ENGINEERING CHANGE NOTICE

1. ECN 637502

Proj. ECN

2. ECN Category (mark one)	3. Originator's Name, Organization, MSIN, and Telephone No.	3a. USQ Required?	4. Date
Supplemental [] Direct Revision [X] Change ECN [] Temporary [] Standby [] Supersedure [] Cancel/Void []	A. M. Ermi, Remote Sensing and Sampling Equipment Engineering, L6-37, 376-5099	[X] Yes [ ] No TF-97-0581	May 20, 1997
	5. Project Title/No./Work Order No.	6. Bldg./Sys./Fac. No.	7. Approval Designator
	Tank 241SY101 Hydrogen Mitigation Test Project / N2045	Tank 241SY101	SQ
	8. Document Numbers Changed by this ECN (includes sheet no. and rev.)	9. Related ECN No(s).	10. Related PO No.
	WHC-SD-WM-CSDD-008 Rev. 2	198629	N/A

11a. Modification Work	11b. Work Package No.	11c. Modification Work Complete	11d. Restored to Original Condition (Temp. or Standby ECN only)
[] Yes (fill out Blk. 11b) [X] No (NA Blks. 11b, 11c, 11d)	N/A	N/A Cog. Engineer Signature & Date	N/A Cog. Engineer Signature & Date

12. Description of Change

Design Baseline Document? X Yes _ No

The previous Rev. 2 document is being replaced by this Rev. 3 document. Numerous changes in field instrumentation and improvements to the graphical interface as requested by Operations are incorporated in this revision. This also resulted in upgrades to the DACS control software (TEST303 to TEST304, and PLC304 to PLC305).

13a. Justification (mark one)

Criteria Change	[]	Design Improvement	[X]	Environmental	[]	Facility Deactivation	[]
As-Found	[]	Facilitate Const	[]	Const. Error/Omission	[]	Design Error/Omission	[]

13b. Justification Details

Since the issuance of the Rev. 2 document, changes to field instrumentation and improvements to the operator graphical interface have made it necessary to revise the software, and reissue the document to reflect the changes.

Design verification performed by informal review per WHC-CM-6-1, EP-4.1

14. Distribution (include name, MSIN, and no. of copies)

See distribution list.

RELEASE STAMP

AUG 15 1997

DATE: STA: 37 HANFORD RELEASE ID: 20

ENGINEERING CHANGE NOTICE

1. ECN (use no. from pg. 1): 637502

16. Design Verification Required	17. Cost Impact				18. Schedule Impact (days)	
	ENGINEERING		CONSTRUCTION			
[ ] Yes	Additional	[ ] $	Additional	[ ] $	Improvement	[ ]
[X] No	Savings	[ ] $	Savings	[ ] $	Delay	[ ]

19. Change Impact Review: Indicate the related documents (other than the engineering documents identified on Side 1) that will be affected by the change described in Block 13. Enter the affected document number in Block 20.

SDD/DD	[X]	Seismic/Stress Analysis	[ ]	Tank Calibration Manual	[ ]
Functional Design Criteria	[ ]	Stress/Design Report	[ ]	Health Physics Procedure	[ ]
Operating Specification	[ ]	Interface Control Drawing	[ ]	Spares Multiple Unit Listing	[ ]
Criticality Specification	[ ]	Calibration Procedure	[ ]	Test Procedures/Specification	[ ]
Conceptual Design Report	[ ]	Installation Procedure	[ ]	Component Index	[ ]
Equipment Spec.	[ ]	Maintenance Procedure	[ ]	ASME Coded Item	[ ]
Const. Spec.	[ ]	Engineering Procedure	[ ]	Human Factor Consideration	[ ]
Procurement Spec.	[ ]	Operating Instruction	[ ]	Computer Software	[X]
Vendor Information	[ ]	Operating Procedure	[ ]	Electric Circuit Schedule	[ ]
OM Manual	[ ]	Operational Safety Requirement	[ ]	ICRS Procedure	[ ]
FSAR/SAR	[ ]	IEFD Drawing	[ ]	Process Control Manual/Plan	[ ]
Safety Equipment List	[ ]	Cell Arrangement Drawing	[ ]	Process Flow Chart	[ ]
Radiation Work Permit	[ ]	Essential Material Specification	[ ]	Purchase Requisition	[ ]
Environmental Impact Statement	[ ]	Fac. Proc. Samp. Schedule	[ ]	Tickler File	[ ]
Environmental Report	[ ]	Inspection Plan	[ ]	I/O Channel List	[X]
Environmental Permit	[ ]	Inventory Adjustment Request	[ ]		[ ]

20. Other Affected Documents: (NOTE: Documents listed below will not be revised by this ECN.) Signatures below indicate that the signing organization has been notified of other affected documents listed below.

Document Number/Revision	Document Number/Revision	Document Number Revision
WHC-SD-WM-SDD-045 / Rev. 1		
WHC-SD-WM-EL-001 / Rev. 5		

21. Approvals

	Signature	Date		Signature	Date
Design Authority	W. G. Brown Warren G. Brown	7/21/97	Design Agent		
Cog. Eng.	G. J. Gauck Gregory J. Gauck	7/21/97	PE		
Cog. Mgr.	G. L. Dunford	7/21/97	QA		
QA	R. R. True R. R. True	7.21.97	Safety		
Safety	L. S. Krogsrud L S Krogsrud	7/23/97	Design		
Environ.			Environ.		
Other	A. M. Ermi A. M. Ermi	5-20-97	Other		
Other	C. E. Hanson Carl Hanson	8/11/97			
Other	R.W. Truitt (Informal Review) R.W. Truitt	8/14/97	DEPARTMENT OF ENERGY		
			Signature or a Control Number that tracks the Approval Signature		
			ADDITIONAL		

UNREVIEWED SAFETY QUESTION SCREENING/DETERMINATION FORM

(Per WHC-IP-0842)

USQ Tracking No. TF-97-0581

Rev. 0

AREA: [] East [X] West [] General

Facility: [] 242-A [X] DST [] SST [] LERF
[] Aging Waste [] Other

ECN No. 637502 PCA No. N/A

Work Pkg No. N/A Other (Specify) HNF-SD-WM-CSDD-008, Rev 3

TITLE: COMPUTER SYSTEM DESIGN DESCRIPTION FOR SY-101 HYDROGEN MITIGATION TEST PROJECT DATA ACQUISITION AND CONTROL SYSTEM (DACS-1); HNF-SD-WM-CSDD-008, Rev.3, (ECN 637502)

Description of the Proposed Activity/REPORTABLE OCCURRENCE or PIAB:

This ECN changes the computer systems design description support document describing the computers system used to control, monitor and archive the processes and outputs associated with the Hydrogen Mitigation Test Pump installed in SY-101.

Introduction:

There is no new activity or procedure associated with the updating of this reference document. The updating of this computer system design description maintains an agreed upon documentation program initiated within the test program and carried into operations at time of turnover to maintain configuration control as outlined by design authority practicing guidelines. Any changes made to controlled components in the field will be updated after the time of implementation to support the engineers and operators understand, maintain, train to and operate the system.

There are no new credible failure modes associated with the updating of information in a support description document. The failure analysis of each change was reviewed at the time of implementation of the Systems Change Request for all the processes changed. This document simply provides a history of implementation and current system status.

Scope:

This USQ screening examines the CSDD described above, HNF-SD-WM-CSDD-008, Rev 3 and no other.

Authorization Basis:

WHC-SD-WM-ISB-001 Vol 1 Rev 0-M Sec. 6.0 Table 6. Hanford Site Tank Farm Facilities Interim Safety Basis

LA-UR-92-3196, REV 14a, Los Alamos National Laboratory 1995

West Tank Farms Standing Order 97-01 Rev 2

HNF-SSD-TWR-TI-003, Rev 2, Supporting Documentation for Requested Exceptions to Standing Order 96-36 (East) and 96-34 (West)

Conclusion:

UNREVIEWED SAFETY QUESTION SCREENING/DETERMINATION FORM (Continued)	Page 2 of 3
	USQ Tracking No. TF-97-0581
	Rev. 0

The update and revision of the computer system design description does not represent a USQ and the conditions surrounding the change which occurred in order to change this document have been analyzed in the authorization basis documents. Therefore no USQ determination is required. The incorporation of information into this revision of the Systems Design Description has been examined against the flammable gas JCO 97-01 latest rev and found not to be applicable because there is no field work associated with this ECN.

References

DE&S Hanford Inc., Internal Memo, T.C. Geer to J.H. Wicks and W.M. Funderburke, " CLARIFICATION OF STANDING ORDER 96-36 (EAST) AND 96-34 (WEST)," dated Dec. 2, 1996

Van Vleet, R.J., 1994 "Safety Basis for Activities in Double-Shell Flammable Gas Watch List Tanks," WHC-SD-WM-SARR-002, Westinghouse Hanford Company, Richland WA.

Straalsund E.K. & Mendoza R.E., June 19, 1995, "SYSTEM DESIGN DESCRIPTION FOR SY-101 HYDROGEN MITIGATION TEST PROJECT DATA ACQUISITION AND CONTROL SYSTEM (DACS-1)," USQ Screening Against ECN 198628

Gauck, G.J. & Brown, W.G. February 18, 1997, "SYSTEM DESIGN DESCRIPTION FOR SY-101 HYDROGEN MITIGATION TEST PROJECT DATA ACQUISITION AND CONTROL SYSTEM (DACS-1)," USQ Screening (TF-97-0174) against ECN 637503

USQ Screening:

A. Does the PROPOSED ACTIVITY represent a change to the facility as described in the AUTHORIZATION BASIS?

[X] No [] Yes [] N/A

Basis: The document, HNF-SD-WM-CSDD-008, Rev 3, makes no changes to the facility as described in the authorization basis. The hardware system design description is being updated to reflect current status of the DACS system and do not drive any change to the SY-101 tank farm facility. The Safety Assessment, LA-UR-92-3196, REV 14a, Los Alamos National Laboratory 1995, requirements for DACS system hardware and software control testing and verification are discussed in Appendix S. The updating of this document is in compliance with the maintaining of current revisions of the software which operates the pump and provides alarms, shutdown sequences and surveillance/archiving functions.

B. Does the PROPOSED ACTIVITY represent a change to procedures as described in the AUTHORIZATION BASIS?

[] No [] Yes [X] N/A

Basis: This is not a change to a procedure nor control associated with the operation of the pump or its recording surveillance devices.

UNREVIEWED SAFETY QUESTION SCREENING/DETERMINATION FORM (Continued)	Page 3 of 3
	USQ Tracking No. TF-97-0581
	Rev. 0

C. Does the test or experiment represent a test or experiment not described in the AUTHORIZATION BASIS documentation?

[] No [] Yes [X] N/A

Basis: This document up date is not a test nor experiment.

D. Does the PROPOSED ACTIVITY or REPORTABLE OCCURRENCE, impact:

- OSRs or IOSRs?
- Approved IOSR Compliance Implementation Plan?

[X] No [] Yes [] N/A

Basis: There are no changes in any OSR, IOSR, or TSR documents or requirements governing the operation or configuration of this pump control system or this document outlining the systems design descriptions.

E. Does the REPORTABLE OCCURRENCE or PIAB involve analytical errors, omissions, and/or deficiencies in the AUTHORIZATION BASIS?

[] No [] Yes [X] N/A

Basis: This is a change to a systems design description document not documentation noting a reportable occurrence or PIAB.

USQE No. 1 G.J. Gauck (Print Name) — Signature: G J Gauck — Date: 7-21-97

USQE No. 2 Warren G Brown (Print Name) — Signature: Warren G Brown — Date: 7/21/97

IF "YES", USQE CONTINUE WITH DETERMINATION BELOW

USQ DETERMINATION:

1. Could the PROPOSED ACTIVITY or USQ ISSUE significantly increase the frequency of occurrence of an accident previously evaluated in the AUTHORIZATION BASIS?

   [] No [] Yes/Maybe

   Basis:

2. Could the PROPOSED ACTIVITY or USQ Issue significantly increase the consequences of an accident previously evaluated in the AUTHORIZATION BASIS?

   [] No [] Yes/Maybe

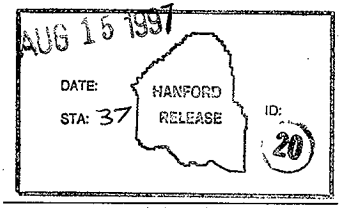
COMPUTER SYSTEM DESIGN DESCRIPTION FOR SY-101 HYDROGEN MITIGATION TEST PROJECT DATA ACQUISITION & CONTROL SYSTEM (DACS-1)

A. M. Ermi
SGN Eurisys Services Corporation, Richland, WA 99352
U.S. Department of Energy Contract DE-AC06-96RL13200

EDT/ECN: 637502 UC: 2030
Org Code: S1200 Charge Code: N2045
B&R Code: EW3120072 Total Pages: 216

Key Words: DACS, DATA ACQUISITION AND CONTROL SYSTEM, 241SY101, MIXER PUMP, SOFTWARE, COMPUTER

Abstract: This document provides descriptions of the components and functions of the automated data acquisition and control system (DACS) in support of hydrogen mitigation for waste tank 241SY101 at the Hanford Nuclear Reservation. The system was designed and implemented by Los Alamos National Laboratory, supplied to Westinghouse Hanford Company, and now operated by Lockheed Martin Hanford Corporation.



Release Approval — Date: 8-15-97

Release Stamp

Approved for Public Release

RECORD OF REVISION	(1) Document Number HNF-SD-WM-CSDD-008	Page 1

(2) Title

Computer System Design Description for SY-101 Hydrogen Mitigation Test Project Data Acquisition and Control System (DACS-1)

CHANGE CONTROL RECORD

(3) Revision	(4) Description of Change - Replace, Add, and Delete Pages	Authorized for Release	
		(5) Cog. Engr.	(6) Cog. Mgr. Date
Rev. 0	(7) EDT 158000, Dec. 22, 1992	---	---
Rev. 1	Change title to: "Computer System Design Description for SY-101 Hydrogen Mitigation Test Project Data Acquisition and Control System (DACS-1)" The document was completely re-worked, with numerous changes made in the arrangement of the information and in the level of detail of the information. Information not available at the time Rev. 0 was released is now included in the Rev. 1 document.	RW Truitt	RE Bauer (original signed 8-24-94)
Rev. 2	Complete revision per ECN 198629, January 4, 1995	GJ Gauck	SH Rifaey (original signed 8-11-95)
Rev. 3 RS	Complete revision with the installation of revised software TEST-3.04 and PLC-3.05 per ECN 637502, May 20, 1997; "WHC" document prefix changed to "HNF"	GJ Gauck 3.21.97.	G. L Dunford 7/21/97

HNF-SD-WM-CSDD-008
Revision 3.0

Computer System Design Description for the SY-101 Hydrogen Mitigation Test Project Data Acquisition and Control System (DACS-1)

May 1997

Prepared by:

S. O. Smith & R. W. Truitt,
PLCs Plus

and

A. M. Ermi,
SGN Eurisys Services Corporation

Prepared for:

G. J. Gauck,
Lockheed Martin Hanford Corporation

TABLE OF CONTENTS

Trademark Credits

Agiler is a trademark of Sysgration Ltd.
3Com is a registered trademark of 3Com Corporation
ARCLINK is a trademark of Datapoint Corporation
ARCNET is a registered trademark of Datapoint Corporation
Bernoulli is a registered trademark of Iomega Corporation
Black Box is a registered trademark of Black Box Corporation
Datataker Plus is a registered trademark of Data Electronics (Aust) Pty. Ltd.
DeCipher Plus is a trademark of Data Electronics (Aust) Pty. Ltd.
DeTerminal is a trademark of Data Electronics (Aust) Pty. Ltd.
DT50, DT200 and DT500 are trademarks of Data Electronics (Aust) Pty. Ltd.
Eaton is a registered trademark of the Eaton Corporation
EMM386 is a trademark of Microsoft Corporation
Ethernet is a trademark of the Xerox Corporation
Expert Mouse is a trademark of Kensington Microware Ltd.
FAMOS is a trademarks of Nicolet Instrument Corporation
Genesis Control Series is a trademark of Iconics, Inc.
Gen-Net is a trademark of Iconics, Inc.
Genius Mouse is a trademark of KYE International Corporation
HP LaserJet is a registered trademark of Hewlett-Packard Company
IBM PC is a trademark of the International Business Machine Corporation
IBM Proprinter is a trademark of the International Business Machine Corporation
Iomega is a registered trademark of Iomega Corporation
Jaz is a registered trademark of Iomega Corporation
Kensington is a registered trademark of Kensington Microware Ltd.
Lotus is a registered trademark of Lotus Development Corporation
Magic I/O is a trademark of Everex Systems, Inc.
Modsoft is a registered trademark of AEG Schneider Automation, Inc.
Modicon, Modbus Plus and 984 PLC are trademarks of AEG Schneider Automation, Inc.
MS-DOS and Windows are registered trademarks of Microsoft Corporation
NETROOM is a trademark of the Helix Software Company, Inc.
Norton DiskLock is a trademark of Symantec Corporation
Nicolet is a registered trademarks of Nicolet Instrument Corporation
SA-85 Modbus Plus Card is a trademark of AEG Schneider Automation, Inc.
Shinko is a trademark of Shinko Technologies, Inc.
SMC is a registered trademark of of Standard Microsystems Corporation
Texas Microsystems is a registered trademark of Texas Microsystems, Inc.

Other product names may be trademarks or registered trademarks of their respective companies and are hereby acknowledged.

1.0 INTRODUCTION

This document provides descriptions of components and tasks that are involved in the computer system for the data acquisition and control of the mitigation tests conducted on waste tank 241-SY-101 at the Hanford Nuclear Reservation. The system was designed and implemented by Los Alamos National Laboratory, supplied to the Westinghouse Hanford Company, and is currently being operated by Lockheed Martin Hanford Corporation. The computers (both personal computers and specialized data-taking computers) and the software programs of the system will hereafter collectively be referred to as the DACS (Data Acquisition and Control System).

1.1 PURPOSE

This document reflects the DACS-1 information for the software currently in use in the DACS trailer; TEST strategy "TEST304", MOTOR strategy "MOTOR301", and Modicon ladder logic software "PLC305".

1.2 SCOPE

The DACS was designed for hydrogen mitigation testing for waste storage tank 241-SY-101. The mitigation testing uses a pump immersed in the waste tank and is directed at certain angles and operated at different speeds and time durations.

The positioning of the pump and operation of the pump is controlled by the DACS, with the test operators commanding the system. There are many instruments used to monitor process variables within the tank at all times. The DACS collects data from these instruments, displays real-time data for operators and archives the data for later analysis; it also interfaces with control elements and safeguards the operation of the test equipment.

The DACS is composed of several components working in parallel to perform the tasks needed for test operation and monitoring.

There is a variable frequency drive that controls a motor that will move the position of the pump to different angles. A second variable frequency drive controls a motor which drives the pump that circulates the waste.

A PLC (Programmable Logic Controller) interfaces with instrumentation in the waste tank and controls elements for safe operation. This PLC has embedded logic to assure safe operation of the pump and to process data for interfacing to a supervisory networked computer system.

A networked computer system, using a multitasking software shell called Genesis, brings together all data gathered from operator input, the PLC, and the variable frequency drives. The Genesis system is used to collect data from the PLC and from the variable frequency drives controlling the motors. Genesis displays this data in real time and in a graphic format to the operators. The system also stores data to files that can be analyzed at any time and these files are archived for later analysis.

The Genesis system also is used to control the variable frequency drives' operation from operator requests. There is embedded logic in the system to warn operators of critical conditions and ensure safe operation of the pump motors.

The Genesis system supports peer-to-peer network communications to enable data to be shared between terminals. The Genesis network also supports host communications to multiple supervisory stations, allowing several consoles to access common data.

1.3 OVERVIEW

This document provides a description of the system in ever-increasing detail. First the general organization of system components is shown in Section 2, "System Layout." Then, in Section 3, "System Functions," the DACS tasks are broken down and an overview of each component's purpose is given. Finally, in Section 4, "Design Details," the intricacies of how each task is accomplished are supplied.

1.4 CONVENTIONS AND DEFINITIONS

This document contains some formalisms to aid in its descriptions. These, as well as acronyms that are used, are given in the sections that follow.

1.4.1 Formatting Conventions

The Genesis multitasking software application is embodied in a database referred to as a **strategy**. Strategies are organized by algorithm blocks, which have a type and a tag name. The block types are denoted by an *italic* font. The block tag names will be encapsulated in angle brackets; for example, <TAGNAME>.

1.4.2 Acronyms

The following acronyms are used:

CSDD	Computer System Design Description
DACS	Data Acquisition and Control System
FTIR	Fourier Transform InfraRed
HLAN	Hanford Local Area Network
HSDD	Hardware System Design Description
MIT	Multi-function Instrument Tree
PLC	Programmable Logic Controller
PNL	Pacific Northwest Laboratory (Battelle)
R&R	Report and Recipe
RSS	Remote Supervisory Station
SDD	System Design Description
VFD	Variable Frequency Drive
WHC	Westinghouse Hanford Company

2.0 SYSTEM LAYOUT

The DACS contains several components that function in parallel to perform data collection and analysis, data display, alarm indication, monitoring, pump control, and archiving functions. Each component is carefully chosen, based on its capabilities, to perform a specific task. Their close choreography is required to meet the performance requirements of the system.

The system is designed to collect data from the field, display the data to operators, and log the data for later analysis. The system has the ability to warn the operators of critical conditions. There are control functions embedded in the system to control the position and operation of the pump. During pump control there are several checks on the critical signals to abort operation when a problem is detected. The system is designed to have enhanced effectiveness by being user friendly, and has built-in protections to make the test operation safe.

Information must be exchanged between components to integrate all of the data and control signals into a workable whole. Communication routines are implemented to read signals and data from the PLC and AF5000 devices and to send control commands to the pump equipment. The display terminals share data over a real-time data network. This network allows multiple users to share information and allows strategies to share critical signals and control commands.

2.1 COMPONENT LAYOUT

The DACS is composed of Modicon PLCs, two Eaton AF5000+ variable frequency drives, a Genesis data collection runtime station, a Genesis pump control runtime station, a backup station to each Genesis runtime station, multiple local consoles for the Genesis data collection station, multiple remote consoles for the Genesis data collection station, a file server interface to the local area network and a station in the trailer for access to and programming of the PLC.

The Eaton AF5000+ variable frequency drives are used to control electrical motors. The units have microprocessors that have logic functions specifically designed to control the motor and provide for its safe operation. The reading of parameters and sending of control commands for the variable frequency drives can be done from the front panel of the unit or from serial communication commands. The operating status of the drive can be viewed from the front panel of the unit or it can be requested by a device that supports serial communications.

A driver has been written to communicate with the Eaton AF5000+ variable frequency drives to request data and send control commands. This driver has been specifically designed to function within a personal computer running the Genesis multitasking software.

The Genesis multitasking software is used to display real-time data from the field, perform logic functions, and send commands to the control devices. Each Genesis application

is embodied in a database which is referred to as a strategy. This strategy contains algorithms that are executed at specific rates. The algorithms in the strategy are graphically selected by placing blocks in the strategy configuration environment. There are different kinds of algorithm blocks that can be used in the strategy to read data from hardware devices and write data to hardware devices. Genesis provides control-based algorithm blocks to build control logic.

Personal computers that use Genesis to update a strategy from hardware devices are considered runtime stations. Real-time displays are dynamically updated from the current values of the parameters in the algorithm blocks in the Genesis strategy. These displays are interactive and are designed to show current signal readings and allow the operators to change operating parameters. The Genesis software is used to bring data from the field together, perform calculations on that data, and then display the results to the operators.

Each Genesis runtime application can share its data to other Genesis stations over a real-time data network called GEN-NET. The GEN-NET is used to transfer selected real-time data and files between personal computers running the Genesis software. The physical layer of the GEN-NET uses ARCNET hardware in a star configuration. All GEN-NET stations are connected to an active hub with coaxial cable.

Two Genesis runtime stations are used to pass data to and from the Modicon PLC. Information is retrieved and sent to the PLC with a communication driver that executes in parallel with the other Genesis runtime functions. One runtime station is called STATION5 on the Genesis GEN-NET. The Genesis strategy executed at this station is called TEST.

The other Genesis runtime station is used for pump control and communicates to the two variable frequency drives used to position and control the pump. This runtime station is called STATION8. It also communicates with the PLC to transfer data from the variable frequency drives. The Genesis strategy that is executed in this station is called MOTOR.

One of the personal computers on the GEN-NET is used as a backup to either STATION5 running the TEST strategy or STATION8 running the MOTOR strategy. This station is called STATION7 on the GEN-NET.

STATION7 can also become a second console to the TEST strategy running in STATION5. As a Remote Supervisory Station (RSS) the console can bring up dynamic displays that update directly from the TEST strategy. STATION6 will serve as a RSS to the TEST strategy at all times.

Four RSS consoles are located outside of the DACS trailer. These are known as STATION11, STATION13, STATION15 and STATION17 to the GEN-NET network software. STATION11 is located in Building MO-278, STATION13 is located in the 306E Building, STATION15 is located at the 2750 Building and STATION17 is located in the office of the software developers. All four remote stations transfer their network data packets over phone lines at 9600 baud using special ARCNET/RS-232 conversion units called Arc-Links.

The Arc-Links can only recognize odd-numbered network addresses (see 2.3.7), so all of the remote stations have odd numbers. When the remote stations were added, stations in the trailer had been assigned the network addresses 5 through 9 so the remote stations were given the next available odd numbers - 11, 13, 15 and 17.

A personal computer is dedicated to transferring files from the GEN-NET to a file server on the Hanford local area network (HLAN). STATION9 on the GEN-NET is logged into the file server and is operated in a GEN-NET file server mode. This allows files from STATION5 to be transferred to the file server without broadcasting real-time data packets over the HLAN.

STATION1 in the trailer runs the commercial package Modsoft. This software allows the PLC to be programmed and configured. It also allows the PLC registers and coils (single bits) to be viewed and manipulated online, by developers only.

Figure 1 is a diagram of the physical connections between components.

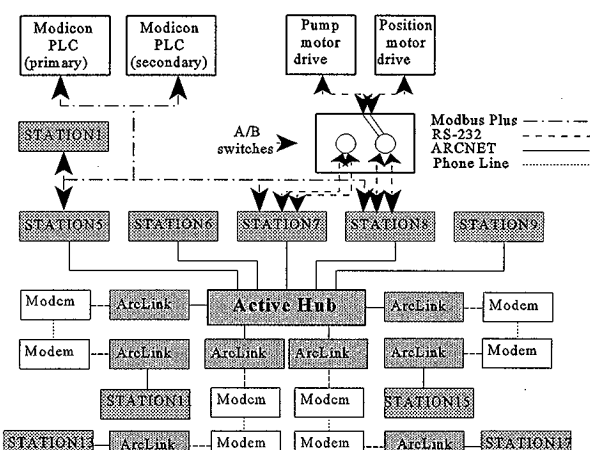


Fig. 1. Diagram of the physical connections between components.

2.2 LOGICAL FLOW OF DATA

Data are retrieved from the PLC registers and status points by the station running the TEST strategy. This updates the TEST strategy algorithm blocks that are used to read data from the PLC. Data are sent back to the PLC from algorithm blocks that are used to write data to the PLC. For backup support STATION7 can run the TEST strategy; for normal operation STATION5 will run the TEST strategy. All discussion related to data flow of the TEST strategy will assume normal operation. For backup operation simply substitute STATION7 for STATION5. The MOTOR strategy is used to communicate bidirectionally with both variable frequency drives. For normal operation STATION8 will run the MOTOR strategy; for backup operation STATION7 can also run the MOTOR strategy. Data which are passed between STATION5 and STATION8 (via registers in the PLC) allow the TEST and MOTOR strategies to share critical data.

The local RSSs linked to the TEST strategy can receive and send values to the TEST strategy. The RSSs outside of the DACS trailer will only be able to receive information from the TEST strategy.

Figure 2 shows the data and control paths between components.

