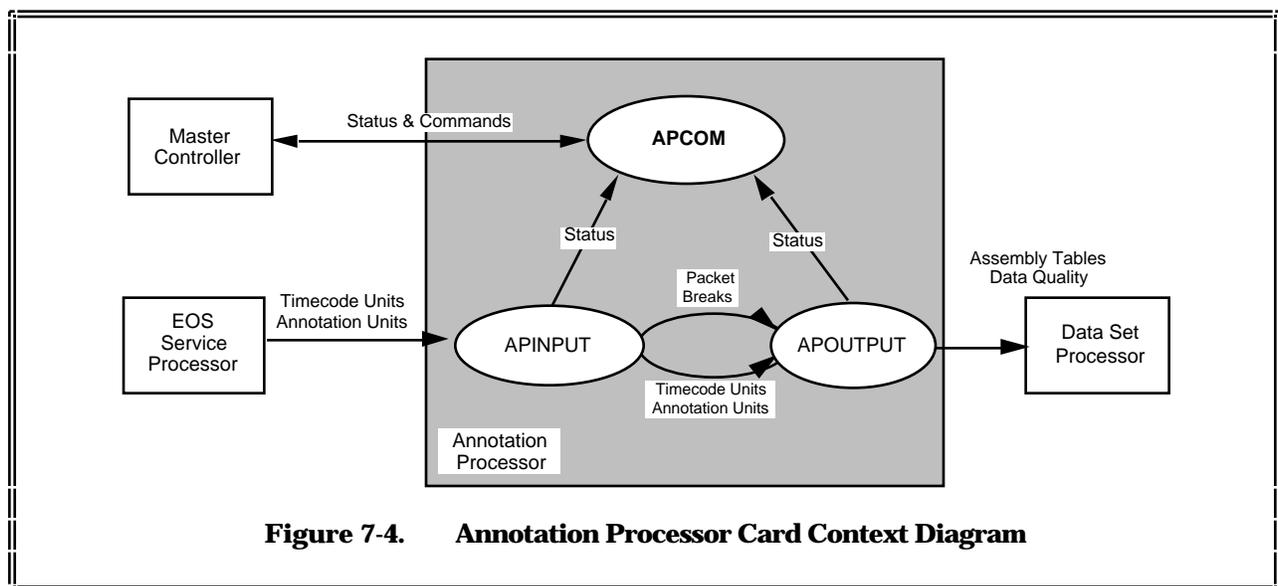
DSP 1 has a private connection process to the FDDI WAN that is not visible to other components in the VHS rack. This connection is used to distribute and receive data sets. These interfaces support TCP/IP transfer protocols.

7.1.6 ANNOTATION PROCESSOR SOFTWARE

The Annotation Processor is a COTS MVME-167 (33 MHz) with a 32-Mbyte onboard memory. Annotation Processor software is essentially data-driven and initiated when the EOS Service Processor software starts transferring timecode and error annotation information. Annotation Processor tasks are described as follows:

- a. APCOM: controls application tasks (APINPUT and APOUTPUT). It receives commands from, and returns status to, the Master Controller.
- b. APINPUT: receives timecode (for packets without errors) and annotation units (for packets with errors) from the EOS Service Processor and stores them on the 4-Gbyte SCSI (annotation) disk. It scans timecode units and records any breaks found in packet sequence counts.
- c. APOUTPUT: generates tables used by the Data Set Processor to assemble packets into data sets. It uses the break information collected by APINPUT to determine boundaries of segments of contiguous data. The task orders segments chronologically, merges segments, and discards redundant packets based on the quality of individual packets. Finally, it constructs assembly tables, which inform the Data Set Processor of the starting location and length of each data segment. Assembly tables are written to an assembly file, along with data quality information.

Figure 7-4 provides the Annotation Processor context diagram.



7.1.7 EOS SIMULATOR CARD SOFTWARE

The EOS Simulator Card outputs a single frame data stream. All data manipulation is handled by the card's hardware. Software initializes and controls the hardware, and keeps track of status counts. All software processes on the EOS Simulator Card reside on the 68040 CPU Mezzanine.

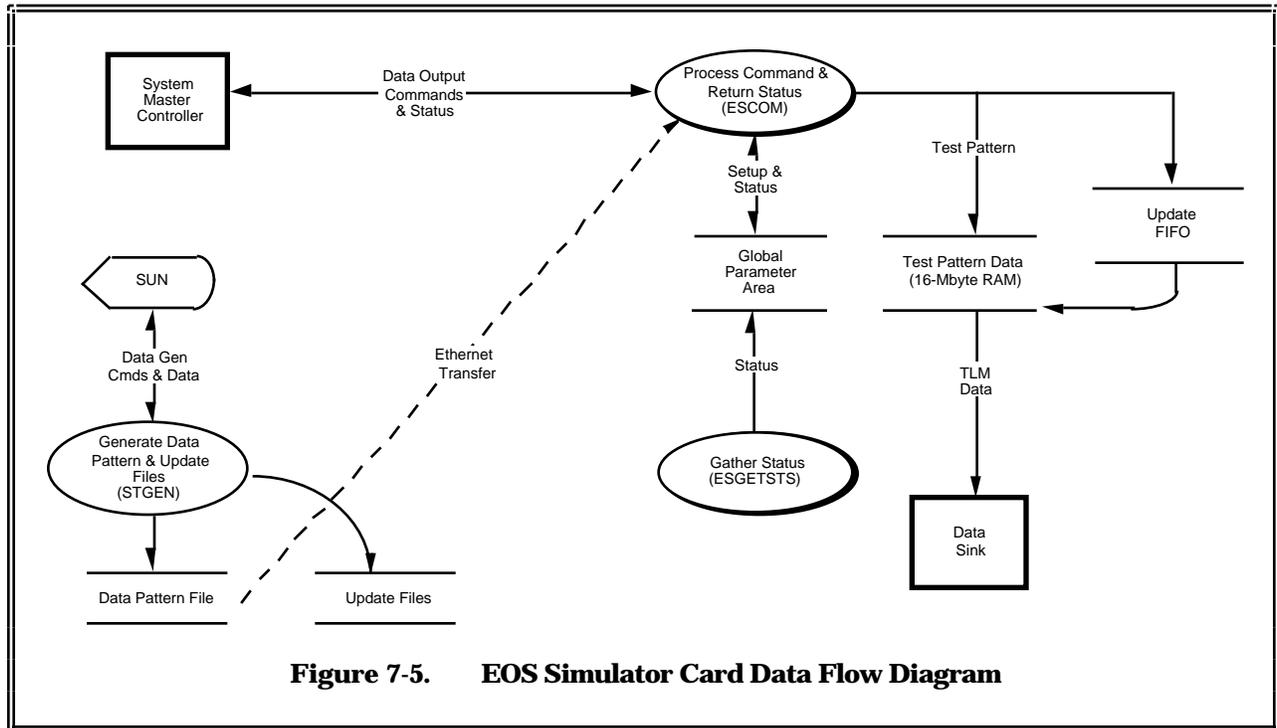
The EOS Simulator Card software runs in MEDS. Thus, it must accept and respond to operator commands sent by the MCC, and it must prepare and store status information for possible display.

The EOS Simulator Card software is divided into the following tasks:

- a. ESCOM (Command Handler): handles card initialization, setup, and self-test. During setup, the ESCOM loads frame data sets generated by SCTGEN into the 4-Mbyte RAM. This task establishes and maintains all communication with the Master Controller.

- b. ESGETSTS (Status Gathering): periodically reads hardware status registers to accumulate status on data flow through the card.
- c. Debug: used for self-test, development, and maintenance purposes.

Figure 7-5 provides the EOS Simulator Card software data flow diagram.



7.1.8 EOS FRAME SYNCHRONIZER CARD SOFTWARE

The EOS Frame Synchronizer Card software is executed by the MZ 8130 controller section. Figure 7-6 illustrates EOS Frame Synchronizer Card software data flow.

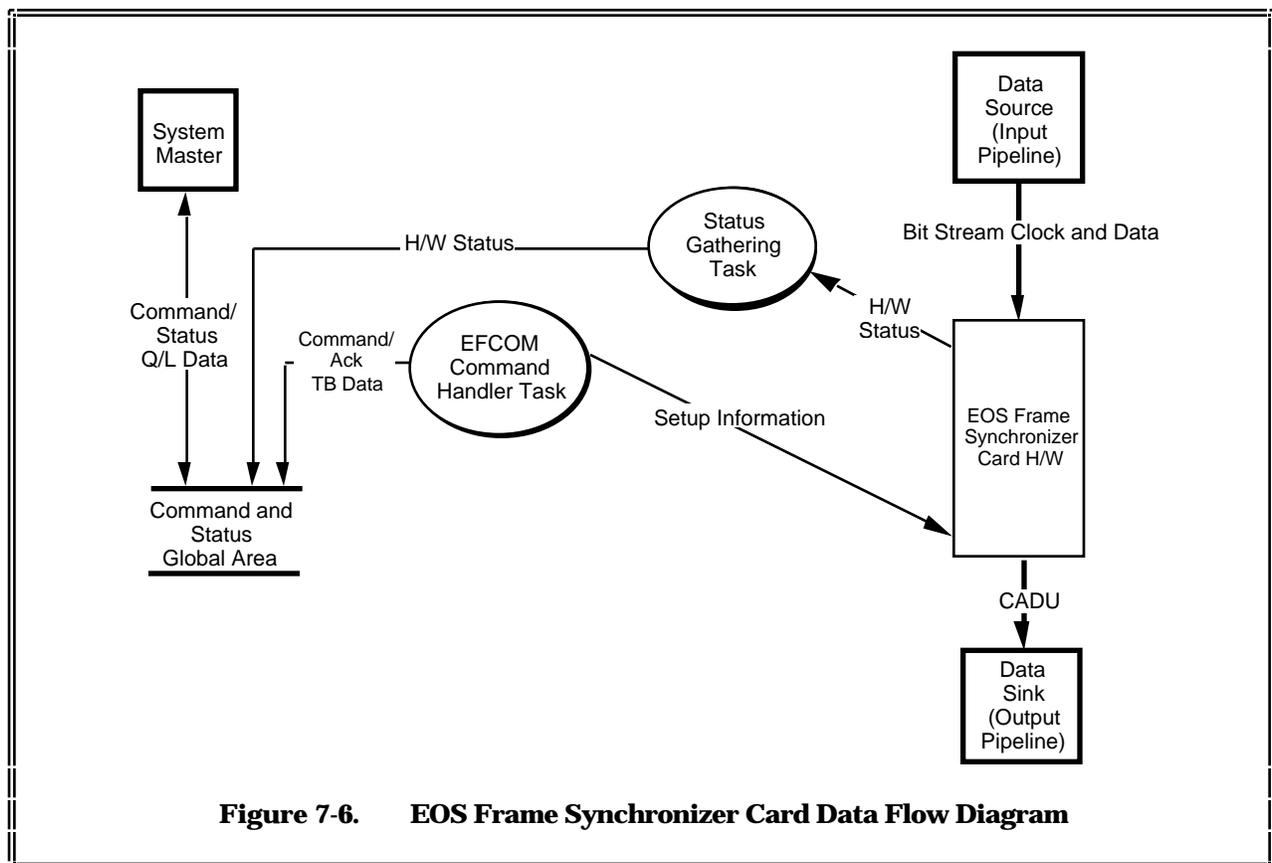


Figure 7-6. EOS Frame Synchronizer Card Data Flow Diagram

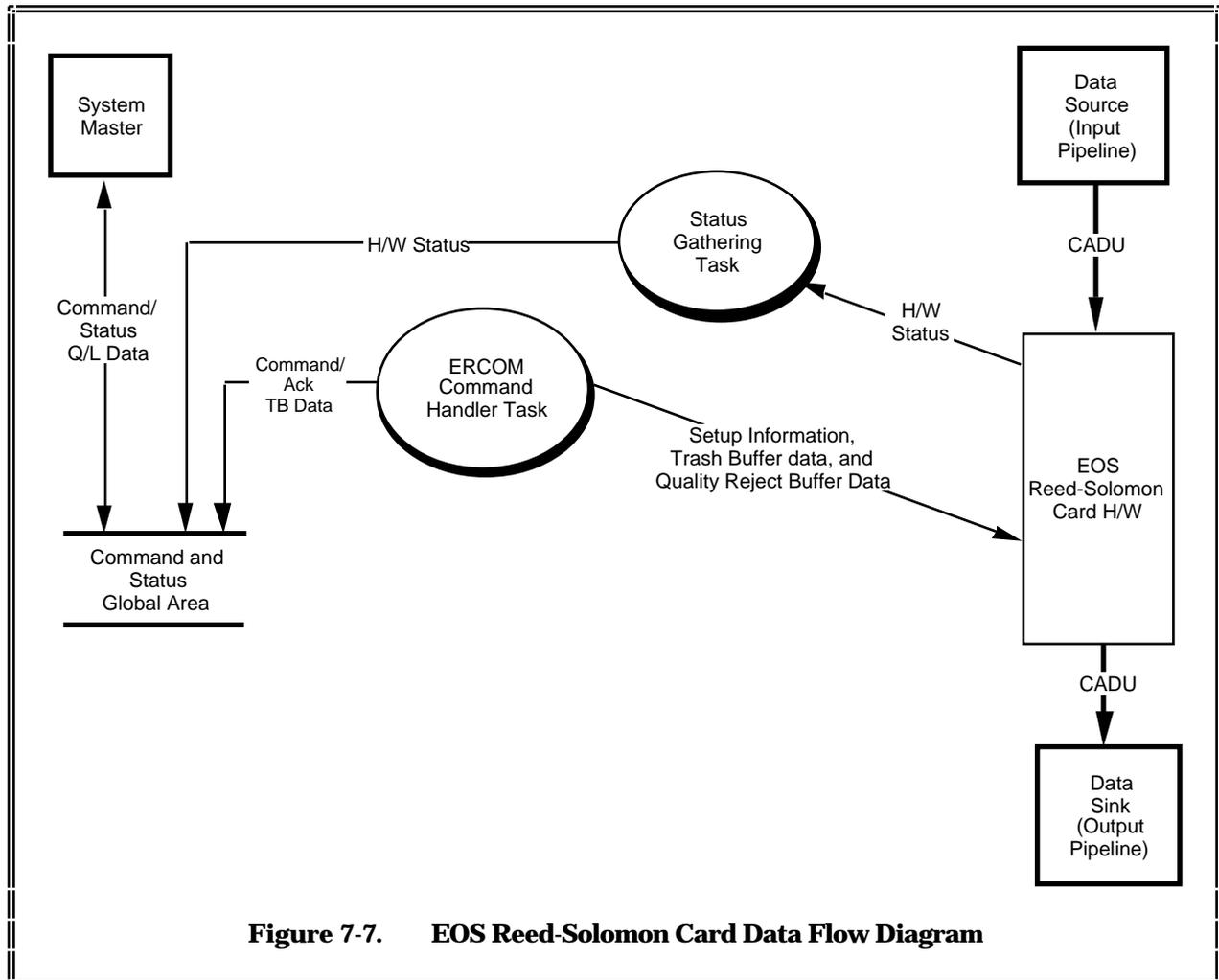
Software is divided into two main modules: EFCOM and EFGETSTS:

- a. EFCOM: handles card initialization, setup, and self-test, including setup of the HRTB I/O ports and timeout LEDs. It executes data transfer and configuration commands received from the system MCC. It uses a global parameter area of its memory to store card setup parameters received from the MCC via the card's command buffer, as well as status parameters from EFGETSTS to be made available to the MCC via the card's status buffer. EFCOM initializes the card's software with the following parameters:
 - (1) Frame size = 1024 bytes = 8192 bits.
 - (2) Bit slip detection and correction disabled.
 - (3) No frame sync pattern errors allowed (best-match strategy).
 - (4) Check mode enabled (programmable: 1-15 frames).
 - (5) Flywheel mode enabled (programmable: 0-15 frames).
 - (6) No CRC check or PN decoding.
 - (7) Polarity correction disabled.
 - (8) Reverse correction disabled.
 - (9) Append trailer containing quality information to each frame.
 - (10) Input from front-panel SMA connectors.
 - (11) True forward synchronization pattern (1ACFFC1D in hexadecimal).

- b. EFGETSTS: runs periodically (every second). It transfers hardware and data flow status to the card's software global parameter area, from where EFCOM copies it to the card's status buffer for transfer to the system MCC.

7.1.9 EOS REED-SOLOMON CARD SOFTWARE

The ERS Card software is executed by the MZ 8130 controller. Figure 7-7 illustrates EOS Reed-Solomon Card software data flow.



Software is divided into two main modules: ERCOM and ERGETSTS:

- a. ERCOM: handles card initialization, setup, and self-test, including setup of the HRTB input/output ports and the RSEC Chip. It executes data transfer and configuration commands received from the system MCC. It uses a global parameter area of its memory to store card setup parameters received from the MCC via the card's command buffer, as well as status parameters from ERGETSTS to be made available to the MCC via the card's status buffer. ERCOM handles card initialization and setup for:
- (1) Reed-Solomon (255, 223) decoding up to 150 Mbps.
 - (2) Reed-Solomon interleave of four (programmable between 1 to 16).

- (3) Enable bypass mode to perform Reed-Solomon error detection only (no correction).
 - (4) Enable trash buffer function to examine invalid VCID CADUs.
 - (5) Enable quality reject buffer function to examine CADUs that are unrouteable, or have Reed-Solomon errors.
 - (6) Acquiring CADU Reed-Solomon correction information in the quality reject buffer.
 - (7) CADU routing for up to four destinations using fully configurable routing table.
- b. ERGETSTS (Status Gathering): runs periodically (every second) and transfers hardware (including the RSEC Chip's periodic status output) and data flow status to the card's software global parameter area.

7.1.10 EOS SERVICE PROCESSOR CARD SOFTWARE

The EOS Service Processor Card software runs on three MC68040 microprocessors: Header, Quality, and Output. Each processor is linked to the others by DPR, has a separate set of internal buses, and executes an independent process. Each microprocessor also has an RS-232 port for debugging and testing through a local terminal. The combination of the software running on all three processors implements the following functions:

- a. A communications process that handles command and status transfers between the system MCC and the EOS Service Processor Card.
- b. Header, quality, and output telemetry processing functions, including:
 - (1) Receive annotated CADUs (at rates up to 50 Mbps).
 - (2) Perform all CCSDS path packet service processing functions.

Figure 7-8 tabulates the EOS Service Processor Card task allocation. The EOS Service Processor Card software can be summarized by five top-level processes: communication, simulation, header, quality, and output. The communication process runs on the Quality Processor and handles card initialization, setup, and self-test (including setup of I/O ports and status transfers between the system MCC and the EOS Service Processor Card via the card's status buffer). The remaining three telemetry data processes convert CADUs received by the EOS Service Processor Card into reassembled source packets and transfer selected packets to the user.

Task	CPU Mezzanine	Header Processor	Quality Processor	Output Processor
Input CCSDS transfer frames or CADUs		√		
Extract transfer frame headers and packet headers, including timecodes		√		√
Check for and report errors		√	√	√
Process forward or reverse sequence-ordered transfer frames or CADUs		√	√	
Interface EOS Service Processor Card with other cards in the system			√	
Accumulate processing status and report to the system master			√	
Assemble spacecraft time and sequence count of each packet into timecode records			√	
Detect and report missing packets			√	
Reassemble source packets				√
Output annotation records				√
Output packets				√

Figure 7-8. EOS Service Processor Card Task Allocation

7.1.10.1 Header Process

Figure 7-9 illustrates EOS Service Processor Card header process data flow. The CADUs received from the ERS Card first move into the Tribuffer Controller, which is managed by the Header Processor. The Header Processor has three data stream outputs:

- a. The Header Processor reads incoming CADUs and extracts frame headers and annotation trailers. Next, it validates frame headers and annotation trailers, and sends frame status information to the quality process via dual-ported buffer memory.
- b. The Header Processor extracts and validates packet headers from CADUs. It then generates a packet data piece list, which summarizes the location of each data piece in the CADU, and passes this list to the output process via dual-ported buffer memory. It also passes packet piece and packet length status to the quality process via dual-ported buffer memory.
- c. CADU data (stripped of previously appended trailers) is output to the RAM Controller, which is controlled by the output process.

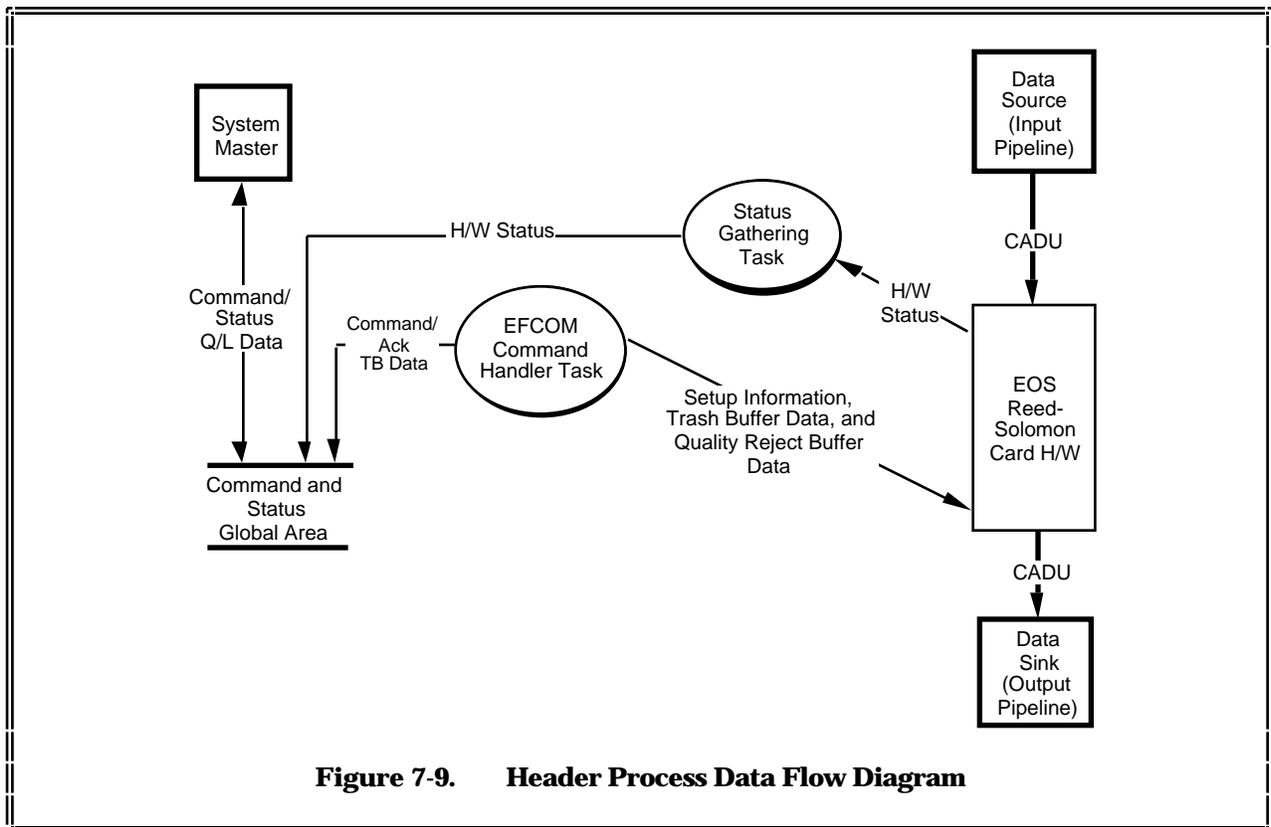
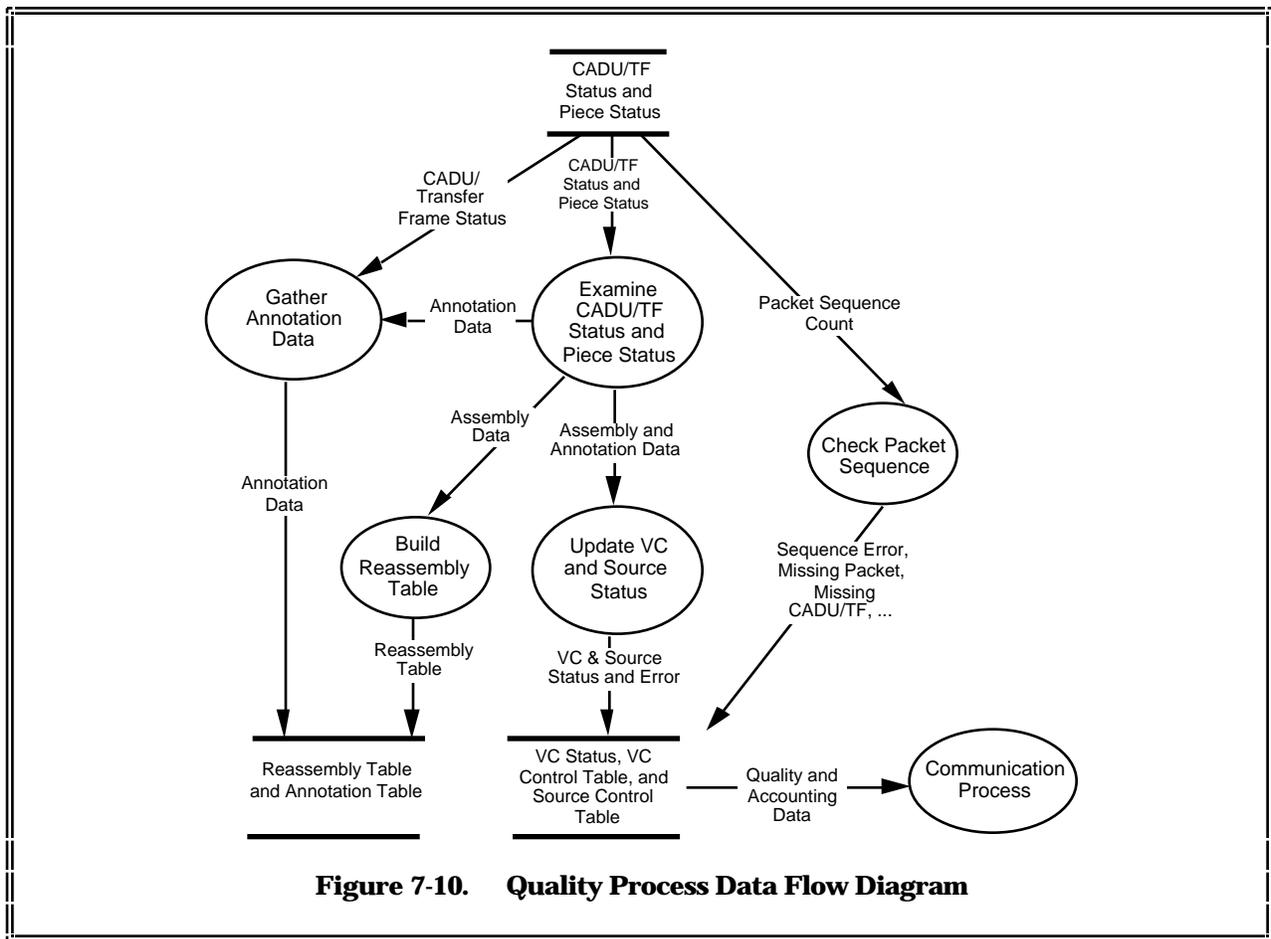


Figure 7-9. Header Process Data Flow Diagram

7.1.10.2 Quality Process

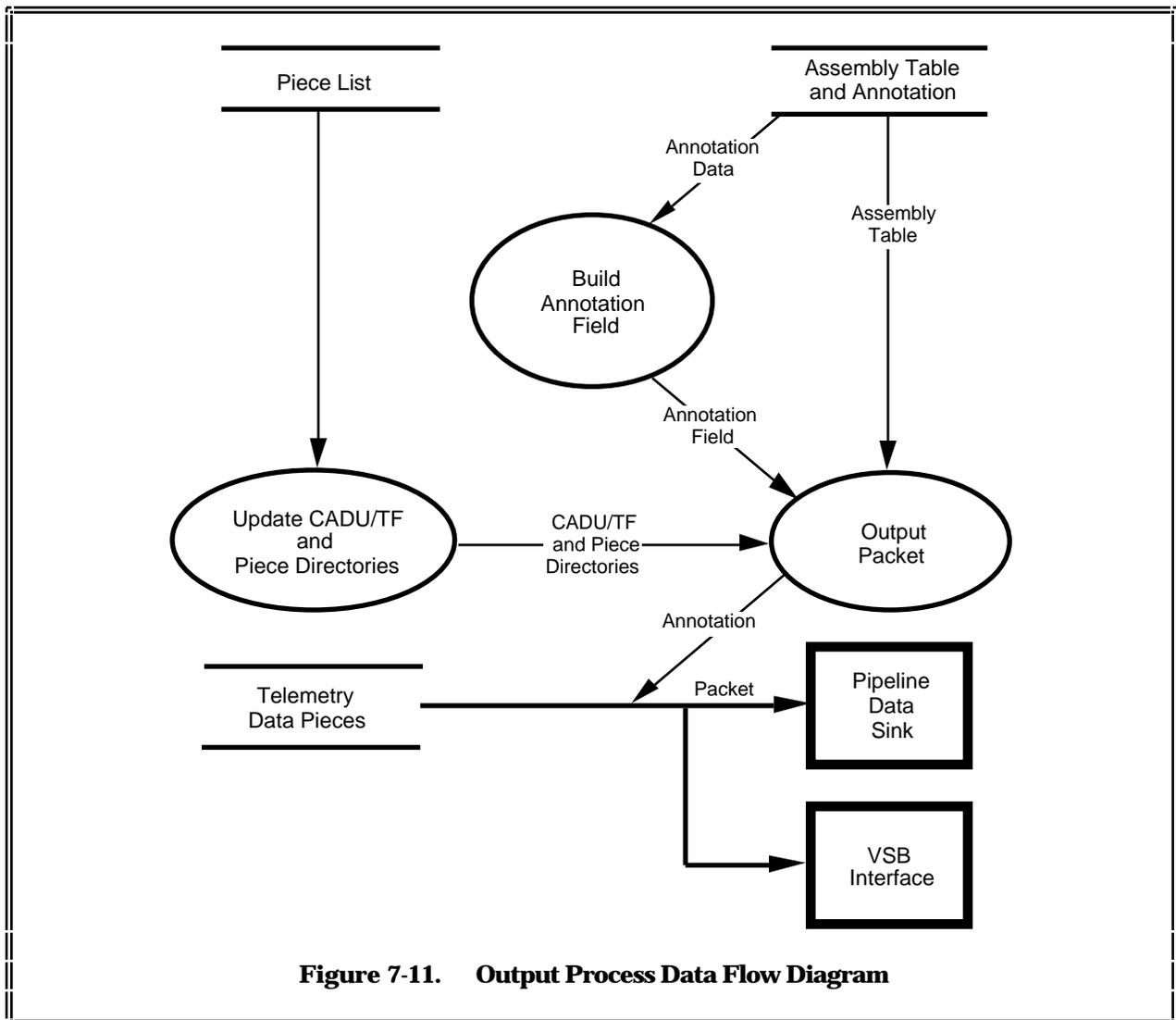
Figure 7-10 illustrates EOS Service Processor Card quality process data flow. Once the Quality Processor receives status information from the Header Processor, it checks the information against predefined criteria, and updates quality and accounting tables in the status buffer, where they become available to the communications process for eventual transfer to the system MCC.

The Quality Processor also tracks the availability of data pieces in the RAM Controller by examining frame and piece status information. When all pieces of a packet are obtained, or when some pieces have to be rejected, the process builds and sends the Output Processor a packet reassembly table that identifies packet pieces and their respective annotation data via dual-ported buffer memory. It also updates the data flow and error status for each VC and packet source, and checks packet sequence counts to determine whether there are any missing packets.



7.1.10.3 Output Process

Figure 7-11 illustrates EOS Service Processor Card output process data flow. The output process is the final stage of the overall packet reassembly process. It maintains a piece directory using the piece list generated by the header process. When a new packet reassembly table (which identifies the packet pieces and corresponding annotation) is received from the Quality Processor, the Output Processor constructs a packet annotation field. If configured by the user, packet annotation is sent out of the card before packet data. Using the piece directory for starting/ending addresses of packet pieces in the telemetry data buffer, the Output Processor finds the first piece of the packet and outputs the packet through the P3 connector, or piece by piece via the VSB until the packet is complete.



7.2 TAPE RECORDING SUBSYSTEM SOFTWARE

ETS will use the TRS to meet the unique data quantity and rate requirements of EOSDIS testing. The TRS allows very large test data sets (up to 96 Gbytes) to be created, archived, retrieved, and played back at rates up to 150 Mbps.

The TRS software architecture (Figure 7-12) consists of one Sony tape drive and one Ampex tape drive. The Sony drive functions as the operational unit in the TRS during EOSDIS testing. The Ampex drive provides media compatibility with the SCITF team, which will supply spacecraft and instrument-generated test data sets on Ampex tapes to ETS. The Ampex drive will be used by ETS users for dubbing test data sets to the Sony drive as an offline function. At the application software level, the TRS will be “seen” as a programmable resource within ETS, allowing applications to manipulate its tape drives, tapes, and data sets. Any ETS software element can manipulate the TRS as long as it adheres to the command protocol of the TRS interface.

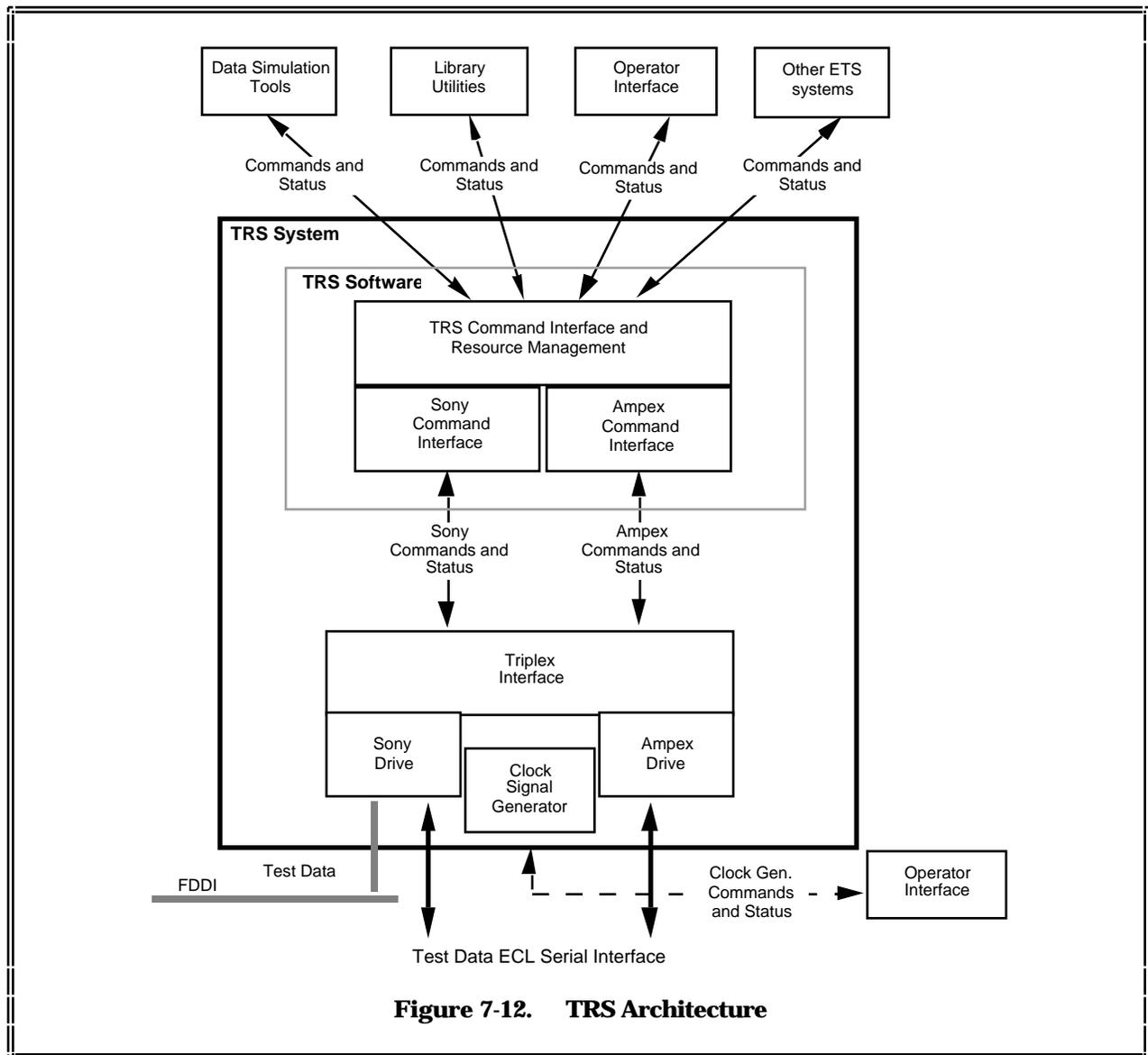
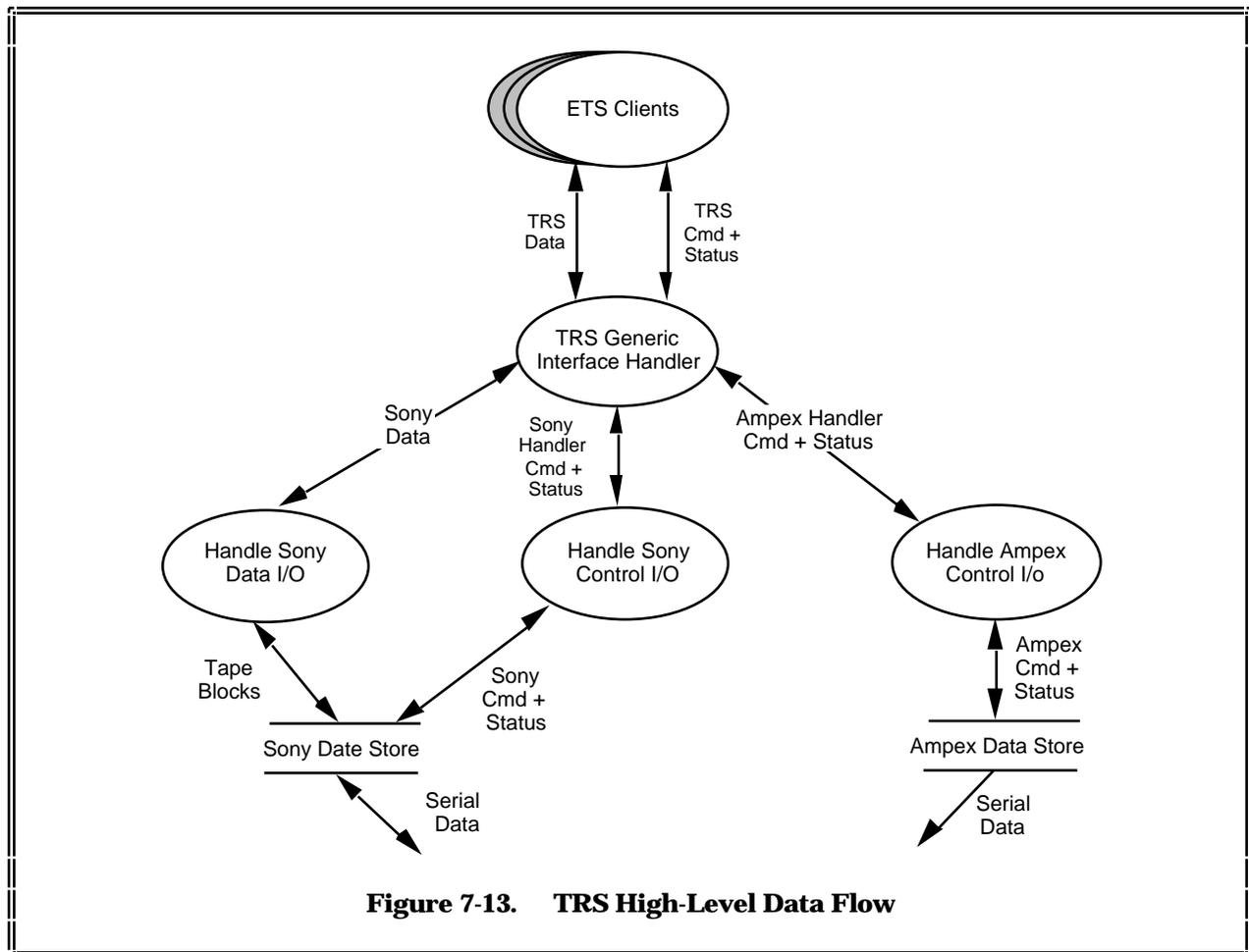