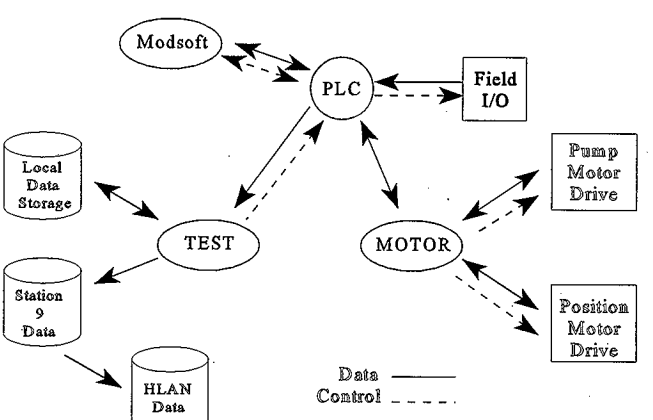


Fig. 2. Data and control paths between components.

2.3 TASK ALLOCATION TO COMPONENTS

Each component performs at least one main task needed for the system. The following section is a breakdown of the tasks that are performed by each component.

2.3.1 Modicon PLC

The functions of the Modicon PLC are to input data from the field, to abort the pump when critical channels are beyond their abort limits, to provide control and alarm signals and the logic to generate them, to provide timers and enable logic for pump testing, and to provide data and status information to Genesis.

2.3.2 Eaton AF-5000+ Variable Frequency Drives

The variable frequency drives control the motors used to position and operate the pump. Each drive controls its respective motor's critical parameters such as speed, acceleration, deceleration, and maximum speed as well as the starting and stopping of the motor. The drive also monitors current, line voltage, actual speed, and current state of the motor.

The variable frequency drives communicate with the Genesis MOTOR strategy via RS-232 serial communications. Each variable frequency drive has a dedicated COM port for these communications. The drive that controls the pump motor is on COM1, and the drive that controls the directional motor is on COM2. Genesis only acts in a supervisory role in controlling the motors; the drives can run the motors independent of Genesis communication.

A special signal from each drive that indicates when the motor is running is wired directly to a discrete input module. This signal is then used by the PLC for accurate timing of the duration of a test run.

2.3.3 TEST Strategy Terminal, STATION5

STATION5 is the primary data acquisition computer and is connected to the Modicon E984 PLC via Modbus Plus. The strategy functions that STATION5 has to perform are: send and retrieve field values from the Modicon E984 I/O drops, write those engineering values to disk as Genesis history files, participate in the watchdog timer with the E984 ladder logic, download abort limits to the PLC and generate operator alarms.

2.3.4 MOTOR Strategy Terminal, STATION8

This personal computer is dedicated to running the Genesis Runtime MOTOR strategy. The GEN-NET node name STATION8 is used to reference this component.

The main task STATION8 performs is to communicate with the AF5000+ variable frequency drives to monitor data from the drives and send commands to them. This is performed with the Eaton AF5000+ Genesis communications driver.

The MOTOR strategy contains logic to control the positioning of the pump. This logic uses operator input, the current status of the pump and directional motors, and the current feedback signal levels of the critical variables in the waste tank to determine what the directional motor should do. This task will command the directional motor's variable frequency drive to move the pump to the operator's desired position when all conditions are correct.

The mixing pump motor is controlled through its variable frequency drive by the logic associated with the pump operation in the MOTOR strategy. The logic uses the current values of the critical variables in the tank to decide if commands from the operators to start or stop the pump motors should be honored. If allowable, commands are sent to change the pump's operating status.

There is logic embedded in the MOTOR strategy to abort the directional or pump motor operation. This task will immediately shut down operation of either motor if any of the abort criteria becomes active. STATION8 will display alarm conditions in the alarm summary and warn the operator with an audible alarm when a signal goes into alarm state.

The tests that are performed on the waste in the tank involve operating the pump at different speeds and at different angles. A test is selected using a display of the MOTOR strategy on the STATION8 console. This selection determines a predetermined set of time durations, speeds, and angles so that the test parameters can be downloaded easily to the active parameters of the position and mixing pump control logic.

PLC communications functions are enabled in this strategy to pass variables to and from the PLC.

2.3.5 Backup Station, STATION7

This station can act as the TEST strategy runtime station or the MOTOR strategy runtime station. The tasks that are embedded in each strategy would be performed by this station if it becomes a runtime station. To make this station a runtime station, the position of a manual switch must be changed and the appropriate strategy initiated in the computer.

2.3.6 Local RSS Terminals, STATION6, and STATION7

Remote supervisory stations (RSS) allow access to a runtime station strategy through graphic displays that were developed for the strategy. These network stations allow the operator to display and change values in the runtime strategy. All RSS terminals on the GEN-NET will be linked to the station runtime TEST strategy. There are two RSS terminals in the DACS trailer: STATION6 and STATION7 (when it is not operating as a runtime station).

2.3.7 Remote RSS Terminals, STATION11, STATION13, STATION15, and STATION17

These terminals are at a site remote to the DACS trailer. They function just as a RSS terminal in the DACS trailer except that the values in the TEST strategy can not be changed from these terminals. This is done by using a different set of key macros which disable functions that would ordinarily allow access to TEST strategy values.

RSS identification numbers must be unique. The broadcast option uses the low-order bit in the identifier. If the bit has the value 1, broadcasts are allowed; if it has the value 0, broadcasts are disabled. The broadcast option is used at DACS, so a 1 is used in the low-order bit making the identifier an odd number and allowing broadcasts to pass through the ARCLINKs. For this reason, all remote RSS terminals using ARCLINKs have odd numbered identification numbers.

2.3.8 File Server Support, STATION9

The data files that are logged on STATION5 must be archived. Every two hours the data files are started over and a file transfer program is executed to transfer data files from STATION5 to STATION9. The data files are copied to a 1GB JAZ drive on STATION9 and are transferred to a file server on the HLAN. STATION9 has both an Ethernet card to access the HLAN and an ARCNET card to receive files via GEN-NET.

3.0 SYSTEM FUNCTIONS

The tasks that are performed by the DACS are allocated to the PLC, the Eaton AF5000+ variable frequency drives, or the Genesis supervisory control software. The following sections give an overview of those tasks for each of the components.

3.1 INSTRUMENT MONITORING AND DATA COLLECTION FUNCTION (TEST STRATEGY)

The functions of the TEST strategy are to monitor the status and values read from the tank instruments, log data, generate alarms, and control the PLC. Essential to these functions is communication with the Modicon PLC, with the MOTOR strategy via the PLC, and with other stations on the network. Given below is an overview of these TEST strategy functions and the communication paths needed to accomplish them. The details are reserved for Chapter 4.

3.1.1 TEST Strategy Communications

The TEST strategy communicates with the Modicon PLC via the Modbus Plus network protocol. It communicates via the ARCNET with the RSS stations for operator displays, and with the STATION9 data archiving computer. Communication with the STATION8 MOTOR strategy is accomplished via the Modicon PLC and Modbus Plus. The type of data communicated via each of these pathways is discussed briefly below.

3.1.1.1 Communications with the Modicon PLC

All of the tank instrumentation data enter the TEST strategy from the PLC via the Modbus Plus network. These include data such as tank temperature readings, gas measurements, flow measurements, and much more. These data enter the TEST strategy through one of two device blocks called DEV 1 and DEV 2. The device blocks accomplish the mapping between the PLC registers and the Genesis TEST strategy I/O blocks. The device blocks together with the I/O blocks scale the data to engineering units. The data are then available to be logged to the hard disk and displayed on any of a number of user display screens.

In addition to the tank instrumentation data, the PLC will provide status information about the PLC hardware and abort status information about any measurements that have aborts associated with them. It will provide instrument failure status information for selected critical measurements.

The communication with the PLC is a two-way communication, and information is sent from the TEST strategy to the PLC. The TEST strategy sends abort limit values to the

PLC for use in the abort comparison logic. It will also send an operator-initiated abort reset signal to the PLC, which resets the PLC abort logic.

3.1.1.2 Communications with the MOTOR Strategy

Communications between the MOTOR strategy and the TEST strategy is accomplished via the PLC. There is no direct communication path between the two strategies.

3.1.1.3 Communications from the TEST Strategy to the RSS Stations

Communication with the RSS stations is via ARCNET . These stations can display any of the TEST strategy screens for data monitoring.

3.1.1.4 Communications from the TEST Strategy to the Data Archiving Computer

Data are first logged locally on STATION5 by the TEST strategy. The data are then transferred to the data archiving computer, STATION9. After a successful transfer, the data are then deleted from STATION5. The computers are linked with ARCNET, and the transfer is accomplished by using a *GENFXR* block located in the TEST strategy. The data archiving computer is networked to both ARCNET and to the HLAN. The data are transferred from STATION5 directly to the network server as well as to the JAZ drive on STATION9.

3.1.2 Data Monitoring

Data monitoring is one of the primary functions of the TEST strategy. Tank instrumentation data and system status information are presented on a number of operator display screens. The operator displays fall into four basic categories plus a MAP screen.

The MAP screen shows the organization of all of the other screens in the system. Three of the screen categories are used for monitoring and are discussed below.

3.1.2.1 Abort Status Monitoring

This group of screens include the screen CSMAIN and those below it on the MAP organization screen. These screens allow the operator to monitor critical safety-related measurements. These are measurements that need to be within predetermined limits before the pump can be operated safely. These measurements generally have automatic abort functions located in the PLC that automatically shut off the pump motor when the abort limit is exceeded. They generally have alarms associated with them that alert the operator if their value is approaching the abort limit. The current measurement value, the alarm and abort limits, and the alarm and abort status of these measurements are displayed on the abort monitoring screens.

3.1.2.2 Tank Measurement Monitoring

These screens include the MSMAIN and ASMAIN screens and the screens below these two on the MAP screen. They allow the operator to view any of the tank instrumentation measurements. Those below MSMAIN allow monitoring of the data in tabular form, on diagrams based on location within the tank, or using graphical representations such as bar graphs. Those below ASMAIN are trend screens. They allow the data to be viewed graphically as it changes through time.

3.1.2.3 System Status Monitoring

These are the screens DACS, IOSTATUS, ABRTENAB, ABRTCHEK, MININ1, MININ2, and ABRTNAME. DACS allows the operator to monitor the status of the trailer power systems and the trailer rack temperatures as well as the outside weather. IOSTATUS allows the operator to monitor the status of the Modicon I/O modules and racks. ABRTENAB, ABRTCHEK, and ABRTNAME allow the operator to monitor the status of various abort coils and enable/disable any such coils. MININ1 and MININ2 allow the operator to monitor the status of the minimum instruments required to perform a pump run.

3.1.3 PLC Control

The TEST strategy provides limited operator control of the PLC. Specifically, from any of the abort monitoring screens (see Section 3.1.2.1) the operator can reset the PLC abort coil. The PLC performs comparisons of certain critical instruments with their abort limits. If one of these measurements exceeds its abort limit then the PLC abort coil is set. The pump cannot run when this coil is set. This abort coil is latched, which means that if a measurement exceeds its abort limit only momentarily the abort coil will remain set even after the measurement has returned to a normal value. The only way to reset the abort coil is via the button on the abort monitoring screens mentioned above.

The TEST strategy is responsible for downloading the abort limits for these measurements to the PLC. This is done once upon startup of the strategy. The only exceptions to these are those limits that change based upon the pump speed. These limits are calculated in the MOTOR strategy and sent to the PLC when a new test is selected.

3.1.4 Data Logging

The TEST strategy is responsible for logging data to the hard drive and transmitting the data to the archiving computer, STATION9. The data are used for long-term tank monitoring and to evaluate the results of the pump tests. Approximately 200 data channels are logged. These are stored in 10 history files at intervals ranging from 6 seconds to 5 minutes.

The TEST strategy creates these history files and assigns file names derived from the time and date the file is created. Data are logged until an even-numbered 2-hour boundary is reached (that is, 2:00, 4:00, 6:00, etc.). At this time the current set of history files is closed and a new set is opened. The old set of history files is then transferred to the STATION9 archiving computer and placed on the HLAN. Upon successful transfer the files are deleted from the STATION5 computer disk. If the file transfer is not successful the files remain on the STATION5 disk and another attempt is made 2 hours later. The files accumulate on the STATION5 disk until the network problem is resolved and they can successfully be transferred to STATION9.

3.1.5 Alarming Functions

An important function of the TEST strategy is to notify the operator of significant events. This is accomplished by using the alarm features of the Genesis software. The strategy can be set up such that measurements which exceed alarm limits encoded in the strategy blocks cause an audible alarm to sound. Information about the alarm is posted to a special Genesis-provided screen called the Alarm/Event Summary Screen.

The system has been designed to produce alarms when critical tank measurements approach safety limits, when critical instruments fail, when communications have been lost between the TEST strategy and the PLC or the TEST strategy and the MOTOR strategy, and when the PLC has detected an abort condition.

3.2 PUMP CONTROL FUNCTION (MOTOR STRATEGY)

The pump is controlled by two variable frequency drives, one for position and one for operation. These variable frequency drives are commanded by the Genesis MOTOR strategy using RS-232 asynchronous communications. This strategy has logic functions that control pump position, pump operation, error checking, operation aborting, test selection, network communications, and alarming.

The following section is a description of each task performed by the MOTOR strategy. Details on how the task is implemented can be found in Section 4.2 of this document.

3.2.1 Position Control

The position control portion of the Genesis strategy involves several tasks operating in parallel. The desired pump position is selected by the operator or by the current phase of an automated test. The pump position is controlled by an Eaton AF5000+ variable frequency drive. Parameters are read and set via the Genesis AF5000 driver. The logic portion of the Genesis strategy tells the directional motor to start or stop; the direction of movement is based on the actual position feedback from the PLC.

The starting of the directional motor is controlled by automated logic and enabled by the operator. The automated logic will seek to move the pump within a desired range. When enabled, the logic will set up the direction the pump needs to move and then start the directional motor if the pump is not within the desired range.

The directional motor cannot be started if the pump motor is running. This is a safeguard to ensure equipment is not damaged. At no time are both the directional motor and the pump motor to operate at the same time.

The actual position of the pump must be outside of the desired range for the directional motor to start. If all safeguard conditions are met and the pump is directed to move by the operator, a command is sent to the Eaton AF5000+ variable frequency drive to start movement of the pump.

The directional motor will stop if the actual position of the pump is within the desired range; it will also stop if the actual position exceeds the limits of movement or overshoots the desired range. An abort condition will also cause the directional motor to stop. The operator can manually stop the directional motor from the console or from the stop button on the VFD control panel. If any logic conditions are met that specify halting of the directional motor, a command is sent to the Eaton AF5000+ variable frequency drive to stop the movement of the pump.

Communications to the Eaton AF5000+ variable frequency drive are enabled at all times. Parameters can only be read from and sent to the drive when the drive is powered up and receiving asynchronous communication requests. The communication task is responsible for sending control commands and desired operating conditions to the variable frequency drive. Feedback status from the directional motor is displayed to the operator and is also passed to the PLC abort logic via Modbus Plus communications.

The current position of the pump is requested from the PLC. When the operator requests a position change, the actual position is compared with the desired position to determine if the directional motor should be started. After being started, it is stopped automatically when it is within range or when it overshoots the desired range. "Within range" is defined by a dead-band of 2 degrees above and below the desired position but this range is adjustable by the operator. When the pump is within this range the directional motor is not started. If the pump is out of the desired range and a request to change the position is made, the direction to move is automatically determined by the position control logic. The direction signal is then passed to the variable frequency drive and the directional motor is started if all critical safety checks are passed.

If for any reason a condition is preventing a successful start of the directional motor an indicator denoted by a question mark will flash on the display. By clicking on the indicator the source of the problem will be displayed to the operator.

When the directional motor is running, limit-checking on the position of the pump assures that the pump will not be moved to a position beyond the maximum bounds set for the pump position. These limits are adjustable by the operator.

The pump can be moved by operator request before or after pump motor operation. There are safeguards in place to disable pump movement during pump operation to protect equipment and test results. There is automated logic that will disable movement after the pump is within the desired range. Any time the directional motor is stopped the movement of the pump is disabled. The logic is set up so the operator must enable position control every time the pump is to be moved.

3.2.2 Pump Control

Several tasks that execute in parallel comprise the pump control. The pump is driven by a motor-controlled Eaton AF5000+ variable frequency drive. Commands to start, stop, and change speeds are processed by a Genesis AF5000 communication driver.

Abort logic is provided to ensure safe operation of the pump. Pump operating status is displayed to the operator and passed on to the PLC for abort logic and for STATION5 to data log critical signals.

An operator request will start the pump provided preventative conditions do not disallow it. An operator can start the pump from the STATION8 console if the directional motor is not operating, all permissives are satisfied, and the enable pump control bit has been set by the operator. If the start logic is satisfied, a command is sent to the Eaton AF5000+ variable frequency drive that controls the pump motor.

If for any reason a condition is preventing a successful start of the pump motor an indicator denoted by a question mark will flash on the display. By clicking on the indicator the source of the problem will be displayed to the operator.

The pump can be halted automatically from the elapsed test timer, an abort condition, or a manual request by an operator. The operator can halt the operation manually by selecting the stop pump button on the STATION8 console or the stop button on the VFD control panel. The stop button on the STATION8 console requires two consecutive presses to prevent accidental stopping of the pump. If the logical conditions to halt the pump motor are met, a command is sent to the Eaton AF5000+ variable frequency drive to stop the pump.

Communications from Genesis to the Eaton AF5000+ variable frequency drive which control the pump motor are enabled at all times, parameters can only be read and sent to the drive when the drive is powered up and receiving asynchronous communication requests. The MOTOR strategy is responsible for sending control commands and desired operating conditions to the pump motor and retrieving feedback status from the pump motor.

The current operating values in the variable frequency drive are compared with the desired operating values set by the operator. If the current operating parameters deviate from the desired operating parameters the operator is warned and the pump is not started. For instance, the desired pump motor speed is compared with the actual pump speed and a warning is displayed to the operator if difference is more than 5 RPM of the desired speed. The acceleration, deceleration, and maximum speed from the pump VFD are also compared with the desired operating conditions.

The predicted motor current draw is calculated based on the desired speed of the motor and an alarm limit is set on the motor current 10 amps above this limit. The operator will be warned during pump operation if the motor current exceeds this limit. An abort limit is set 20 amps above the predicted current draw. The PLC logic will send an abort command if the motor current exceeds this limit. There is a delay of 6 seconds after the pump is started or changes speeds before this alarm and abort logic come into effect for the motor amps.

3.2.3 Abort

The directional motor or pump motor will automatically halt operation if an abort condition becomes active from the abort coil in the PLC, a failure in AF5000+ communications, or a failure in PLC communications to STATION8 or STATION5.

There is limit-checking built into the logic used to safeguard the operation of the pump and directional motors. Feedback signals from the PLC are compared with abort limits and desired operating conditions set by the operator (directly or by test selection). Several instrument signals are also monitored by the PLC to ensure safe operating conditions. The PLC can command the mixing pump control and directional control motors to stop if any of these signals exceed their operating range. There are instrument and motor signals that will cause the abort coil in the PLC to become active, as detailed in Section 4.

VR232050	Mixer pump speed high.
VR232040	Mixer pump motor amps high.
MIP00001	Moisture in pump motor oil.
TIR12A01	Mixer pump motor oil temperature A high.
TIR12A02	Mixer pump motor oil temperature B high.
ZIMPE143	Position CW limit switch.
ZIMPE144	Position CCW limit switch.
WIR12A01	Pump support column strain #1 high.
WIR12A02	Pump support column strain #2 high.
WIR12A03	Pump support column strain #3 high.
WIR12A04	Pump support column strain #4 high.

The status of the PLC communications with STATION5 and STATION8 is monitored at all times, and any time if the PLC communications fails both motors will be halted. A failure in communications with either of the AF5000+ variable frequency drives will also cause an abort condition which will stop both motors. If communications to the VFDs is halted it will probably be unlikely that the stop command will be successfully sent to the VFDs. This condition assumes communications has faulted and revived itself. In any case an alarm warning will be indicated to the operator that an abort condition has occurred and which one of the four sources has caused the motors to stop.

3.2.4 Test Operation

There are different tests that can be executed by the operators. Each test specifies the position of the pump, the speed of the pump, the duration of pump operation, and the time delay between pump operations, all of which have an effect on the critical feedback signals. The MOTOR strategy is responsible for controlling the pump under stringent guidelines set up by the design engineers.

Each test has a specific number associated with the parameter settings for the test. The test engineer cannot override the test parameters that control the pump unless the mixing pump motor display is used, which is a manual control screen for pump operation. The operator can select the test to be executed from the console and must set the values of the test to the active parameters, by selecting the SET VALUES button from the console. The test operation must then be enabled before starting the test.

When a test is active an elapsed timer keeps track of the overall duration of the test. Another timer keeps track of how long the pump is in operation. An early warning is activated before the elapsed time of pump operation reaches the desired operation time. If either of these elapsed timers reach the desired time during pump operation the test will be aborted and the pump will be stopped. The PLC times pump operation and the overall test time. The desired test time values are passed to the PLC from the test selection with network communications to STATION5.

The pump may become active several times during a test at different positions. The operator will be prompted to move the pump when the desired time of pump operation has been achieved at the current position. After pump operation has been completed at all desired positions, the end of the test is delayed until all critical data are monitored for an appropriate period of time after pump operation is suspended.

During test selection and operation there are several procedures the test engineer must follow to begin the test. The system sets control points on the STATION8 console to different colors to show the operators which procedures have been performed and what functions have been enabled.

3.2.5 PLC Communications

PLC communications are enabled in the MOTOR strategy for several functions. One is to receive the actual position of the pump to be used in the position pump logic. Also the PLC abort coil status is read into the strategy to stop either motor and disallow the start of a motor if it is active. The desired operating parameters and abort levels that are set in the MOTOR strategy are sent to the PLC logic to be used for logic. The current values of the critical variables from the VFDs are also sent to the PLC for abort logic and to be transferred to the TEST strategy for display on STATION5.

3.2.6 Alarming

The operators are warned of signals that exceed limits and the status of events with the alarming function of Genesis. When an alarm condition arises the computer drives an amplified audible alarm and the tag name of the signal that caused it is displayed in the lower right corner of the display. A history of alarm conditions can be viewed from the alarm summary.

The alarms set in the motor strategy pertain mostly to the pump motor and the directional motor. There are some additional signals that are received from the PLC that are also alarmed.

Operator activity is also tracked in the alarm/event summary. If a value is changed by the operator, a record of the change is logged into the alarm/event summary.

3.2.7 System Parameters

There are special algorithm functions in the MOTOR strategy that are used to execute Report & Recipe code, and for version tracking. Please refer to Section 4.2.8 of this document for details on these functions.

3.3 MODICON PLC FUNCTIONS

The functions of the Modicon PLC are to input data from the field, to abort the pump when critical channels are beyond their abort limits, to provide control and alarm signals and the logic to generate them, to provide timers and enable logic for pump testing, and to provide data and status information to Genesis. Each of these are discussed functionally in the following sections. The details are in Section 4.3.

3.3.1 Data Collection from the Field

Data enter the PLC via any of a number of Modicon input modules. Different types of modules can accept analog voltage or current inputs, thermocouple inputs, digital inputs of

varying levels or ASCII serial inputs. There are corresponding types of modules for data output.

In order to access data from a module, it must be entered into the PLC configuration table known as the Traffic Cop. This involves specifying the module's type, physical location, and the PLC registers which will be used for communication with the module. The physical location is specified in terms of the Modicon I/O drop, the I/O rack within the drop, and the slot within the rack which contains the module.

For most modules, the PLC I/O hardware places the data into the specified registers. From there, the data can be used within the PLC if further logic operations or abort comparisons are required.

The exceptions to this are the thermocouple input modules and the ASCII/BASIC input modules. Both of these require that PLC logic be provided to program module parameters, control the operation of the modules, and handle the interface between the modules and the PLC. In addition, the ASCII/BASIC module contains a built-in BASIC interpreter. A BASIC program must be loaded into this module to control the input of data into the module as well as the interface between the module and the PLC. The logic and programs which operate these modules will be discussed in Section 4.3.1.1.

The data are copied into a contiguous set of registers before being sent to Genesis. This is done to increase the PLC to Genesis communication efficiency.

3.3.2 Abort Functions

The PLC is responsible for providing an automatic pump shutoff signal (known as the abort signal) whenever certain safety-related or other critical measurements exceed predetermined limits. This function is provided in the PLC rather than in Genesis since the response time of the PLC is much faster (on the order of 50 ms).

The abort limits are sent to the PLC from Genesis (see Section 3.1.5). The PLC then continually compares the incoming critical measurements to their abort limits and sends the abort signal if any of the limits are exceeded.

3.3.3 Control Points and Alarm Outputs

The PLC abort output discussed above is one example of the control and alarm functions of the PLC that is important enough to deserve its own section. The other control outputs provided by the PLC are: a signal which triggers the Nicolet data collection system whenever there are indications of a tank rollover, and a signal which shuts off the VDTT flow meters when a high strain is detected on certain risers.

The PLC also contains a watchdog timer which expects to receive regular updates from Genesis. If communications from Genesis to the PLC fail, a signal is sent which sounds an audible alarm. Also, an oscillating signal is provided from the PLC which resets watchdog timers in Genesis to monitor communication from the PLC to Genesis.

3.3.4 Test Timers and Enable Logic

The PLC contains a timer which times the pump tests. The time allowed for the test is sent from Genesis. The PLC timer then runs whenever the pump is running and shuts down the pump whenever the allowed time is exceeded. This provides a backup means for shutting off the pump when the allowed time for the test has elapsed. In most cases, the operator or the MOTOR strategy will have already shut off the pump before the allowed time has elapsed.

The PLC also contains test enable logic. This is a logic mechanism which ensures that the timer is set and abort parameter changes are accomplished before the pump is run.

3.3.5 Special Data and Status Information

In most cases, data can be made available to Genesis without any further PLC logic other than the module configuration setup (see Section 3.3.1) and copying to contiguous registers. However, some PLC logic is needed to extract data or status information to be sent to Genesis. The status information provides information about the primary and hot standby PLCs and the I/O drop and module status. By extracting only the information needed in the PLC the amount of I/O between the PLC and Genesis can be reduced.

Also, there is PLC logic which monitors the high-frequency strain alarms for momentary triggers which may otherwise occur too fast to cause an alarm in Genesis.

3.4 INSTRUMENTATION

DACS instrumentation provides measurements which can be organized in the following categories: measurements for mixer pump operation, waste tank conditions, riser strain, and DACS trailer status and area monitors. Each of these is discussed in the following sections.

Data enter the DACS via one of four general methods. Knowledge of the method used is necessary in order to determine the resolution of the data.

Most of the analog measurements enter the DACS through Modicon I/O modules which are set up to receive signals ranging from 4-20 mA, 1-5 volts, or 0-5 volts. These modules digitize the signals to a resolution of 12 bits. The range column of the I/O list gives the corresponding range in engineering units. This represents the full range of the input signal. The engineering unit range is used to scale the digitized input, and this scaled value is recorded to disk. The useful range does not always correspond to the full range. In some cases, there is an entry in the subrange column of the channel list which specifies the useful range. If the digitized input signal falls outside of the subrange, the recorded value of the measurement will be pegged at one or the other boundaries of the subrange.

The MIT17B, MIT17C, and tank bottom and side thermocouple measurements enter DACS via Modicon thermocouple modules. These modules are set to produce double

precision readings with 0.1 degree F resolution as their output which is recorded without any further scaling.

Data from several analytical instruments enter the DACS via Modicon ASCII/BASIC modules. These modules have built-in BASIC interpreters. BASIC programs are written through which the modules communicate with the instruments via RS-232. The data are then placed in Modicon PLC registers in whatever format is agreed upon with the experimenters. This method allows considerable flexibility concerning the ultimate resolution of the data. The GC1, GC2, GC3, FTIR, and Photo NH3 data all enter the DACS via ASCII/BASIC modules.

Data from the variable speed drives (speed, current, and voltage) enter the DACS via the AF5000 device driver. This is the only category of data which does not enter via the PLC.