Figure 7-12. TRS Architecture

7.2.1 HIGH-LEVEL DESIGN

The TRS is a distributed system that consists of software running on a number of heterogeneous workstations controlling a collection of heterogeneous tape recorders and associated interfacing hardware. The TRS design hides the details of the manufacturer-specific tape recorders and other devices that interface with them from client programs. This concept includes the TRS GUI and TRS Applications Program Interface (API), which supply a generic tape recorder model for operator interaction. Figure 7-13 illustrates, without regard to implementation constraints, the general flow of information through the TRS and its levels of abstraction.



A high-level abstract tape recorder is presented to the client. Commands and status are translated from the abstract recorder interface to the device-specific interface, and finally to device commands.

Recorder data may have one of two possible sources and destinations. Data may be received or sent to the client program initiating control of the device. Or, data may leave the system on the serial data interface bound for an unknown, external entity connected to the serial interface. Note that the middle level of abstraction provides a “handler” for each device type. This layer allows for various implementation methods of handling the device-specific control and data I/O. For instance, Sony recorder control handling can be implemented by Remote Shell (RSH) calls to Triplex utility programs, or it may be handled by a custom server task that makes *ioctl* calls to the Sony recorder device driver.)

7.2.2 HIGH-LEVEL DESIGN DESCRIPTION

Figure 7-14 illustrates some of the realities hidden by the high-level abstractions discussed in Section 7.2.1. TRS clients are separated into TRS GUI and “Other Clients”—this is for two important reasons. First, the GUI must have access to both tape recorders, whereas Other Clients only need access to the Sony drive via the API. Second, these clients may run on different machines, which dictates that the TRS control interface at the “generic tape recorder level” be handled via Ethernet/TCP/IP to allow the application to be distributed across many workstations.

Figure 7-14 also includes the CSG, which is shared by both tape recorders.

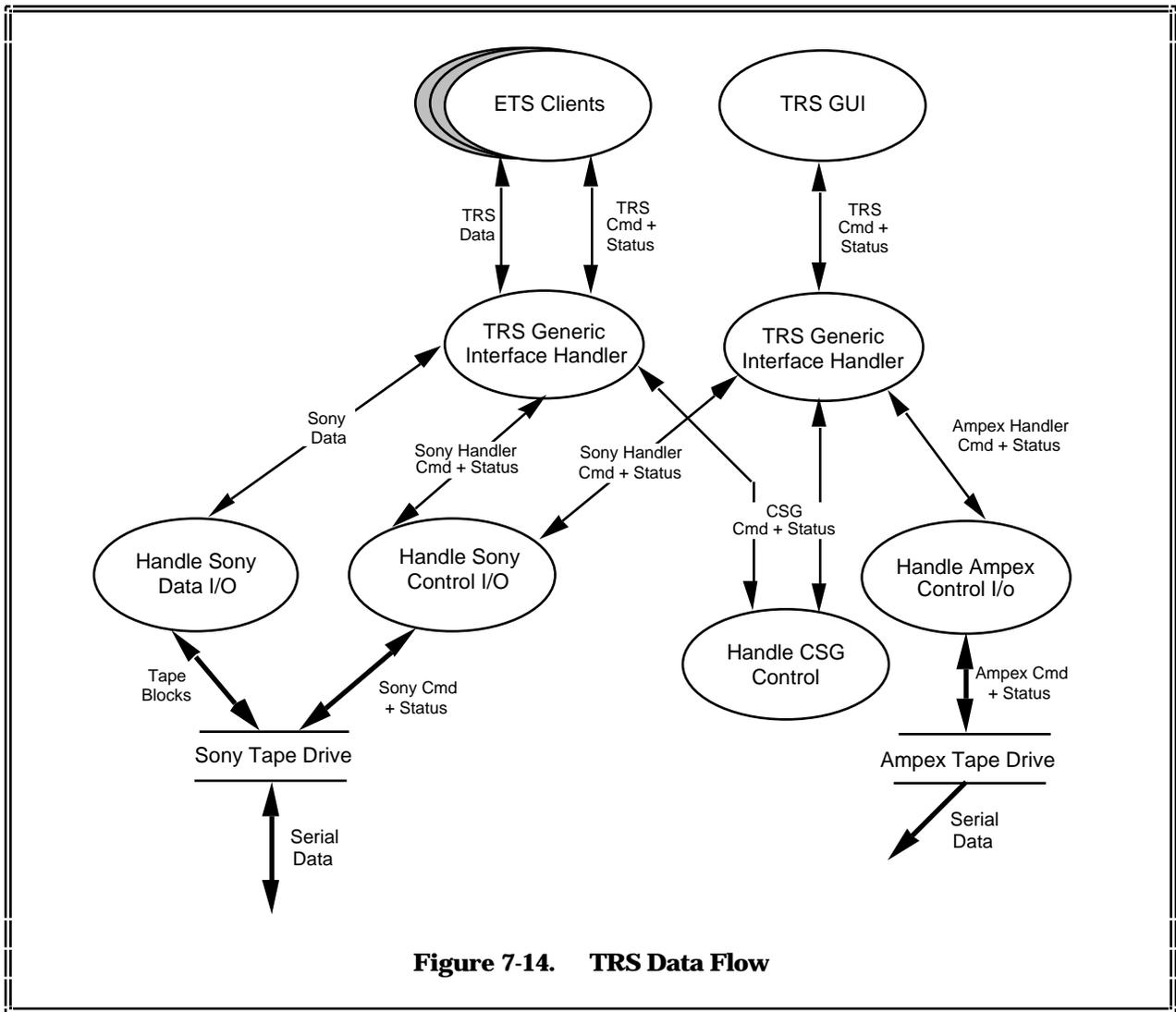


Figure 7-14. TRS Data Flow

The TRS GUI must run on the CDS. Therefore, this process is separated from “Other ETS Clients,” which are not constrained in this way. They may/may not run on the CDS, and are therefore shown separately. The GUI also has features that make it unique among TRS clients. For example, the GUI has only control/status connections with the tape drives. It does not source or sink data (as other clients may), and only the GUI interacts with the Ampex drive.

Because the Ampex drive is controlled through an RS-232 port, it is connected directly to the CDS so that software controlling the Ampex can directly communicate with the RS-232 port without need of a client/server link to another workstation. Linkage between the GUI, generic TRS handler, and Ampex handler processes will actually be function calls within the same program (shown as solid lines). The various processes will be implemented as libraries to which the client code will link.

The Triplex interface controls the Sony drive, which is locally connected to it. This forces the Sony handler process to be resident on the Triplex, while the TRS generic handler process will be part of each client. All control interaction with the Sony goes through this path. The generic TRS control process will make a socket connection to the Sony handler on the Triplex to exchange control and

status information.

Data connections between “Other ETS Clients” and the Triplex will be socket-based on a FDDI network. The data socket will be opened as part of a file open, and will exist until the file is closed.

7.2.3 DETAILED DESIGN

7.2.3.1 TRS Generic Interface Handler

The TRS receives commands and returns status to a number of possible controlling entities, including operator interface, SCTGEN, library utilities, HRS, and possibly other subsystems. These functions are handled by the generic interface handler. This software presents a generic tape recorder to TRS clients, while hiding the details of TRS hardware/software interfaces and differences between the Sony and Ampex tape drives. It is used by ETS clients including the operator interface to access basic TRS functions, in the form of a library to which the client software will link.

Implied in the partitioning of ETS is the ability of the ETS human operator and ETS subsystems to manipulate the TRS. During EOSDIS testing, the TRS is normally controlled by the ETS operator; however, in some tests, and in the production of test data, other ETS subsystems may be the controlling entities. The view of the TRS is slightly different from the human operator interface than from the API, based on the functionality available to each class of user.

The human operator’s functional view of the TRS will be similar to a common audio tape recorder, but with an added directory capability. The operator will have two drives to work with—one Sony drive and an Ampex drive for dubbing. This is the only area in the TRS where dubbing Ampex to Sony is needed. Functions at this level will operate only on entire blocks of test data. Operations will include the following:

- a. Directory, Cue, Play, Record, Dub, Stop, Eject: the API will support the creation and modification of test data by ETS programs. The API operates at a lower level than the human operator, and must support test data I/O at the record level. Specifically, programs must have the ability to read/write records or variable-length byte streams to/from test data on a tape in a particular drive.
- b. Open, Read, Write, Position, Close: the API must also support those functions supported by the human operator interface to allow program control of test scenarios. The API is a super-set of the human operator interface capabilities providing access to the Ampex drive. At the API, the Ampex drive is not needed. Test data will be created and played on Sony tapes only.

From the user point of view, there are five types of entities to be manipulated in the TRS: tape drives, tapes, directories, test data, and data records. Each drive can be manipulated separately. Only one user is allowed per drive. Once access to a drive is granted, others will be locked out until the drive is explicitly unlocked.

A set of commands will be supplied to the user to manipulate these entities. Examples of operations and parameters that may be supported by the TRS include:

- a. -LockDrive(driveID, timeout).
- b. -FreeDrive(driveID).
- c. -EjectTape(driveID).

- d. -SearchDirectory(driveID).
- e. -CueDataset((driveID,datasetID).
- f. -PlayDataset(driveID,datasetID,outputChnl,dataRate).
- g. -RecordDataset(driveID,datasetID,inputChnl).
- h. -DubDataSet(fromDriveID,datasetID,outputChnltoDriveID,datasetID,inputChnl).
- i. -RewindTape(driveID,position).
- j. -FastForwardTape(driveID,position).
- k. -Open(driveID,datasetID).
- l. -Position(driveID,datasetID,recordNum).
- m. -Read(driveID,datasetID,outputChnl).
- n. -Write(driveID,datasetID,inputChnl).
- o. -Close(driveID,datasetID).

7.2.3.2 Handle Sony Command and Status Process

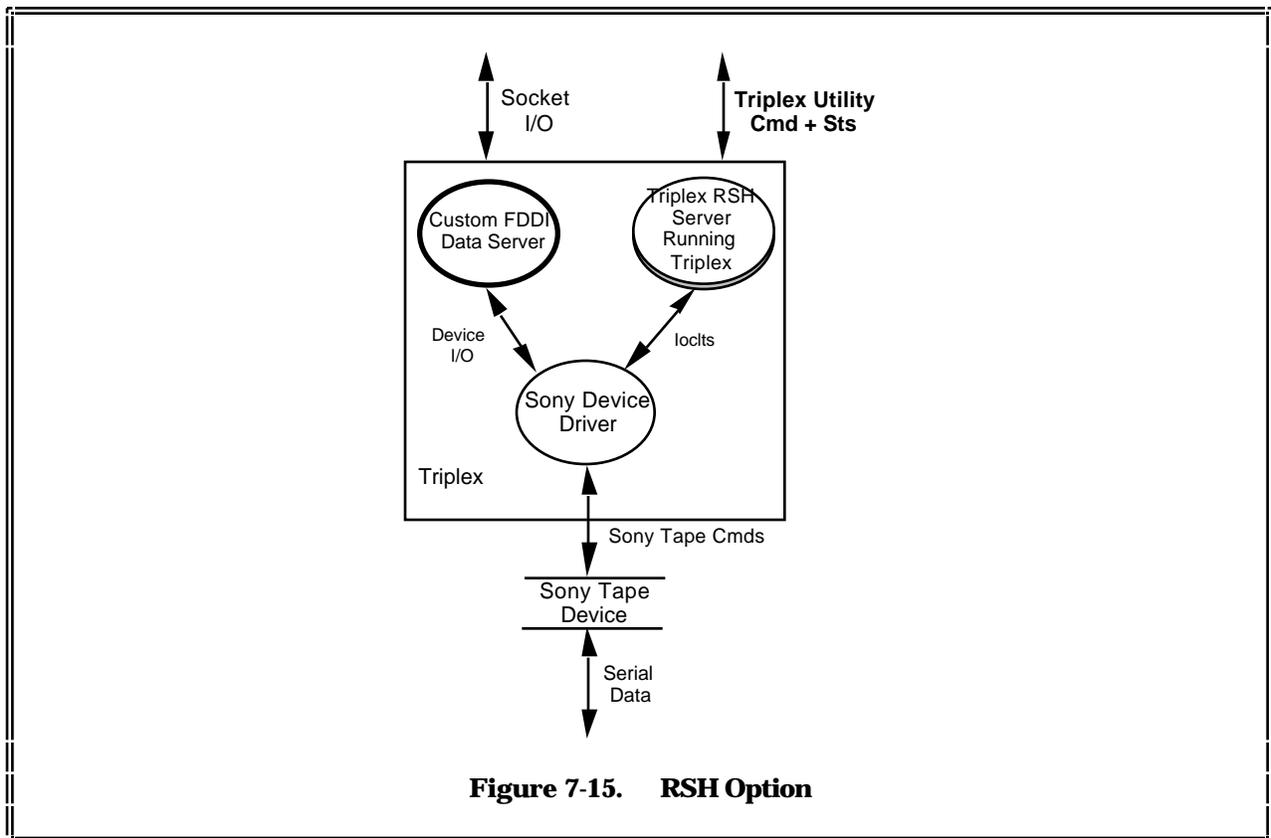
Figures 7-15 and 7-16 expand on the process depicted in Figure 7-14. There are two possible ways to handle this process:

- a. Using UNIX built-in features of the RSH.
- b. Building a custom Sony control server.

In both cases, the software implementing this process and path is distributed between clients running on the CDS and the server running on the Triplex.

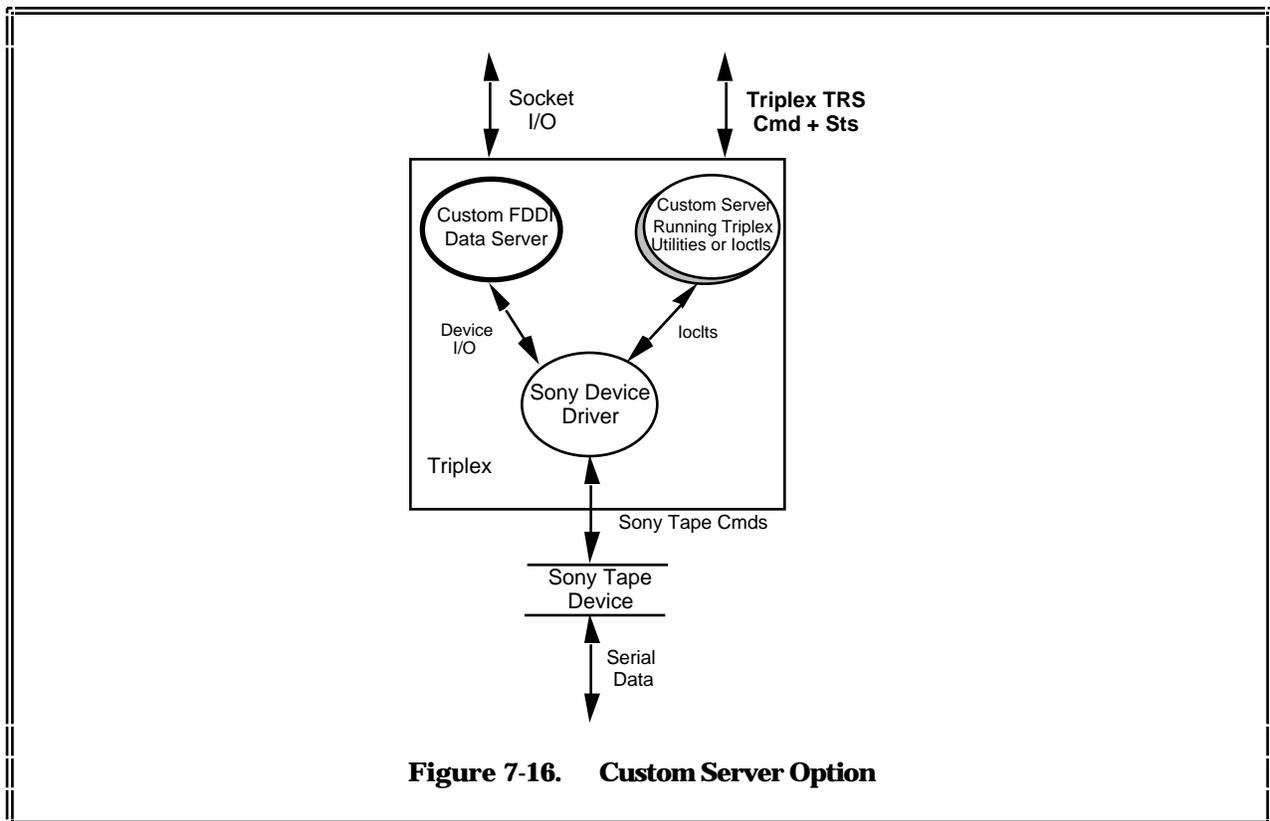
7.2.3.2.1 Handle Sony Command and Status Via RSH

The fastest, least implementation-intensive method of handling this process is by using the UNIX RSH to pass Triplex utility (MT, TMT, LPMT) commands from the controlling entity to the Triplex, and ultimately to the Sony drive. Note in Figure 7-15 that the Triplex utility *Cmd* and *Sts* flow is highlighted to show that the content of the control flow is actually Triplex utility commands and status returned from running them.