3.4.1 Mixer Pump Operation Measurements

DACS provides measurements which monitor the mixer pump operation. There are measurements for the operating characteristics of the pump motors and for detecting the characteristics and motion of the waste at the pump discharge nozzles and within the pump column.

3.4.1.1 Mixer Pump Motor Operation

These measurements provide information about the operating state of the mixer pump motor and the pump directional motor. Most of these measurements originate with the variable speed drive and first enter the MOTOR strategy over the RS-232 link. There are 12 measurements giving the motor speed (RPM), the motor voltage, current, percent of load voltage frequency and set-point speed for each motor.

In addition there are measurements for mixer pump motor oil temperature and directional motor position which enter via the Modicon PLC. There are also digital signals which detect moisture in the pump motor oil and when the pump rotational position is at one of its limits.

3.4.1.2 Discharge Nozzle and Pump Column

DACS provides a set of measurements for detecting the motion and characteristics of the waste as it moves through the mixer pump. These include measurements for discharge nozzle waste temperature, flow and pressure, nozzle tap pressures, and pump column pressure. There are a total of 16 measurements sampled once every 6 seconds.

3.4.2 Tank and Waste Measurements

The bulk of DACS measurements provide information about the characteristics of the tank waste and the conditions inside the tank.

3.4.2.1 Tank and Waste Temperatures

There are two multifunction instrument "trees" (MIT) within the tank called MIT17B and MIT17C. Each of these instrument trees contain 22 thermocouples spaced along the length of the tree for measuring the temperature of the waste at various levels within the tank. Both MIT17B and MIT17C use type K thermocouples. These are sampled by Genesis every 12 seconds.

In addition there are 26 type J thermocouples which measure temperatures around the bottom and side of the inner waste tank. These are sampled every 30 seconds.

3.4.2.2 Tank Waste Level

There is one radar gauge measurement of tank level, a wire level gauge indicating tank level, and a digital measurement which provides a signal if the tank level exceeds a manual set point.

3.4.2.3 Tank Pressures

There are two instruments which measure dome space pressure.

3.4.2.4 Vent Header

There are measurements for vent header flow rate, temperature and relative humidity, and hydrogen concentration.

3.4.2.5 Gas Concentrations

There are three gas chromatographs, called GC1, GC2 and GC3, which provide hydrogen concentration measurements to DACS. These enter the PLC as serial ASCII data through the Modicon ASCII/BASIC modules (see Section 4.3.1.3). The data from GC1 and GC2 are compiled together and enter through one of the ASCII-BASIC modules. The GC3 data is compiled with the infrared spectrometer and photo-NH3 data and sent as another ASCII stream which enters the other ASCII/BASIC module (see Section 4.3.1.2). Information sent with the hydrogen concentrations gives the time of the sample, file ID numbers which index the sample, and hydrogen retention time.

An infrared spectrometer provides measurements of NH3 and N2O concentrations. This is sent along with the gas chromatograph data and includes the file ID and sample time numbers.

A separate set of measurements monitor the gas chromatograph equipment and the IR spectrometer. This information enters DACS via standard Modicon input modules.

Four hydrogen monitors which measure hydrogen concentration in various parts of the dome space and the vent header complete the set of gas-measurement equipment.

3.4.2.6 Miscellaneous Tank Measurements

There are measurements for the in-tank camera enclosure pressure and nitrogen supply. There are four fluid velocity measurements for measuring waste movement within the tank.

3.4.3 Strain Measurements

There are measurements which provide strain readings for the instrument trees and the pump column. These measure the strain on these trees which can be produced by movement of the tank waste. There are measurements for low- and high-frequency strains. There are a total of 11 low-frequency strain measurements; 4 for the pump column, 3 for riser 1B, and 2 each for risers 17C and 17B. There are also two pump column vibration measurements.

The high-frequency strain measurements (using the same transducers as the low-frequency strain measurements) are recorded during tank rollover and during pump operation using the Nicolet storage oscilloscope. If these measurements exceed alarm and abort limits, they are sent to the PLC to cause an alarm or abort condition.

3.4.4 Trailer and Area Monitors

Environmental measurements provide information about the DACS trailer status. Sensors inside the trailer measure rack temperatures and the power system status. Other devices outside the trailer indicate weather conditions outside the trailer such as air temperature, wind speed and direction, barometric pressure and relative humidity.

There is also an area radiation monitor which provides tank farm radiation readings and alarms.

4.0 DESIGN DETAILS

This section presents the details of the DACS design. It includes a discussion of the TEST strategy, the MOTOR strategy, and the PLC, focusing upon how each of these accomplishes their various tasks.

4.1 INSTRUMENT MONITORING AND DATA COLLECTION FUNCTION (TEST STRATEGY)

In Section 3.1, the functions of the TEST strategy were discussed, including communications with the Modicon PLC and with the MOTOR strategy, data and status monitoring, alarm generation, data logging, and PLC control. In this section, the details of how these functions are accomplished will be presented.

Some of these functions are accomplished by the TEST strategy blocks, some by the display screens, and some via built-in features of the Genesis system. For those functions that are accomplished via strategy blocks, sample portions of the strategy are shown and explained in the following sections. Figure 3 is a general overview of the TEST strategy that delineates functional regions. This is provided as an aid in placing the sample strategy portions into the context of the strategy as a whole.

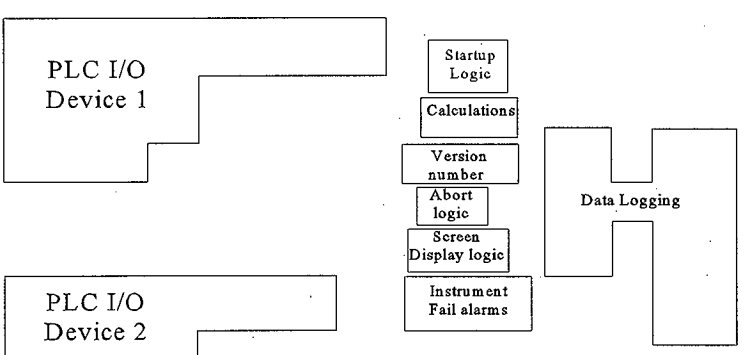


Fig. 3. General overview of TEST strategy functional regions.

Some of these regions correspond directly to the TEST strategy functions; others are strategy regions that perform supporting roles to these functions. Some brief comments about the regions follow:

- **PLC I/O Regions (Device 1 and Device 2):** The blocks in these are involved in communications with the Modicon PLC. This will be discussed in detail in Section 4.1.1.1.

  The PLC I/O Device 1 regions contains most of the standard instrumentation I/O. It contains the abort limit blocks as well. These blocks are set to the appropriate abort limit values via a report on startup of the strategy (see Section 4.1.3). The values are then sent to the PLC where the actual abort comparison takes place.

  The PLC I/O Device 2 region contains I/O for the ASCII/BASIC modules and thermocouple modules. The ASCII/BASIC modules can be used to interface a variety of serial scientific devices (see Section 4.3.1). This region contains blocks that implement the Genesis portion of the Genesis-to-PLC communications watchdog timer (see Section 4.1.5). Input via Device 2 is PLC status information used to monitor the status of the PLC and all of the PLC I/O modules (see Section 4.1.2).

- **Startup Logic Region:** This region contains blocks that initiate data logging and the downloading of abort limits to the PLC upon startup (see Sections 4.1.4 and 4.1.3).

- **Calculations Region:** These blocks perform calculations related to providing operator alarms and data logging (see Sections 4.1.4 and 4.1.5).

- **Version Number:** This in an enlarged dummy logic block whose tag name is set to the current strategy version number.

- **Abort Logic Region:** This region produces operator alarms when pump problems occur. It also contains the abort coil reset logic (see Section 4.1.5).

- **Screen Display Logic Region:** This contains logic pertaining to operator displays (see Section 4.1.2).

- **Instrument Fail Alarms Region:** These blocks provide instrument failure alarms; they will be discussed in Section 4.1.5.

- **Data Logging Region:** These blocks implement the Data Logging function, to be discussed in Section 4.1.4.

4.1.1 Communications

The TEST strategy communicates with the Modicon PLC, with the MOTOR strategy (indirectly), and with the other stations on the ARCNET. The following is a discussion of the details of each of these.

4.1.1.1 Communications between TEST Strategy and the Modicon PLC

The TEST strategy communicates with the Modicon PLC via the Modbus Plus network protocol.

Communications between Genesis and the PLC is accomplished via two device blocks, called DEV 1 and DEV 2. DEV 1 is set to scale 12-bit raw data (0-4095) and DEV 2 is set to scale 16-bit raw data (0-65535). The Modicon PLC uses 16-bit registers, but most of the data from the output modules are 12-bit data. In general, DEV 1 is used for most data, and DEV 2 is used when a full 16-bit PLC register is to be transferred.

Each device block has 16 groups that can be set up to accomplish a mapping between PLC registers and Genesis I/O blocks. For each group to be used, one must specify the type of Genesis I/O block to be connected, the type of PLC register, the starting PLC register address, the Modbus Plus node address of the source data, and the number of connection points within the group. The Genesis I/O block types available are *AIN*, *AOUT*, *DIN*, *DOUT*, *PAIN*, *PAOT*, *PAIO*, *PDIN*, *PDOT*, and *PDIO*. In the above terminology "A" stands for analog, "D" for digital, "P" for packed, "IN" for input, "OUT" or "OT" for output, and "IO" for bidirectional input output. The PLC registers available are: CS (Coil/Status bits–0xxxxx), IS (Input/Status Bits–1xxxxx), IR (Input Registers–3xxxxx), and HR (Holding Registers– 4xxxxx). The starting address specifies the xxxxx part of the above registers.

The Genesis I/O block and the PLC register type must be compatible. Coil/Status and Input/Status registers must be paired with digital input or output blocks. Input registers must be paired with analog input blocks, and holding registers can be paired with either analog or digital input or output blocks.

The amount of data that can be transferred within one group depends upon the type of I/O block and PLC register that is used. Each group can have up to 16 connections (numbered 0-15). Nonpacked blocks transfer one register or one bit per connection depending on whether the block is analog or digital. Packed blocks can transfer 16 analog registers or 256 (16x16) digital bits. These bits could be 256 coil/status (0xxxx) bits or 16 holding registers (4xxxx) interpreted digitally. Packed communication is much more efficient and should be used whenever possible.

The device blocks have several configuration parameters which must be specified. One of these is the device # parameter. This codes for the memory base address of the SA-85

interface card. The SA-85 card is the Modbus Plus network interface card which allows the PCs to be connected to the Modbus Plus network. Although the SA-85 card can be set for memory base addresses ranging from C0000-EF800, the Modbus Plus driver only allows four choices: DD000, DD800, DE000, and DE800.

The Scan Task parameter assigns a scan number which corresponds to COM Port number (1 to 4). Although the SA-85 card does not use a COM port, the COM port corresponding to the scan task parameter must be set to INSTALLED in the COMM CNFG menu of the Genesis configurator (under System Configuration). The Scan Period which is defined here determines how often the Modbus Plus driver is executed. The scan task parameter is currently set to 0.5 second.

The device block configuration menu also has parameters for Modbus plus network routing (not used in our system), BCD interpretation of data (also not used) and to specify the bit-order of holding registers (4xxxxx) interpreted bitwise.

Following are the device setup parameters and the group assignments for DEV 1 and DEV 2 for the TEST Strategy:

DEV 1

Device #	2	Scan Task	Task 1
AIN Raw Count-Zero	0	AOUT Raw Count - Zero	0
AIN Raw Range	4095	AOUT Raw Range	4095
BCD	NO	Bit Order	16-1

Group	I/O Type	PLC Type	Register Range	Connections															
				0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	PAIN	HR	402001-402128	y	--	y	y	y	--	--	y	y	y	y	y	y	y	--	--
2	PDIN	CS	002001-002256	y	y	y	y	--	--	--	--	--	--	--	--	--	--	--	--
3	DOUT	CS	000241-000256	y	--	--	--	--	--	--	--	--	y	--	--	--	--	--	--
4	PAOT	HR	400200-400255	y	--	--	y	y	y	y	--	--	--	--	--	--	--	--	--
5	PDIO	CS	002052-002147	y	y	y	y	y	y	--	--	--	--	--	--	--	--	--	--
6	PDIO	CS	000476-000731	--	--	y	y	y	y	--	--	--	--	--	--	--	--	--	--
7	PDIO	CS	000220-000475	y	y	y	y	y	y	y	y	--	--	--	--	--	--	--	--
8	---	---	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
9	PAIN	HR	400200-400255	y	y	--	y	y	y	y	--	--	--	--	--	--	--	--	--
10	PDOT	CS	002150-002165	y	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
11	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
12	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
13	---	---	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
14	---	---	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
15	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

DEV 2

Device #	2	Scan Task	Task 1
AIN Raw Count-Zero	0	AOUT Raw Count - Zero	0
AIN Raw Range	65535	AOUT Raw Range	65535
BCD	NO	Bit Order	16-1

Group	I/O Type	PLC Type	Register Range	Connections															
				0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	DOUT	CS	000213	y	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
2	PAIN	HR	402129-402256	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	--
3*	PDIN	HR	400108-400123	y	y	y	y	--	--	--	--	--	--	--	--	--	--	--	--
4	PDIN	CS	400112-400127	y	y	y	y	--	--	--	--	--	--	--	--	--	--	--	--
5	PDIN	CS	400116-400131	y	y	y	y	--	--	--	--	--	--	--	--	--	--	--	--
6	PDIN	CS	400120-400135	y	y	--	--	--	--	--	--	--	--	--	--	--	--	--	--
7	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
8	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
9	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
10	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
11	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
12	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
13	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
14	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
15	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
16	---	--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

*Accurate for corrected Modbus Plus driver. Alternate grouping currently in effect pending Modbus Plus driver update.

Figure 4 shows a portion of the PLC I/O section of the STATION5 strategy. For efficient communications, most I/O is done via packed blocks. The packed blocks are connected to the device block. To allow each channel to have its own tag name and description, regular *AIN* blocks are connected to the packed blocks as shown in the figure. The I/O blocks are (with few exceptions) organized by device block group such that each horizontal row in the PLC I/O area contains the I/O for one device block group starting with Group 1 at the top.

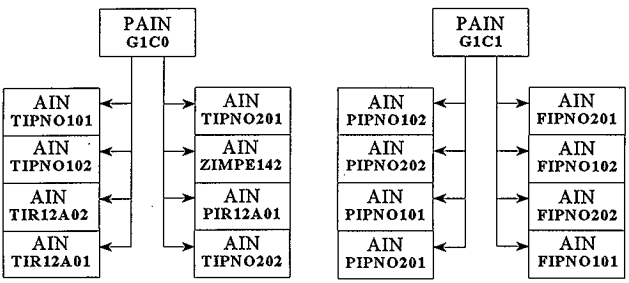


Fig. 4. PLC I/O strategy detail.

The packed blocks are named to reflect their device block group and connection number within the group. Thus, the *PAIN* block <G1CO> is connected to Group 1, connection 0 of device block 1. The packed blocks connected to device 1 are named beginning with the letter "G." Those connected to device 2 begin with the letter "H." Thus, a block called "H6C3" would be connected to Group 6, connection 3 of device block 2.

The regular *AIN* blocks connected to the packed blocks are arranged by connection number of the packed block starting at the upper left and reading down. Thus, in the figure above, <TIPNO101> is connection 1, <TIPNO102> is connection 2, <TIR12A01> is connection 3, <TIR12A02> is connection 4, <TIPNO201> is connection 5, <ZIMPE142> is connection 6, <PIR12A01> is connection 7, and <TIPNO202> is connection 8. This convention has been followed throughout.

The *PAIN* blocks include range information for the incoming raw values that allow them to be scaled to engineering units. This scaling information is duplicated in the *AIN* blocks. In addition, the *AIN* blocks are configured to include the engineering units, the measurement description, alarm values (if any), and scan period.

The scan period specifies the time between measurement updates. Genesis only allows scan periods of 0.05, 0.1, 0.25, 0.5, 1, 2, 6, 12, and 30 seconds. However, the scan period actually achievable depends upon the number of blocks in the system and the communication rate with the PLC. In addition, the overall communication scan period for each communication link must be specified. The scan periods of individual blocks cannot exceed the overall communication scan period. The communication link with the PLC is currently set to a 0.5 second scan period. Most of the measurements are set for 6-, 12-, or 30-second scan periods.

There are two cases where additional manipulation is needed to present the data in engineering format. The measurements are <ARMGAMMA> and <PHO-NH3>. The

signal sent to Genesis for <ARMGAMMA> is actually an exponent, so a calculation in a Genesis *F(X)* block is done to create the actual measurement value from the exponent. The signal <PHO-NH3> is sent originally as two halves, a most significant and least significant portion. These are combined into one measurement value in the TEST strategy using a *F(X)* block.

4.1.1.2 Communications with the MOTOR Strategy

There is no direct communication path between the TEST and MOTOR strategies. Instead, communication between them is accomplished via the PLC and Modbus Plus. Data relating to the variable speed drives for pump motor control and directional motor control are sent from the MOTOR strategy to the PLC, then are relayed to the TEST strategy.

Also, the TEST strategy receives alarm and abort limit values for pump motor speed, pump motor current and pump discharge pressure. These values are calculated in the MOTOR strategy and sent to the PLC. The actual abort and alarm comparisons are done in the PLC, but the calculated limit values are sent from the PLC to the TEST strategy.

Also, both the TEST and MOTOR strategies have watchdog timers to monitor the communication between the strategies and the PLC. When one of these communication paths fails, the other strategy is notified of the communication failure.

4.1.1.3 Other Network Communications

The TEST strategy communicates over the ARCNET to RSS stations for remote screen display information and to the STATION9 computer for data archiving. As mentioned above, the details of the ARCNET configuration can be found in Section 4.4. The data archiving function will be discussed in Section 4.1.4.

4.1.2 Data Monitoring

The data and status monitoring functions of the TEST strategy are accomplished by displaying information to the operators on any of a number of displays including Genesis-provided system runtime trend and history displays. In addition to providing the monitoring function, some displays allow the operator to input control signals that can initiate processes in Genesis or ultimately in the PLC. In addition, all displays have a button which, when clicked on with the mouse, allow other displays to be brought up on that particular monitor.

Displays can be brought up independently on the STATION5, STATION6, and STATION7 computers in the DACS trailer, as well as the four remote stations (STATION11 in Building MO-278, STATION13 in the 306E Building, STATION15 in Building 2750 and STATION17 in the office of the software developers). All data that travel to and from the remote displays (in RSS stations) go through the master computer (STATION5) via ARCNET.

The displays are organized in a hierarchy based on their function; other displays can be accessed from the current one. At the top of the hierarchy is the WELCOME display. This display appears when the system is first brought up. It has buttons that allow access to the MAP, CSMAIN, MSMAIN, and ASMAIN displays.

- The MAP display shows a representation of the hierarchy and allows access to any other display in the system.

- CSMAIN is the overall abort status monitoring display. It allows access to the other displays listed in the "Abort Status Monitoring Screens" table below, which provide more detailed information about the abort status of various instruments. All the members of this group include a button that allows the user to reset the PLC abort coil. This function is discussed in Section 4.1.5.

- MSMAIN has a schematic representation of the tank showing the locations of the various instrument trees. It allows the user to bring up displays that show data from the instruments on these trees. Its buttons allow the user to call up other tank measurement monitoring displays which present the data in a variety of formats. These displays are listed in the "Tank Measurement Monitoring Screens" at the end of this section excluding those indicated as runtime trend screens.

- ASMAIN allows the operator to access runtime trend and history displays. These are displays listed in the "Tank Measurement Monitoring Screens" table below. Runtime trends plot various tank measurements as a function of time, beginning with the point when the screen is first brought up. The graph grows from this point until the time allotted for the graph has passed. After that, the older values are scrolled off the edge of the graph as new values become available. If another screen is brought up on the same display, then the process must start over again. This differs from the system runtime screen discussed below.

Accessible from the MAP display are those displays listed in the "System Status Monitoring and Miscellaneous Screens" table at the end of this section. IOSTATUS and DACS allow monitoring of the PLC racks and modules and the trailer status respectively. The displays named ABRTENAB, ABRTNAME, and ABRTCHEK are used for monitoring and/or enabling/disabling coils used for aborts by the PLC. MININ1 and MININ2 are for convenience and can be used to check minimum instrument status. The "other" displays are non-monitoring screens, MAP and WELCOME, discussed above.

All of the displays have a similar header bar. The header bar has the title, the current time and date, the application name and version number, and buttons that allow the user to bring up the MAP, CSMAIN, MSMAIN, or ASMAIN displays as well as the display one level above in the hierarchy (by pressing the Page Back button). The displays have been designed so that anything in blue can be clicked on with the mouse. These are either buttons

that bring up another display or measurement values that bring up a tag details sub-window. The tag details sub-window appears at the bottom of the display and gives further information about the measurement.

Genesis provides two other facilities useful for data monitoring. One is the system runtime trend. This allows the display of runtime graphs for up to 20 predefined measurements. These differ from the runtime trend screens discussed above in that the prior data are not lost when the display is exited momentarily.

Additionally, Genesis provides the ability to examine, graphically or in tabular format, any data that are being logged to a history file on the current computer. STATION5 is the only computer that has history files available, so this feature only works on STATION5. Data logging will be discussed further in Section 4.1.4.

TEST Strategy Screens

Tank Measurement Monitoring Screen

SCREEN	SCREEN TITLE	COMMENTS
ASMAIN	Runtime Trend Selection Screen	Selection screen for runtime trend displays
GASSUM	Gas Summary	Collected gas concentrations and related items
MIT17B	Instrument Tree (1) Riser 17B (MIT)	Tank Temperatures
MIT17C	Instrument Tree Riser 17C (MIT)	Tank Temperatures
MSMAIN	Main Riser Profile	Selection screen for monitoring displays
NEW72HR	Current 72 hour history trend.	History trend - critical measurements
OLD72HR	Previous 72 hour history trend.	History trend - critical measurements
PUMP	Mixer Pump Riser 12A	Mixer pump instrument readings
PUMPOPS	Pump Operations	Runtime trend - pump operation measurements
ROLLOVER	Imminent Rollover	Runtime trends and temperature profiles to monitor for possible rollover
SUMMARY	Summary Information	Vent Header, In-tank parameters, MIT temperatures
TBSTC	Tank Bottom and Side Thermocouples	Temperature readings
TEMPRFL	MIT Temperature Profiles	Risers 17B and 17C temperature bar graphs

Abort Status Monitoring Screens

SCREEN	SCREEN TITLE	COMMENTS
CSMAIN	Automatic Alarms and Aborts	Overall alarm and abort summary display
HVTALARM	Hydrogen-Vent Header Tank	Alarm and abort details
MANABRT	Manual Aborts	H2 and NH3 concentrations
PUMPALRM	Pump Parameters	Alarm and abort details
STRNALM	Strain Gauges	Alarm and abort details
TEMPALM	Temperature	Alarm and abort details

System Status Monitoring and Miscellaneous Screens

SCREEN	SCREEN TITLE	COMMENTS
DACS	DACS Facilities Management	DACS trailer power, trailer temperature and weather station
IOSTATUS	I/O Health Status	Modicon rack and module status
MAP	Map	Shows organization of all screens and allows operator to call up any screen
WELCOME	DST 241-SY-101 Data Acquisition and Control System (DACS)	Screen that appears upon system startup
ABRTENAB	Abort Enable Checklist	Allows enable/disable of PLC abort coils and indicates coil status
ABRTCHEK	Abort Limit Checklist	Used for pump run
ABRTNAME	Abort Coil Tag Names	Identifies PLC abort coils and indicates coil status
MININ1	Minimum Instrumentation Checklist #1	Convenience
MININ2	Minimum Instrumentation Checklist #2	Convenience

4.1.3 PLC Control

The TEST strategy provides limited control of the PLC. Primarily, it allows the operator to reset the PLC's abort coil. This function was discussed in Section 4.1.2 and the strategy logic to accomplish it will be shown and discussed in Section 4.1.5.

The TEST strategy is also responsible for downloading the abort limits to the PLC. These are the values that the PLC will compare the incoming measurement value with to determine if an abort condition exists. The abort limits are sent via *AIN* blocks located in the

PLC I/O section of the strategy. These *AIN* blocks get their values from a report SETLIMS.RPS, which contains the actual abort limit values. This report is run at startup to load the *AIN* blocks with the abort limit values.

Figure 5 show the startup logic. It shows the mechanism for generating a pulse immediately upon startup as well as a delayed startup pulse. This is implemented by connecting output of the <ON> *DIN* block to the <PULON> pul block and the <DELAYON> *TON* block. The <ON> block is wired to always be 1. The <OFF> block is wired to be always 0. Since at startup, the inputs of all blocks are initialized to 0, the only time the <PULON> and <DELAYON> blocks will see a 0 to 1 transition on their inputs is at startup. This will trigger the *pul* block <PULON> and cause the *TON* block <DELAYON> to produce its output after the programmed 45-second delay elapses.

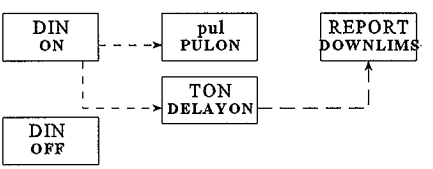


Fig. 5. Startup logic.

The <PULON> block initiates the history file naming the phasing process that was discussed in Section 4.1.4. The *TON* block <DELAYON> is the block that runs the DOWNLIMS report via the <DOWNLIMS> *REPORT* block. The 45-second delay is necessary to allow time for Genesis to PLC communication to be established before the abort limits are downloaded.

The <OFF> block is used to prevent data logging in a *HIST* block (see section 4.1.4).

4.1.4 Data Logging

The TEST strategy is responsible for logging data to the hard disk and sending the data files over the network to the archiving computer.

The data logging portion is accomplished in Genesis by the use of *HIST* blocks. The files are known in Genesis terminology as history files. Figure 6 shows an overall view of the TEST strategy history/data logging section. More detailed views of portion of this strategy section are shown later.

The top twelve large blocks of the rightmost column in the diagram above are the *HIST* blocks. The remainder of the blocks are involved with timing the creation of the history files, phasing the creation of the files and logging of the data to avoid system overload, naming the files, and copying the files automatically over the network and to different directories on the system.

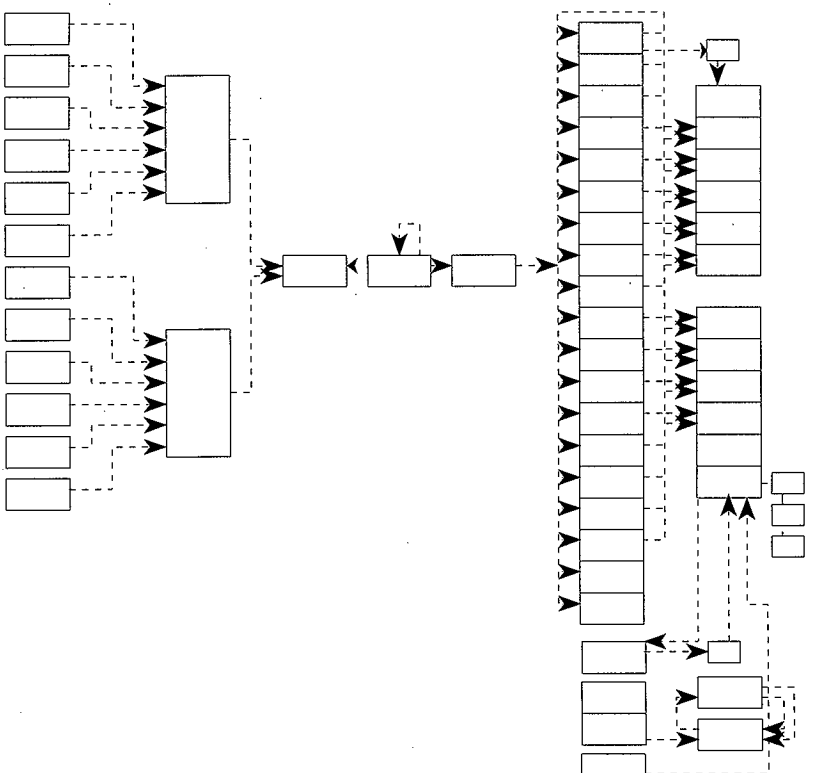


Fig. 6. *History data logging overall view.*

Two of the twelve *HIST* blocks are not used for the standard data logging which is described below. The blocks <OLD72HR> and <NEW72HR> provide a way to have at least 72 hours of readings available which can be viewed using the history display capabilities of Genesis. These history files contain measurements which can characterize the tank status for the time between pump runs when the DACS trailer is not manned.