7.2.3.2.2 Handle Sony Command and Status Via Custom Server

A more “elegant,” yet implementation-intensive method of handling this process is by using a custom RSH to pass commands from the controlling entity to the Triplex, and ultimately to the Sony drive. This set of commands and status flows provides a completely transparent interface for all tape drives. Note in Figure 7-16 that the custom TRS commands and status flow is highlighted to show that the content of the control flow is a custom command set that the server is designed to understand.



This option has the following advantages:

- Uses a generic command set for all tape drives. Code can be reused for any tape drive by connecting to the socket for that drive.
- There is a server for each drive that is accessed via TCP/IP sockets; it can be located anywhere.
- Command set supplies a simple high-level interface to controlling the tape drive.
- Assumes the CSG is connected to Triplex. If not, the design is complicated by the fact that CSG manipulation will have to go through a CSG server wherever it exists.

7.2.3.3 Sony Data I/O Handler Process

This program is based on skeleton code supplied by Triplex. It will listen on a socket (or be spawned to listen on a socket by the control server). The client code will actively connect to the socket to retrieve data. The client will obtain the socket port number from the control server when it executes a *trsFilePlay* or *trsStrmOpen*. The socket connection will be closed by *trsFileStop* or *trsStrmClose*.

The Triplex sample data server for FDDI has a dedicated read/write socket. A connection to a socket implies that the user wants to read or write from the current tape position.

7.2.3.4 Ampex Command and Status Handler

The Ampex tape recorder supplies a simpler and less-featured interface than the Sony/Triplex. It is controlled via ASCII commands sent over RS-232. There are no directories on the Ampex, or the ability to position files based on file name, as on the Sony/Triplex.

Design issues to be considered for development of this module are as follows:

- a. **Location and Access to Ampex:** the current plan is to locate the Ampex on the CDS and communicate with it directly from the TRS GUI via the generic recorder command set mapped to Ampex-specific commands. Note that only the TRS GUI, and not other clients, accesses the Ampex. Therefore, there is no need to make this a server architecture that is accessible by other clients.
- b. **Directories:** the Ampex has no directories. Discussions with Ampex technical representatives revealed that a directory could be built. However, this will only be of use if the SCITF uses the same directory structure. An agreement on this is being drafted, and will be finalized when all information is available. The default will be to position the tape to a particular, preknown block position, using the block locator option provided by Ampex for writing on tapes, and begin to play or record data. The actual start/end positions of the tape will have to be manually recorded, or programmatically recorded on disk, but not on the tape itself.
- c. **Ampex Software Interface:** the current plan is that it will be a linkable library that is directly called by high-level TRS commands. However, this may still be implemented as a server if there are advantages in code reuse, uniformity, and flexibility to allow other clients not running on the CDS to use the Ampex.

7.2.3.5 Clock Signal Generator Handler Process

The CSG produces a clock signal that is required for outputting serial data from a tape recorder. The tape recorders do not source a clock signal themselves—only the data. Therefore, a CSG is required.

The CSG is controlled through an IEEE 488 interface, which means it could be placed on the CDS (where the TRS GUI runs), or it could be placed directly on the local Triplex workstation. The option used is to put it on the CDS, and constrain all TRS clients to be resident on the CDS so that their code can access the CSG through local IEEE 488 port. The generic *trsOpen* and *trsPlay* commands will be written such that they will communicate through the local IEEE 488 bus to hide details from clients. All clients are required to run on the CDS.

SECTION 8

LOCAL AREA NETWORK AND CONFIGURATIONS

Two types of network communication are used by the ETS HRS: FDDI and Ethernet. Typically, the Ethernet interface is used to set up the system, monitor system status, and output real-time packets. FDDI is used to transfer data sets and test data files in/out of the system.

8.1 COMMUNICATION PROTOCOLS

A protocol provides formulas for passing messages; it includes the message format and rules for error conditions. The ETS HRS uses FTP, TCP/IP, and User Datagram Protocol/Internet Protocol (UDP/IP), which provide a standard method to regulate data transmission between the ETS HRS and a host workstation. Although FTP, TCP, UDP, and IP are used in conjunction throughout this document, they are separate protocols that perform separate functions.

FTP ensures the successful transmission of data files. It records what is sent, retransmits anything not successfully transmitted, and returns data to its proper order after transmission. TCP ensures the successful transmission of commands to the receiver. It records what is sent, retransmits anything not successfully transmitted, and returns data to its proper order after transmission. UDP is at the same level as TCP. IP routes data to its correct destination.

In the following discussion, an octet is defined as 8 bits, and a datagram is a collection of data that is sent as a single message during a data transfer. Information in the following discussion was obtained from "Introduction to the Internet Protocols," produced by Rutgers University; the material copyright belongs to Charles Hedrick.

8.1.1 FILE TRANSFER PROTOCOL

FTP allows users to access and retrieve files from remote computers. A client program running on one system allows the user to communicate with other systems—it translates commands into requests for information from other programs, or servers, running on remote systems. FTP is used by the Data Set Processor to transfer data sets as binary files to specified users. A typical FTP session is summarized by the following procedures:

- a. Start up FTP client program.
- b. Assign FTP connection address to client.
- c. Provide remote site with user identification.
- d. Provide remote site with user password.
- e. Search directory for files.
- f. Change directories.
- g. Set transfer mode (optional).
- h. Transfer desired files.
- i. Quit.

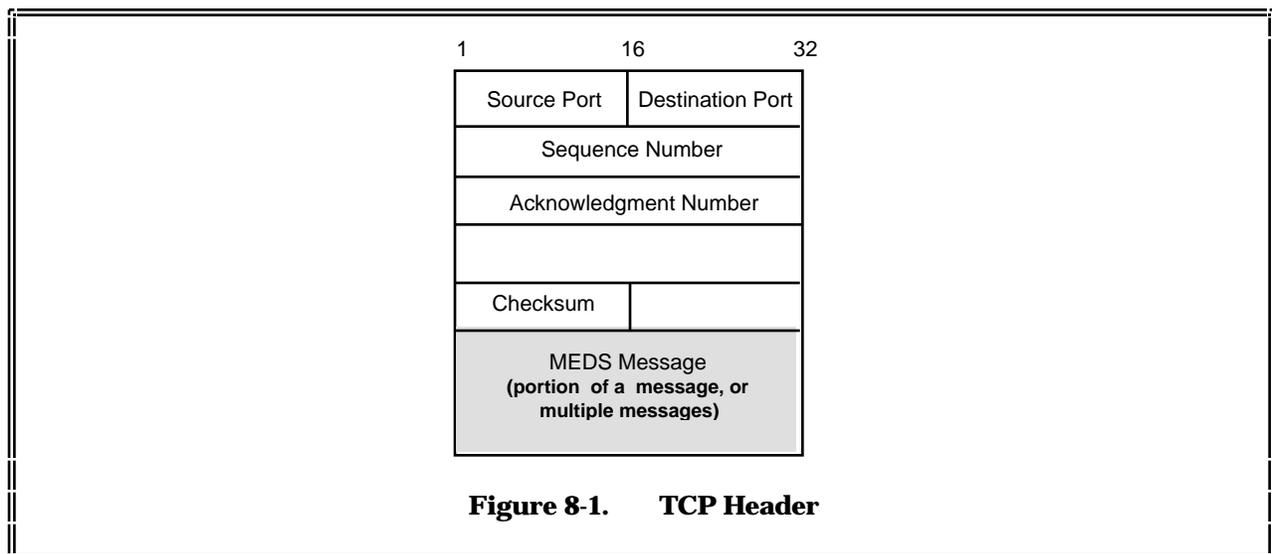
8.1.2 TRANSMISSION CONTROL PROTOCOL

TCP is summarized by the following capabilities:

- a. Transfers a stream of octets in both directions between the ETS HRS and a workstation by packaging a number of octets into segments for transmission.
- b. Handles errors: damaged, lost, or duplicated data, and data delivered out of order.
- c. Allows many processes to simultaneously transfer data by assigning addresses (ports) to each process.
- d. Maintains status information for each data stream.

TCP breaks the data stream into manageable chunks, and attaches a header—this constitutes one datagram. In the ETS HRS, the data field of the datagram is a MEDS message, a portion of a MEDS message, or multiple MEDS messages. Primary header fields are defined as follows:

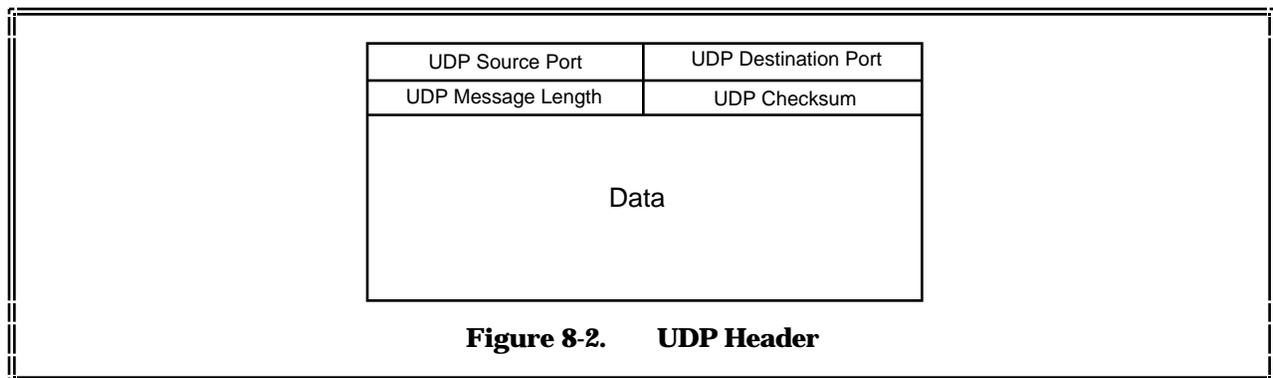
- a. Source and Destination Port: port numbers identify different conversations. Each end of the communication assigns a different port number to a conversation; source and destination port fields identify these port numbers (Figure 8-1).
- b. Sequence Number: ensures that data is assembled in the correct order and without missing portions when it arrives at the destination side of the transfer. The sequence is maintained by the number of octets passed; not the number of datagrams passed. Therefore, this number would increment by 500 if each datagram contained 500 octets.
- c. Acknowledgment Number: ensures that a datagram arrived at the receiving side. When a datagram arrives, the receiving side returns a datagram with an acknowledgment number that reflects the total number of octets received; the field value indicates that all octets have been received up to that number.
- d. Checksum: holds the sum of all octets in the datagram. The receiving end recomputes this number upon the datagram's arrival and checks it against the header data field. If the numbers do not agree, the datagram is retransmitted.



8.1.3 USER DATAGRAM PROTOCOL

Each UDP message, which consists of a UDP header and UDP data field, is called a datagram. UDP does not use acknowledgments to ensure the safe arrival of data. It primarily identifies the exact location of the datagram recipient. UDP header (Figure 8-2) fields are defined as follows:

- a. Source Port: identifies port to which a reply should be sent (optional; set to zero if not used).
- b. Destination Port: provides exact port to which datagram is to be delivered when it arrives at the host.
- c. Message Length: length in octets of the UDP datagram: UDP header and data.
- d. Checksum: allows receiving end to verify that data was received without error (optional; set to zero if not used).



8.1.4 INTERNET PROTOCOL

TCP and UDP send each datagram to IP. IP finds a route and transports the datagram to its destination. To accomplish this, IP must add its own header. Primary header fields (Figure 8-3) are defined as follows:

- a. Source and Destination Addresses: reflect Internet addresses of sending/receiving systems.
- b. Protocol: number that instructs receiving end to send the datagram to TCP.
- c. Header Checksum: sum of IP header. The receiving end recomputes this value, compares it against the header value, and discards (which requires retransmission of datagram) any datagram for which the two values do not agree.

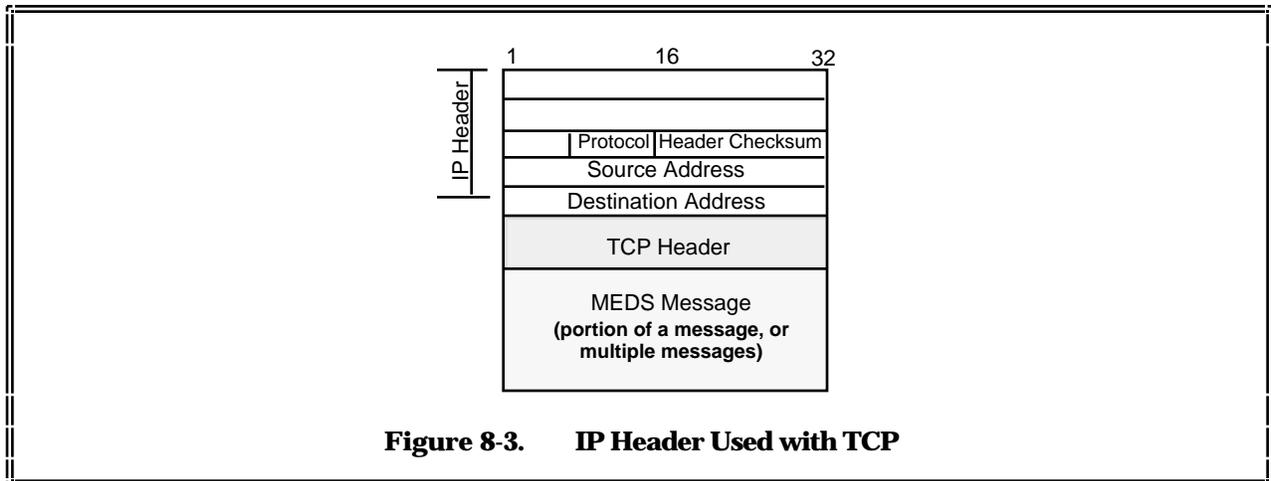


Figure 8-3. IP Header Used with TCP

8.2 FIBER DISTRIBUTED DATA INTERFACE

The ETS HRS has two FDDI interfaces. The local area FDDI, or internal interface, is used by the ETS HRS for the transfer of large-volume data files (up to 2 Gbytes) between the VHS and the CDS. The protocols used for these transfers are TCP, IP, and FTP. The IP layer transfers data between a pair of hosts on the Internet. The TCP layer differentiates among multiple sources or destination ports within one host.

The wide area FDDI, or external interface, to the ETS HRS is used for the high-rate transfer of data from ETS HRS to EBnet. The protocols used are TCP, IP, and FTP.

8.2.1 FDDI MODULE

For high-rate data transfers to EBnet, ETS HRS uses a FDDI interface controller card. To transfer data from the FDDI interface, the Data Set Processor opens a socket through the FDDI WAN interface port. For the transfer of test data files and data sets between the CDS and ETS HRS, the FDDI interface controller card will be accessed by the Master Controller and Data Set Processor, respectively.

8.2.2 FDDI FRAME

The final layer before transfer by FDDI is a FDDI frame. The total frame (FDDI header, FDDI data field, and FDDI trailer [16-bit CRC]) may vary in size from 60 to 4500 bytes. The IP and UDP application layers are encapsulated within the FDDI frame (see Figure 8-4).

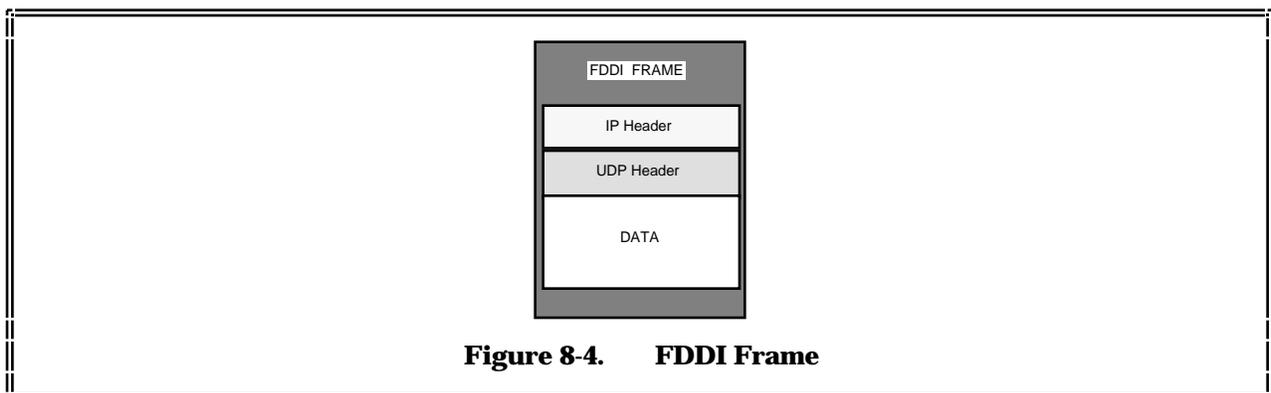


Figure 8-4. FDDI Frame

8.3 ETHERNET INTERFACE

Commands and data have a bidirectional flow on the Ethernet interface between ETS HRS and remote workstation(s). The Master Controller manages these communications; the physical link between it and the workstation is Ethernet. Essentially, the Master Controller software acts as a mailman that passes messages back and forth between the workstation and the ETS HRS. The communication protocols used for these data transfers are TCP/IP. The information passed may include commands, status, and real-time data, but it is always packed into MEDS messages.

The Master Controller and CDS maintain a handshaking relationship; every message transfer requires an acknowledgment that the message was received without error. If an error occurs, the message is retransmitted.

The ETS HRS uses layering, which is a method that uses protocols at different levels. MEDS, TCP, and IP are separate layers that access the services of the level below it. The Master Controller (MEDS) accesses the TCP library of routines, TCP accesses the IP library of routines, and IP calls the Ethernet routines. Figure 8-5 illustrates the format of data after it is processed by these levels.