The functionality of the history section of the strategy is by column, and the general flow of the logic is left to right. The leftmost column consists of *TIMER* blocks controlling the timing of the opening and closing of the history files. These feed into two *OR* blocks that feed into a *REPORT* block. The *REPORT* block runs a report (HISTFNAM) that creates file names for the about-to-be-opened history files based upon the date and time.

When this process is complete the report sets the *pul* block, the next block to the right above. This block generates a pulse that causes the *SHOT* block (the next one over) to generate a one-shot pulse. This pulse starts the *TON* blocks (the long column of 19 blocks). It is detrimental to start all the history blocks at the same time, so for this reason, the *TON* (timer-on-delay) blocks lie in-line with every history block. Each *TON* block has a different delay, spreading the load on the hard disk drive over the total time interval. These delay timer blocks are set to produce their outputs at varying times from the start of the one-shot pulse. They implement the phased stopping and starting of the history files by feeding into the start and stop inputs of the *HIST* blocks, the rightmost column of blocks. The one exception to this is the <RGA5> block. It is started when new data is received form the RGA5 computer and stopped after it has logged just one sample. This is done to avoid multiple logging of the same data.

These blocks control the timing of the running of another report (HISTCOPY), which copies history files to two directories on the disk \HIST and \HIST2. The *GENFXR* block at the bottom of the rightmost column is controlled by the *TON* blocks. The *GENFXR* block copies history files over the network. More details concerning this process follow.

Figure 7 shows a portion of the history strategy section that initiates a history file changeover. The system is designed to produce history files that contain 2 hours worth of data each. Every 2 hours throughout the day, the current set of history files is closed and a new set is opened. The timer blocks are set to create a pulse at a specific time of day. In this case, the timer blocks are set to create their pulses on the hour spaced 2 hours apart. When one of the timers goes off, the signal feeds through the *OR* block and causes the report HISTFNAM to be run.

This report implements the history file-naming convention. The files are named "ymdhFFFF.PRN," where

y	is the year, with 2=1992, 3=1993, etc.
m	is the month, numbered 1-9 and A-C
d	is the day, numbered 1-9 and A-V
h	is the hour, numbered 0-9 and A-N, and

FFFF is the file code that can be any one to four alphanumeric characters.

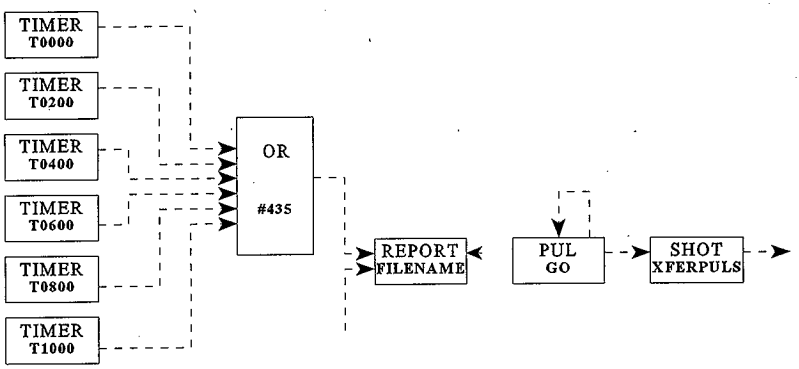


Fig. 7. Timer, report, and pulse portion of history strategy section.

HISTFNAM reads the system time/date and creates these file names. It loads the file name field of the *HIST* blocks with the new file name. It then sets the *PUL* block <GO> which in turn initiates the one-shot pulse from the *SHOT* block <XFERPULS>. The HISTFNAM report is run at system startup.

History File Names and Contents

File Name	Sample Period (sec.)	Contents				
ymdhPUMP.PRN	6 or 300	PCR12A01, VR232040,	PDPBASE, VR232050,	PITNO111, ZIMPE112,	PITNO110, ZIMPE142	VR232020
ymdhMIT2.PRN	12	TIR17B09, TIR17B14, TIR17C12,	TIR17B10, TIR17B15, TIR17C13,	TIR17B11, TIR17C09, TIR17C14,	TIR17B12, TIR17C10, TIR17C15	TIR17B13, TIR17C11,
ymdhTBSI.PRN	300	TBSTC01, TBSTC06, TBSTC11,	TBSTC02, TBSTC07, TBSTC12,	TBSTC03, TBSTC08, TBSTC13	TBSTC04, TBSTC09,	TBSTC05, TBSTC10,

History File Names and Contents

File Name	Sample Period (sec.)	Contents				
ymdhGAS-PRN	150	FT-FILE, FT-NH3C, PHO-TIME,	FT-TIME, GC3-TIME, PHO-NH3,	FT-N2OA, GC3-FILE, RGA5TND1,	FT-N2OC, GC3-RT, RGA5TND2	FT-NH3A, GC3-H2,
ymdhRGA5.PRN	per sample	RG-RUN, GC1-RT, FITMSY17, PDTMSY12,	RG-STAT, GC2-H2, TICMSY18, PITMSY07,	RG-TIME, GC2-AREA, PITMSY13, PITMSY10	GC1-H2, GC2-RT, TITMSY15,	GC1-AREA, PITMSY04, PITMSY16,
ymdhMIT3-PRN	12	TIR17B16, TIR17B21, TIR17C19,	TIR17B17, TIR17B22, TIR17C20,	TIR17B18, TIR17C16, TIR17C21,	TIR17B19, TIR17C17, TIR17C22	TIR17B20, TIR17C18,
ymdhTBS2.PRN	300	TBSTC14, TBSTC19, TBSTC24,	TBSTC15, TBSTC20, TBSTC25,	TBSTC16, TBSTC21, TBSTC26	TBSTC17, TBSTC22,	TBSTC18, TBSTC23,
ymdhSTRN.PRN	150	WIR12A01, WIR17C02,	WIR12A02, WIR1BA01,	WIR12A03, WIR1BA02,	WIR12A04, WIR1BA03	WIR17C01,
ymdhVOL.PRN	12	LIR13A01, PIR11B01, NITJSY06, WSWDIR,	FTE50001, PIR17B04, NIR17B01, WSWSPD,	FTE50002, FTE50003, WSP1, TIR12A01,	TT10001, NIRO5A01, WSH1, TIR12A02,	MT10001, NITKSY06, WST1, ENRAF
ymdhMIT1.PRN	12	TIR17B01, TIR17B06, TIR17C03, TIR17C08	TIR17B02, TIR17B07, TIR17C04,	TIR17B03, TIR17B08, TIR17C05,	TIR17B04, TIR17C01, TIR17C06,	TIR17B05, TIR17C02, TIR17C07,
NEW72HR.PRN	300	FTE50001, GC3-H2S, WYR17C01,	FTY50002, GC2-H2S, FTE50003,	PYR17B04, NIR17B01, WYR12A01	PYR11B01, WYR1BA01,	LIR13A01, WYR17B01,

Figure 8 shows some of the *TON* blocks that implement the phased starting and stopping of the *HIST* blocks. This phasing is necessary in order to spread out the load on the system imposed by the data logging operations. The system has limited processing time available for data logging as well as limited buffer space for file operations. If the strategy attempts to write too much data at once, this buffer overflows and a system message appears indicating a loss of data.

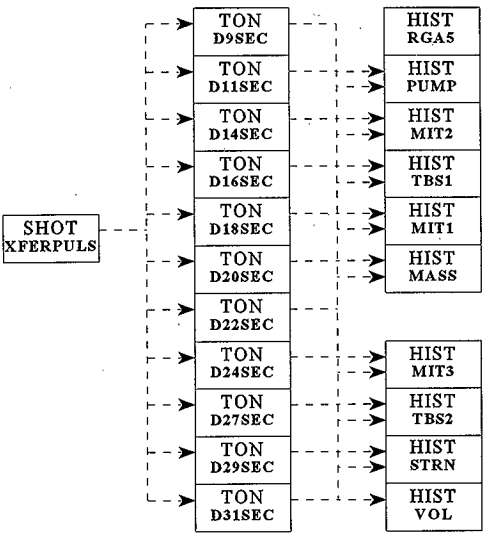


Fig. 8. Phasing and HIST blocks.

Besides the buffer overflow problem, the system resources are strained when files are opened and closed. The phasing scheme is designed to spread out the load on the system as much as possible. The scheme provides that during file changeover, no two files are started or stopped simultaneously. In addition, between changeovers, the sampling of data should be spread out as much as possible. In the current system, no more than two *HIST* blocks sample their data in any one scan period. The scheme is dependent upon the sample period in the *HIST* blocks. If the sample period of any of the blocks is changed, the phasing scheme must be adjusted accordingly. Currently, the changeover occurs as follows:

Second	Action
0	PUMP Stop
1	PUMP Start
5	MIT2 Stop

7	TBS1 Stop, PUMP Sampled
9	MIT2 Start
11	MIT1 Stop
13	PUMP Sampled
14	TBS1 Start
16	GAS Stop
18	MIT1 Start
19	PUMP Sampled
21	MIT2 Sampled
22	GAS Start
24	MIT3 Stop
25	PUMP Sampled
29	MIT3 Start
30	MIT1 Sampled
31	TBS2 Stop, PUMP Sampled
33	STRN Stop, MIT2 Sampled
35	TBS2 Start
37	PUMP Sampled
38	VOL Stop
40	STRN Start
41	MIT3 Sampled
43	VOL Start, PUMP Sampled

Given the start times above for each one of the files and the data sampling periods, one can determine when each of the samples are taken. In this case, for a 12-second period, the samples are:

Second	Files Sampled
1	PUMP
2	TBS1 (every 25th pass beginning with pass 2)
4	MASS (every 25th pass beginning with pass 15) STRN (every 25th pass beginning with pass 4)

5	MIT3
6	MIT1
7	PUMP, VOL
9	MIT2
10	MASS (every 25th pass beginning with pass 2) STRN (every 25th pass beginning with pass 17)
11	TBS2 (every 25th pass beginning with pass 3)
12	
1	(pattern repeats at this point)

The RGA5 file does not sample regularly as the others do. This file is opened, a sample taken, and closed again only when a new sample is received.

Fig. 9 shows the *GENFXR* block and an *OR* block. It also shows a few of the *TON* blocks. The *TON* block D1MIN initiates the report, HISTCOPY, 1 minute after the file changeover begins. This report copies the recently closed files to both the \HIST and \HIST2 directories then deletes them from the \TEST directory. This is done to avoid having history files build up in the directory. The HISTCOPY report renames (using the history file naming convention) copies, then deletes the alarm and event summary files.

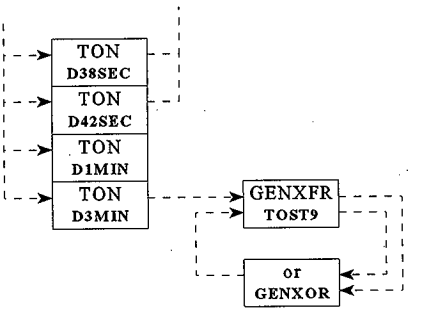


Fig. 9. *GENFXR* detail.

The *GENFXR* block is set up to copy the history files over the network. This block is initiated 3 minutes after a file changeover begins. The first pass attempts to copy files from

the \HIST directory to the \HIST directory on drive C: of STATION9. If this file transfer is successful, the files are deleted from the \HIST directory on STATION5.

The *OR* block detects one of two status outputs: either successful transfer or failure. In either case, it means that the first operation is complete. This signal is used to initiate the second file transfer operation in the *GENFXR* block. This time, files are copied from the \HIST2 directory over the network to the Jaz drive (Drive E:) of STATION9. Once again, the files are deleted from the STATION5 directory if the transfer is successful.

The history file ymdhRGA5 differs from the others in that a new sample is logged only when a new reading is received by Genesis. This avoids duplication of samples in the file. The instruments are set to produce a new sample every three minutes. If the instruments are not working properly or if there is some other problem, no data will be logged and the file could be empty. The file is still closed every two hours no matter how many samples have been taken during that time.

A discussion of the logic which implements this follows. The *F(X)* block, <FILEUPDT>, continually scans the <RG-RUN> tag. The value of this tag will change whenever a new sample is received. The <FILEUPDT> block contains logic to compare the previous value of <RG-RUN> with the current value. It sets one of its outputs (DO3) to be high (1) if the current and previous scans are equal, and low (0) otherwise. When this output transitions from 0 to 1, it starts a 45 second timer (*TON* block <RGSTLOG>). When the 45 seconds have elapsed, the sample is logged (by starting data logging for the <RGA5> *HIST* block. This 45 second delay is to insure that the value of PHO-NH3 which is calculated from MSB and LSB portions, has enough time to execute before the values are logged.

The *HIST* blocks <OLD72HR> and <NEW72HR> provide a way to have at least 72 hours of readings available which can be viewed using the history display capabilities of Genesis. These history files contain measurements which can characterize the tank status for the time between pump runs when the DACS trailer is not manned. Only <NEW72HR> is set to actually log data. When a 72 hour period has elapsed the tag <CHNG72OK> is set. The report HISTCOPY.RPS detects this and copies the file NEW72.PRN to OLD72.PRN. New data is then logged to the NEW72.PRN file. The data in the OLD72.PRN file can be examined using the history file display capabilities of Genesis.

4.1.5 Alarming Functions

Operators are notified of significant events by using the alarm capabilities of the Genesis software. Genesis provides a special screen called the alarm/event summary screen. Alarms, operator actions, and other system events are posted to this screen as they occur. In addition, the PC speaker sounds when an alarm arrives. These speakers have been amplified in the trailer producing an effective audible alarm. The operator must hit a keyboard key to silence the alarm. Information about the alarm can be found on the alarm/event summary screen. From there the alarms can be acknowledged by the operator.

Most of the alarms are generated by measurement I/O blocks. The alarm values are programmed into the block and the alarm priority is set. When the measurement value exceeds one of the alarm limits the alarm occurs. Other alarms require additional strategy logic. Portions of the TEST strategy related to the alarming function are shown in Fig. 10 and discussed below:

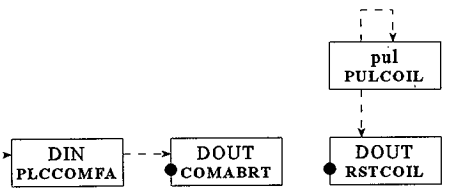


Fig. 10. Abort logic detail.

This strategy section is responsible for causing a Genesis alarm if a PLC communication failure is detected. It also contains logic for resetting the PLC abort coil.

Since the PLC abort coil remains latched following an abort, the operator needs some means of resetting the coil after the problem has been resolved. This is provided by the <PULCOIL> and <RSTCOIL> blocks shown above. When the Reset Abort Coil buttons on the abort coil status screens (see Section 4.1.2) are clicked upon, they cause the <PULCOIL> block to produce a pulse. This is sent via the <RSTCOIL> digital output block to the PLC where it causes a reset of the PLC abort coil, as well as all of the individual aborts in the PLC.

The <PLCCOMFA> block is set when Genesis detects a communication failure from the PLC to Genesis. This can be detected by the Modicon driver and accessed by a connection to the FAIL bit of an I/O block. The <PLCCOMFA> block causes a Genesis alarm whenever this condition occurs. It also attempts to send this information to the PLC so that the abort coil can be set. This is accomplished via the <COMABRT> block shown above. This is useful in the unlikely event that the communication failure has been only one way and the PLC has not detected a corresponding Genesis to PLC failure.

Figure 11 shows a portion of the strategy that generates instrument failure alarms. These blocks generate alarms when a critical instrument reads at or near zero indicating a possible instrument failure. They are also wired to the instrument failure detection

mechanisms in the PLC (see Section 4.3.2) and cause an alarm when the PLC detects an instrument failure as well.

Each of the *OR* blocks shown in Fig. 11 implements an instrument fail alarm for the measurements whose names correspond most closely to the names of the *OR* blocks. In most cases, the measurement name has an I in place of the Z in the above names.

Four of these instrument failure alarm blocks have to be handled differently. These are the ones shown as <GC1-INST>, <GC2-INST>, <GC3-INST>, <NH3-INST>, and <FT-INST> in Fig. 11. These originate from the gas measurement equipment and arrive at the PLC after having traveled through several other computers and devices. An instrument failure in this case could mean either a failure of the instrument itself or of the communication pathways involved in getting the measurement to the PLC.

The first case is handled by detecting when the measurements involved read zero. This is slightly complicated by the fact that <GC1-H2>, <GC2-H2> and <GC3-H2> already have low alarm values associated with them, so separate blocks (<GC1-ZVAL>, <GC2-ZVAL>, and <GC3-ZVAL>) have been provided to detect the zero case.

The second case is handled by the logic shown in Fig. 12. It is used to detect a failure in the communications path between the instrument and the PLC. The gas measuring equipment sends a value to the PLC known as the File ID. This value is incremented (independently for FTIR, PHOTO NH3, GC1, GC2 and GC3 data) whenever a new measurement has been taken. The *F(X)* block, <FILEUPDT>, receives this file ID from the measurements. As long as the current value is equal to the previous value, the *TON* blocks are allowed to run. When a new File ID arrives, the *TON* blocks are reset. If the *TON* blocks ever reach their predefined timeout values, the instrument failure alarm occurs. The block <RGSTLOG> is used to start the RGA5 data logging. It was discussed in section 4.1.4.

Values for FTIR, PHOTO NH3, GC1, GC2, and GC3 are forced to zero by Genesis if the TON blocks time out. This is an obvious indication to operators that the instrument has a problem. The gas values are actually forced to zero in the PLC ladder logic by use of a PDOT block (G10C0). As soon as updated data is sent to the PLC from the gas equipment, the TON blocks will reset and the display will indicate valid readings.

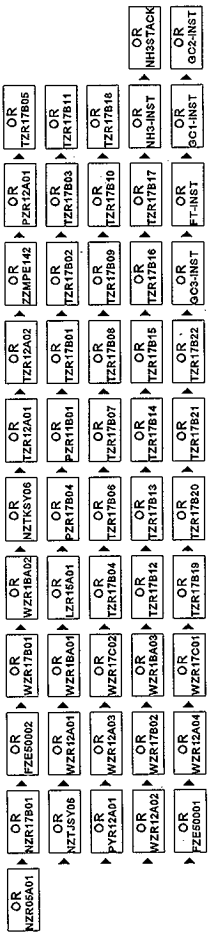


Fig. 11. Instrument fail detail.

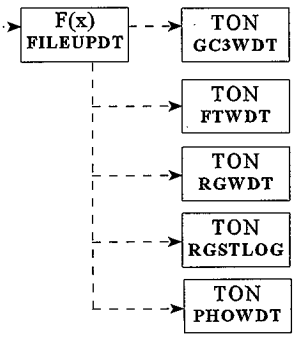


Fig. 12. Communication pathway failure detection.

4.2 PUMP CONTROL FUNCTION

The signals that actually control the pump are assigned by the Eaton AF5000+ variable frequency drives. There is one drive to control the motor that positions the pump and one drive to control the motor that operates the pump. These variable frequency drives receive commands in the form of ASCII strings from a communication driver. This ASCII driver is imbedded in the Genesis MOTOR strategy and is represented by two AF5000+ hardware communication device blocks, one for each VFD.

There are several different tasks that are performed in parallel to control the position and operation of the pump. The results of these tasks are sent as commands to the variable frequency drives or as indications to the operators. The following subsections of Section 4.2 supply the details of how the separate tasks of position control, pump control, aborting, and test operation are implemented in the MOTOR strategy. These will be followed by discussions of PLC communications, alarms, system parameters, and indication of failure of executing tasks.

4.2.1 Position Control

The following are the details related to controlling the position (i.e., direction) of the pump.

4.2.1.1 Start

The directional motor can move the pump only if the actual position is outside the desired range, the actual position is within the safe boundary, the pump is not running, and the operator enables the pump to move. The following tags of the MOTOR strategy are used to execute the start of the directional motor.

ABORTOR	BACKWARD	DISMOVE	DMOTOR
FORWARD	FSTRTDEL	GOFORWRD	GOREVERS
HILIMAND	LOLIMAND	NOTRUN	POSCNTRL
POSITION	PRUN	PSTOP	RSTRTDEL
START	STARTAND	STARTSHT	STRTAND

When the output of the *one shot* block <STARTSHT> goes high, the directional motor will be commanded to start by setting <DMOTOR.start> high. <STARTSHT> is set high when the output of the small logic *and* block <STARTAND> goes high. <STARTAND> and <STRTAND> are used as the safeguard to disable a start of movement under improper conditions. All of the following conditions must be true before the directional motor can be started.

- <ABORTOR.QNOT> is high if no abort condition is active.
- <START.DOUT> is high indicating a start request is active from the direction control logic.
- <HILIMAND.QNOT> is high indicating the position of the pump is not beyond the maximum boundary.
- <LOLIMAND.QNOT> is high indicating the position of the pump is not beyond the minimum boundary.
- <DISMOVE.QNOT> is high indicating position control is not disabled.
- <PSTOP.DOUT> is high indicating that pump motor is stopped.
- <PRUN.QNOT> is high indicating the pump motor is not running.

The small logic *or* block <START> is the block that will go high when the directional motor is successfully requested to move. <START> has two inputs to move the pump forward or backward.

- <FSTRTDEL.DOUT> goes high when the pump is to move in the forward direction. <FSTRTDEL> will go high when both of the following inputs are true.
  - <GOFORWARD.DOUT> which goes high when the pump is to go forward.
  - <DMOTOR.forward> is high when the position motor is set to move in the forward direction.
- <RSTRTDEL.DOUT> goes high when the pump is to move in the reverse direction. <RSTRTDEL> will go high when both of the following inputs are true.
  - <GOREVERS.DOUT> which goes high when the pump is to go backward.
  - <DMOTOR.reverse> is high when the position motor is set to move in the reverse direction.

4.2.1.2 Stop

The directional motor will be stopped automatically if the position of the pump is within the desired range, the position of the pump exceeds the safe boundary, or an abort condition arises. The directional motor also can be stopped by an operator request from the STATION8 console.

The following signals are involved with the stop task in the MOTOR strategy.

BUTPULSE	DISP1	DMOTOR	FOROVER
FOVRSTOP	GOSTOP	HILIMIT	LOLIMIT
REVOVER	ROVRSTOP	STOP	STOP2

The output of the small logic *or* block <STOP2> controls <DMOTOR.stop>, which commands the directional motor to halt if the signal transitions from 0 to 1.

<STOP.DOUT> is also a small logic *or* block to help compile stop conditions. <STOP2.DOUT> will go high if any one of the following conditions become true.

- <STOPTEST.DOUT> which goes high when the operator requests to stop the pump, an abort condition occurs, or the pump operation time reaches the desired time.

- <GOSTOP.DOUT> goes high, which is automatically set when the pump position comes within the desired range while the directional motor is moving.

- <FOVRSTOP.DOUT> goes high, which indicates an overshoot condition while the pump is moving forward.

- <ROVRSTOP.DOUT> goes high, which indicates an overshoot condition while the pump is moving in reverse.

- <HILIMIT.DOUT> goes high, which indicates the actual pump position has exceeded the maximum boundary. This will be discussed in further detail in the position feedback task in section 4.2.1.4.

- <LOLIMIT.DOUT> goes high, which indicates the actual pump position has exceeded the minimum boundary. This will be discussed in further detail in the position feedback task in section 4.2.1.4.

- <BUTPULSE.DOUT> goes high, which indicates the operator requested the motor to stop.

Each time the position motor is stopped the movement is disabled by the output of <STOP2> setting input 1 of <DISP1> high to execute the macro DISMOV1. The macro DISMOV1 simply forces <DISMOVE> high. The disable movement task will be discussed in further detail in section 4.2.1.6.

An overshoot condition causes the directional motor to stop operation. This condition is recognized when <POSCNTRL.DO2> goes high and the pump is moving forward or <POSCNTRL.DOUT> goes high and the pump is moving in reverse.

The small logic *and* block <FOROVER> goes high to drive the *one-shot* block <FOVRSTOP> that stops the directional motor when all three of the following conditions are true.

- <DMOTOR.inforwd> is high, indicating the motor is moving in the forward direction.

- <POSCNTRL.DO2> is high when the actual position is beyond the desired range and the motor should be moved in reverse to achieve the desired position.

- <DMOTOR.inrun> is high, indicating the directional motor is running.

The small logic *and* block <REVOVER> goes high to drive the *one-shot* block <ROVRSTOP> that stops the directional motor when all three of the following conditions are true.

- <DMOTOR.inrevers> is high, indicating the motor is moving in the reverse direction.
- <POSCNTRL.DOUT> is high when the actual position is beyond the desired range and the motor should be moved in forward to achieve the desired position.
- <DMOTOR.inrun> is high, indicating the directional motor is running.

4.2.1.3 AF5000+ Communications

The AF5000 device driver is a communication task that runs in parallel with all other Genesis tasks. The device driver is responsible for sending information to and collecting information from the variable frequency drive. The device block with the tag name <DMOTOR> processes all communications for the variable frequency drive controlling the directional motor.

The following signals are used to perform the AF5000+ communications to the position motor.

AF5COMFA	COMMOR	DACCEL	DATSPEED
DDECEL	DFAULT	DFORWARD	DLINE
DMAXSPD	DMOTOR	DREADY	DREVENAB
DREVERSE	DRUN	DSTOP	VR232070
VR232080	VR232090	VR232100	VR232110
VR232120	GOFORWRD	GOREVERS	STARTSHT
STOP2			

The following tags are used to issue commands to the variable frequency drive to change operating value and operating condition.

TAG	VARIABLE	DESCRIPTION
GOFORWRD	forwrd.do0	Changes direction of motor to forward
GOREVERS	revers.do0	Changes direction of motor to reverse
STARTSHT	start.do0	Causes motor to start
STOP2	stop.do0	Causes motor to halt

The following tags are used as directional motor feedback signals from the AF5000+ device block. Some of these signals are passed to STATION5 with the PLC communication task and some are used for parameter verification.

TAG	AF5000+ Variable	Description
COMMOR.INP1	ACCEL.fail	Indicates a failure in communications
DACCEL	ACCEL.inp	Desired acceleration rate
DDECEL	DECEL.inp	Desired deceleration rate
VR232090	FREQ.inp	Stator frequency high range
DLINEV	LINEV.hrng	Line voltage
VR232070	LOAD.inp	Motor load
DMAXSPD	MAXSPD.inp	Maximum speed
VR232100	MOTORA.inp	Motor current
VR232080	MOTORV.inp	Motor voltage
VR232120	SPDSET.inp	Desired speed of motor
VR232110	SPEED.inp	Actual speed of motor
DATSPEED	inatspd.di0	Actual speed is within ±5% of desired speed
DFAULT	infault.di0	Variable frequency drive is faulted
DFORWARD	inforward.di0	Direction of motor movement is forward
DREADY	inready.di0	Drive is ready to run
DREVERSE	inrevers.di0	Direction of motor movement is reverse
DRUN	inrun.di0	Drive is running
DSTOP	instoppd.di0	Drive is stopped
DREVENAB	reven.di0	Reverse enable

The following is a list of variables in the AF5000+ device block <DMOTOR>.