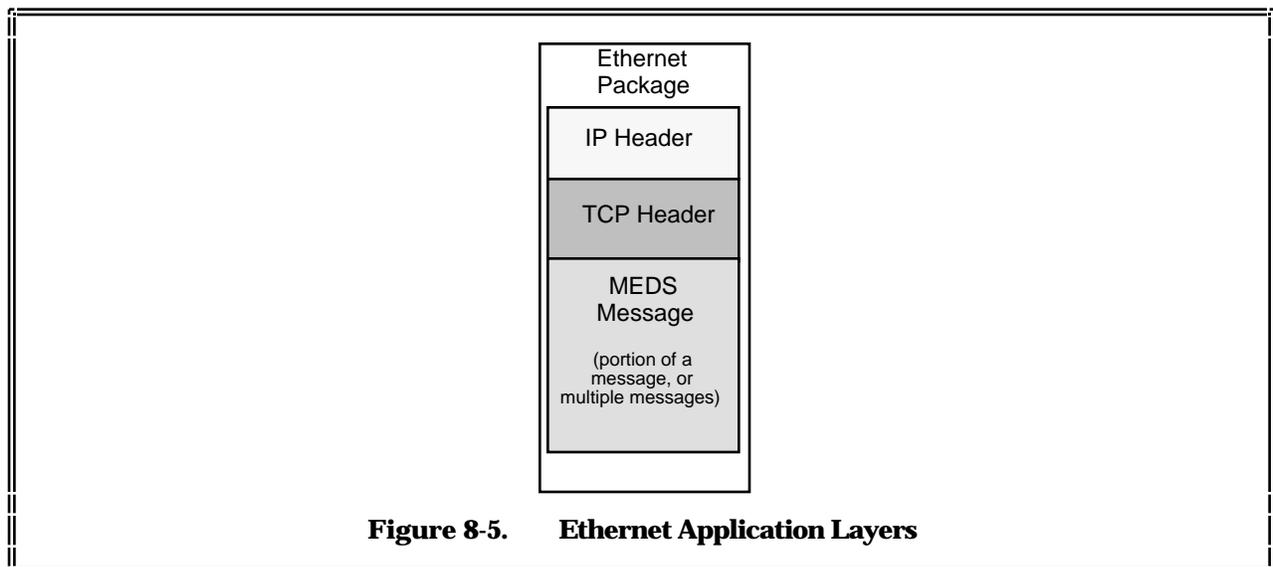


Figure 8-5. Ethernet Application Layers

The ETS HRS uses Ethernet hardware to transfer data from the system. Ethernet is the final layer in the data transfer. After the TCP and IP headers are appended, Ethernet wraps the datagram and headers in a frame (i.e., it appends a header and trailer). Three fields are of primary interest: Ethernet Destination Address, Ethernet Source Address, and Ethernet Checksum (Figure 8-6).

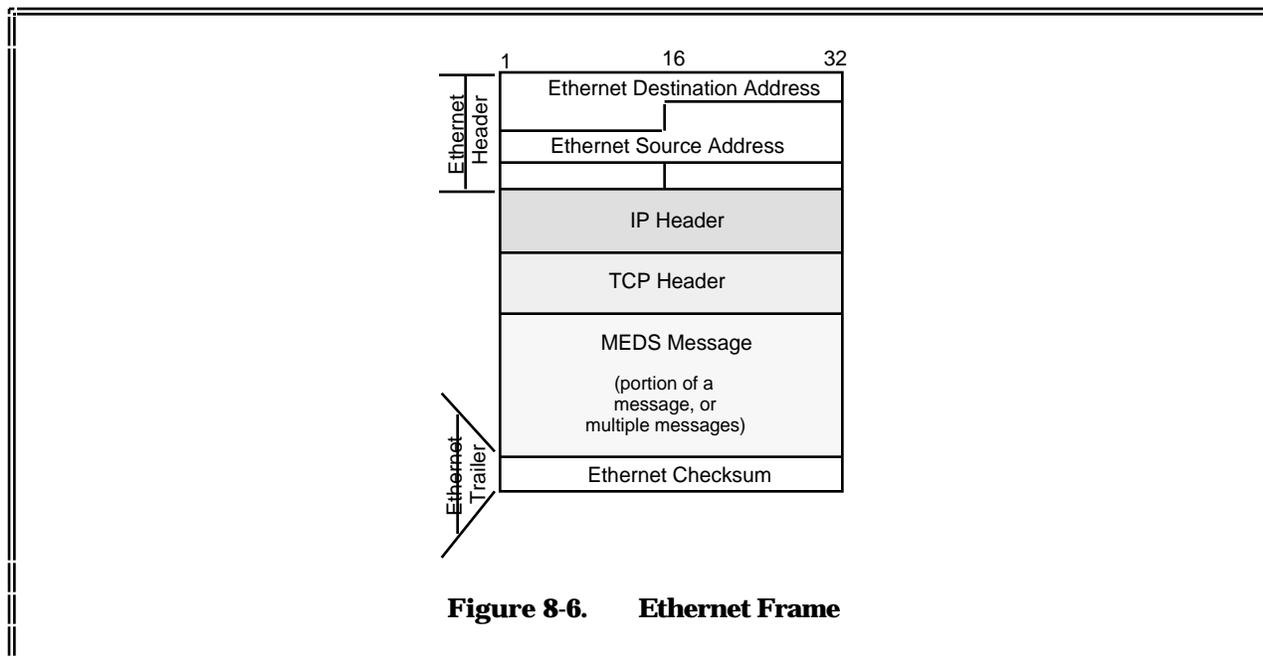


Figure 8-6. Ethernet Frame

Each system has an Ethernet address; the Ethernet Source and Destination Addresses identify the frame's origination and destination by system. The Ethernet Checksum, located in the trailer, is a checksum of the entire frame. When the frame arrives at its destination, the checksum of the frame is computed. If it does not agree with the trailer value, the frame is discarded and retransmitted.

8.4 ETS HRS SYSTEM CONFIGURATION

The ETS HRS consists of the CDS, VHS, and TRS. A physical Ethernet connection exists between the three systems. Each system supports a full suite of TCP/IP protocols. The VHS has seven Internet addresses assigned to the VME rack. Each physical card needs an address to boot. Only the Master Controller Card has a physical Ethernet connection. All other cards use the Shared Memory Network scheme, which works like Ethernet, except that it is implemented over the VME backplane. All CPU cards on the Shared Memory Network are assigned an IP address and support network protocols. The VHS contains the Master Controller Card, EOS Frame Synchronizer Card, EOS Service Processor Card, EOS Simulator Card, EOS Reed-Solomon Card, two FDDI interface cards, Annotation Processor, and two Data Set Processor Cards.

The CDS is a UNIX-based system with full TCP/IP support. Its address is known to the VHS TRS and EOC to establish connections as needed. CDS may use more than one physical Ethernet connection and address, if performance is an issue. A priority scheme is implemented to support this case.

8.4.1 MASTER CONTROLLER AND CDS CONFIGURATION

The CDS makes three dedicated socket connections to the Master Controller at the beginning of a mission. These connections are for MCRCommand, MCRStatus, and MCREvent. These connections are used to transfer and receive commands, status, and event messages generated by CDS and Master Controller Card. This is the primary communication between the CDS and VHS. Control of the VHS is accomplished through this connection.

8.4.2 EOS SIMULATOR CARD AND CDS CONFIGURATION

This connection provides a way to transfer test files from CDS disks to the SCSI-2 disk drive controlled by the EOS Simulator Card. A DOS file system installed on the SCSI device is cross-mounted to CDS over Ethernet using TCP/IP protocol. The CDS mounts the remote SCSI disk at the beginning of a mission and downloads test files by copy.

8.4.3 TRS AND CDS CONFIGURATION

The CDS has four different interfaces with the TRS. The FDDI socket connection permits read and write from/to tape drives. Control interfaces for the Ampex, Sony, and clock generator are RS-232, Ethernet, and IEEE 488, respectively.

SECTION 9 SYSTEM TEST APPROACH

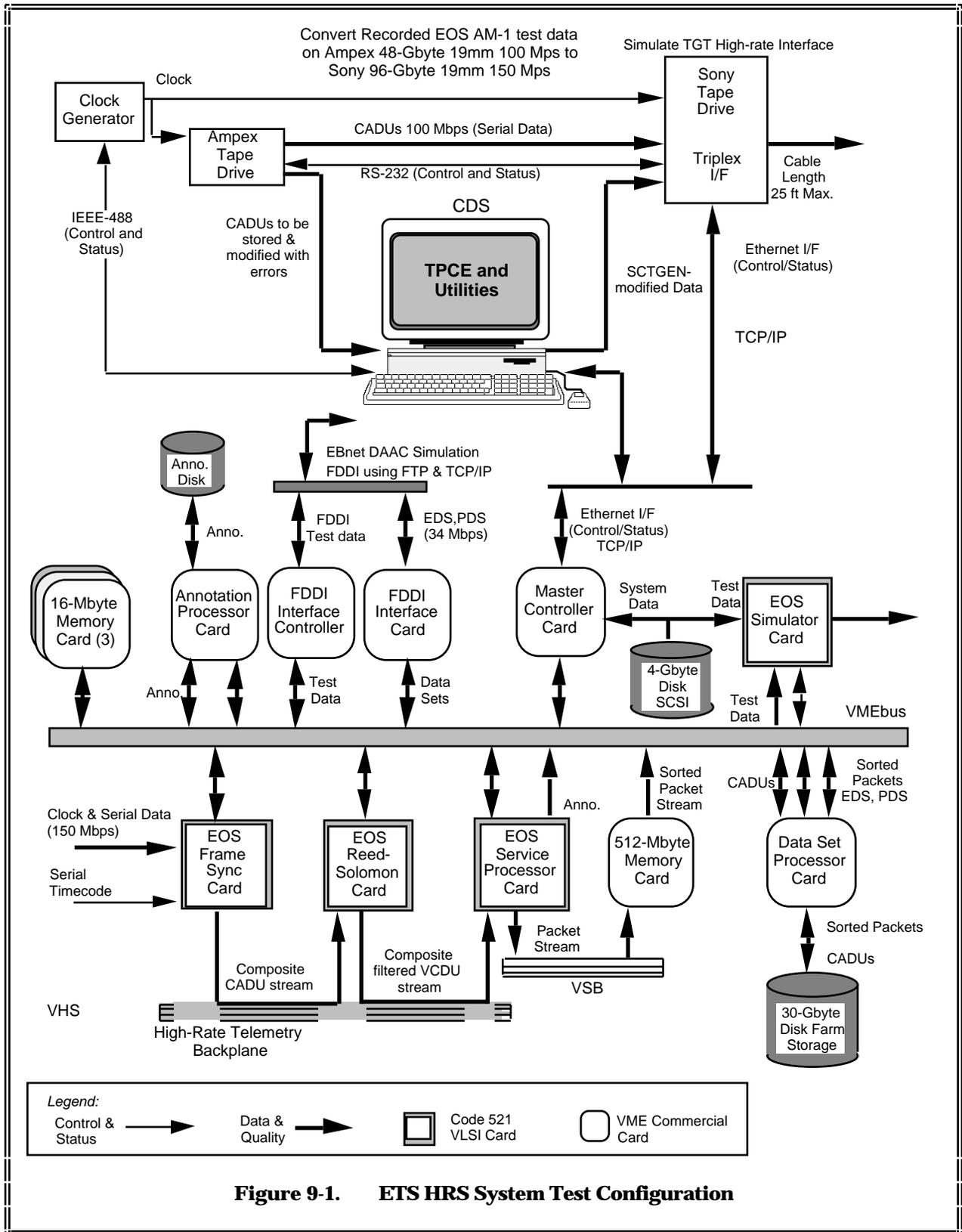
9.1 INTRODUCTION

A system test plan will be developed to validate the ETS HRS in an environment that simulates a real-world scenario. System testing will verify that the ETS HRS is ready for delivery to the ETS project, and will include user interface and functional capabilities tests. System testing will determine whether the software complies with functional, performance, operational, and interface requirements while operating in a test environment. The intent is to verify that the system satisfies the requirements specified in the ETS HRS Requirements Document. Figure 9-1 depicts the system test configuration.

Initial tests will concentrate on the following subsystem elements:

- a. Master Controller Card.
- b. EOS Simulator Card.
- c. EOS Frame Synchronizer Card.
- d. EOS Service Processor Card.
- e. Data Set Processor/SCSI-II interface.
- f. Annotation Processor.
- g. FDDI interface cards.
- h. Sony/Triplex tape drive.
- i. Ampex tape drive.

The fully integrated system will be evaluated using the EOS Simulator Card, as well as external test equipment for total functional compliance.



9.2 **SYSTEM TESTS**

System test case acceptance criteria are defined as a part of the system test plan. System tests are conducted by the System Test Team; results are documented in a system test report. The System Test Team, in conjunction with the development team, will use the verification criteria to evaluate test results and determine system acceptability. Discrepancy reports will be written for any problems or anomalies encountered.

System testing, as a minimum, will demonstrate that:

- a. The ETS HRS satisfies all operational requirements under nominal and capacity conditions.
- b. Each system function affected by newly developed and newly modified software included in the ETS HRS is correctly processed and produces responses consistent with the function.
- c. The ETS HRS responds correctly to simulated anomalies.
- d. Outstanding problems found in previous software deliveries are resolved satisfactorily in the current release/build.

9.3 **TEST GUIDELINES**

Tests will be conducted to prove functional compliance with operational requirements of the referenced specifications. The following guidelines will be used to measure the success and thoroughness of a test:

- a. All minimum functions defined by the established standard protocols are executed correctly at least once.
- b. Verification tests will be automatically executed to accelerate testing and reduce errors.
- c. The capability will be provided to perform step-by-step analysis for comparison between expected and actual results.
- d. A dynamic test modification capability will be provided to:
 - (1) Retest specific sequences that previously exhibited problems.
 - (2) Enable manual inputting of augmented and/or new test sequences for the identification/evaluation of potential problem areas.

9.4 **TEST APPROVAL**

All test procedures, test configuration, and test equipment will be approved by the ETS HRS Project Manager.

9.5 **ENVIRONMENTAL TEST CONDITIONS**

ETS HRS equipment will be tested in the Code 521 Microelectronic Laboratory. Tests will be performed under the following environmental conditions:

- a. Temperature at the test location will be within the range of 0°C to +36.7°C.
- b. Relative humidity at the test location will be within the range of 20% to 80%.

9.6 DISCREPANCY REPORTING AND RETEST

Any constraints and irregularities observed during a prescribed test will be recorded on a Discrepancy Report (DR) form. Every DR will be evaluated and assigned a priority.

9.7 FAILURE DURING TEST

If an ETS HRS component fails during the performance of any test, the following actions will be taken:

- a. Test will be immediately terminated and the failure recorded in the Problem Reporting System (PRS).
- b. Project management will review problem reports and classify problems according to severity.

9.8 RETEST AND REGRESSION TESTING

Upon completion of problem resolution, the test will be resumed at the point or portion of the test cycle where the failure occurred, or all necessary tests will be repeated as specified by the evaluator(s).

Prior to each new release, all necessary tests that were successfully executed under previous release may be repeated as determined by the evaluator(s).

9.9 TEST DEPENDENCIES

The test will be conducted after satisfying the following dependencies:

- a. Availability of test equipment (e.g., local control terminal or workstation) and tools listed for each test.
- b. Availability of test procedures that have been reviewed and approved by the ETS HRS Project Manager.
- c. Completion of mechanical and electrical assembly inspection after the replacement of a board.

SECTION 10 REQUIREMENTS TRACEABILITY MATRIX

This section traces requirements from the ETS HRS System Requirements Specification to the hardware/software CIs described in this document. The reference will point to the hardware or software module that implements the functions necessary to meet the requirement(s).

10.1 ETS HRS SRS TRACED TO ETS DDS

Rqmt. Number	ETS HRS Requirement Description	F&P Rqmt. Number	ETS DDS
2.2.1.1	The VHS shall provide one input serial port and two output serial ports. Electrical and timing characteristics of these ports shall comply with the TGT-EDOS ICD. As stated in this document, ports shall comply with the 100K ECL standard with differential data and clock signals. The maximum length for each connection (source or destination) shall be 25 feet.	5.2.1	VHS, TRS
2.2.1.2	The VHS shall use the input port to receive test data and clock from a data source at data rates up to 150 Mbps. A data source can be the TRS, test data generated by the HRS itself, or other external sources as specified in the TGT-EDOS ICD.	5.2.1	VHS
2.2.1.3	The VHS shall use the serial output ports to output test data and clocks to the EDOS or TGT as specified in the TGT-EDOS ICD. The output port shall support data rates up to 150 Mbps per port.	5.2.1	VHS
2.2.2.1	The VHS shall provide an EBnet interface. This interface shall comply with the EDOS-EBnet and EDOS-DAAC ICDs.	4.8.1, 4.9.1	VHS
2.2.2.2	The VHS shall use the EBnet interface to exchange data sets with EOSDIS elements under test such as EDOS and DAAC.	4.8.1, 4.9.1	VHS
2.2.2.3	The VHS shall output data sets to a user-specified DAAC via EBnet interface using FTP and TCP/IP.	4.8.1, 4.9.1	VHS
2.2.2.4	The VHS shall output data sets to a single destination at rates up to 34 Mbps via EBnet interface.	5.3.1, 5.3.2	VHS
2.2.2.5	The VHS shall receive data sets from the EDOS via EBnet interface using FTP and TCP/IP.	5.4.1, 5.4.2	VHS
2.2.2.6	The VHS shall receive data sets from the EDOS at rates up to 34 Mbps via EBnet interface.	5.4.1, 5.4.2	VHS
2.2.3.1	The VHS shall support a LAN based on Ethernet.	Derived	VHS
2.2.3.2	The VHS shall transmit data through Ethernet using FTP and TCP/IP protocols.	Derived	VHS
2.2.4.1	The VHS shall support a LAN based on FDDI.	Derived	VHS
2.2.4.2	The VHS shall transmit data through FDDI using FTP and TCP/IP protocols.	Derived	VHS

2.3.1.1	The TRS shall read test data off Ampex tape media as recorded by EOS AM-1 developers at SCITF. The data format recorded on tapes is specified in the Ampex Data Format MOU.	3.1.2, 3.1.13	TRS
2.3.2.1	The TRS shall provide one output serial port. Electrical and timing characteristics of the ports shall comply with the TGT-EDOS ICD. As stated in this document, this port shall comply with the 100K ECL standard with differential data and clock signals. The maximum length for the connection shall be 25 feet.	5.2.1	TRS
2.3.2.2	The TRS shall use the serial output port to output test data and clock to the VHS when simulating a TGT operational scenario. The output port shall support data rates up to 100 Mbps.	5.2.1	TRS
2.3.3.1	The TRS shall provide an IEEE-488 port for control of its clock generator.	Derived	TRS
2.3.3.2	The TRS shall allow the CDS to issue commands to and collect status from the clock generator through the IEEE-488 port.	Derived	TRS
2.3.4.1	The TRS shall provide an RS-232 port for control of its Ampex tape drive.	Derived	TRS
2.3.4.2	The TRS shall allow the CDS to issue commands to and collect status from the Ampex tape drive through the RS-232 port.	Derived	TRS
2.4.1.1	The CDS shall support a LAN based on Ethernet.	Derived	CDS
2.4.1.2	The CDS shall transmit data through Ethernet using FTP and TCP/IP protocols.	Derived	CDS
2.4.2.1	The CDS shall support a LAN based on FDDI.	Derived	CDS
2.4.2.2	The CDS shall transmit data through FDDI using FTP and TCP/IP protocols.	Derived	CDS
2.4.3.1	The CDS shall provide an IEEE 488 port to control the clock generator in the TRS.	Derived	CDS
2.4.3.2	The CDS shall provide interaction to enable the user to issue commands to and collect status from the clock generator through the IEEE 488 port.	Derived	CDS, TRS
2.4.4.1	The CDS shall provide an RS-232 port to control the Ampex tape drive in the TRS.	Derived	CDS
2.4.4.2	The CDS shall provide interaction to enable the user to issue commands to and collect status from the Ampex tape drive through the RS-232 port.	Derived	CDS, TRS
3.1.1	The HRS shall provide file systems on its mass storage devices.	3.1.2, 5.1.10	VHS, TRS
3.1.1.1	The CDS shall provide a UNIX file system.	Derived	CDS