Variable	Description
ACCEL	Desired acceleration rate
DECEL	Desired deceleration rate
FREQ	Stator frequency
LINEV	Line voltage
LOAD	Motor load
MAXSPD	Maximum speed
MOTORA	Motor current
MOTORV	Motor voltage
SPDSET	Desired speed of motor
SPEED	Actual speed of motor
forwrd	Changes direction of motor to forward
inatspd	Actual speed is within ±5% of desired speed
infault	Variable frequency drive is faulted
inforward	Direction of motor movement is forward
inready	Drive is ready to run
inrevers	Direction of motor movement is reverse
inrun	Drive is running
instoppd	Drive is stopped
reven	Reverse enable
revers	Changes direction of motor to reverse
start	Causes motor to start
stop	Causes motor to halt

If a failure of communications arises from the VFD, the parameter <DMOTOR.ACCEL.fail> will go high which is passed on <COMMOR> which alarms <AF5COMFA>. Communications failure is an abort condition which causes either motor drive to be automatically stopped as discussed further in the abort task section 4.2.3.

4.2.1.4 Position Feedback

There are maximum limits set on the actual position of the pump. The actual position signal comes from the block <POSITION>. The high-alarm value of <POSITION> is set to 190 degrees, and the low-alarm value is set to 15 degrees.

The following signals are integral to the position feedback task in the MOTOR strategy.

ANION	HILIMAND	HILIMIT	INBAND
LOLIMAND	LOLIMIT	POSITION	

The high-alarm bit from <POSITION> will go high when the actual position exceeds the high alarm value. This high-alarm bit is passed on to a small logic *and* block <HILIMAND>, which is used to disable the start of the directional motor.

The output of <HILIMAND> must be low for the directional motor to be enabled to start.

- <POSITION.HALM> is high indicating the pump has exceeded the maximum boundary.
- <DMOTOR.inforwrd> is high indicating the variable frequency drive is enabled to move the directional motor forward.

The high-alarm bit from <POSITION> is currently passed to a small logic *and* block <HILIMIT>, which is used to stop the directional motor and disable movement of the pump if it is running and the actual position exceeds the high limit. The output of <HILIMIT> is passed to the block <STOP>, which will cause the directional motor to stop. The output of <HILIMIT> will go high when the following three conditions are true.

- <POSITION.HALM> is high, indicating the pump has exceeded the maximum boundary.
- <DMOTOR.inrun> is high, indicating the directional motor is running.
- <DMOTOR.inforwrd> is high, indicating the directional motor will move forward when it is started.

The low-alarm bit from <POSITION> will go high when the actual position is less than the low-alarm value. This low-alarm bit is passed on to a small logic *and* block <LOLIMAND>, the output of which must be low for the directional motor to be enabled to start. The output of <LOLIMAND> will go high when both of the following conditions are true.

- <POSITION.LALM> is high, indicating the pump has exceeded the minimum boundary.
- <DMOTOR.inrevers> is high, indicating the variable frequency drive is enabled to move the directional motor in reverse.

The low-alarm bit from <POSITION> is also passed to a small logic *and* block <LOLIMIT>, which is used to stop the directional motor and disable movement of the pump, if it is running and the actual position falls below the low limit. The output of <LOLIMIT> is passed to the block <STOP>, which will cause the directional motor to stop. The output of <LOLIMIT> will go high when the following three conditions are true.

- <POSITION.LALM> is high, indicating the pump has fallen below the minimum boundary.
- <DMOTOR.inrun> is high, indicating the directional motor is running.
- <DMOTOR.inrevers> is high, indicating the directional motor will move in reverse when it is started.

The position feedback alarming is disabled on startup of Runtime by controlling the inhibit alarm bit of <POSITION> by <ANTON.QNOT> which will go low 10 seconds after the strategy is started. When the feedback position is within the deadband of the desired position <INBAND> is set high.

4.2.1.5 Direction Control

The actual pump position is monitored from the PLC by STATION8 with the block called <POSITION>. The output of the block <POSITION> is passed to the *DGAP* control block <POSCNTRL> as the measured process variable.

The *DGAP* block <POSCNTRL> is used to control forward movement or reverse movement of the directional motor based on the comparison of desired position with actual

position. The *DGAP* block is an on-off control block that sets two digital outputs to various states depending on where the measured position is in relationship to the deadband. Figure 13 is a diagram of these two outputs.

The high gap and low gap of the *DGAP* block <POSCNTRL> is set to a 2-degree dead band around the desired position.

The first output of the *DGAP* block <POSCNTRL.DOUT> goes high when the measured position is less than the desired position minus the low-gap offset. The output goes to a one-shot block <GOFORWRD>, used to start forward movement of the pump.

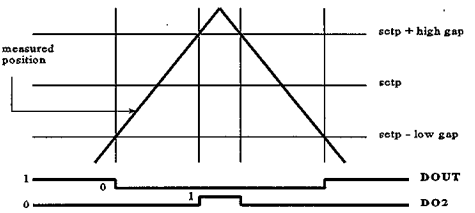


Fig 13. DGAP block output

The output of <GOFORWRD> will stay high for 15 seconds unless the block is reset. The reset of the block <GOFORWRD> is controlled by the inverse of the small logic *and* block <FORWARD>. When the output of <FORWARD> is low, the output of <GOFORWRD> will always be low and will not start forward movement.

The block <FORWARD> has two inputs that must go high for the output to go high to allow the pump to move forward.

- The output of <DISMOVE> must be low. This is the bit used to disable movement mentioned in the disable movement task.

- The output of <NOTRUN> must be high, which is the inverse of the feedback signal from the directional motor variable frequency drive that goes high when the motor is running.

When the output of <GOFORWRD> is allowed to go high it sets <DMOTOR.forwrd> high, which commands the variable frequency drive to move the motor forward when it is running. The output of <GOFORWRD> also starts a 6-second *time-on delay* block <FSTRTDEL>.

The 6-second delay of <FSTRTDEL> is to ensure that the forward command is sent to the variable frequency drive before the motor is commanded to start. The output of <FSTRTDEL> sets a small logic *or* block <START> high, which is the beginning of the start task of position control.

The second output of the *DGAP* block <POSCNTRL.DO2> goes high when the measured position is greater than the desired position plus the high-gap offset. The output goes on to a *one-shot* block <GOREVERS> used to start reverse movement of the pump.

The output of <GOREVERS> will stay high for 15 seconds unless the block is reset. The reset of the block <GOREVERS> is controlled by the inverse of the small logic

and block <BACKWARD>. When the output of <BACKWARD> is low the output of <GOREVERS> will always be low and will not start reverse movement.

The block <BACKWARD> has two inputs that must go high for the output to go high and allow the pump to move backward:

- The output of <DISMOVE> must be low. This is the bit used to disable movement mentioned in the disable move task.

- The output of <NOTRUN> must be high. This is the inverse of the feedback signal from the directional motor variable frequency drive that goes high when the motor is running.

When the output of <GOREVERS> is allowed to go high it sets <DMOTOR.revers> high, which tells the variable frequency drive to move the motor in reverse when it is running. The output of <GOREVERS> also starts a 6-second *time-on delay* block <RSTRTDEL>.

This delay ensures that the reverse command is sent to the variable frequency drive before the motor is commanded to start. The output of <RSTRTDEL> sets a small logic *or* block <START> high, which, as noted above, imitates the position control task.

An overshoot condition causes the directional motor to stop operation. This condition is recognized when <POSCNTRL.D02> goes high and the pump is moving forward or <POSCNTRL.DOUT> goes high and the pump is moving in reverse.

The small logic *and* block <FOROVER> goes high to drive the *one-shot* block <FOVRSTOP> that stops the directional motor when all three of the following conditions are true:

- <DMOTOR.inforwrd> is high, indicating the motor is moving in the forward direction.

- <POSCNTRL.DO2> is high when the actual position is beyond the desired range and the motor should be moved in reverse to achieve the desired position.

- <DMOTOR.inrun> is high, indicating the directional motor is running.

The small logic *and* <REVOVER> goes high to drive the *one-shot* block <ROVRSTOP> that stops the directional motor when all three of the following conditions are true:

- <DMOTOR.inrevers> is high, indicating the motor is moving in the reverse direction.

- <POSCNTRL.DOUT> is high when the actual position is beyond the desired range and the motor should be moved forward to achieve the desired position.

- <DMOTOR.inrun> is high, indicating the directional motor is running.

4.2.1.6 Disable Movement

The pump is disabled from moving unless the operator enables movement from the console. Each time the pump stops or goes beyond the desired range the pump is disabled from moving automatically.

The following tags in the MOTOR strategy are used to execute the disable movement task.

DISMOV	DISMOV1	DISP1	PBPOSAN2
DISMOVE	DISP1	INBAND	ON
STOP2			

The *DIN* block <DISMOVE> must have a 0 value in order to start the directional motor; This is the value set when the operator enables pump movement from the STATION8 console. The value of <DISMOVE> is set to 1 any time the directional motor is stopped or when the MOTOR strategy is first started. This is accomplished with two custom macros in EXTRA.KMS. The macro DISMOV0 is used to set <DISMOVE.DOUT> low and the macro DISMOV1 is used to set <DISMOVE.DOUT> high which will disable movement of the pump.

The following signal will execute the macro DISMOV0.

- <PBPOSAN2.DOUT> which goes high if the operator selects to position the pump during a full integrated test.

The movement of the pump is also enabled manually by the operator from the DDISPLAY display.

The following signals will execute the macro DISMOV1.

- <STOP2.DOUT> will go high any time the directional motor is stopped.

- <ON.DOUT> when the MOTOR strategy is first started.

- <DISMOV1.DOUT> which goes high any time there is a problem with starting the position motor, the position motor is not running, and the position of the pump is not within the desired deadband.

The *function* block <DISMOV> is used to stop the directional motor and disable movement when the position of the pump is within the desired range. This block also checks the disable movement bit from <DISMOVE.DOUT>, the forward movement bit from <POSCONTRL.DOUT>, and the reverse movement bit from <POSCNTRL.D02>. When all three conditions are not true an internal control bit <DR1> starts an internal counter <AR1>. This counter is used to delay the stop of the pump by 2 seconds after the position is within the dead band. This allows the pump to move closer to the desired position before shutting down movement. When the internal counter reaches 2 seconds the directional motor is commanded to stop and movement is disabled by control bit <DISMOV.DOUT>.

The *function* block <DISMOV> also has a logic equation to determine if the pump is within the desired range. <DISMOV.D02> is passed to <INBAND>, which is used to indicate the pump is within the dead band range of 2 degrees.

4.2.2 Pump Control

The following are the details related to controlling the operation of the mixing pump to actually mix the contents of the tank.

4.2.2.1 Start

The AF5000+ device block variable <PMOTOR.start> requires a transition from 0 to 1 to start operation of the pump motor. The <PMOTOR.start> variable is driven by the output of the block with the tag name <PBSTRTDN>. <PBSTRTDN> will go high when the operator successfully starts a fully integrated test for pump operation. Test operation will be explained in detailed sections to follow.

The following blocks are involved with the start pump control task.

PBSTRTDN PMOTOR

4.2.2.2 Stop

The AF5000+ device block variable <PMOTOR.stop> requires a transition from 0 to 1 to stop operation of the pump motor. The <PMOTOR.stop> variable is driven by the output of a small logic *or* block with the tag name <STOPTEST>.

The <STOPTEST> block will go high when the pump operation is stopped automatically from an abort condition, from the elapsed time exceeding the desired times, the operator manually stopping pump operation, or failure in communications to either VFD.

The following blocks are involved with the stop pump control task.

PMOTOR STOPTEST

4.2.2.3 AF5000+ Communications

The AF5000+ device driver is a communication task that runs in parallel with all other Genesis tasks. The device driver is responsible for sending information to and collecting information from the variable frequency drive. The device block with the tag name <PMOTOR> processes all communications for the variable frequency drive controlling the pump motor.

The following signals are used to perform the communications task for the pump control.

AF5COMFA	COMMOR	EQUPSPD	PACCEL
PATSPEED	PDECEL	PFAULT	PFORWARD
PLINEV	PMAXSPD	PMOTOR	PREADY
PREVENAB	PREVERSE	PRUN	PSTOP
VR232020	VR232030	VR232040	VR232050
VR232060	VSDRS232		

The following tags are used to issue commands to the variable frequency drive to change operating values and operating conditions.

TAG	AF5000+ Variable	Description
COMMOR.QNOT	forwrd.do0	Changes direction of motor to forward
PBSTRTDN	start.do0	Causes motor to start
STOPTEST	stop.do0	Causes motor to halt

The following tags are used as pump feedback signals from the AF5000+ device block. Some of these signals are passed to STATION5 with the PLC communication task, and some are used for parameter verification.

TAG	AF5000+ Variable	Description
COMMOR.INP2	ACCEL.fail	Indicates a failure in communications
PACCEL	ACCEL.inp	Desired acceleration rate
PDECEL	DECEL.inp	Desired deceleration rate
VR232030	FREQ.hrng	Stator frequency high range
PLINEV	LINEV.inp	Line voltage
VSDRS232	LOAD.inp	Motor load
PMAXSPD	MAXSPD.inp	Maximum speed
VR232040	MOTORA.inp	Motor current
VR232020	MOTORV.inp	Motor voltage
VR232060	SPDSET.inp	Desired speed of motor
VR232050	SPEED.inp	Actual speed of motor
VSDRS232	LOAD.inp	VSD% load
PATSPEED	inatspd.di0	Actual speed is within ±5% of desired speed
PFAULT	infault.di0	Variable frequency drive is faulted
PFORWARD	inforward.di0	Direction of motor movement is forward
PREADY	inready.di0	Drive is ready to run
PREVERSE	inrevers.di0	Direction of motor movement is reverse
PRUN	inrun.di0	Drive is running
PSTOP	instoppd.di0	Drive is stopped
PREVENAB	reven.di0	Reverse enable

The following is a list of variables in the AF5000+ device block <PMOTOR>.

Variable	Description
ACCEL	Desired acceleration rate
DECEL	Desired deceleration rate
FREQ	Stator frequency
LINEV	Line voltage
LOAD	Motor load
MAXSPD	Maximum speed
MOTORA	Motor current
MOTORV	Motor voltage
SPDSET	Desired speed of motor
SPEED	Actual speed of motor
forwrd	Changes direction of motor to forward
inatspd	Actual speed is within $\pm 5\%$ of desired speed
infault	Variable frequency drive is faulted
inforward	Direction of motor movement is forward
inready	Drive is ready to run
inrevers	Direction of motor movement is reverse
inrun	Drive is running
instoppd	Drive is stopped
reven	Reverse enable
revers	Changes direction of motor to reverse
start	Causes motor to start
stop	Causes motor to halt

If a failure of communications arises from the VFD the parameter <PMOTOR.ACCEL.fail> will go high which is passed on <COMMOR> which alarms <AF5COMFA>. Communications failure is an abort condition which should cause either motor drive to be automatically stopped as discussed further in the abort task section 4.2.3.

4.2.2.4 Parameter Verification

The following blocks are used for parameter verification in the MOTOR strategy.

CALCUR	RCALDISP	RCURABR	TCURALRM
CURPRED	DISPRABT	DISPRALM	DISPRPRE
EQUPSPD	VR232050	VR232060	

The *function* block <EQUPSPD> contains an equation to compare the desired speed set in the variable frequency drive <VR232060> with the actual speed <VR232050>. The variable <EQUPSPD.DRI> will be high if the actual speed is within 5% of the desired speed. This will be displayed to the operator as a light green indication on the pump speed bar graphs.

The desired speed value is passed to the function block <CALCURR> uses <CURPRSPD> as the desired speed and automatically calculates the predicted motor current, which is passed to <CURPRED>.

The predicted motor current plus 20% is passed to <CURALRM> as the alarm limit. The output of <CURALRM> sets the high alarm value of <VR232040> and is passed to the PLC for the high motor alarm.

The predicted motor current plus 40% is passed to <CURABRT> as the abort limit. The output of <CURABRT> is passed to the PLC as the abort limit for motor current.

The desired pump speed is also passed to the *function* block <CALDISPR> which automatically calculates the predicted discharge pressure, which is passed to <DISPRPRE> for operator display.

The predicted discharge pressure plus 20% is passed to <DISPRALM> as the alarm limit, and the output of <DISPRALM> is passed to the PLC to set the high alarm value of the discharge pressure signal.

The predicted discharge pressure plus 40% is passed to <DISPRABT> as the abort limit, and the output of <DISPRABT> is passed to the PLC as the abort limit for the discharge pressure.

4.2.3 Abort

Certain conditions are monitored continuously in the MOTOR strategy as abort criteria to stop both motors automatically. Also abort conditions will prevent the start of either motor.

The following blocks are involved with the abort task.

A5ABRT	ABORT	ABORTOR	ABORTPUL
COMMOR	NODEFAIL	PLCFAIL	

The current abort status is indicated by the small logic *or* block <ABORTOR> which has the following input conditions.

- <ABORT.DOUT> is high if the abort coil in the PLC is active.
- <COMMOR.DOUT> is high if either VFD has a failure in communications to the MOTOR strategy.
- <NODEFAIL.DOUT> is high if a failure in communications from STATION5 to the PLC is recognized.
- <PLCFAIL.DOUT> is high if a failure of communications to the PLC is recognized from the MOTOR strategy.

The output of <ABORTOR> is passed to a small logic *pulse* block <ABORTPUL> which is passed on to <STOPTEST> which will stop either VFD with a communication command.

<COMMOR> is a small logic *or* block with two inputs from each hardware device block. If communications to either variable frequency drive goes bad, indicated by a fail bit of the first parameter selected from each hardware device block, <COMMOR.DOUT> will go high. This signal is passed to the PLC with the *digital output* block <A5ABRT>.

4.2.4 Test Operation

This section provides the details of the actual test operation, including their selection, how the elapsed time is calculated, and the information provided to the operator.

4.2.4.1 Test Creation and Selection

The following blocks are related to test creation and selection.

ANGLECHG	CHANGED	CHGCHK1	CLRMESS
DECELCHG	DELPUL	HRSCHG	INFILE
MANUAL	MAXINDX	MINSCHG	NEXTINDX
NEXTPUL	OACCEL	OANGLE	ODECEL
OHRS	OLDTEST	OMINS	ORESTIM
OSECS	OSPEED	PBTESTNO	PREVPUL
RESCHG	SAVEPUL	SECSCHG	SPEEDCHG
TESTCHG	TESTSET	TESTSET2	UACCEL
UANGLE	UDECEL	UHRS	UMINS
UMSG1	URESTIM	USECS	USPEED
UTESTNO			

All test setups are in a file called USERSET.TDF (TDF stands for Test Data File). The test setups can be accessed from the pump operation display PUMPRUN. There is a display called TESTSET which allows the creation and modification of the test setup file USERSET.TDF. This display is not part of the STATION8 menu system but can be accessed via the F1 function key. The TESTSET display calls report and recipe code which performs basic operations on the test setup file such as test setup loading, scrolling, insertion and deletion.

The details of all of this are described below.

USERSET.TDF file format. This is an ASCII file which contains records with the following information for each test setup:

test number (or end of file marker)
angle (degrees)
pump speed (RPM)
test duration (hours minutes seconds)
pump motor acceleration (RPM/Sec)
pump motor deceleration (RPM/Sec)
test description
Reset overall test timer indication (1 = yes, 0 = no)

The format for each record is:

```
tt.t aaa.a rrrr.r hh  mm  ss ccc.c ddd.d eeeeeeeeeeeeeeeeeeee f
```

where tt.t is the test number, aaa.a is the angle, rrrr.r is the pump speed, hh is test duration hours, mm is test duration minutes, ss is test duration seconds, ccc.c is pump motor acceleration, ddd.d is pump motor deceleration, e...e is the test description and f is reset overall timer indication.

Each record is separated by a carriage return/line feed. The end of file record has 999 as the test number.

The reset overall test timer field is used to indicate whether the overall test time should be reset at the beginning of a test. This allows for multiple-angle tests; the timer should be reset for the first of the series of tests and not for the subsequent tests.

TESTSET Screen. The TESTSET screen is used to create and maintain the setup file USERSET.TDF. It contains data entry fields for the above parameters as well as buttons to save and delete test setups, and buttons to go to the previous or next record in the file. A test setup can also be loaded by typing the test number in the test number data entry field. There is an indication on the screen telling whether or not the current test number exists in the file.

The buttons on the screen initiate the execution of reports which do the actual file manipulation. The Save Test button executes the report SAVE.RPS, the Delete Test button, DELETE.RPS, the Next button, NEXT.RPS, and the Previous button, PREV.RPS. In addition, there are reports CLRMESS.RPS, ULOAD.RPS and ILOAD.RPS. These reports are discussed in the next section.

Reports for test setup file manipulation. SAVE.RPS is executed by clicking on the Save Test button on the TESTSET screen. It inserts the test setup which has been entered into the data entry fields on that screen into the USERSET.TDF file. The new setup is inserted so as to keep the records in numeric order by test number. If a record already exists with the same test number, it is replaced.

DELETE.RPS is executed by clicking on the Delete Test button on the TESTSET screen. It deletes the current test from the file, if it exists in the file.

NEXT.RPS loads the setup for the next test (in numeric order). This setup is displayed on the screen. It is executed by clicking on the Next button on the TESTSET screen or the up-arrow button on the PUMPRUN screen. If the current test is not in the test setup file, the next button will load the next test from the last test loaded which was in the file.

PREV.RPS works the same as NEXT.RPS only the previous test setup (in numeric order) is loaded. It is executed by clicking on the Previous button, or the down-arrows.

ILOAD.RPS loads the first setup in the test setup file and initializes the housekeeping blocks in the strategy. It is executed upon strategy startup.

ULOAD.RPS loads the setup for the current test number. It is executed when the strategy blocks detect that the user has entered a new test number.

CLRMESS.RPS clears the message field on the TESTSET screen. The reports SAVE and DELETE produce messages on the TESTSET screen giving the status of the save or delete operation. These messages are cleared by CLRMESS after a delay. It is executed by the strategy block CLRMESS going high. This bit is controlled from within the SAVE and DELETE reports.

SETOLD.RPS sets a set of strategy blocks corresponding to all of the test setup parameters to be equal to the current test setup parameters. This allows the strategy to detect when the user changes a parameter. It is executed from within any of the above reports which load a new predefined test setup.

Strategy Blocks. The reports are run from the report blocks <TESTSET> and <TESTSET2>. <TESTSET> executes SAVE.RPS from input 1, DELETE.RPS from input 2 and CLRMESS.RPS from input 3. <TESTSET2> executes ULOAD.RPS from input 1, NEXT.RPS from input 2, PREV.RPS from input 3, and ILOAD.RPS from input 4.

The test setups are loaded into a set of AIN and DIN blocks called <UTESTNO>, <UANGLE>, <USPEED>, <UHRS>, <UMINS>, <USECS>, <UACCEL>, <UDECEL>, <UDESC> and <URESTIM>. There is a corresponding set of blocks with the first letter of "O" instead of"U" (except for the one corresponding to <UTESTNO>, which is called <OLDTEST> and the one corresponding to <UDESC> which is called <TDESC>). These sets of blocks are set to be equal when a new test setup is loaded (see SETOLD.RPS above). A set of alarm blocks detects when any of the pairs of blocks contain differing values. This indicates that the user has changed a setup parameter and this information is displayed on the screens. In addition, if the test number is changed, the report ULOAD.RPS is executed to load in the setup corresponding to the new test number if it exists. The block <INFILE> is maintained by the reports. It is 1 if the current test is in the file, 0 otherwise. The blocks <NEXTINDX> and <MAXINDX> are maintained by the reports. They keep track of the record number of the current test and the record number of the highest numbered test in the file respectively. The block <UMSG1> is used by the reports to display messages in the message field of the TESTSET screen. The block <CHANGED> is 1 if any of the test setup parameters have been changed since last loaded and 0 if not. The block manual is set when a test setup is actually loaded for use if either <CHANGED> is 1 or <INFILE> is 0. This indicates that a setup is being used that was not predefined.

4.2.4.2 Elapsed Time Calculation

There are two elapsed timers that are used during test operation. The overall timer for test operation keeps track of the total test time. There is a second timer to time pump operation time.

The following blocks are integral is performing the elapsed time calculation task.

DF65535	DFLT0	DISP4	PB30SEC
PBCALSEC	PBDNTM	PBHR	PBMIN
PBRAMP	PBRAMPHL	PBSEC	PBSPLIT
RESETPB	RSECPB	SPLITPB	TSTRUNNG

The *RAMP* block <RSECPB> totals the test time while a test is in operation. When ramp rate 1 becomes active the output of <RSECPB> increments by one every second.

A test is in operation when the *rs flip-flop* block <TSTRUNNG> is high. This signal controls <RSECPB.RPlO>, which activates ramp rate 1. The elapsed timer <RSECPB> is reset at the beginning of test operation by <RESTPB> which is set high with the RR code PBSETVAL.RPS when a new test is selected. The elapsed test time is broken down from seconds into hours, minutes, and seconds by the *function* block <SPLITPB>.

There is a second elapsed timer used to keep track of the pump operation time at each position. The *RAMP* block <PBRAMP> totals the time the pump is in operation. When ramp rate 1 becomes active, the output of <PBRAMP> increments by one every second.

The pump is in operation when the *rs flip-flop* block <TSTRUNNG> is high. This signal controls <RSECPB.RPlO>, which activates ramp rate 1.

The elapsed timer <PBRAMP> is reset at the beginning of pump operation by <SETVALDN> which goes high when the test values have been successfully set. The high limit of <PBRAMP> is controlled by <PBRAMPHL> which will pass either the value 0 from <DFLT0> or the maximum pump operation time from <DF65535>. <PBRAMPHL> will pass the value of <DF65535> when <SETVALDN> is high when the test values are successfully loaded.

The desired pump operation time is set during test selection by setting the test values. The *AIN* blocks <PBHR>, <PBMIN>, and <PBSEC> contain the desired minutes and seconds for pump operation. The outputs of these blocks are passed to a *function* block <PBCALSEC>, which merges the values into one desired pump operation time value in seconds. This value is passed to the *function* block <PBSPLIT>, which is used to stop pump operation when the elapsed time from <PBRAMP> exceeds the desired time from <PBCALSEC>.

The control bit <PBSPLIT.DOUT> will go high when the elapsed time exceeds the desired time and is passed to the *DIN* <PBDNTM>. The output of this block resets the *rs flip-flop* block <PBRSFF> mentioned above. The output of <PBDNTM> is passed to logic which stops pump operation by setting <STOPTEST> high. <PBSPLIT> also breaks seconds of pump operation into hours, minutes, and seconds for operator display.

The desired pump operation times are reset by the output of <STOPTEST> executing the macro RESPTIM which is referenced in <DISP4>.

The *alarm* block <PB30SEC> is used to warn the operator 30 seconds before the end of pump operation. The high alarm value of <PB30SEC> is set by the second analog output of <PBCALSEC> which is set to 30 less than the desired pump operation time during test operation and to a very large number when a test is not in operation.

4.2.4.3 Operator Test Information

The PUMPRUN screen can be used for any test. The operator actions required for the PUMPRUN screen are discussed below. The operations are:

1) Select a test setup or manually enter a pump test configuration
2) Set the selected test values (set values).
3) Position the pump (if required)
4) Enable the pump test.
5) Start the pump test.
6) Stop the pump test (performed automatically when the preset time elapses or at any time by the operator.

Test Setup. Tests can be selected by typing the test number in the Test entry field on the screen. This will access the USERSET.TDF file to load the parameters for this test into the other fields (see 4.2.4.1). These values can be used as is, or can be changed by entering different values into any of the fields. If new values are entered the test is considered to be a manual test. A message will appear in the lower left corner of the screen indicating that the test values have been changed.

Set Values. The test parameters do not take effect until the SET VALUES button is pressed. The button is only effective if it is green in color. Pressing SET VALUES begins the pump test process, copying the test setup parameters to operational parameters. After the SET VALUES button is pressed the STOP TEST button will turn green. This button will abort the current test and return the system to the test setup mode. This can be done at any time, however the button functions differently if the pump is actually running (see below). After the SET VALUES button is pressed the POSITION PUMP button will turn green if the pump is not oriented at the specified angle. Otherwise the ENABLE TEST button will turn green. If there is a problem with setting the values, a flashing yellow question mark will appear at

the bottom of the screen. Clicking on this question mark will bring up a screen showing what the problem was. Similar problem screens exist for all of the operations.