3.1.1.2	The VHS shall provide file systems on all its local mass storage devices that shall: a) Be supported by the VxWorks operating system. b) Be NFS-mountable from the CDS.	Derived	VHS
3.1.1.3	The VHS shall transfer test data files (base set, update, and data set files) from the CDS to its local storage.	3.1.23, 5.4.1, 5.4.2, 5.4.3	VHS
3.1.1.4	The CDS shall transfer test data files (data set files) from the VHS to its local storage.	3.1.23, 3.4.1, 3.4.2	CDS
3.1.2	The HRS shall provide a control environment with automation features.	3.1.6, 3.1.16, 3.5.20	CDS, TPCE
3.1.2.1	The CDS shall provide a primary user interface, as specified in Section 4 of this document.	3.5.20, 5.1.7, 5.1.8	CDS, TPCE
3.1.2.2	The VHS shall provide a local user interface through an RS-232 port, mainly used for diagnostics and contingency operations.	3.1.12, 3.1.6	OPMAN
3.1.2.2.1	The local user interface shall allow users to issue commands.	3.5.20, 5.1.7, 5.1.8	OPMAN
3.1.2.2.2	The local user interface shall allow users to monitor system health and data processing status.	3.1.12	OPMAN
3.1.2.2.3	The local user interface shall allow users to edit configuration sets.	3.1.19	OPMAN
3.1.3	The HRS shall provide self-test and self-diagnostic capabilities.	3.3.3	VHS, TPCE,
3.1.3.1	The HRS shall perform self-test upon bootup and provide results to the user.	Derived	VHS, TPCE
3.1.4	The HRS shall provide a 4mm tape drive for data recording, storage, and playback.	Derived	CDS
3.2.1	The HRS shall output test data serially based on a predefined test pattern with predefined updates.	3.1.8, 3.1.20, 3.2.1, 3.2.1, 3.3.1, 3.3.2, 5.2.1, 5.3.1, 5.3.2, 5.3.3	VHS

3.2.1.1	The VHS shall output the test pattern defined in a test data base set file through the serial port.	3.1.8, 3.1.20, 3.2.1, 3.2.1, 3.3.1, 3.3.2, 5.2.1, 5.3.1, 5.3.2, 5.3.3	VHS
3.2.1.2	The VHS shall output the test pattern defined in a test data base set file repeatedly as specified by the size of the test data file needed by the user.	3.1.8, 3.1.20, 3.2.1, 3.2.1, 3.3.1, 3.3.2, 5.2.1, 5.3.1, 5.3.2, 5.3.3	VHS
3.2.1.3	The VHS shall modify the test pattern in the test data base set file being transmitted based on locations and update values specified in a test data update file.	3.1.8	VHS, CDS
3.2.1.4	The VHS shall encode telemetry frames or VCDUs with Reed-Solomon code based on the CCSDS Telemetry Channel Coding Recommendation.	3.1.1, 3.1.6, 3.1.7	VHS, CDS
3.2.1.5	The VHS shall support Reed-Solomon encoding with interleave from 1 to 5.	3.1.1, 3.1.6, 3.1.7	VHS, CDS
3.2.1.6	The HRS shall simulate bit slip errors in the test data it is transmitting.	3.1.19	VHS
3.2.1.6.1	The HRS shall allow a user to specify the location of bit slip errors on frame boundaries.	3.1.19	VHS
3.2.1.6.2	The HRS shall allow a user to specify the number of bits to be slipped up to 3 bits.	3.1.19	VHS
3.2.1.6.3	The HRS shall allow a user to specify the direction of bit slips.	3.1.19	VHS
3.2.2	The HRS shall output a predefined test data file serially.	3.1.12, 3.2.1, 3.2.2, 3.3.1, 3.3.2, 3.3.3	VHS, TRS
3.2.2.1	The VHS shall output the test pattern in a test data file through the serial port.	Derived	VHS, TRS

3.3.1	The HRS shall transfer data set files from its local storage to an external destination through the EBnet interface.	3.3.13, 3.3.1, 3.3.2, 4.9.1, 5.3.1, 5.3.2, 5.3.3	VHS
3.3.1.1	The VHS shall transfer data set files using FTP.	Derived	VHS
3.3.1.2	The VHS shall send an event message to the CDS upon completion of each file transfer with the following information: a) File name. b) Destination. c) PDS transfer start time and date. d) PDS transfer end time and date. e) Transfer session summary.	3.3.3	VHS
3.4.1	The HRS shall receive data set files from an external source through the EBnet interface.	3.1.13, 3.1.20, 4.9.1, 5.4.1, 5.4.2, 5.4.3	VHS
3.4.1.1	The VHS shall receive data set files using FTP.	Derived	VHS
3.4.1.2	The VHS shall store the data set files it receives in its local mass storage.	5.4.3	VHS
3.4.1.3	The VHS shall transfer received data set files to the CDS.	3.1.13	VHS, CDS
3.4.1.4	The VHS shall send an event message to the CDS upon receipt of a data set with the following information: a) File name. b) Source. c) Transfer start/end time. d) Size. e) Transmission status.	5.1.6	VHS
3.4.2	The HRS shall discard a data set if there is not enough space on the file system for storage.	Derived	VHS
3.5.1.1	The HRS shall receive return link data from an external source via ECL serial input port.	3.1.20,	VHS
3.5.1.2	The HRS shall support all service options for the CCSDS AOS VCDU service applicable to the EOS AM-1 mission, based on EOS AM ICD-106, Section 1.3.3	3.1.1, 3.3.1, 3.3.2, 4.8.1, 5.3.1	VHS

3.5.1.3	The HRS shall detect and synchronize on the following modes of data: a) Normal polarity, forward bit order. b) Inverted polarity, forward bit order.	Derived	VHS
3.5.1.4	The HRS shall correlate to sync patterns up to 32 bits in length.	Derived	VHS
3.5.1.5	The HRS shall perform search/check/lock/flywheel synchronization strategy with error tolerance between 0-7 bits. (NOTE: EDOS requires tolerance between 0-15 bits. EDOS also requires check/flywheel tolerance between 0-15 CADUs.)	Derived	VHS
3.5.1.6	The HRS shall invert the bits of each CADU detected to have inverted polarity.	Derived	VHS
3.5.1.7	The HRS shall correct bit slips, selectable between 0 and ± 3 bits, in a CADU, by truncating or padding to the proper length.	Derived	VHS
3.5.1.8	Upon detection of the first locked frame in a session, the VHS shall notify the CDS of synchronization. The number of search frames before achieving lock is user-selectable.	Derived	VHS
3.5.1.9	The HRS shall perform Reed-Solomon error detection and correction on each CVCDU.	Derived	VHS
3.5.1.10	The HRS shall discard, as a user option, any VCDU that failed Reed-Solomon decoding.	Derived	VHS
3.5.1.11	The HRS shall store VCDUs that failed Reed-Solomon decoding.	Derived	VHS
3.5.1.12	The HRS shall discard and account for fill VCDUs.	Derived	VHS
3.5.1.13	The HRS shall time-stamp each VCDU.	3.1.15	VHS
3.5.1.14	The HRS shall maintain the following counts as CADU processing quality and accounting information for each processing session: a) CADUs received. b) Back-to-search CADUs. c) Search CADUs. d) Check CADUs. e) Lock CADUs. f) Flywheel CADUs. g) Forward inverted CADUs. h) Forward true CADUs. i) CADUs with bit slip detected. j) CADUs with errors in sync pattern. k) Fill CADUs.	3.3.3	VHS

3.5.1.15	The HRS shall allow a user to zero CADU processing quality and accounting counts.	5.1.7, 5.1.8	VHS, CDS
3.5.1.16	The HRS shall maintain the following frame Reed-Solomon quality and accounting information for each processing session: <ul style="list-style-type: none"> a) Operation mode. b) Interleave depth. c) Codeword length. d) CADUs output without Reed-Solomon processing. e) Uncorrectable and filled CADUs filtered. f) CADUs output. g) CADUs with header error counts. h) CADUs with headers corrected. i) CADUs with uncorrectable headers. j) Reed-Solomon codeword error counts. k) Codewords corrected. l) Uncorrectable codewords. m) CADUs with corrections. n) Uncorrectable CADUs. 	3.3.3	VHS
3.5.1.17	The HRS shall allow a user to zero Reed-Solomon processing quality and accounting counts.	5.1.7, 5.1.8	VHS, CDS
3.5.1.18	The HRS shall maintain counts for each VC, including as a minimum: <ul style="list-style-type: none"> a) VCDUs received. b) VCDUs with correctable Reed-Solomon errors. c) Reed-Solomon symbols corrected. d) VCDUs with sequence errors. e) Missing VCDUs. 	3.3.3	VHS
3.5.1.19	The HRS shall allow a user to zero VC data quality counts.	5.1.7, 5.1.8	VHS, CDS
3.5.1.20	The HRS shall reject a VCDU if it contains a header error, or a packet piece that can not be processed because of the detected error. However, all packets before the erroneous packet piece shall be processed and output.	Derived	VHS

3.5.1.21	The HRS shall reject a VCDU that contains any of the following errors: a) Invalid SCID. b) Invalid VCID. c) Invalid first header pointer. d) Invalid APID. e) Packet length error. f) No packet header (including secondary header) found for a piece in the VCDU. g) Reed-Solomon uncorrectable errors.	Derived	VHS
3.5.1.22	The HRS shall generate annotation for each rejected VCDU with the following information as a minimum: a) Frame synchronization and Reed-Solomon decoding quality trailers. b) Byte location of detected error in the VCDU. c) Flags defining error condition(s).	3.1.5	VHS
3.5.1.23	The HRS shall keep all rejected VCDUs of a session in a reject data file in the order received. This file shall be available to the user post-pass as a reject VCDU data set.	Derived	VHS, CDS
3.5.1.24	The HRS shall identify VCDUs of VCs that request AOS VCDU service.	Derived	VHS
3.5.1.25	The HRS shall group identified VCDUs by VC in VCDU data sets in the order received.	Derived	VHS
3.5.1.26	The HRS shall store all captured CADUs during a test session after frame synchronization and before Reed-Solomon decoding. No header or trailer shall be added.	Derived	VHS
3.5.1.27	The HRS shall store all CADUs available after Reed-Solomon decoding during a test session, prior to packet-level processing. Quality annotation shall be included.	Derived	VHS
3.5.1.28	The HRS shall transfer stored CADUs to the CDS or a user-specified destination through its network interface.	3.1.23, 3.2.1, 5.4.3	VHS, CDS
3.5.2.1	The HRS shall perform processing in support of CCSDS AOS path service for return link data.	3.1.1	VHS
3.5.2.2	The HRS shall support the packet service option for CCSDS AOS path service.	3.1.1	VHS
3.5.2.3	The HRS shall identify independent sources by any combination of SCID, VCID, and APID.	3.1.6, 3.1.7	VHS
3.5.2.4	The HRS shall extract packet pieces from VCDUs requesting AOS path service and reassemble packets.	Derived	VHS
3.5.2.5	The HRS shall process source packets with variable length.	Derived	VHS

3.5.2.6	The HRS shall generate fill data using a selectable fill pattern to complete packet fragments that have complete packet headers, corresponding to the packet length defined by the valid packet header.	3.1.8	VHS
3.5.2.7	The HRS shall discard and account for fill packets.	3.1.7, 5.1.6	VHS
3.5.2.8	The HRS shall discard any packet fragment without an associated valid header.	3.1.6, 3.1.7	VHS
3.5.2.9	The HRS shall check for packet sequence discontinuities and determine the minimum number of packets missing in sequence gaps.	3.1.6, 3.1.7	VHS
3.5.2.10	The HRS shall generate an ESH, as specified in the EDOS External ICD Data Format Control Document listed in Section 1.3.3, for every packet as follows: a) ESH version number. b) Time and date of processing. c) Source VCDU-ID, copied from the VCDU. d) Corrected source VCDU indicator. e) Error-free source VCDU indicator. f) Source VCDU sequence counter discontinuity. g) Indicator that VCDU is playback data, based on replay flag in the primary header. h) Packet source sequence counter discontinuity indicator (not applicable for VCDU service). i) Location of first byte of EDOS-generated fill data for a packet (not applicable for VCDU service). j) Recovery processing indicator. k) Test data indicator.	3.1.20	VHS
3.5.2.11	The HRS shall generate an EDU by concatenating an ESH with each return link VCDU.	3.1.20	VHS
3.5.2.12	The HRS shall generate an EDU by concatenating an ESH with each return link packet.	3.1.20	VHS
3.5.2.13	The HRS shall identify forward-ordered playback VCDU data based on the replay flag in the VCDU primary header.	3.1.6, 3.1.7	VHS
3.5.2.14	The HRS shall identify packets for quicklook processing based on source ID as defined by a user.	3.1.6, 3.1.7	VHS
3.5.2.15	The HRS shall identify packets for quicklook processing based on the quicklook flag in the packet secondary header.	3.1.6, 3.1.7	VHS
3.5.2.16	The HRS shall store all EDUs as specified in the ETS HRS User's Guide (TBD), as listed in Section 1.3.3.	3.1.23	VHS
3.5.2.17	The HRS shall transfer stored EDUs by source and session to user-specified destinations upon request.	3.1.23, 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS, CDS

3.5.2.18	<p>The HRS shall maintain the following cumulative packet processing quality and accounting records by VC and source for each test session:</p> <p>By Virtual Channel:</p> <ul style="list-style-type: none"> a) SCID. b) Number of VCDUs processed. c) Number of VCDUs rejected. d) Number of APIDs. e) Number of packets processed. f) Number of VCDU sequence discontinuities. g) Number of missing VCDUs. h) Number of invalid application packets rejected. i) Number of Reed-Solomon uncorrectable VCDUs. j) Number of Reed-Solomon corrected VCDUs. k) Number of packet source sequence counter discontinuities. l) Number of missing packets. m) Number of packets with fill. <p>By Source:</p> <ul style="list-style-type: none"> a) SCID. b) APID. c) Spacecraft time of first packet. d) Spacecraft time of last packet. e) Number of packets processed. f) Number of length errors. g) Number of packets from VCDUs with errors. h) Number of packets with fill. i) Number of packet source sequence counter discontinuities. j) Number of missing packets. k) Number of EDUs output during the current session. 	3.3.3	VHS
3.5.2.19	The HRS shall zero cumulative packet processing quality and accounting records.	Derived	VHS, CDS
3.5.2.20	The HRS shall output EDUs from user-selected sources as they become available during a session.	3.1.23, 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS, CDS

3.5.2.21	The HRS shall output EDUs in the same order as received.	3.1.23, 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS
3.5.2.22	The HRS shall retain EDUs that are output for further data set processing on the system disk storage until the next test session, or on removable media for a user-specified time.	3.1.23, 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS, CDS
3.5.2.23	The HRS shall transfer EDUs to a single user-specified destination.	3.1.23, 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS
3.5.3.1	The HRS shall sort packets by source, specified by users and identified by the SCID, VCID, and APID, to construct data sets. In general, a source may correspond to an onboard experiment with certain attributes (e.g., real time, playback, quicklook).	3.1.1	VHS
3.5.3.2	The HRS shall sort in forward time order packets based on CCSDS binary timecode in the packet secondary header.	3.1.15	VHS
3.5.3.3	The HRS shall generate data sets by sensor, specified by users, and may consist of one or more sources. In general, a sensor will correspond to an onboard experiment identified by SCID and APID.	3.1.20, 3.3.1, 3.3.2	VHS
3.5.3.4	<p>The HRS shall perform the following functions for production data processing as specified in the EDOS Functional and Performance Specifications Document listed in Section 1.3.3:</p> <ul style="list-style-type: none"> a) Order packets by source sequence count. b) Resolve source sequence count order ambiguities using packet secondary header time values. c) Provide packet quality indicator (yes vs. no) for all packets with the following quality conditions, as a minimum: <ul style="list-style-type: none"> 1. Discrepancies between packet header length field and actual packet length. 2. Packet source sequence count discontinuity. 3. Packet contains fill. 4. Playback indicator. d) Identify identical packets by comparing APID, packet source sequence count, and packet secondary header time values. 	3.1.20, 3.3.1, 5.3.1	VHS

	<ul style="list-style-type: none"> e) Exclude redundant packets when more than one copy of a packet exists by using the best quality packet as determined by the packet quality indicator. f) Merge data from multiple sessions. g) Exclude ESH for all packets. h) Include a PDS construction record with each PDS. i) Include a PDS end record with each PDS. 		
3.5.3.5	<p>The HRS shall generate a PDS construction record for each PDS as follows:</p> <ul style="list-style-type: none"> a) Software version number. b) PDS ID. c) APID associated with the PDS. d) SCID and VCID associated with the PDS. 	3.3.1, 5.3.1	VHS, CDS
	<ul style="list-style-type: none"> e) List of scheduled session start time for the PDS. f) List of scheduled session stop time for the PDS. g) Time of PDS completion. h) ESH time and date annotation of first packet in the PDS. i) ESH time and date annotation of last packet in the PDS. j) List of packet gaps for the PDS. k) Count of packets from VCDUs with errors corrected by Reed-Solomon decoding. l) Count of packets in the PDS. m) PDS size in bytes excluding the construction and end records. n) List of packets containing fill with location of first fill byte for each packet. o) Count of packets with fill. p) Count of total fill bytes. q) Count of packets with length errors. r) Count of packets with sequence count discontinuities. s) Count of playback packets. t) CCSDS binary timecode of first packet. u) CCSDS binary timecode of last packet. v) Packet source sequence count of first packet. w) Packet source sequence count of last packet. 		