Position Pump. Pressing this button will cause the directional motor to begin to orient the pump to the desired angle. The current angle and the directional motor speed (in RPMs), as well as a directional motor running indicator are displayed above the button. When the pump is in position the POSITION PUMP button will turn gray and the ENABLE TEST button will turn green. As with all of the buttons, if there is a problem with pump positioning, clicking on the flashing question mark will bring up a problem screen.

Enable Test. This button enables a pump test. At this time test times and pump speed based alarm and abort limit values are sent to the PLC. The START TEST button will turn green. The operator has 60 seconds to start the test after the enable button is pressed. If the 60 seconds elapses, the ENABLE TEST button will once again turn green.

Start Test. This button starts the pump. The elapsed time indication on the screen will begin to increment and the pump motor running indicator will turn green. The pump speed is also displayed. The legend for the STOP TEST button will change to read ENABLE STOP.

Stop Test. This button functions to reset the system when the pump is not running. It returns the system to the test setup mode (SET VALUES button is green). When the pump is running, the legend for this button is ENABLE STOP. When clicked upon, the legend of the button will change to STOP TEST. After two seconds, the button will begin to flash. At this point, the user has five seconds to click the button again. This will stop the pump and reset the system to the test setup mode. If the 5 seconds elapses, the legend will change once again to ENABLE STOP. The pump will stop automatically when the specified time for the test has elapsed. There is also a timer in the PLC to shut off the pump as a backup to this. The STOP TEST button gives the operator a means to manually stop the pump before the allotted time has elapsed. When the pump is stopped either manually or at the end of the time allotted, the system will return to the test setup mode.

4.2.4.4 Test Operation Strategy Details

There is logic within the MOTOR strategy for indication of test operation status. There are status bits for test values set correctly, pump at desired position, test enabled to start, and test running. If a function is selected by an operator and the task was not successfully executed there is logic to indicate what the problem is. The following sections describe each main task execution and status feedback for indication of success of operation.

Set Test Values. The desired test values selected by the operator must be downloaded to the control parameters in the MOTOR strategy. This is done by the operator

with the set values button. The blocks in the following table are related to the set test values task.

NORUN	PBENABLE	PBREPORT	PBSETAND
SETTEST	SETVALDN	STOPTEST	TSTRUNNG
VALSET			

The small logic *pulse* block <SETTEST> starts the execution of the set values task. Its output is passed to the small logic *and* block <PBSETAND> which has the following input conditions to be true before executing the RR code PBSETVAL.RPS by setting input 1 of <PBREPORT> high.

- <SETTEST.DOUT> is set high by the operator to set the test values.
- <PBENABLE.QNOT> is high if a test is not already enabled.
- <TSTRUNNG.QNOT> is high if a test is not in progress.
- <NORUN.DOUT> is high if both the pump motor and position motor are stopped.
- <VALSET.QNOT> is high if the test values are not already set.

At the end of the RR code PBSETVAL.RPS <SETVALDN> is set high to indicate the test values are successfully set. <SETVALDN> has a pulsed output and sets the *rs flip-flop* block <VALSET> high. <VALSET.DOUT> is the status bit used to indicate that the test values have been successfully set. <VALSET> is reset when <STOPTEST> goes high when a test is stopped.

Problem with Setting Test Values. If the test values are requested to be set and the task is not successfully completed then the blocks in the following table are used to indicate what the problem with setting test value variables is.

DISP3	NORUN	NOSEPROB	PBBUTSTP
PBPROB	PDISP	RSETPROB	SEDISP
SETPROB	SETPROBA	SETPROBR	

The *rs flip-flop* block <SETPROB> is used to indicate a problem with setting the test values. <SETPROB> is set high if the output of <SETPROBA> is high and the reset input <SETPROBR> is low.

<SETPROBA> is a small logic *and* block with the following inputs.

- <SETTEST.DOUT> is high if the test values are requested to be set.
- <NORUN.DOUT> is high if both motors are stopped.
- <VALSET.QNOT> is high if the test values are not set.

<SETPROBR> will reset <SETPROB> if any of the following conditions are true.

- <RSETPROB.DOUT> which is set high by the operator to reset the problem indication.
- <PBBUTSTP.DOUT> which goes high when the operator manually stops a test.
- <NOSEPROB.DOUT> which is a small logic *and* block with the following input conditions.
  - <PBENABLE.QNOT> is high when a test is not enabled.
  - <TSTRUNNG.QNOT> is high when a test is not running.
  - <NORUN.DOUT> is high when both motors are stopped.

When <SETPROB> goes high <PBPROB> is set high which indicates a test operation problem. The small logic *and* block <SEDISP> is used to bring up the problem indication display for setting test values. When <SETPROB> is high and the operator sets <PDISP> high to view the problem the display SETPROB is brought up via the *display* block <DISP3>.

Position Pump Selection. When the test values are set the desired pump position is compared with the actual position. If the pump is not at the desired position the pump will need to be positioned before a test can be enabled. The blocks in the following table are the key variables in the pump position selection.

ABORTOR	INPOSI	NORUN	PBENABLE
PBPOSAN2	PBPOSAND	PBPOSPMP	POSSET
TSTRUNNG	VALSET		

<PBPOSPMP> starts the execution of the pump position selection. Its output is passed to the small logic *and* block <PBPOSAND> which together with <PBPOSAN2> has the following input conditions before <PBPOSAN2> will start the pump position control by executing the macro DISMOV0 as discussed earlier in the disable movement task section.

- <TSTRUNNG.QNOT> which is high if a test is not running.
- <ABORTOR.QNOT> which is high if an abort conditions is not active.
- <PBENABLE.QNOT> is high if a test is not enabled.
- <VALSET.DOUT> is high if the test values are set.
- <NORUN.DOUT> is high if both motors are stopped.
- <PBPOSPMP.DOUT> goes high when the operator selects to position the pump.
- <POSSET.QNOT> which is an *AND* block with the following input conditions.
  - <VALSET.DOUT> is high when the test values are set.
  - <INPOSI.DOUT> which is high when the pump is at the desired position.

Problem with Position Pump Selection. If the pump position selection does not successfully execute, the blocks in the following table are used to indicate what the problems with the position pump variables are.

ABORTOR	DISP3	NOPOPROB	NORUN
PBBUTSTP	PBENABLE	PBPOSPMP	PBPROB
POSDISP	POSPROB	POSPROBA	POSPROBR
POSSET	RPOSPROB	TSTRUNNG	VALSET

The *rs flip-flop* block <POSPROB> is used to indicate a problem with selecting to position the pump. <POSPROB> is set high if the output of <POSPROBA> is high and the reset input <POSPROBR> is low. <POSPROBA> is a small logic *and* block with the following inputs.

- <PBPOSPMP.DOUT> is set high by the operator to position the pump.
- <POSSET.QNOT> which is high when the pump is not in position or the test values are not set.
- <NORUN.DOUT> is high when both motors are stopped.
- <VALSET.DOUT> is high if the test values have been set.

<POSPROBR> will reset <POSPROB> if any of the following conditions are true.

- <RPOSPROB.DOUT> which is set high by the operator to reset the problem indication.
- <PBBUTSTP.DOUT> which goes high when the operator manually stops a test.
- <NOPOPROB.DOUT> which is a small logic *and* block with the following input conditions.
  - <ABORTOR.QNOT> which is high if no abort condition is active.
  - <TSTRUNNG.QNOT> is high if a test is not running.
  - <PBENABLE.QNOT> is high if a test is not enabled.
  - <VALSET.DOUT> is high if the test values are set.
  - <NORUN.DOUT> is high if both motors are stopped.

When <POSPROB> goes high <PBPROB> is set high which indicates a test operation problem. The small logic *and* block <POSDISP> is used to bring up the problem indication display for position pump selection. When <POSPROB> is high and the operator sets <PDISP> high to view the problem the display <POSPROB> is brought up via the *display* block <DISP3>.

Position Pump Operation Problem. When the position motor is to be started to position the pump and a problem prevents the actual movement of the pump, the blocks in the following table are used to indicate what the problem with position pump operation variables is.

ABORTOR	DDISP	DISMOV1	DISMOVE
DISP1	DISP3	DMDISP	DMPROB
DMPROBD	IDMPROBR	DRUN	DSTOP
ENMOVPUL	ENMOVTON	HILIMAND	HOLDENMO
INBAND	LOLIMAND	NODMPROB	PBDMPROB
PBPROB	PBPROB2	PDISP	POSDISP
PRUN	PSTOP	RDMPROB	

The *rs flip-flop* block <DMPROB> is used to indicate a problem with positioning the pump. <DMPROB> is set high if the small logic *pulse* block <ENMOVPUL> is high and the reset input <DMPROBR> is low. <ENMOVPUL> is set high by <ENMOVTON> which delays <HOLDENMO> for 4 seconds. <HOLDENMO> stays high for 5 seconds after the pump is requested to move by <DISMOVE> going low. <DMPROBR> will reset <DMPROB> if any of the following conditions are true.

- <RDMPROB.DOUT> which is set high by the operator to reset the problem indication.
- <NODMPROB.DOUT> which is a small logic *and* block with the following input conditions.
  - <INBAND.QNOT> which indicates the pump is not within the desired position.
  - <ABORTOR.QNOT> which is high if no abort condition is active.
  - <HILIMAND.QNOT> is high if the pump position is not beyond the high range.
  - <LOLIMAND.QNOT> is high if the pump position is not beyond the low range.
  - <PSTOP.DOUT> is high if the pump motor is stopped.
  - <PRUN.QNOT> is high pump motor is not running.

If there is a problem indicated by <DMPROB> and the test values are set and the pump is not in position <PBDMPROB> goes high which is then passed to <PBPROB> through <PBPROB2>. The small logic *and* block <DMPROBDI> is used to bring up the problem indication display for positioning the pump. When <PBDMPROB> is high and the operator set <PDISP> high to view the problem and there is no problem with the position selection from <POSDISP.QNOT>, the display DMPROB will be brought up via the *display* block <DISP3>.

The small logic *and* block <DISMOV1> will execute the macro DISMOV1 to disable movement of the pump. <DISMOV1> has the following input conditions.

- <DISMOVE.QNOT> is high if movement is enabled.
- <DRUN.QNOT> is high if the position motor is not running.
- <DSTOP.DOUT> is high if the position motor is stopped.

The small logic *and* block <DMDISP> is used to bring up the problem indication display for positioning the pump. When <DMPROB> is high and the operator sets <DDISP> high to view the problem from the MOTOR display the display DMPROB is brought up via the *display* block <DISP1>.

Test Enable. Before test operation can be started a test must be enabled. The enable task is used to check the values set for a test to ensure proper test operation. The blocks in the following table are the variables involved with the test enable function.

BPOSGATE	CHKCOLOR	DO1-6DEV	DO1-7DEV
DO1DEV	DO2DEV	DO3DEV	DO4DEV
DO5DEV	DO6DEV	DO7DEV	ENCHECK
NORUN	PBCHK1	PBCHKCLR	PBENABLE
PBENAND	PBENOR	PBENREST	PBENRSRS
PBENTIMO	PBENTON	POSORBUM	RSCHECK
RSCHECKR	STOPTEST	TSTRUNNG	

The small logic *pulse* block <CHKCOLOR> starts the execution of the test enable task. Its output is passed to the small logic *and* block <PBCHKCLR> which works with <PBENAND> to require the following input conditions before a test can be enabled by setting <PBENABLE> high.

- <CHKCOLOR.DOUT> which is set high by the operator to enable a test.
- <NORUN.DOUT> is high if both motors are stopped.

- <TSTRUNNG.QNOT> is high if a test is not running.
- <PBENABLE.QNOT> is high if a test is not enabled.
- <POSORBUM.DOUT> is high if the pump is in position.
- <DO1-7DEV.QNOT> is high when all of the following conditions are not active. <DO1-6DEV> is also used to compile the conditions.
  - <PBCHK1.DALM> is high when the PLC test time does not match the desired test time.
  - <BPOSGATE.DOUT> is high if the pump is not in position.
  - <DO1DEV.DALM> is high if the pump is not in position.
  - <DO2DEV.DALM> is high if the desired pump speed does not match the pump speed in the VFD.
  - <DO3DEV.DALM> is high if the desired pump acceleration does not match the acceleration set in the VFD.
  - <DO4DEV.DALM> is high if the desired pump deceleration does not match the deceleration set in the VFD.
  - <DO5DEV.DALM> is high if the desired pump maximum speed does not match the VFD feedback.
  - <DO6DEV.DALM> is high if the set speed alarm does not match the PLC feedback.
  - <DO7DEV.DALM> is high if the pump speed abort limit does not match the PLC feedback.

The test enable bit <PBENABLE> is reset when <PBENOR> has one of the following input conditions are true.

- <STOPTEST.DOUT> goes high when a test is stopped.

- <PBENREST.DOUT> is a 4 second time delay of the small logic *and* block <PBENTIMO>. <PBENTIMO> has the following input conditions.
  - <PBENTON.DOUT> will go high if a test is enabled and a test is not started within 60 seconds.
  - <TSTRUNNG.QNOT> is high if a test is not running.

<ENCHECK> is a status bit used to enable color control on the test operation displays for critical parameters. This bit comes from the *rs flip-flop* block <RSCHECK> which is set high by <PBCHKCLR> and reset by <RSCHECKR> which will go high when any of the following conditions go high.

- <SETVALDN.DOUT> goes high when the test values are set.
- <STOPTEST.DOUT> goes high when a test is stopped.

Problem with Test Enable. If the test enable function was desired but not successfully completed the blocks in the following table are used to indicate what the problem with the test enable variables is.

CHKCLRTN	DISP3	DO1-7DEV	NOPENAB
NORUN	PBBUTSTP	PBENABLE	PBENADIS
PBENRSRS	PBPROB	PDISP	PENAB
PENABA	PENABR	POSORBUM	RPENAB
TSTRUNNG			

The *rs flip-flop* block <PENAB> is used to indicate a problem with enabling a test. <PENAB> is set high if the output of <PENABA> is high and the reset input <PENABR> is low. <PENABA> is a small logic *and* block with the following inputs.

- <PBENABLE.QNOT> is high if a test is not enabled.
- <NORUN.DOUT> is high if both motors are stopped.
- <CHKCLRTN.DOUT> will go high 1 second after the operator tries to enable a test.

- <POSORBUM.DOUT> is high if the pump is in position.

<PENABR> will reset <PENAB> if any of the following conditions are true.

- <RPENAB.DOUT> which is set high by the operator to reset the problem indication.
- <PBBUTSTP.DOUT> which goes high when the operator manually stops a test.
- <NOPENAB.DOUT> which is a small logic *and* block with the following input conditions.
  - <DO1-7DEV.QNOT> is high if no problem exists with the feedback parameters.
  - <NORUN.DOUT> is high if both motors are stopped.
  - <TSTRUNNG.QNOT> is high if a test is not running.
  - <POSORBUM.DOUT> is high if the pump is at the desired position.
  - <PBENRSRS.DOUT> is high if the test enable has timed out.

When <PENAB> goes high, <PBPROB> is set high which indicates a test operation problem. The small logic *and* block <PBENADIS> is used to bring up the problem indication display for enabling a test. When <PENAB> is high and the operator sets <PDISP> high to view the problem the display, PBENPROB is brought up via the *display* block <DISP3>.

Test Start. After test parameters are downloaded and a test is enabled the operator can start a test which will operate the pump for a desired period of time. The blocks in the following table are related to the test start task.

ABORTOR	NORUN	PBENABLE	PBENTON
PBSPLIT	PBSTRAN2	PBSTRAND	PBSTRTDN
PBSTTEST	PLCPBENA	STOPTEST	TSTRUNNG
VALSET			

The small logic *pulse* block <PBSTTEST> starts the execution of the test start task. Its output is passed to the small logic *and* blocks <PBSTRAND> and <PBSTRAN2> which must have the following input conditions true before starting a test.

- <PLCPBENA.DOUT> which is high if the test enable status bit is high in the PLC.
- <PBSTTEST.DOUT> is set high by the operator to start a test.
- <PBENABLE.DOUT> is high if a test is enabled.
- <TSTRUNNG.QNOT> is high if a test is not running.
- <VALSET.DOUT> is high if the test values are set.
- <NORUN.DOUT> is high if both motors are stopped.
- <ABORTOR.QNOT> is high if no abort condition is active.
- <PBENTON.DO2> is high when the test enable has not timed out.
- <PBSPLIT.DO3> is high if the elapsed test time is not greater than or equal to the desired time.

The output of <PBSTRAN2> is passed onto <PBSTRTDN> which starts the pump motor and sets the *rs flip-flop* block <TSTRUNNG> high. <TSTRUNNG> is reset from <STOPTEST> when a test is stopped.

Problem with Starting a Test. If the start of a test is not permitted, the blocks in the following table are used to indicate problems with the start test variables.

ABORTOR	DISP3	NOPSPRO2	NOPSPROB
NORUN	PBBUTSTP	PBENABLE	PBENTON
PBPROB	PBSPLIT	PBSTTEST	PLCPBENA
PSDISP	PSPROB	PSPROBA	PSPROBR
RPSPROB	TSTRUNNG	VALSET	

The *rs flip-flop* block <PSPROB> is used to indicate a problem with starting a test. <PSPROB> is set high if the output of <PSPROBA> is high and the reset input <PSPROBR> is low. <PSPROBA> is a small logic *and* block with the following inputs.

- <PBSTTEST.DOUT> is set high by the operator to start a test.
- <TSTRUNNG.QNOT> is high if a test is not running.
- <PBENABLE.DOUT> is high if a test is enabled.

<PSPROBR> will reset <PSPROB> if any of the following conditions are true.

- <RPSPROB.DOUT> which is set high by the operator to reset the problem indication.
- <PBBUTSTP.DOUT> which goes high when the operator manually stops a test.
- <NOPRPRO2.DOUT> which is a small logic *and* block that works with <NOPSPROB> to compile the following input conditions.
  - <PBENABLE.DOUT> is high if a test is enabled.
  - <PLCPBENA.DOUT> is high if the test enable bit in the PLC is high.
  - <ABORTOR.QNOT> is high if no abort condition is active.
  - <VALSET.DOUT> is high if the test values are set.
  - <NORUN.DOUT> is high if both motors are stopped.
  - <PBENTON.DO2> is high if the test enable did not time out.
  - <PBSPLIT.DO3> is high if the elapsed test time is less than the desired time.

When <PSPROB> goes high <PBPROB> is set high which indicates a test operation problem. The small logic *and* block <PSDISP> is used to bring up the problem indication display for starting a test. When <PSPROB> is high and the operator sets <PDISP> high to view the problem the display PSPROB is brought up via the *display* block <DISP3>.

4.2.5 PLC Communications

The MOTOR strategy has communications directly to the PLC for update and control of registers and bits in the PLC. The following table is a list of tags that are used for communications to the PLC.

PLC REGISTER	TAGNAME	DESC	BLOCK_TYPE
242	ST8WDRES	WATCHDOG BIT	DIGITAL OUTPUT
249	PBENAB	PHASE B TEST ENABLE TO PLC	DIGITAL OUTPUT
251	A5ABRT	AF5000 COMM ABORT	DIGITAL OUTPUT
2001-2016	J2C0		PAK DIGITAL INPUT
2015	ZIMPE144	CCW POSITION LIMIT	DIGITAL INPUT
2016	ZIMPE143	CW POSITION LIMIT	DIGITAL INPUT
2017-2032	J2C1		PAK DIGITAL INPUT
2017	MIP00001	MOISTURE IN PUMP MOTOR OIL	DIGITAL INPUT
2033-2048	J2C2		PAK DIGITAL INPUT
2042	ABORT	PLC ABORT COIL SET	DIGITAL INPUT
2043	PLCPBENA	TEST ENABLE FROM PLC	DIGITAL INPUT
2044	PLCWD	WATCH DOG FROM PLC	NOT
2048	NODEFAIL	STATION5 COMM FAIL	DIGITAL INPUT
40208-40215	I2C1		PAK ANALOG OUTPUT
40208	SWCABRT	PUMP CURRENT ABORT	SWCH
40209	DISPRABT	DISCH PRESSURE ABORT	ANALOG INPUT
40210	PBCALSEC	DESIRED HOURS FROM GENESIS	FX
40211	SWCALRM	PUMP CURRENT ALARM	SWCH
40212	PBCALSEC	DESIRED SECONDS FROM GENESIS	FX
40213	SWCSPDAB	PUMP SPEED ABORT SET	SWCH
40214	SWCSPDAL	PUMP SPEED ALARM SET	SWCH
40215	VR232020	PUMP MOTOR VOLTAGE	ANALOG INPUT
40216-40223	I2C2		PAK ANALOG OUTPUT
40216	VR232080	ROTATION MOTOR VOLTAGE	ANALOG INPUT
40217	VR232100	ROTATION MOTOR CURRENT	ANALOG INPUT
40218	VR232110	ROTATION MOTOR SPEED	ANALOG INPUT
40219	VR232040	PUMP MOTOR CURRENT	ANALOG INPUT
40220	VR232050	PUMP MOTOR SPEED	ANALOG INPUT
40223	DISPRALM	DISCH PRESSURE ALARM	ANALOG INPUT
42001-42008	J1C0		PAK ANALOG INPUT
42003	TIR12A02	MIX PUMP MOTOR OIL TEMP 2	ANALOG INPUT
42004	TIR12A01	MIX PUMP MOTOR OIL TEMP 1	ANALOG INPUT
42129-42136	I1C0		PAK ANALOG INPUT

42131	POSITION	POSITION FROM PLC	ANALOG INPUT
42217-42224	I1C11		PAK ANALOG INPUT
42217	PLCAMPAL	FROM PLC, AMPS ALARM LIMIT	ANALOG INPUT
42218	PLCAMPAB	FROM PLC, AMPS ABORT LIMIT	ANALOG INPUT
42219	PLCDPRAL	FROM PLC, DISCH PRESS ALARM	ANALOG INPUT
42220	PLCDPRAB	FROM PLC, DISCH PRESS ABORT	ANALOG INPUT
42221	HPSPDLIM	HIGH SPEED ABORT LIMIT	ANALOG INPUT
42222	VR050HAV	HIGH SPEED ALARM LIMIT	ANALOG INPUT
42223	PLCTIADD	PLC DESIRED HOURS	ANALOG INPUT
42224	PLCTIADD	PLC DESIRED SECONDS	ANALOG INPUT
42225-42232	I1C12		PAK ANALOG INPUT
42231	PLCCURAB	FROM PLC, PUMP CURRENT ABORT	ANALOG INPUT
42232	PLCCURAL	FROM PLC, PUMP CURRENT ALARM	ANALOG INPUT

To report a failure in the PLC, the *DIN* block <PLCFAIL> will alarm in the MOTOR strategy. <PLCFAIL> receives the signal from <PLCFAIL2> which is a small logic *or* block that has three input conditions.

- <PLCWDON.DOUT> goes high if the PLC toggle bit <PLCWD stays high for more than 2 seconds.
- <PLCWDOFF.DOUT> goes high if the PLC toggle bit <PLCWD> stays low for more than 2 seconds.
- <I1C0.FAIL> which will go high if there is a failure in communications in the ModPlus Genesis driver.

4.2.6 Alarming

All algorithm blocks with an alarm priority greater than 0 are enabled as alarming signals. If an alarm level is exceeded the tag and description will show up in the alarm summary to tell the operator which signal is in alarm. Refer to the data dictionary of the MOTOR strategy for all alarm signals with alarm priorities greater than 0.

4.2.7 System Parameters

There are several Report blocks used to execute R&R code. Following is a list of Report blocks in the strategy with the R&R code that each block serves. Refer to Appendix C for a description of each R&R code.

TAG	SOURCE1	SOURCE2	SOURCE3	SOURCE4
TESTSET2	ULOAD	NEXT	PREV	ILOAD
PBREPORT	PBSETVAL			
TESTSET	SAVE	DELETE	CLRMESS	
INITENAB	INITENAB			

The *d flip-flop* block in the center of the database contains the version of the strategy as its tag name. For example, <V3.00> would be the tag name for version 3.00 of the strategy. The block <VERSION> contains the version number in the descriptive field.

4.2.8 Stop Test

The stop test task will stop operation of either motor and reset operator status feedback. The following blocks are used for the stop test task.

ABORTPUL DISP4 PBBUTSTP PBDNTM
STOPTEST

The operator can manually stop a test by setting <PBBUTSTP> high. The output of <PBBUTSTP> is passed to <STOPTEST> which stops the motors and resets the desired pump operation time with the macro RESPTIM in the *display* block <DISP4>. <STOPTEST> also resets the operator feedback status bits for test operation. <STOPTEST> also is controlled by <ABORTPUL.DOUT> which goes high if an abort condition becomes active. <PBDNTM.DOUT> goes high if the elapsed pump operation time exceeds the desired time and sets <STOPTEST> high.

4.3 MODICON PLC FUNCTIONS

The Modicon PLC is responsible for inputting data from the field instrumentation, providing an abort signal when the values of critical measurements exceed their abort limits, providing control signals to field instruments, providing a timer and enable logic for pump operation, sending data and status information to Genesis and providing a directional motor simulation. Details of these functions will be presented in the following sections.

Access to and programming of the PLC is provided by the program Modsoft. Modsoft is run on STATION1 in the DACS trailer and communicates with the PLC via the Modbus Plus network.

Using Modsoft, the PLC I/O configuration can be defined, the PLC registers can be allocated and named, and the ladder logic can be produced. All of this can be done offline and later downloaded to the PLC, or can be done online in the PLC as it operates. The latter is not a wise policy for making permanent changes, but can be useful for debugging ladder logic.

Modsoft allows the ladder logic programmer to impose a structure on the ladder logic for ease of understanding. It allows the programmer to break the ladder logic into modules known as objects and to connect the objects using flow control logic. Each object can contain any number of ladder logic networks. This structure is converted by Modsoft to an equivalent flat nonstructured form when the logic is downloaded to the PLC. In addition, the ladder logic can be distributed in different logic segments. The segments provide another way to organize the ladder logic program. The logic in each segment can be set to be solved either on each pass or conditionally. Also, an I/O drop can be associated with a segment; the drop inputs and outputs are processed when the associated segment is active.

The current PLC program has logic in the first four segments. The first segment contains only the hot standby block. This block controls the hot standby functions of the PLCs. The second segment contains most of the operational logic and will be discussed extensively below. The third segment contains the block copy logic for creating the Genesis I/O region. The fourth segment contains the directional motor simulator logic.

Segment two is the only segment in which the logic is divided into objects. It currently has seven objects (P000, P001, and P003 through P007) which execute sequentially (no control structures mediating them). Their functions are listed below:

- P000 (15 Networks) - PLC status logic, test timer and enable logic, instrument control logic.
- P001 (1 Network) - PLC rack and module status logic.
- P003 (64 Networks) - Abort logic and instrument fail logic.
- P004 (10 Networks) - RGA5 ASCII/BASIC module control logic.

- P005 (13 Networks) - GC3 and FTIR ASCII/BASIC module control logic.
- P006 (5 Networks) - High frequency strain alarm filtering.
- P007 (54 Networks) - Thermocouple module setup and control logic.

Modsoft has a documentation feature which allows information about the PLC configuration and ladder logic to be either saved to a file or printed.

4.3.1 Data Collection From the Field

The PLC is responsible for collecting data from the field instruments. These data enter the PLC through any of a number of Modicon I/O modules. There are several different types of modules which accept different types of input. The modules used in the system and their descriptions are listed below. Both input and output modules are listed in this table. The output modules are used for various control functions discussed in Sections 4.3.2 and 4.3.3.

Module	Description	Signals Accepted
B875-101	8-channel analog input	Configurable for various voltage or current input ranges
B827-024	32-channel digital input	24-V digital input
B885-002	ASCII/BASIC module	RS-232 serial ASCII data
B829-016	16-channel digital input	5-V TTL digital input
B883-200	Thermocouple module	Thermocouple types B, E, J, K, N, R, S, & T
B865-002	TTL Register input	5-V digital output
B824-016	16-chan. digital output	24-V digital output
B828-016	16-chan. digital output	5-V TTL digital output

Each of these modules has a physical location and a set of registers through which it communicates to the PLC. This location and the exact registers used for communication are specified in the configuration module of the Modsoft program. For all but the B885 ASCII/BASIC module and the B883 thermocouple module, this is all that needs to be done in order for the PLC to have access to the data from these modules. Both the ASCII/BASIC module and the thermocouple require additional ladder logic to set up and operate the module. The operation of these modules will be discussed in the next two sections.

Most of the measurements from the field enter the system through the B875 analog input module. This module can be set to accept inputs with differing characteristics. Commonly used in this system are 1-5 V, 0-5 V and 4-20 mA signals. These inputs are digitized with 12 bits of resolution. This 12-bit value is stored in the lower order 12 bits of the Modicon register corresponding to that channel. The most significant bit of the same

register is the out-of-range bit. The B875 module will set this bit if it detects an incoming signal which is out of the range specified when the module is set up. For example, if the module has been set to receive 1 to 5-V inputs and then receives an input of 0 Vs, it will set the out-of-range bit. This out-of-range bit is used for instrument failure detection, a topic which is discussed in Section 4.3.5.

4.3.1.1 ASCII/BASIC Module Operation

The Modicon B885 ASCII/BASIC Module allows ASCII data from a computer or from analytical equipment to be read into the Modicon PLC. The ASCII data are presented via one of two RS-232 ports on the front of the module. The module contains a stripped-down BASIC interpreter. BASIC programs can be written that read the ASCII data from the ports and pass it to the PLC via the command interface registers.

The Module has two operational modes: RUN and PROGRAM. These are selected via a switch on the front panel. In PROGRAM mode, the module communication parameters can be set and the BASIC interpreter accessed for programming. In the RUN mode, the module is under the control of the PLC ladder logic via the command/status registers or under control of its own internal scheduler. These are used to determine which BASIC program stored in the module's memory to execute, and when.

Details of the module configuration, programming, and operation are given in the Modicon B885 ASCII/BASIC Module User Guide.

The DACS uses two ASCII/BASIC Modules: one for the RGA5 gas data (RGA5), and one for gas chromatograph, infrared spectrometer and photo gas concentration data (GC3) (see Appendix J for complete file listings). Both of these require a BASIC program to read the ASCII stream, strip off unwanted characters, format the data for the PLC registers, and send this data to the PLC. They also require ladder logic to control and monitor the operation of the BASIC programs, receive the data from the interface registers, and place it in the appropriate final destination registers.

The BASIC programs and ladder logic are therefore quite similar in concept but differ due to the differing data formats and conventions used by the devices. The RGA5 ladder logic is in Segment 2, Object P004 and the gas chromatograph ladder logic is in Segment 2, Object P005.

4.3.1.2 ASCII/BASIC Module Communications with the Gas Chromatograph

The gas monitoring computer sends its data to the ASCII/BASIC module as a stream of ASCII characters beginning with a STX (start of transmission) character and followed by 14 four-digit ASCII encoded numbers separated by carriage returns (↵) (see table below). These communication parameters are: 1200 Baud, 8 bits, 1 stop bit, no parity.

The BASIC program (filename GC3.BAS) in the ASCII/BASIC module is responsible for reading these incoming characters, interpreting them, placing them in

registers and sending them to the PLC through the module's command/data register interface.

GC3 Data Format		
Label	Description	Value Transmitted
STX	Start of Transmission	2
GC3-TIME	GC3 Time of Sample	nnnn↵
GC3-AREA	GC3 Peak area	nnnn↵
GC3-RT	GC3 H2 Retention Time	nnnn↵
GC3-H2	GC3 H2 Concentration	nnnn↵
GC3-FILE	GC3 File ID	nnnn↵
FT-TIME	FTIR Time of Sample	nnnn↵
FT-N2OA	FTIR N2O Area	nnnn↵
FT-N2OC	FTIR N2O Concentration	nnnn↵
FT-NH3A	FTIR NH3 Area	nnnn↵
FT-NH3C	FTIR NH3 Concentration	nnnn↵
FT-FILE	FTIR File ID	nnnn↵
PHO-TIME	Photo NH3 Time of Sample	nnnn↵
PHO-MSB	Photo NH3 Conc. - most significant byte	nnnn↵
PHO-LSB	Photo NH3 Conc. - least significant byte	nnnn↵

nnnn↵ = four ASCII numeric characters followed by a carriage return.

The PLC ladder logic is responsible for running the BASIC program, issuing commands to read the status of the module, and to read data from the module and placing the data into the proper registers so it can be sent to Genesis.

Communication between the PLC and the BASIC module is through the 6 input and 6 output interface registers assigned to the module during configuration. In addition, there is an array of 100 registers internal to the BASIC module which can be accessed by the PLC (up to 5 at a time) through these interface registers. There are also eight status bits which can be set or reset directly by the BASIC program and read by the PLC along with the hardware status bits.

In general, the normal sequence of operation of the system is as follows (for more specifics, see the ladder logic comments, the BASIC program listing and the ASCII/BASIC module manual):

PLC Operation

1. The ladder logic checks the ASCII/BASIC module status register. If there is an error condition, it resets the module. If the BASIC program is not running, it issues the command to run the BASIC program.

2. The ladder logic continues to monitor the status register for error conditions and checks the data ready user flag (GC3_SND8). This is the signal from the BASIC program that a set of data has been read and is ready to be transferred to the PLC.

3. When the data ready flag has been received, the ladder logic goes through a sequence of commands which transfer the data from the ASCII/BASIC module's internal register array. The data are transferred one to four registers at a time and stored in registers in the PLC for transference to Genesis. The 14 data items to be transferred are stored in the ASCII/BASIC module's internal registers 0-13. The PLC ladder logic issues 7 commands in sequence to transfer these 14 data items.

4. The ladder logic sets the ASCII/BASIC module's internal register 30 to 1 (data transfer complete flag). This is the signal to the BASIC program that the data transference is complete.

5. Go to step 1.

BASIC Program Operation

1. The program is initialized by the PLC ladder logic. It immediately sets the data ready flag to zero (although it should already be zero) and the data transfer complete register (register 30) to zero.

2. The program reads characters from serial port 1 until a STX character is received. The start of transmission character signals the beginning of the gas data stream.

3. After the STX is received, the program reads 14 numeric values and stores them in internal registers 0 through 13.

4. The program sets the data ready flag [SND(8)], indicating to the PLC that there is a set of data in the registers to be transferred.

5. The program waits until the data transfer complete flag is set (Register 30). This will indicate that the PLC has transferred the data.

6. The program clears the data ready flag, then exits. When the PLC detects that the program has exited, it will rerun the program.

4.3.1.3 ASCII/BASIC Module Communications with the RGA5 Computer

Following is the data format for the RGA5 ASCII/BASIC module communication:

```
Run=nnn,  Stream =cccccccc, Date=nn-nn-nn, Time=nn:nn
H2A=nnnn.n, H2A Area=nnnnnnnnn, H2ART=nnn, H2Arf=nnnnn.n
H2B=nnnn.n, H2B Area=nnnnnnnnn, H2BRT=nnn, H2Brf=nnnnn.n
```

where, n = any numeric character and c = any printable ASCII character. *Run* is the run number and is incremented by one with each data stream. *Stream* is set to Tank, Flush or Calib to indicate the source of the data. The BASIC program only looks at the first character of this field. *Date* is the sample date. This is ignored by the BASIC program. *Time* is the time of sample in the format hours:minutes. *H2A* is the GC-1 hydrogen concentration in ppm. *H2A Area* is the GC-1 hydrogen peak area. *H2ART* is the GC-1 hydrogen retention time in seconds. *H2Arf* is the GC-1 hydrogen retention factor. This is ignored by the BASIC program. *H2B* refers to the same information as above except it pertains to GC-2 instead of GC-1. The numeric data will be right justified with leading zeros inserted to preserve the field widths.

The communications parameters are: 1200 baud, 8 bits, no parity, 1 stop bit. The stream must enter the ASCII module via port B. The XON/XOFF protocol should be enabled.

The BASIC program operates by searching the ASCII stream for the word "Run", indicating the start of the data. It then issues reads to the ASCII message processor to input the data. After the data has been input, the program places the data into contiguous registers to be read by the PLC. In some cases a datum is placed unchanged into a register. In other cases, the data is scaled or manipulated in some way in order to make it fit into a 16 bit register. The PLC is then notified that data is available. When it has read the data, it notifies the BASIC program that the data has been read. The program then exits and is run anew by the PLC.

The issuing of ASCII read messages by the program means that these message formats must be loaded into the memory of the ASCII portion of the ASCII/BASIC module. The following are the ASCII read message formats:

```
RMSG1:   10   S0,D3
RMSG2:   10   S1,A1
RMSG3:   10   S2
RMSG4:   10   S2,D2,D2
```

```
RMSG5:    10    S4,F6.1
RMSG6:    10    S6,D1,D4,D4
RMSG7:    10    S9,D3
RMSG8:    10    S10
RMSG9:    10    S10,F6.1
RMSG10:   10    S12,D1,D4,D4
RMSG11:   10    S15,D3
RMSG12:   10    S16
```

In addition, the program uses the prefix capability of the ASCII module to determine the beginning point of the ASCII reads. The prefix instructs the ASCII message processor to ignore all incoming characters until the prefix string is read, then begin the formatted read. In this case, the prefix string of the module must be set to "=." The command to be issued is: PR# 3Dh. Also, the delimiter character should be set to null: DL# 0.

The program must be loaded into RAM 2 of the ASCII/BASIC module.

4.3.1.4 Thermocouple Module Operation

The B883 thermocouple modules receive thermocouple inputs from the MIT17B and MIT17C tank temperatures, and the tank bottom and side thermocouples. Three groups of modules are used: three modules to input the MIT17B tank temperatures (22 measurements), three modules to input the MIT17C tank temperatures (22 measurements), and three to input the tank bottom and side temperatures (26 measurements). These modules require ladder logic to set up and operate the modules.

The ladder logic can be broken into three major sections: (1) creation of the setup table and command table, (2) configuring the module, and (3) operating the module. The configuration of the three modules of each group is the same so only two setup tables are needed. The following is an overview of the ladder logic needed to operate one of the modules.

Figure 14 shows a portion of the ladder logic used to create the thermocouple module setup table. Since no predefined constants are available in ladder logic, the contents of the configuration table must be created from scratch when the PLC is first brought online. The SUB blocks are used to load the values in the top node of the SUB blocks into the registers indicated in the bottom node of the SUB blocks. These registers form the setup table. They are grouped in threes, since these represent command parameters which will later be sent to the thermocouple modules. In the figure the #0301-#0000-#0000 sequence is the thermocouple module STOP command, and the #0290-#0000-#0000 sequence is the thermocouple module ENTER CONFIGURATION MODE command. A complete list of the available commands can be found in the thermocouple module manual.

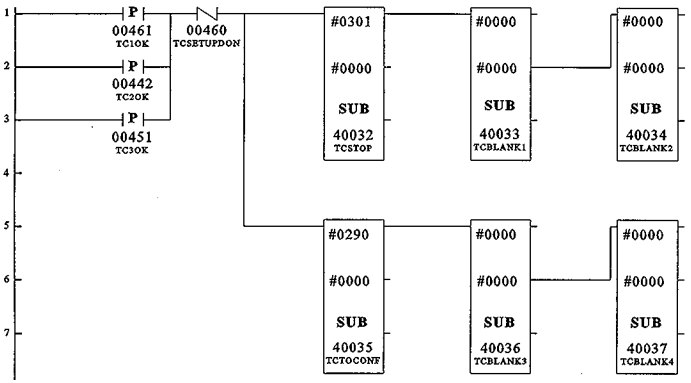


Fig. 14. Ladder logic for thermocouple setup table creation.

The setup table, when complete, contains a set of commands which will be issued sequentially to each module in the group during configuration mode. These commands will configure the module, giving it the thermocouple type to expect upon input and the data format in which to report the channel values. Each channel must be configured separately so, although in our case all of the channels are configured the same, we must still set the configuration parameters for each channel.

Figure 15 shows a portion of the ladder logic for command table creation. The command table is a list of commands which will be issued to the module one after another when the system is in operational mode. In our case, the command table consists of 10 read channel value commands, one for each of the ten channels available on a single module. In this case, *ADD* blocks are used to set the channel parameters. The 984A PLCs do not allow constants larger than 999 in a block, so, for example, the "read channel one value" command, 1001, is created by adding 2 to 999.

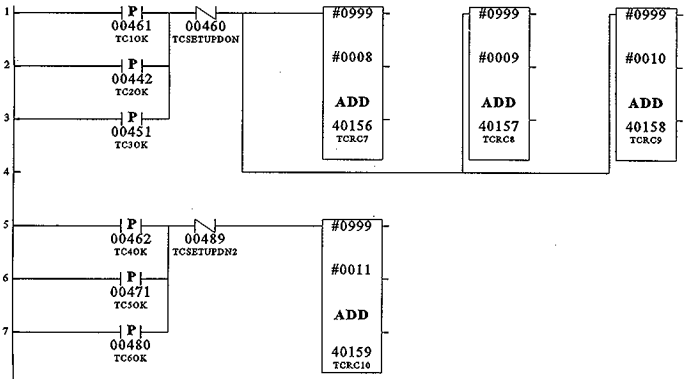


Fig. 15. Ladder logic for thermocouple command table creation.

After these two tables have been created, the modules must then be configured according to the commands now in the setup table. Figure 16 shows the logic which issues these configuration commands. Register 40173 is the setup table pointer. This logic will load the three register command sequences into the command registers, 40091-40093, incrementing the setup table pointer each time. This is done until the end of the setup table is reached and coil 435 is set, indicating that configuration is done. The next set of commands is issued when coil 434 goes high indicating that the previous command has been accepted by the module. The setup sequence can also be initiated when a module first comes online (coil 461 goes high) or after an error has occurred (coil 437 goes high).

After the configuration is complete, the thermocouple modules enter operation mode. In this mode, the commands in the command table are executed. When the end of the table is reached, the system resets the command table pointer to the beginning of the table and goes through the table again. The command table consists of commands to read each of the 10 thermocouple data values one after the other.

In Figure 17, register 40150 is the beginning of the command table. Register 40171 is the index into the command table, and register 40172 is the register which receives the value pointed to by the combination of 40150 plus the offset in 40171. This value is loaded into the thermocouple module command interface register, 40091, thereby issuing the command.

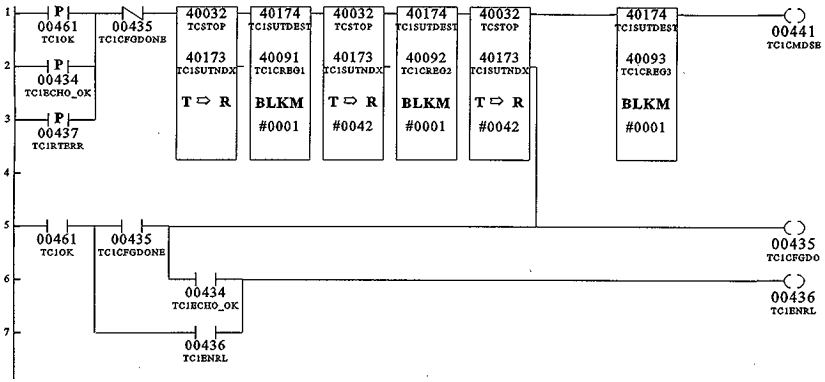


Fig. 16. Ladder logic for configuring thermocouple modules.

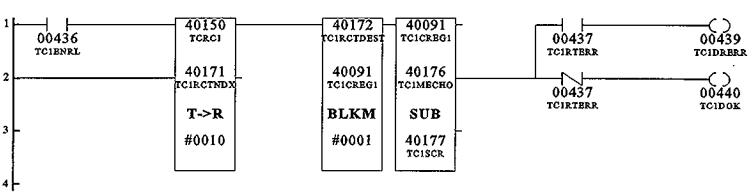


Fig. 17. Thermocouple operation: extraction of commands from command table.

Figure 18 shows the ladder logic for reading the data values from the module after the read channel value command has been issued and successfully echoed. Register 40160 is the index to a table called the data value table. This table is where the thermocouple temperature values are stored after they are read. Register 30277 is the interface register which contains the data value from the module. This is placed in the position in the table given by register 40160. This register is automatically incremented. The *ADD* block increments the command table index so that the next command can be issued. If the end of the table has been reached, the *SUB* blocks are activated. These blocks reset the value table index and the command table index to the beginnings of their respective tables so the process can repeat.

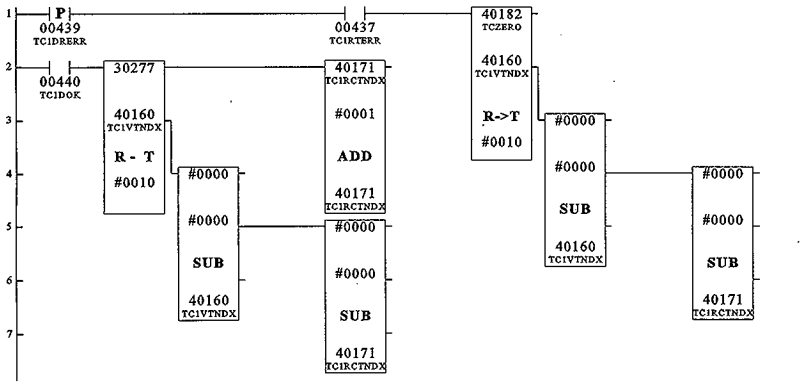


Fig. 18. The thermocouple modules reading thermocouple data values.

The upper pathway is activated when there is an error (coil 437, runtime error, is high). This places a zero into the data value table and resets the data value pointer and the command table pointer.

4.3.2 Abort Functions

One of the primary monitoring functions of the Modicon 984 PLC in the DACS system is to check for over- or under-range conditions on various safety parameters. These parameters are sensed by the 984 series I/O at drops or I/O stations distributed around tank 241-SY-101 at surface level, and then transferred to the 984 PLC crate inside the DACS trailer. Once received by the PLC, these critical values are stored in 30xxx registers. These are two-byte registers that function as repositories for incoming data. Current values of the important parameters are now loaded in the PLC, and are ready for comparison with limit values. This comparison takes place in the ladder logic in the PLC and is of the form below in Figure 19. Register A is the current value register.

Register B is the 40xxx register containing the limit to be compared with the current value. Register C is the 40xxx register containing the difference between the values in A and B. The subtract block actually has three outputs available for user logic. The top output is

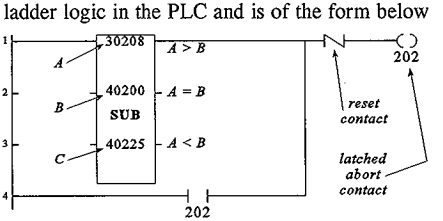


Fig. 19. Abort limit testing.

activated if A is greater than B. This is the case in which the limit abort logic is mainly interested. With this line high and the reset contact sense in the nonreset condition, the latched abort contact 202 is closed. When it closes, the contact sense in the bottom leg holds or latches the contact closed. Nothing can interrupt power to contact 202 except if the reset contact should close. The reset contact in the above diagram is a "not" of the true reset contact. This means that if the reset contact is open, the ladder element conducts power (is closed).

Values for the abort and alarm limits stored in the 4xxxx registers are written from the TEST strategy in STATION5 to the PLC.

In the PLC, there exists ladder logic similar to the above for every limit abort parameter; however, the latched abort contact is distinct for each parameter. The reset contact is common to all the logic. Figure 20 shows how the many abort contacts are OR'ed together to set a single abort contact. Here the reset contact is identical to that in the previous diagram and all other limit abort logic. The composite abort contact is held closed by the latching leg containing the sense of contact 021. It can only be reset by making the reset contact close. This is done from Genesis. Notice below that the abort contact 202 from the logic above is represented as only one of many such contacts combined to set the composite abort contact.

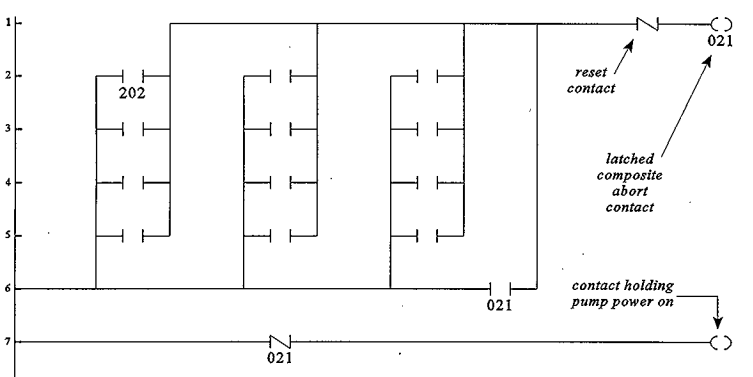


Fig. 20. Composite abort contact.

When the latched composite abort contact closes, there is a contact shown in the bottom right-hand corner of the logic above that is opened upon an abort. This power contact is directly connected to a digital output Modicon module that normally holds the

power breaker to the pump and directional motors closed. When this contact is opened, the power to the motors is interrupted.

When any of the latched abort contacts closes, a display in STATION5 called CSMAIN shows the class of the abort along with the alarm limit value, the abort limit value, and a button to another screen with more detail on the abort.

The class of the abort will be one of the following:

High H2 Concentration	High Waste Tank Level
High Pump Flow	High Pump Speed
High Duration Limit	High Waste Temperature
High Pump Discharge Pressure	High Pump Motor Current
High Pump Motor Oil Temperature	High Pump Motor Oil Moisture
High Pump Column Strain	High Pump Column Vibration
Low Pump Column Gas Pressure	MIT Column Strain
VDTT Column Strain	Low Ventilation Flow
High Tank Dome Pressure	

There is also an indicator of the status of the PLC abort coil and a reset button for it. The text on this screen signifying the class of the abort becomes red if a tag in its class has caused an abort. If the operator presses the button next to a class, another more detailed screen is called up. On this next level of display tag names, tag descriptions, alarm and abort limit values, tag current values, and units are listed. In addition, there is an indicator that other alarms or abort conditions exist as well as an indicator for the PLC abort contact.

As a safety factor, the composite abort contact is read by the Genesis strategy in STATION8. The tag name of the particular block holding this state is <LIMABORT>. When the MOTOR strategy in STATION8 detects a problem, a report executes that issues stop commands to both the pump motor drive and the directional motor drive.

4.3.3 Control Points and Alarm Outputs

The PLC provides a few control and alarm signals other than the pump motor shutoff signal. They are: a signal which triggers the Nicolet data collection system and a signal which causes an audible alarm to sound when communications from either Genesis STATION5 or STATION8 stops.

The Nicolet trigger signal is set whenever any of the following occur: (1) there is a high-frequency strain alarm condition, (2) there is a low frequency strain alarm condition, or (3) the pump motor is running. The Nicolet system captures high-speed strain data. This allows the system to be triggered during a tank rollover.

Figure 21 shows the Genesis to PLC watchdog timer ladder logic. Genesis sends an alternating digital signal which comes into on PLC coil 213. This signal alternately resets one of the two timers while allowing the other to run. If communication from Genesis ceases, one of the two timers will be stuck in the run state and will eventually time out, setting coil 20. This coil is wired to an audible alarm circuit, which produces an alarm when the coil is set.

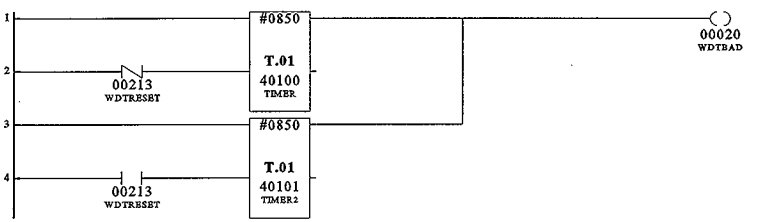


Fig. 21. Genesis to PLC watchdog timer.

4.3.4 Test Timers and Enable Logic

The PLC contains ladder logic which implements a test enable mechanism and timer for timing pump tests. Figure 22 shows this logic.

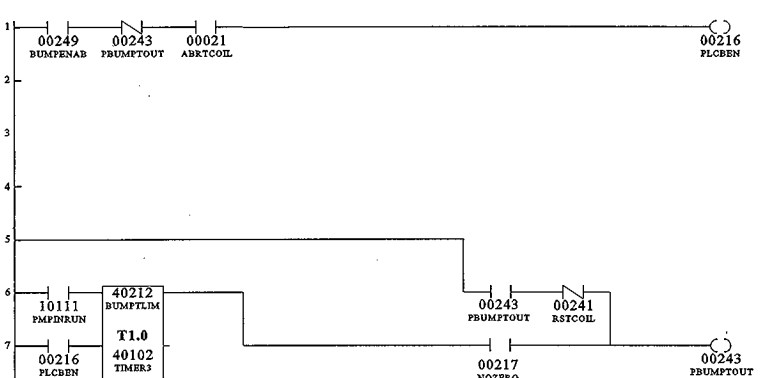


Fig. 22. Test enable and timer logic for pump bump.

The pathway in the upper part of the figure is the test timer enable mechanism. The test enable signal originates from STATION8. The operator presses the test enable button on one of the pump run screens. The enable signal is passed from the MOTOR strategy to the PLC where it becomes coil 249. This enable signal starts the test timer as long as there is no abort current in the PLC logic (indicated by coil 21). The enable signal must remain set continuously in STATION8 for the duration of the run. Coil 243 causes the timer to reset when the test time has elapsed regardless of the state of the enable from STATION8. This coil is also incorporated into the pump abort logic, causing the pump to shut off when the allowed test time has elapsed.

The timer block is set to time the duration of a pump run and will shut off the pump if the elapsed time exceeds the allowed time. The allowed time is loaded into register 40212 by the Motor strategy. The elapsed time is in register 40102. The timer will run only if the test has been enabled (coil 216 is high) and the pump is running (input 10111 is high). If the allowed time elapses the output of the timer will go high. This sets coil 243 high provided that coil 217 is high. Coil 217 is set only if the timer count and elapsed time are both nonzero. This prevents the timer from causing an abort condition before the test has begun.

4.3.5 Data Transfer and Status Information

Most data to be transferred from the PLC to Genesis is organized into contiguous blocks before transfer. This increases the efficiency of the data transfer. This is accomplished in segment 3 of the ladder logic by using block moves.

Other than the ASCII/BASIC module data and the thermocouple data discussed in Sections 4.3.1.1 and 4.3.1.2, there is one other case in which additional PLC processing is needed to prepare data for Genesis. This is required to slow down the possibly momentary high-frequency strain alarms so that Genesis can have enough time to detect them before they disappear.

Figure 23 shows the PLC ladder logic used to extend the duration of the high-frequency strain alarms. An alarm can appear as a high on any of the input register 10193, 10195, 10197, etc. When a positive-going pulse is detected on any of these channels, and if an extended alarm is not already present for that channel, coil 167 is set. This coil indicates that a new alarm has arrived.

In addition, one of the coils 209-215 is set and latched. The coil set is the one corresponding to the alarm received. These are the coils which are sent to Genesis to become the high-frequency strain alarms. The incoming signal on the 10xxx register could have been just a momentary signal that disappeared before the alarm was transferred to Genesis. The latching mechanism ensures that the alarm will remain long enough to be sent to Genesis.

To reset the latches, a timer block is provided. This timer will begin to run if an alarm is received and will time out after 2 seconds causing any latched alarms to be reset. If a second alarm is received before the timer has timed out, the timer count will be reset and the timer will continue to run. This ensures that all latched alarms will remain latched for at least 2 seconds.