3.5.3.6	<p>The HRS shall construct PDSs containing packets of a single sensor according to the following selectable attributes:</p> <ul style="list-style-type: none"> a) Fixed number of packets per PDS directed by storage size. b) All packets from a single session. c) All packets from a specified number of sessions within a 24-hour time period. d) All packets from a specified spacecraft time period up to 25 Gbytes in size. 	3.3.1, 5.3.1	VHS
3.5.3.7	The HRS shall recover, upon restoration of functionality following a failure, all production data processing performed prior to the failure.	Derived	VHS, CDS
3.5.3.8	The HRS shall perform quicklook processing by constructing EDSs for packets selected from a single source during a single session.	3.1.23, 3.3.2, 5.3.2	VHS, CDS
3.5.3.8.1	The HRS shall select packets for quicklook data processing by user-specified sources, or packet secondary header quicklook flag.	3.1.23, 3.3.2, 5.3.2	VHS, CDS
3.5.3.9	<p>The HRS shall perform the following functions for quicklook data processing as specified in the EDOS Functional and Performance Specifications Document listed in Section 1.3.3:</p> <ul style="list-style-type: none"> a) Order packets by source sequence count. b) Resolve source sequence count order ambiguities using packet time values, if available. c) Provide packet quality indicator for all packets with any of the following quality conditions, as a minimum: <ul style="list-style-type: none"> 1. Discrepancies between packet header length field and actual packet length. 2. Packet source sequence count discontinuity. 3. Packet contains fill. 4. Playback indicator. d) Exclude ESH for all packets. 	3.3.2, 5.3.2	VHS
	e) Include an EDS construction record with each EDS.		
	f) Include an EDS end record with each EDS.		

3.5.3.10	<p>The HRS shall generate an EDS construction record for each EDS as specified in the EDOS Functional and Performance Specifications Document listed in Section 1.3.3 as follows:</p> <ul style="list-style-type: none"> a) Software version number. b) EDS ID. c) APID associated with the EDS. d) SCID and VCID associated with the EDS. e) Scheduled session start time for the EDS. f) Scheduled session stop time for the EDS. g) Time of EDS completion. h) ESH time and date annotation of first packet in the EDS. i) ESH time and date annotation of last packet in the EDS. j) List of packet gaps for the EDS (not applicable for packet secondary header quicklook flagged data). k) Count of packets from VCDUs with errors corrected by Reed-Solomon decoding. l) Count of packets in the EDS. m) EDS size in bytes excluding construction and end records. n) List of packets containing fill with location of first fill byte for each packet. o) Count of packets with fill. p) Count of total fill bytes. q) Count of packets with length errors. r) Count of packets with sequence count discontinuities. s) Count of playback packets. t) CCSDS binary timecode of first packet. u) CCSDS binary timecode of last packet. v) Packet source sequence count of first packet. w) Packet source sequence count of last packet. 	3.3.2, 5.3.2	VHS, CDS
3.5.3.11	The HRS shall retain packets used in quicklook data processing for immediate post-pass production processing, or if requested offline for later production processing.	3.3.1, 5.3.1	VHS, CDS
3.5.3.12	The HRS shall support the CCSDS binary timecode applicable to the EOS AM-1 mission.	3.1.15, 3.1.18	VHS

3.5.3.13	The HRS shall transfer generated PDSs and EDSs to user-specified destinations.	3.3.1, 3.3.2 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS, CDS
3.5.3.14	The HRS shall generate a PDS delivery record for each PDS transferred as follows: a) PDS ID. b) Destination ID. c) PDS transfer start time and date. d) PDS transfer end time and date. e) Summary of size, quality, and accounting status of PDS transferred during the session.	3.3.2, 4.8.1, 4.9.1, 5.3.2	VHS, CDS
3.5.3.15	The HRS shall generate an EDS delivery record for each EDS transferred as follows: a) EDS ID.	3.3.1, 4.8.1, 4.9.1,	VHS, CDS
	b) Destination ID. c) EDS transfer start time and date. d) EDS transfer end time and date. e) Summary of size, quality, and accounting status of EDS transferred during the session	5.3.1	
3.5.3.16	The HRS shall transfer data set (PDS and EDS) delivery records to the CDS.	3.3.1, 3.3.2 4.8.1, 4.9.1, 5.3.1, 5.3.2	VHS, CDS
3.5.3.17	The HRS shall store EDSs and PDSs up to a maximum of 12 Gbytes until the next test session, or offline for a user-specified period of time.	3.1.23, 5.1.3, 5.4.3	VHS, CDS
3.5.3.18	The HRS shall overwrite EDSs and PDSs stored in its local storage when there is no space for new data. Data received first will be overwritten first on a session basis. Optionally, data will be protected by offloading the same to user-provided physical media.	Derived	VHS, CDS
3.6.1	The HRS shall read Ampex 19mm cartridge tapes.	3.1.2	TRS
3.6.2	The HRS shall provide a clock signal programmable to variable rates, from 1 kHz to 150 MHz.	Derived	TRS
3.6.3	The HRS shall provide control of the TRS through the CDS.	3.1.23	TRS, CDS
3.6.4	The HRS shall provide the Ampex tape drive and clock generator status to the user, as specified in the ETS HRS User's Guide (TBD) listed in Section 1.3.3.	Derived	TRS

3.6.5	The HRS shall provide CUE command to position a tape to a user-specified record.	Derived	TRS
3.6.6	The HRS shall provide PLAY command to read data back from tape with user-specified tape position and clock rate.	Derived	TRS
3.6.7	The HRS shall provide RECORD command to record data onto tape with user-specified tape position and block size.	Derived	TRS
3.6.8	The HRS shall provide STOP command to allow users to stop current playback or recording functions.	Derived	TRS
3.6.9	The HRS shall provide EJECT command to eject the tape from the Ampex tape drive.	Derived	TRS
3.6.10	The HRS shall provide REWIND command to rewind a tape.	Derived	TRS
4.1.1	The HRS shall support a mass storage of 30 Gbytes.	Derived	VHS
4.1.2	The HRS shall complete a warm boot in no more than 10 minutes.	5.1.9	VHS, TRS, CDS
4.1.3	The HRS shall complete a cold boot in no more than 15 minutes.	5.1.9	VHS, TRS, CDS
4.1.4	The HRS shall have a Mean Time Between Failure (MTBF) of at least 706 hours.	5.1.1	HRS
4.1.5	The HRS shall have a Mean Time To Restore (MTTR) of at most 8 hours.	5.1.2	HRS
4.1.6	The HRS shall transfer test data files at data rates up to 8 Mbps via external interfaces.	Derived	TRS, CDS
4.1.7	The HRS shall transfer a test data file with a size up to 2 Gbytes.	5.3.3	VHS, CDS
4.2.1	The HRS shall transmit test data through the serial port at rates up to 150 Mbps.	5.2.1	VHS, TRS
4.2.2	The HRS shall allow a user to select data output rates from 0 to 150 Mbps with 1-kbps resolution.	5.2.1	CDS, TRS, VHS
4.2.3	The HRS shall update a test pattern at data rates up to 30 Mbps.	Derived	VHS
4.2.4	The HRS shall output a single test data file with a size up to 25 Gbytes.	Derived	VHS, TRS
4.3.1	The HRS shall transfer a user-specified data set file consisting of one or more files, each with a maximum size of 2 Gbytes, up to a total size of 25 Gbytes.	5.3.1	VHS
4.3.2	The HRS shall transmit a data set to a user-specified destination through the EBnet interface at sustained data rates up to 34 Mbps.	5.3.1, 5.3.2	VHS
4.4.1	The HRS shall receive a data set from an external source through the EBnet interface at sustained data rates up to 34 Mbps.	5.4.1, 5.4.2	VHS

4.4.2	The HRS shall receive data set files, each up to a maximum of 2 Gbytes in size, and store a maximum of 25 Gbytes.	5.4.3	VHS
4.5.1	The HRS shall perform CADU-level return link processing for statistics only at sustained data rates up to 150 Mbps.	Derived	VHS
4.5.2	The HRS shall perform packet-level return link processing at sustained data rates up to 30 Mbps over a duration of 50 minutes.	Derived	VHS
4.5.3	The HRS shall perform path service processing at a rate of up to 10,000 packets per second at 30 Mbps over a duration of 50 minutes.	Derived	VHS
4.5.4	The HRS shall transfer EDUs from user-selected low-rate sources to user-specified destinations at rates up to 3 Mbps.	Derived	CDS, VHS
4.5.5	The HRS shall allow a user to specify output data rates at 1-kbps intervals from 0 to 150 Mbps.	5.5.1, 5.5.2, 5.5.3	CDS, VHS, TRS
4.5.5.1	The HRS VHS shall output test data serially at user-specified data rates up to 150 Mbps.	5.5.1, 5.5.2, 5.5.3	VHS, TRS
4.5.6	The HRS shall complete the generation of an EDS after receipt of all return link data comprising the EDS.	Derived	VHS
4.5.7	The HRS shall degrade no more than 1 data set out of 10,000 data sets processed. (A data set, PDS, or EDS is degraded when one or more packet is missing or unprocessable due to HRS processing error.)	Derived	VHS
4.5.8	The HRS shall support the CCSDS binary timecode in the packet secondary header with a resolution of 1 microsecond.	3.1.1, 3.1.15	VHS
4.5.9	The HRS shall initiate the transfer of a stored data set, PDS, or EDS following construction of the data set.	Derived	VHS, TPCE
4.5.10	The HRS shall initiate the transfer of a data set (PDS or EDS) delivery record within 5 seconds following delivery of the data set.	Derived	VHS, TPCE
4.5.11	The HRS shall process packets at rates up to 21 packets per any 64 Kbits at 30 Mbps.	Derived	VHS
4.5.12	The HRS shall deliver packets from user-selected sources within 10 ms (at a 30-Mbps ingest rate) of receipt.	Derived	VHS
4.5.13	The HRS shall process a single input stream.	Derived	VHS
4.5.14	The HRS shall process up to 8 spacecraft IDs with ID values from 0 to 255.	Derived	VHS
4.5.15	The HRS shall process up to 31 VCs with ID values from 0 to 63.	Derived	VHS
4.5.16	The HRS shall process up to 512 application processes with ID values from 0 to 2046.	Derived	VHS
4.5.17	The HRS shall process up to 512 sources.	Derived	VHS

4.5.18	The HRS shall process up to 60 data segments per source per session.	Derived	VHS
4.6.1	The HRS shall play back Ampex tapes at rates up to 100 Mbps.	3.1.2	TRS
4.6.2	The HRS shall provide a clock signal at rates up to 100 MHz.	3.2.2	TRS
4.6.3	The HRS shall allow users to specify clock rate at an interval of 1 MHz.	Derived	TRS, CDS
4.6.4	The HRS shall record data on Ampex tapes at rates up to 100 Mbps.	Derived	TRS
4.6.5	The HRS shall store up to 48 Gbytes of data on a single tape cartridge.	Derived	TRS
5.1.1.1	The GUI shall conform to a set of display style guidelines that comply with the X-Windows/OSF Motif Style Guide, Version 2.	Derived	TPCE
5.1.1.2	The GUI shall provide interaction to enable the user to exercise all functions associated with the ETS HRS.	Derived	TPCE
5.1.1.3	The GUI shall display a TBD icon response to a user command within 2 seconds of command entry.		TPCE
5.1.1.4	The GUI shall display a TBD icon response regarding execution of a command within 5 seconds of command entry.		TPCE
5.1.1.5	The GUI shall provide windows allowing the user to view status at system and subsystem levels, as specified in the ETS VHS User's Guide (TBD) listed in Section 1.3.3.	Derived	TPCE
5.1.1.6	The GUI shall provide interaction to enable the user to configure the HRS to simulate TGT high-rate return link output.	Derived	TPCE
5.1.1.7	The GUI shall provide interaction to enable the user to configure the HRS to simulate EDOS data set output to a DAAC.	Derived	TPCE
5.1.1.8	The GUI shall provide interaction to enable the user to configure the HRS to simulate a DAAC front-end for receiving data sets.	Derived	TPCE
5.1.1.9	The GUI shall provide interaction to enable the user to configure the HRS to generate EDOS data sets from spacecraft test data.	Derived	TPCE
5.1.1.10	The GUI shall provide interaction to enable the user to configure the HRS to generate CADU files from spacecraft test data.	Derived	TPCE
5.1.1.11	The GUI shall provide interaction to enable the user to control tape operations.	Derived	TPCE
5.1.2.1	The GUI shall provide interaction to enable the user to view the activity schedule.	Derived	TPCE
5.1.2.2	The GUI shall provide interaction to enable the user to add new session events to the activity schedule.	Derived	TPCE

5.1.2.3	The GUI shall allow users to delete selected session events from the activity schedule.	Derived	TPCE
5.1.2.4	The GUI shall allow users to modify selected session events from the activity schedule.	Derived	TPCE
5.1.3.1	The GUI shall provide interaction to enable the user to create a new configuration set.	Derived	TPCE
5.1.3.2	The GUI shall provide interaction to enable the user to delete a configuration set.	Derived	TPCE
5.1.3.3	The GUI shall provide interaction to enable the user to view a configuration set.	5.1.5	TPCE
5.1.3.4	The GUI shall allow the operator to edit configuration sets that define the operation to be performed by the VHS.	Derived	TPCE
5.1.3.5	The GUI shall provide interaction to enable the user to associate a configuration data set to a session event.	Derived	TPCE
5.1.4.1	Status, data quality, and accounting information, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3, shall be available to users, with displays of status for ongoing sessions updating continuously at least once every 5 seconds.	3.1.12, 5.1.6, 5.1.7	TPCE
5.1.4.2	The GUI shall provide interaction to enable the user to view current system-level session status, including errors.	3.1.12	TPCE
5.1.4.3	The GUI shall provide window(s) allowing the user to view current subsystem-level session status (e.g., Frame Sync, Reed-Solomon, Simulator), as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.12	TPCE
5.1.4.4	The GUI shall provide window(s) allowing the user to view status of available ports on the Ethernet LAN, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.12	TPCE
5.1.4.5	The GUI shall provide window(s) allowing the user to view status of available ports on the FDDI LAN, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.12	TPCE
5.1.4.6	The GUI shall provide window(s) allowing the user to view status of available ports on the EBnet, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.12	TPCE
5.1.4.7	The GUI shall generate session reports summarizing session system status, quality, and accounting information, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.3.3	TPCE
5.1.4.8	The GUI shall provide window(s) allowing the user to view session reports, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	Derived	TPCE
5.1.5.1	The GUI shall log user directives in the session journal, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.16	TPCE

5.1.5.2	The GUI shall log all commands sent to the VHS in the session journal, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.16	TPCE
5.1.5.3	The GUI shall log all command responses received from the VHS in the session journal.	3.1.16	TPCE
5.1.5.4	The GUI shall log all system events received from the VHS in the session journal.	3.1.16	TPCE
5.1.5.5	The GUI shall log all system errors produced and detected in the session journal.	3.1.16	TPCE
5.1.5.6	The GUI shall time-tag the start of each system event to the nearest minute.	3.1.16	TPCE
5.1.5.7	The GUI shall provide a window for the user to view the session journal.	3.1.16	TPCE
5.1.5.8	The GUI shall provide a window for the user to annotate individual events in the session journal.	3.1.16	TPCE
5.1.5.9	The GUI shall provide interaction to enable the user to print the session journal.	3.1.16	TPCE
5.1.6.1	The GUI shall provide window(s) allowing user to run card-level diagnostic tests, subsystem-level self-tests, and system-level self-tests.	Derived	TPCE
5.1.6.2	The GUI shall provide interaction to enable the user to view results of selected card(s) or subsystem(s) self-tests.	Derived	TPCE
5.1.6.3	The GUI shall provide a capability allowing the user to log the results of selected card(s) or subsystem(s) self-tests.	Derived	TPCE
5.1.6.4	The GUI shall provide a capability allowing the user to annotate self-test results.	Derived	TPCE
5.1.6.5	The GUI shall provide a capability allowing the user to view selected self-test results in the self-test log.	Derived	TPCE
5.2.1	The GUI shall provide interaction to enable the user to select data channel(s).	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.2	The GUI shall provide interaction to enable the user to select test data files to be transmitted on each data channel.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.3	The GUI shall provide interaction to enable the user to select a test data rate on each data channel.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.4	The GUI shall provide interaction to enable the user to select a number of repetitions, if desired, on one data channel.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE

5.2.5	The GUI shall provide interaction to enable the user to specify bit slip error insertion on one data channel, including location and number of bits.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.6	The GUI shall provide interaction to enable the user to activate the transmission of test data on each channel independently.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.7	The GUI shall provide interaction to enable the user to monitor data transmission status, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.2.8	The GUI shall notify the user when the simulation session terminates.	3.1.12, 3.5.20, 5.1.5, 5.1.6,	TPCE
5.3.1	The GUI shall provide interaction to enable the user to select a data destination.	5.3.1, 5.3.2, Derived	TPCE
5.3.2	The GUI shall provide interaction to enable the user to select the data set files to be transferred.	5.3.1, 5.3.2, Derived	TPCE
5.3.3	The GUI shall provide interaction to enable the user to specify a sequence in which data set files will be transferred.	5.3.1, 5.3.2, Derived	TPCE
5.3.4	The GUI shall provide interaction to enable the user to activate the transfer of data set files.	5.3.1, 5.3.2, Derived	TPCE
5.3.5	The GUI shall provide interaction to enable the user to monitor data set transfer status, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3, including average data transfer rates.	5.3.1, 5.3.2, Derived	TPCE
5.3.6	The GUI shall notify the user when the data set transfer session terminates.	5.3.1, 5.3.2, Derived	TPCE
5.3.7	The GUI shall generate reports summarizing status of the data set transfer session, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	5.3.1, 5.3.2, Derived	TPCE
5.4.1	The GUI shall provide interaction to enable the user to configure the system to be ready to receive EDOS data sets.	5.4.1, 5.4.2, Derived	TPCE
5.4.2	The GUI shall notify the user of the start of a data set transfer session to the VHS.	5.4.1, 5.4.2, Derived	TPCE
5.4.3	The GUI shall notify the user via the CDS when a data set is completely received at the VHS.	5.4.1, 5.4.2, Derived	TPCE

5.4.4	The GUI shall provide interaction to enable the user to view the directory of data sets received and stored on the VHS file system.	5.4.1, 5.4.2, Derived	TPCE
5.4.5	The GUI shall provide interaction to enable the user to select the received data sets and transfer them from the VHS to the CDS .	5.4.1, 5.4.2, Derived	TPCE
5.4.6	The GUI shall provide interaction to enable the user to delete data sets from the VHS file system.	5.4.1, 5.4.2, Derived	TPCE
5.4.7	The GUI shall notify the user via the CDS when a data set transfer to the VHS failed because of insufficient space in the VHS file system.	5.4.1, 5.4.2, Derived	TPCE
5.5.1	The GUI shall provide interaction to enable the user to specify spacecraft test data files to be used.	3.5.20	TPCE
5.5.2	The GUI shall provide interaction to enable the user to specify end data products (i.e., CADU files, EDSs, and PDSs).	3.1.20, 5.2.1, 5.3.1, 5.3.2, 5.3.3	TPCE
5.5.3	The GUI shall provide interaction to enable the user to specify processing parameters through user-generated ASCII files referred to as configuration sets.	Derived	TPCE
5.5.4	The GUI shall provide interaction to enable the user to configure the VHS with the user-selected configuration set.	Derived	TPCE
5.5.5	The GUI shall provide interaction to enable the user to play back the user-selected spacecraft test data file.	Derived	TPCE
5.5.6	The GUI shall provide interaction to enable the user to monitor data processing status.	3.1.12	TPCE
5.5.7	The GUI shall notify the user via the CDS when the processing session terminates.	5.1.5	TPCE
5.5.8	The GUI shall notify the user via the CDS as and when each CADU file, EDS, or PDS is generated by the VHS.	5.1.5, 5.1.6	TPCE
5.5.9	The GUI shall provide interaction to enable the user to view the directory of data sets available for transmission from the VHS.	Derived	TPCE
5.5.10	The GUI shall generate a session status report, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3, summarizing processing session status.	3.3.3, 5.1.6	TPCE
5.5.11	The GUI shall provide interaction to enable the user to transfer the CADU file, EDS, and PDS to a user-specified destination via network.	5.2.1, 5.3.1, 5.3.2	TPCE
5.6.1	The GUI shall provide window(s) allowing the user to exercise the functions associated with the Ampex tape drive, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.1.2, 3.1.13, Derived	TRS

5.6.2	The GUI shall provide window(s) allowing the user to exercise the functions associated with the clock generator, as specified in the VHS User's Guide (TBD) listed in Section 1.3.3.	3.2.2, Derived	TRS
5.6.3	The GUI shall provide interaction to enable the user to select spacecraft test data for playback by a tape ID and pseudo file names associated with corresponding record numbers.	3.1.2, 3.1.13, Derived	TRS
5.6.4	The GUI shall provide interaction to enable the user to play back selected spacecraft test data.	3.1.2, 3.1.13, Derived	TRS
5.6.5	The GUI shall provide interaction to enable the user to specify a clock rate at which the selected spacecraft test data is played back.	3.2.2, Derived	TRS

10.2 ETS F&P REQUIREMENTS TRACED TO ETS SRS

F&P Req. Number	Requirement Description	ETS HRS Vol. 3 Req. Number
3.1.1	ETS shall conform to CCSDS Recommendations for Space Data System Standards applicable to the EOSDIS project as specified in this document.	3.2.1.4, 3.2.1.5, 3.5.1.2, 3.5.2.1, 3.5.2.2, 3.5.3.1, 4.5.7,
3.1.2	ETS shall provide the capability to accept and distribute test data by electronic transmission and physical media.	2.3.1.1, 3.1.1, 3.6.1, 4.6.1, 5.6.1, 5.6.3, 5.6.4,
3.1.6	ETS shall provide the capability to read and interpret flags in all headers.	3.1.2, 3.1.2.2, 3.2.1.4, 3.2.1.5, 3.5.2.3, 3.5.2.8, 3.5.2.9, 3.5.2.13, 3.5.2.14, 3.5.2.15,
3.1.7	ETS shall provide the capability to validate all headers of received data.	3.1.2.2, 3.2.1.4, 3.2.1.5, 3.5.2.3, 3.5.2.8, 3.5.2.9, 3.5.2.13, 3.5.2.14, 3.5.2.15
3.1.8	ETS shall provide the capability to generate static test data, dynamic test data, or a combination of both.	3.2.1, 3.2.1.1, 3.2.1.2,
3.1.12	ETS shall provide the capability to monitor and display ETS system configuration and test status.	3.1.2.2, 3.1.2.2.2, 3.2.2, 5.1.3.1, 5.1.3.2, 5.1.3.3, 5.1.3.4, 5.1.3.5, 5.1.3.6, 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6, 5.2.7, 5.2.8, 5.5.6,
3.1.13	ETS shall provide dumps of received or generated test data on electronic and physical media.	2.3.1.1, 3.3.1, 3.4.1, 3.4.1.3, 4.5.7, 5.6.1, 5.6.3, 5.6.4,
3.1.15	ETS shall use the Universal Time Coordinated (UTC) format for time-of-day-related data, as required by the test.	3.5.1.13, 3.5.1.2.2, 3.5.3.2, 4.5.7,

3.1.16	ETS shall provide the capability to log operator dialog to maintain an operations audit trail, as specified in the test script.	3.1.2, 5.1.4.1, 5.1.4.2, 5.1.4.3, 5.1.4.4, 5.1.4.5, 5.1.4.6, 5.1.4.7, 5.1.4.8, 5.1.4.9,
3.1.18	ETS shall provide the capability to receive, store, and use the AM-1 spacecraft data base.	3.2.1.3, 3.5.2.6, 3.5.3.12,
3.1.19	ETS shall provide the capability to insert errors in data from spacecraft- and instrument-generated tapes.	3.2.1.6, 3.2.1.6.1, 3.2.1.6.2, 3.2.1.6.3, 3.2.1.6.4,
3.1.20	ETS shall provide the capability to generate PDSs and EDSs from spacecraft- and instrument-generated tapes.	3.2.1, 3.2.1.1, 3.2.1.2, 3.4.1, 3.5.1.1, 3.5.2.10, 3.5.2.11, 3.5.2.12, 3.5.3.3, 3.5.3.4, 5.5.2,
3.1.23	ETS shall provide the capability to store transmitted and received test data as directed by operator control.	3.1.1.3, 3.1.1.4, 3.5.1.28, 3.5.2.16, 3.5.2.17, 3.5.2.20, 3.5.2.21, 3.5.2.22, 3.5.2.23, 3.5.3.8, 3.5.3.17, 3.6.3,
3.2.1	ETS shall provide the capability to generate and transmit a TGT high-rate return link in the form of CADUs.	3.2.1, 3.2.1.1, 3.2.1.2, 3.2.2, 3.5.1.28,
3.2.2	ETS shall provide the capability to generate and transmit the TGT return link clock.	3.2.2, 4.6.2, 5.6.2, 5.6.5,
3.3.1	ETS shall provide the capability to generate and transmit PDSs using ETS-generated test data.	3.2.1, 3.2.1.1, 3.2.1.2, 3.2.2, 3.3.1, 3.5.1.2, 3.5.3.3, 3.5.3.4, 3.5.3.5, 3.5.3.6, 3.5.3.11, 3.5.3.13, 3.5.3.15, 3.5.3.16,
3.3.2	ETS shall provide the capability to generate and transmit EDSs using ETS-generated test data.	3.2.1, 3.2.1.1, 3.2.1.2, 3.2.2, 3.3.1, 3.5.1.2, 3.5.3.3, 3.5.3.8, 3.5.3.8.1, 3.5.3.9, 3.5.3.10, 3.5.3.13, 3.5.3.14, 3.5.3.16,
3.3.3	ETS shall provide the capability to generate summary status, quality, and accounting data reporting of EDOS services for processing of ETS-generated data.	3.1.3, 3.2.2, 3.3.1.2, 3.3.1.3, 3.3.1.4, 3.5.1.14, 3.5.1.16, 3.5.1.18, 3.5.2.18, 5.1.3.7, 5.5.10,
3.4.1	ETS shall provide the capability to receive PDSs through EBnet.	3.1.1.4
3.4.2	ETS shall provide the capability to receive EDSs through EBnet.	3.1.1.4
3.5.20	ETS shall provide the operator the real-time capability of enabling/disabling any element of the command validation process, entering spacecraft commands, controlling transmission of telemetry, and including a limited number of telemetry errors.	3.1.2, 3.1.2.1, 3.1.2.2.1, 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6, 5.2.7, 5.2.8, 5.5.1

4.1.1	ETS equipment shall operate in an operational ambient temperature range of 50 degrees to 80 degrees Fahrenheit, and ambient humidity range of 20 percent to 70 percent (noncondensing).	
4.1.2	ETS equipment shall require a physical space of no more than 10 feet by 20 feet.	
4.1.3	ETS equipment shall require approximately 10 kilowatts (± 10 percent) for power.	
4.8.1	ETS interface to the EDOS when ETS is simulating a DAAC shall comply with interface formats and protocols specified for DAACs in the EDOS to EGS Interface Requirements Document.	2.2.2.1, 2.2.2.2, 2.2.2.3, 3.5.1.2, 3.5.3.13, 3.5.3.14, 3.5.3.15, 3.5.3.16, 3.5.2.17, 3.5.2.20, 3.5.2.21, 3.5.2.22, 3.5.2.23,
4.9.1	ETS interface with EBnet shall comply with the interface formats and protocols specified in the EBnet to ETS Interface Requirements Document.	2.2.2.1, 2.2.2.2, 2.2.2.3, 3.3.1, 3.4.1, 3.5.3.13, 3.5.3.14, 3.5.3.15, 3.5.3.16, 3.5.2.17, 3.5.2.20, 3.5.2.21, 3.5.2.22, 3.5.2.23,
5.1.1	ETS hardware configured for actual testing shall have an MTF greater than 396 hours.	4.1.4,
5.1.2	ETS hardware configured for actual testing shall have an availability greater than 0.98.	4.1.5,
5.1.5	ETS shall provide the capability to issue system configuration and test status reports on a periodic basis, in selectable multiples of 1 minute.	5.1.2.3, 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6, 5.2.7, 5.2.8, 5.5.7, 5.5.8,
5.1.6	ETS shall update status, data quality, and accounting information once every 10 seconds, at a minimum.	3.4.1.4, 3.5.2.7, 5.1.3.1, 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6, 5.2.7, 5.2.8, 5.5.8, 5.5.10,
5.1.7	ETS shall acknowledge a request from a local user within 2 seconds of its entry.	3.1.2.1, 3.1.2.2.1, 3.5.1.15, 3.5.1.17, 3.5.1.19, 5.1.3.1,
5.1.8	ETS shall start the execution of a local user request within 5 seconds of its entry.	3.1.2.1, 3.1.2.2.1, 3.5.1.15, 3.5.1.17, 3.5.1.19,
5.1.9	ETS shall be ready for operational use within 20 minutes of poweron.	4.1.2, 4.1.3,
5.1.10	ETS shall provide the capability to store high-rate data during a testing session up to 2 Gbytes.	3.1.1,
5.2.1	ETS shall provide the capability to generate and transmit up to two TGT return link data streams, including clock, in the form of CADUs, each at rates up to 150 Mbps.	2.2.1.1, 2.2.1.2, 2.2.1.3, 2.3.2.1, 2.3.2.2, 3.2.1, 3.2.1.1, 3.2.1.2, 4.2.1, 4.2.2, 5.5.2, 5.5.11,

5.3.1	ETS shall provide the capability to generate and transmit PDSs to a single destination through EBnet at a rate up to 34 Mbps.	2.2.2.4, 3.2.1, 3.2.1.1, 3.2.1.2, 3.3.1, 3.5.1.2, 3.5.2.17, 3.5.2.20, 3.5.2.21, 3.5.2.22, 3.5.2.23, 3.5.3.4, 3.5.3.5, 3.5.3.6, 3.5.3.11, 3.5.3.13, 3.5.3.15, 3.5.3.16, 4.3.1, 4.3.2, 4.5.4, 5.3.1, 5.3.2, 5.3.3, 5.3.4, 5.3.5, 5.3.6, 5.3.7, 5.5.2, 5.5.11,
5.3.2	ETS shall provide the capability to generate and transmit EDSs to a single destination through EBnet at a rate up to 34 Mbps.	2.2.2.4, 3.2.1, 3.2.1.1, 3.2.1.2, 3.3.1, 3.5.1.2, 3.5.2.17, 3.5.2.20, 3.5.2.21, 3.5.2.22, 3.5.2.23, 3.5.3.8, 3.5.3.8.1, 3.5.3.9, 3.5.3.10, 3.5.3.13, 3.5.3.14, 3.5.3.16, 4.3.2, 4.5.4, 5.3.1, 5.3.2, 5.3.3, 5.3.4, 5.3.5, 5.3.6, 5.3.7, 5.5.2, 5.5.11,
5.3.3	ETS shall provide the capability to generate and transmit a data set up to 2 Gbytes in size that spans multiple TDRSS sessions.	3.2.1, 3.2.1.1, 3.2.1.2, 3.3.1, 4.1.7, 5.5.2,
5.4.1	ETS shall provide the capability to receive PDSs through EBnet from a single source at rates up to 34 Mbps.	2.2.2.5, 2.2.2.6, 3.1.1.3, 3.4.1, 4.4.1, 5.4.1, 5.4.2, 5.4.3, 5.4.4, 5.4.5, 5.4.6, 5.4.7,
5.4.2	ETS shall provide the capability to receive EDSs through EBnet from a single source at rates up to 34 Mbps.	2.2.2.5, 2.2.2.6, 3.1.1.3, 3.4.1, 4.4.1, 5.4.1, 5.4.2, 5.4.3, 5.4.4, 5.4.5, 5.4.6, 5.4.7,
5.4.3	ETS shall provide the capability to receive, store, and verify a data set of to 2 Gbytes in size.	3.1.1.3, 3.4.1, 3.4.1.2, 3.5.1.28, 4.4.2,
5.5.1	ETS shall provide the capability to generate and transmit CADUs at a rate of 256 kbps.	3.2.3, 3.2.3.1,
5.5.2	ETS shall provide the capability to generate and transmit CADUs at a rate of 16 kbps.	3.2.3, 3.2.3.1,
5.5.3	ETS shall provide the capability to generate and transmit CADUs at a rate of 1 kbps.	3.2.3, 3.2.3.1,

10.3 BUILD PLAN FOR ETS HRS

The build plan for development and delivery of the ETS HRS is shown in Table 10.1.

Table 10-1. ETS HRS Projected Build Plan

Build	Date	Function	Hardware
1	5/8	<ol style="list-style-type: none"> 1) Create and store test data. 2) Transmit test data in CADUs at 150 Mbps. 3) Generate and store EDSs and PDSs. 4) Transmit EDSs and PDSs via Ecom @ 34 Mbps. 5) Capture and store EDSs and PDSs @ 34 Mbps. 6) Play back user-provided test data from an Ampex tape drive. 7) Generate EDSs and PDSs from user-provided test data. 8) Generate CADU files from user-provided test data. 9) Partial workstation interface (control and status). 	Rack with all cards, some of which may be prototype. An Ampex tape drive with workstation interface.
2	9/30	<p>In addition to functions in Build 1:</p> <ol style="list-style-type: none"> 1) Verify EDSs and PDSs generated by EDOS. 2) Generate second 150-Mbps CADU stream through a disk array. 3) Full workstation interface (OMD, Report generation, Library). 4) Transfer use-provided test data from a tape to a disk array. 	Prototype cards replaced with production cards. Upgrade rack.

ACRONYMS AND ABBREVIATIONS

<u>Term</u>	<u>Definition</u>
ANSI	American National Standards Institute
AOS	Advanced Orbiting Systems
API	Applications Program Interface
APID	Application Process Identifier
ASCII	American Standard Code for Information Interchange
ASIC	Application-specific Integrated Circuit
BaSE	Base System Environment
BER	Bit Error Rate
BOT	Beginning of Tape
BTD	Bit Transition Density
CADU	Channel Access Data Unit
CCB	Configuration Control Board
CCSDS	Consultative Committee for Space Data Systems
CDS	Control and Display Subsystem
CI	Configuration Item
CLIPS	C Language Integrated Production System
CODA	Customer Operations Data Accounting
COTS	Commercial Off-the-Shelf
CP	Continuous Pattern
CPU	Central Processing Unit
CRC	Cyclical Redundancy Check
CSG	Clock Signal Generator
CVCDU	Coded Virtual Channel Data Unit
DAAC	Distributed Active Archive Center
DCN	Documentation Change Notice
DIF	Data Interface Facility
DMA	Direct Memory Access
DPR	Dual-ported Random Access Memory
DR	Discrepancy Report
DRAM	Dynamic Random Access Memory
DSAT	Data Set Assembly Table
EBnet	EOSDIS Backbone Network
EDOS	EOS Data and Operations System
EDS	Expedited Data Set
EOC	EOS Operations Center
EOD	End-of-Data
EOF	End-of-Frame
EOF	End-of-Frame
EOSDIS	Earth Observing System Data and Information System
EOT	End of Tape
EPROM	Eraseable Programmable Read-only Memory
ERS	EOS Reed-Solomon Card
ETS	EOSDIS Test System
F&PR	Functional and Performance Requirement
FAST	Fast Auroral Snapshot Explorer
FDDI	Fiber Distributed Data Interface
FIFO	First-in, First-out
FPGA	Field-programmable Gate Array
FT	Forward True

FTP	File Transfer Protocol
GMT	Greenwich Mean Time
GOTS	Government Off-the-Shelf
GSFC	Goddard Space Flight Center
GTDG	GaAs Test Data Generator
GTFS	GaAs Telemetry Frame Synchronizer
GUI	Graphical User Interface
HDD	Hardware Definition Document
HiPPI	High-performance Parallel Interface
HRS	High-rate System
HRTB	High-rate Telemetry Backplane
HSI	High-speed Interface
I&T	Integration and Testing
I/O	Input/Output
IC	Integrated Circuit
ICD	Interface Control Document
IEEE	Institute of Electrical and Electronic Engineers
IPC	Interprocessor Communication
ISO	International Standards Organization
LAN	Local Area Network
LED	Light-emitting Diode
LRS	Low-rate System
LZP	Level Zero Processing
MAC	Media Access Control
MCC	Master Controller Card
MEDS	Modular Environment for Data Systems
MP	Multiple Pattern
MSB	Microelectronic Systems Branch
MT	Magnetic Tape
MTBF	Mean Time Between Failure
MTTR	Mean Time To Repair
M_PDU	Multiplexing Protocol Data Unit
NCO	Numerically Controlled Oscillator
NFS	Network File Server
OPMAN	Operations Manager
OSF	Open Software Foundation
PAL	Programmable Array Logic
PAT	Packet Assembly Table
PCA	Processor Communications Area
PDOS	Power Disk Operating System
PDS	Production Data Set
PHY	Physical Layer Protocol
PLD	Programmable Logic Device
PMD	Physical Layer Medium Dependent
PN	Pseudonoise
PPS	Packet Processing System
PRS	Problem Reporting System
RAM	Random Access Memory
RI	Reverse Inverted
RSEC	Reed-Solomon Error Correction
RSH	Remote Shell
RT	Reverse True
SBC	Single-board Computer
SCID	Spacecraft Identifier

SCITF	Spacecraft Integration and Test Facility
SCSI	Small Computer System Interface
SDS	System Design Specification
SDU	Service Data Unit
SFE-FEP	Science Formatting Equipment Front-end Processor
SMA	Subminiature Assembly
SOL	Segment Order List
SP	Single Pattern
SRAM	Static Random Access Memory
SRS	System Requirements Specification
TCP/IP	Transmission Control Protocol/Internet Protocol
TDRSS	Tracking and Data Relay Satellite System
TF	Transfer Frame
TPCE	Telemetry Processing Control Environment
TRS	Tape Recording Subsystem
TTL	Transistor-to-Transistor
TV	Thermal and Vacuum
UART	Universal Asynchronous Receiver-Transmitter
UDP	User Datagram Protocol
UI	User Interface
UTC	Universal Timecode
VC	Virtual Channel
VCDU	Virtual Channel Data Unit
VCID	Virtual Channel Identifier
VHS	VME High-rate Subsystem
VME	Versa Module Eurocard
VS	VME Subsystem Bus
WAN	Wide Area Network