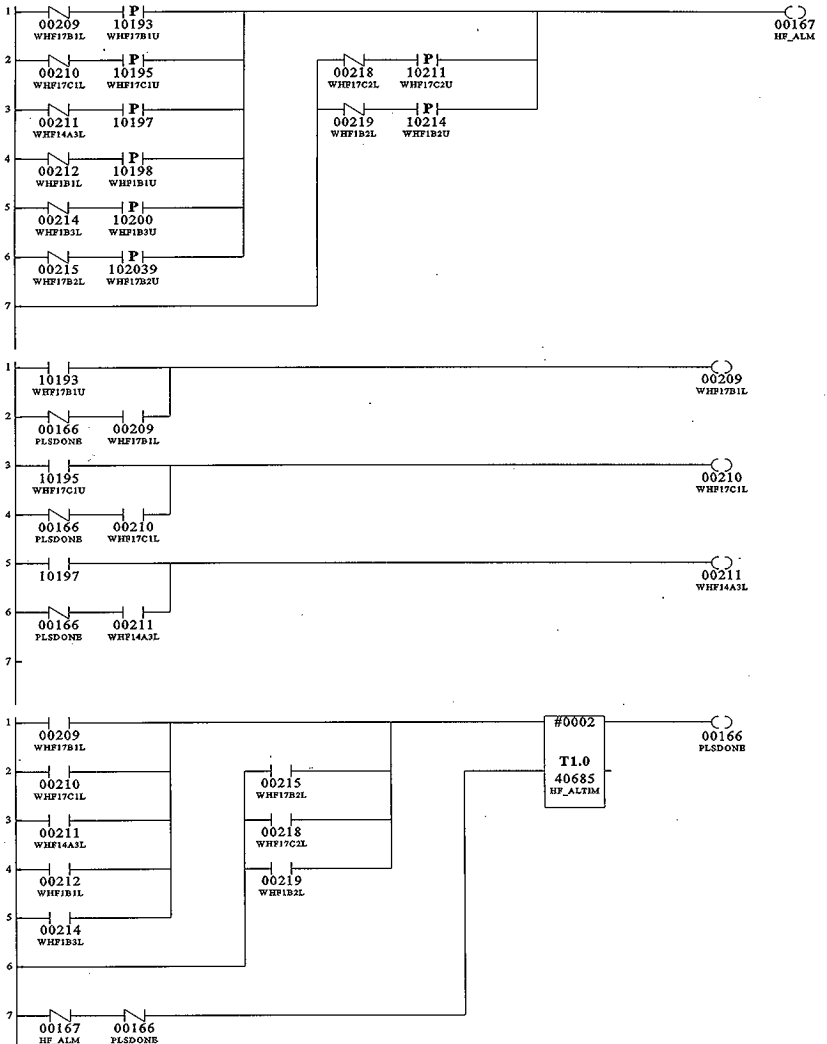


Fig. 23. High-frequency strain alarm monitoring logic.

The PLC also provides status information to Genesis. It provides a signal which monitors the PLC and the hot standby PLC. This signal is 1 if both PLCs are okay, and 0 if either of them is offline. This information comes from extracting information from the PLC status table which begins at register 40301. Refer to the Modsoft PLC manual "Modicon 984 Programmable Controller Systems Manual" for the details of the status table.

The PLC also provides information to Genesis about the health status of I/O modules in the system. The following four conditions must be met for a module to indicate good health:

- The slot has been configured in the Traffic Cop.
- The slot contains a module with the correct personality.
- Valid communications exist between the module and the J890.
- Valid communications exist between the J890 and the 984 PLC.

Module health is stored in registers in the Modicon PLC. Different PLCs use different registers. To standardize access to this information, a STAT block is added to the ladder logic. During runtime, the status information is moved to registers defined by the STAT block.

None of the drops are full of modules. Thus, not all of the status registers are of use. A STAT block register contains the I/O health of one crate (alternately referred to as a rack) of modules on a drop. All told, 13 racks are currently in use.

To minimize the communication load, the status registers that correspond to existing racks are copied into a set of transmit registers. It is the transmit registers that Genesis views.

The movement of the STAT registers to the transmit registers is accomplished by using BLKMs (block moves). The order of the STAT registers is by drop (1 through 32) and by rack (1 through 5) within each drop. Only in cases where two crates have been used within a given drop are the registers contiguous and thus can be handled by a single block move, as shown in Figure 24.

To streamline communication, the Genesis TEST strategy gathers the transmit registers in as packed data. The 13 registers fit within 13 packed digital inputs (PDIN) fanned out from DEV 4, GC11 (group 11). The names of the *PDIN* blocks reflect the drop and crate numbers of the involved modules. The drop number is prefaced by a D, the crate number is prefaced by an R (for Rack). Thus, the *PDIN* block for drop 2 crate 1 modules is named D2R1.

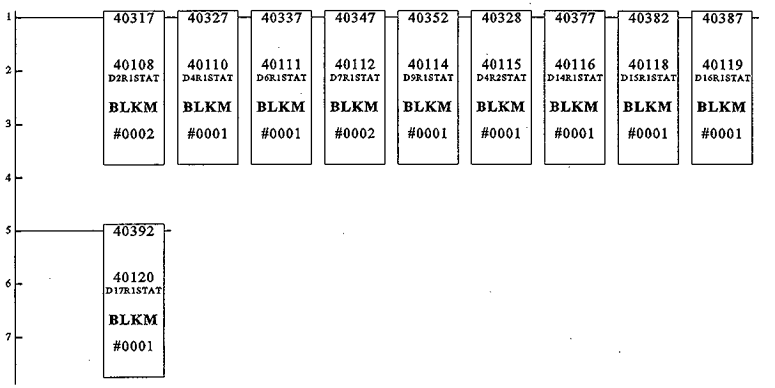


Fig. 24. Block moves for transmitting I/O health status.

Each *PDIN* block contains 16 bits. Each block represents modules 1-11 in a given crate (plus 5 unused bits). Because Modicon defines its bits opposite of Genesis, this corresponds to bits 16 through 6 in Genesis. (The data also require byte swapping, which is provided in DEV 4.) Note that alarms for *PDIN* blocks are reported by bit number.

Display of the I/O health is available on the graphic IOSTATUS. Each point in the matrix is tied to the *PDIN* block bit corresponding to the module's health. Thus, the matrix point for drop 2 crate 1 slot (or module) 4 is tied to D2Rl.DO13.

Good health is indicated by a green OK; bad health is indicated by a red BAD. On startup of Genesis, the graphic will show all the modules in bad health until one scan of transmit registers has been made. The scan rate of the transmit registers is 30 seconds.

A complete table of these PLC I/O status registers is given in the "I/O Health Status Implementation Table" in Appendix A.

4.3.6 Directional Motor Simulator

Segment four of the PLC program contains the directional motor simulator logic. This logic simulates the rotation of pump in the lab environment. This logic receives the directional motor acceleration setting, the directional motor speed setpoint and whether or not the motor is in forward or reverse from the Motor strategy. It then increments or decrements the value of ZIMPE112, the position indication.

The simulator is enabled by setting coil 171 to 0 using Modsoft. It also requires the real position encoder (ZIMPE112) be at 0. If there is a non-zero reading for ZIMPE112, then the simulation is disabled and the real value is passed to Genesis.

The simulation logic calculates the number of seconds required to move the directional motor 1 degree. This calculation is based upon the directional motor simulated speed (RPM). The value is calculated using a value of 13.09 degrees for each revolution of the pump motor.

4.4 NETWORK LAYOUT

The GEN-NET network is composed of eight GEN-NET network stations with ARCNET in a star configuration, as illustrated in Figure 25.

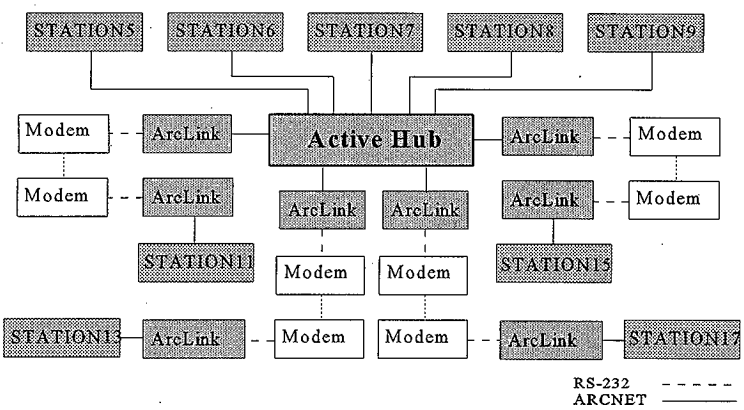


Fig. 25. GEN-NET network configuration

The Genesis system allows peer-to-peer communications between Genesis applications. STATION5 serves data to all other stations on the GEN-NET. STATION6, STATION7, STATION11, STATION13, STATION15 and STATION17 all link directly to STATION5 and share the database TEST strategy running on STATION5. These stations accomplish this by running Remote Supervisory Software with the TEST strategy on each station.

STATION5 uses GEN-NET software to enable networking to a Runtime master station. STATION8 is connected to the network for offline file transfer but is not connected when the motor Standby is running.

STATION7 normally serves as a Remote Supervisory Station but is also used as a backup for STATION5 or STATION8. In this instance STATION7 serves as a Runtime master station.

STATION9 acts as a file server on the network to store data files. STATION9 is also connected to the HLAN and is connected to a file server using Novell software.

Figure 26 shows the current network data flow.

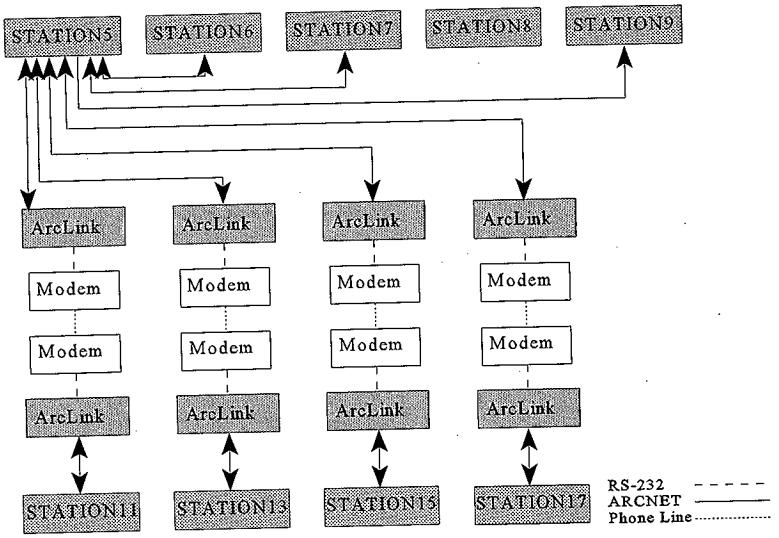


Fig. 26. Network data flow.

Each station is addressed on the GEN-NET by the network card address on the ARCNET. A station name is linked to a physical address in the GEN-NET software. Refer to Appendix G for details on the network layout.

Appendix A: PLC I/O Status Registers

The Modicon I/O health status alarms Genesis when a module is no longer in good health. In Section 4.3.5, the facility for Genesis determining the health of the I/O modules in the system was discussed. This included the conditions for good health, the STAT blocks used in the ladder logic to define the module health registers that are transmitted to Genesis, and the *PDIN* blocks in the Genesis strategy that receive them.

The following is a complete list of the PLC registers and their corresponding Genesis strategy variable names.

Table B-1. I/O Health Status Implementation Table

Drop	Rack	Slot	STAT block Register	Transmit Register	PDIN NAME	Output
2	1		40317	40108	H3C0	
		4				DO13
		5				DO12
		6				DO11
2	2		40318	40109	H3C1	
		3				DO14
		4				DO13
4	1		40327	40110	H3C2	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
4	2		40328	40115	H3C3	
		3				DO14
		4				DO13
		5				DO12
		6				DO11
		7				DO10
6	1		40337	40111	H3C3	
		4				DO13
		5				DO12
		6				DO11
		7				DO10

8	1		40347	40112	H4C0	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
8	2		40348	40113	H4C1	
		3				DO14
		4				DO13
		5				DO12
9	1		40352	40114	H4C2	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
13	1		40372	40120	H6C0	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
		8				DO9
14	1		40377	40116	H5C0	
		4				DO13
		5				DO12
		6				DO11
14	2		40378	40117	H5C1	
		3				DO14
		4				DO13
		5				DO12
		6				DO11
		7				DO10
15	1		40382	40118	H5C2	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
16	1		40387	40119	H5C3	
		4				DO13
		5				DO12
		6				DO11
		7				DO10
		8				DO9

Appendix B: PLC Register List

The Modsoft program allows a symbol table to be created. This symbol table allows the programmer to assign symbolic names and comments to the registers used by the PLC. The symbol table is accessed under the Utility Menu of Modsoft. A hard copy print-out can be obtained by using the File I/O and Export menu choices once the symbol table has been called up.

The symbol table is essential for the ladder logic programmer. It is the definitive record of which PLC registers have been used and which are available for use. Any entry in the symbol table which has a symbolic name has been used by the PLC. If there is no entry in the symbol table for a register, or if there is a register entry without a symbolic name, that register is available for use.

It is essential that the symbol table be kept up to date. Any change which adds or deletes PLC registers (including adding I/O modules) must be reflected in the symbol table also.

Appendix C: Report and Recipe File Listings

Report and Recipe (R&R) code is controlled by the state of bits in the strategy. These bits are connected to *REPORT* blocks which reference the code as a source file.

R&R code is composed of a command and function set provided by the R&R software option. These commands and functions must follow a syntax format explained in the R&R Option document provided by Iconics.

Following is a description of each R&R code's function for the MOTOR and TEST strategies and when it is executed. Refer to the code itself for details on how the functions are carried out.

MOTOR Strategy Reports

```
R&R file:       CLRMESS.RPS
REPORT block:   TESTSET (File 3)
Controlled by:  CLRMESS.DOUT   Set high within the reports DELETE and SAVE.
Purpose:        This clears the message window on the TESTSET screen.
Related task:   Test setup.

@MSG(UMSG1.DESC," ")
@EXIT(CURRENT)


R&R file:       DELETE.RPS
REPORT block:   TESTSET (File 2)
Controlled by:  DELPUL.DOUT   Set high by the operator when deleting a test
setup.
Purpose:        This deletes a test setup from USERSET.TDF, the test setup
                file.
Related task:   Test setup.
```

```
;_____________________________________________________________________
; Code:          "DELETE.RPS"
;
; Purpose:       This code uses deletes the test indicated in the block,
;                UTESTNO from the test setup file (if it is there).
;
; Related task: Test Setup
;
; Author:        Jeff Martin     6/7/94
; Modifications:
;
; This code is executed by:
;                DELPUL.DOUT is set high by the operator selecting DELETE TEST
;                       in the TESTSET display.
;
; Reference:     TESTSET???.SOURCE_FILE1
;_____________________________________________________________________
@FLOAT(testno,i)  ;
@DIGITAL(flg)
@msg(umsg1.desc,"Checking...")
@if(access("userset.tdf"==1))
   @open(f1,"userset.tdf",ASCII,INPUT,80)
   @open(f2,"newset.tdf",ASCII,OUTPUT,80)
   @set(i,0)
   @set(flg,0)
   @set(testno,f1[i,0]:4)
   @loop((testno <> 999) and (not(eof(f1))))
      @if(flg==0)
         @if(((testno-0.05) < UTESTNO.out) and ((testno+0.05) > UTESTNO.out))
            @msg(UMSG1.desc,"Deleting test.")
            @if(MAXINDX.out > 0)
               @set(MAXINDX.out,MAXINDX.out-1)
            @endif
            @if(NEXTINDX.out > 0)
               @set(NEXTINDX.out,NEXTINDX.out-1)
            @endif
            @wait(10)
            @set(flg,1)
         @else
            @if(UTESTNO.out < testno-0.05)
               @set(flg,1)
               @msg(UMSG1.desc,"Test not found.")
               @out(f2,f1[i,0]:80)
               @wait(10)
            @else
               @out(f2,f1[i,0]:80)
            @endif
         @endif
      @else
         @out(f2,f1[i,0]:79)
```

```
      @endif
      @set(i,i+1)
      @set(testno,f1[i,0]:4)
   @endloop
   @if(flg==0)
      @msg(UMSG1.desc,"Test not found.")
   @endif
   @out(f2," 999",nl)
   @close(f1)
   @close(f2)
   @wait(20)
   @COPY("newset.tdf","userset.tdf")
   @set(INFILE.out,0)
@else
   @msg(UMSG1.desc,"Test file not found.")
@endif
@set(clrmess.out,1)
@exit(current)
```

R&R file: ILOAD.RPS
REPORT block: TESTSET2 (File 4)
Controlled by: ON.DOUT Executes at startup only.
Purpose: This initializes the test setup blocks.
Related task: Test setup.

```
@FLOAT(testno,i)
@set(OLDTEST.out,UTESTNO.out)
@if(access("userset.tdf"==1))
   @open(f1,"userset.tdf",ASCII,INPUT,80)
   @set(testno,f1[0,0]:4)
   @if(testno<>999)
      @set(i,0)
      @set(UTESTNO.out,f1[i,0]:4)
      @set(UANGLE.out,f1[i,5]:5)
      @set(USPEED.out,f1[i,11]:6)
      @set(UHRS.out,f1[i,19]:2)
      @set(UMINS.out,f1[i,23]:2)
      @set(USECS.out,f1[i,27]:2)
      @set(UACCEL.out,f1[i,30]:5)
      @set(UDECEL.out,f1[i,36]:5)
      @msg(UDESC.desc,f1[i,42]:21)
      @set(URESTIM.out,f1[i,75]:1)
      @set(INFILE.out,1)
      @set(NEXTINDX.out,i)
      @loop((testno<>999) and (not(eof(f1))))
         @set(i,i+1)
         @set(testno,f1[i,0]:4)
      @endloop
```

```
      @set(MAXINDX.out,i)
      @load("setold")
      @close(f1)
   @else
      @set(MAXINDX.out,0)
      @set(NEXTINDX.out,0)
      @set(INFILE.out,0)
      @close(f1)
   @endif
@else
   @open(f1,"userset.tdf",ASCII,OUTPUT,80)
   @out(f1," 999",nl)
   @close(f1)
   @set(MAXINDX.out,0)
   @set(NEXTINDX.out,0)
   @set(INFILE.out,0)
@endif
@exit(current)
```

R&R file: NEXT.RPS
***REPORT* block:** TESTSET2 (File 2)
Controlled by: NEXTPUL.DOUT Set high by the operator when using the down arrow button to access the next test in the test setup file.
Purpose: This accesses the next test in the test setup file.
Related task: Test setup.

```
;_____________________________________________________________________
; Code:          "NEXT.RPS"
;
; Purpose:       This code uses the file index block, NEXTINDX, to load
;                test setup values for the next test in the file
;                USERSET.TDF.
;
; Related task: Test Setup
;
; Author:        Jeff Martin     6/7/94
; Modifications:
;
; This code is executed by:
;                NEXTPUL.DOUT is set high by the operator selecting NEXT
;                in the TESTSET display or by selecting the up-arrow
;              . in the PUMPOPS or PBTESTS displays.
;
; Reference:     TESTSET???.SOURCE_FILE1
;_____________________________________________________________________
@FLOAT(testno,i)      ;setup local variables. testno for the test number, i for the file index
@if(MAXINDX.out == 0)   ;are there any records in the file?
```

```
   @exit(current)       ;if not, exit
@endif
@if(access("userset.tdf"==1))        ;does the file exist?
   @open(f1,"userset.tdf",ASCII,INPUT,80)  ;yes. open it.
   @set(i,NEXTINDX.out+1)                  ;increment the file index pointer
   @set(testno,f1[i,0]:4)                  ;read the test number
   @if(testno <> 999)                      ;are we at the end of file?
      @set(UTESTNO.out,testno)             ;no. set the test parameter blocks
      @set(UANGLE.out,f1[i,5]:5)
      @set(USPEED.out,f1[i,11]:6)
      @set(UHRS.out,f1[i,19]:2)
      @set(UMINS.out,f1[i,23]:2)
      @set(USECS.out,f1[i,27]:2)
      @set(UACCEL.out,f1[i,30]:5)
      @set(UDECEL.out,f1[i,36]:5)
      @msg(UDESC.desc,f1[i,42]:21)
      @set(URESTIM.out,f1[i,75]:1)
      @set(INFILE.out,1)                   ;set in file flag
      @set(NEXTINDX.out,i)                 ;set new value of index
      @load("setold")                      ;queue setold.rps
      @close(f1)                           ;close and exit.
      @exit(current)
   @else
      @set(i,0)              ;we were at the end so wrap to the beginning of the file
      @set(testno,f1[i,0]:4) ;get test number
      @if(testno <> 999)     ;still at end of file?
         @set(UTESTNO.out,testno)   ;no. set the test parameter blocks.
         @set(UANGLE.out,f1[i,5]:5)
         @set(USPEED.out,f1[i,11]:6)
         @set(UHRS.out,f1[i,19]:2)
         @set(UMINS.out,f1[i,23]:2)
         @set(USECS.out,f1[i,27]:2)
         @set(UACCEL.out,f1[i,30]:5)
         @set(UDECEL.out,f1[i,36]:5)
         @msg(UDESC.desc,f1[i,42]:21)
         @set(URESTIM.out,f1[i,75]:1)
         @set(INFILE.out,1)            ;set the in-file flag
         @set(NEXTINDX.out,i)          ;set the index pointer
         @load("setold")               ;queue setold.rps
         @close(f1)                    ;close and exit.
         @exit(current)
      @endif
   @endif
@endif
@exit(current)
```

R&R file: PBSETVAL.RPS
***REPORT* block:** PBREPORT (File 1)
Controlled by: SETTEST.DOUT Set high by the operator selecting to set values.
Purpose: This code will download the desired test values to the critical parameters in the MOTOR.
Related task: Set test values

```
;_____________________________________________________________________
; Code:          "PBSETVAL.RPS"
;
; Purpose:       This code is used to update setpoints, alarm limits, and abort
;                limits based on test selection.
;
; Related task: Test Operation
;
; Author:        Darrel Holt     8/3/93
; Modifications:Ken Eldridge     1/23/94
;                       To provide additional comments to code.
;                S.G. McNeece    3/08/94
;                       CR304 to remove duplicate strategy blocks
;
;                S.G. McNeece    4/21/94
;                       CR334 to remove demo mode
;                Ken Eldridge    5/25/94
;                       Integrate status indication to strategy
;                Ken Eldridge    6/6/94
;                       Incorporate setting of alarm setpoints for feedback check.
;                Jeff Martin     6/7/94
;                       Values for each test are now maintained in a file (userset.tdf)
;                       and loaded into algorithm blocks.  The actual test settings
;                       are loaded by this report from these blocks.
;                Jeff Martin     7/28/94
;                       Add position deadband sets.
; This code is executed by:
;                PBSETAND.DOUT is set high by the operator selecting to set values
;                       in the phase B test display.
;
; Reference:     PBREPORT.SOURCE_FILE1
;_____________________________________________________________________


                        ;REMOVE THIS WHEN PUMP BUMP SCREEN GOES AWAY
@IF(BUMPENAB.OUT == 1)                  ; Is this a pump bump or a phase b test?
          @SET(DO3DEV.SETP,PMOTOR.ACCE) ; Pump acceleration check for PBTESTS indication.
          @SET(DO4DEV.SETP,PMOTOR.DECE) ; Pump deceleration check for PBTESTS indication.
          @SET(RESETPB.OUT,1)           ; Reset overall test time.
          @SET(DO1DEV.SETP,POSCNTRL.SETP)   ; Position check for PUMP BUMP indication.
          @SET(PMPSPDAL.OUT,PMOTOR.SPDS + 10)   ; Set pump motor alarm limit.
          @SET(PMPSPDAB.OUT,PMOTOR.SPDS + 20)   ; Set pump motor abort limit.
          @SET(DO2DEV.SETP,PMOTOR.SPDS) ; Desired speed check for PBTESTS indication.
          @SET(DO5DEV.SETP,PMOTOR.SPDS) ; Pump maximum speed check for PBTESTS indication.
          @SET(DO6DEV.SETP,PMOTOR.SPDS+10) ; Pump speed alarm check for PBTESTS indication.
          @SET(DO7DEV.SETP,PMOTOR.SPDS+20) ; Pump speed abort check for PBTESTS indication.
@ELSE
          @SET(DMOTOR.SPDS,100)         ; Set desired speed of position motor.
          @SET(DMOTOR.ACCE,500)         ; Set acceleration of position motor.
```

```
        @SET(DMOTOR.DECE,500)          ; Set deceleration of position motor.
        @SET(PMOTOR.ACCE,UACCEL.OUT) ; Set acceleration of pump motor.
        @SET(PMOTOR.DECE,UDECEL.OUT) ; Set deceleration of pump motor.

        @SET(DO3DEV.SETP,UACCEL.OUT) : Pump acceleration check for PBTESTS indication.
        @SET(DO4DEV.SETP,UDECEL.OUT) : Pump deceleration check for PBTESTS indication.
        @MSG(TDESC.DESC,UDESC.DESC)  : Test description
        @IF(URESTIM.OUT == 1)          ; Check if overall test time should be reset.
                    @SET(RESETPB.OUT,1)       ; Reset overall test time.
        @ENDIF

        @SET(POSCNTRL.HGAP,2.2)           ; Set pump position deadband (high)
        @SET(POSCNTRL.LGAP,2.2)           ; Set pump position deadband (low)
        @SET(POSCNTRL.SETP,UANGLE.OUT)    ; Set desired pump position.
        @SET(PBMIN.OUT,UMINS.OUT)       ; Set desired minutes for pump operation.
        @SET(PBSEC.OUT,USECS.OUT)       ; Set desired seconds for pump operation.
        @SET(PBHR.OUT,UHRS.OUT)         ; Set desired hours for test operation.

        @SET(DO1DEV.SETP,UANGLE.OUT)    ; Position check for PBTEST indication.
        @SET(PMOTOR.MAXS,USPEED.OUT)    ; Set pump motor maximum speed.
        @WAIT(10)                       ; 1 second wait needed for maximum speed to
                                  ; be set first before desired speed.
        @SET(PMOTOR.SPDS,USPEED.OUT)        ; Set pump motor desired speed.
        @SET(PMPSPDAL.OUT,USPEED.OUT+10)    ; Set pump motor alarm limit.
        @SET(PMPSPDAB.OUT,USPEED.OUT+20)    ; Set pump motor abort limit.

        @SET(DO2DEV.SETP,USPEED.OUT) ; Desired speed check for PBTESTS indication.
        @SET(DO5DEV.SETP,USPEED.OUT) ; Pump maximum speed check for PBTESTS indication.
        @SET(DO6DEV.SETP,USPEED.OUT+10) ; Pump speed alarm check for PBTESTS indication.
        @SET(DO7DEV.SETP,USPEED.OUT+20) ; Pump speed abort check for PBTESTS indication.

        @if((INFILE.OUT == 0) or (CHANGED.OUT == 1)) ;check for test in file or changed.
        @SET(MANUAL.OUT,1)       ;test changed or not in file. indicate manual test.
        @SET(PBTESTNO.OUT,0)     ;and clear test number.
        @ELSE
                    @SET(MANUAL.OUT,0)             ;Indicate valid test in file.
                    @SET(PBTESTNO.OUT,UTESTNO.OUT) ;Set current test number
        @ENDIF

@ENDIF

@SET(SETVALDN.OUT,1)    ; Set indication that values are set successfully.
```

R&R file: PREV.RPS
REPORT block: TESTSET2 (File 3)
Controlled by: PREVPUL.DOUT Set high by the operator when using the up arrow button to access the previous test in the test setup file.
Purpose: This accesses the previous test in the test setup file.
Related task: Test setup.

```
;_____________________________________________________________________
; Code:          "PREV.RPS"
;
; Purpose:       This code uses the file index block, NEXTINDX, to load
;                test setup values for the previous test in the file
;                USERSET.TDF.
;
; Related task: Test Setup
;
; Author:        Jeff Martin     6/7/94
; Modifications:
;
; This code is executed by:
;                PREVPUL.DOUT is set high by the operator selecting PREVIOUS
;                       in the TESTSET display or by selecting the down-arrow
;                       in the PUMPOPS or PBTESTS displays.
;
; Reference:     TESTSET???.SOURCE_FILE1
;_____________________________________________________________________

@FLOAT(testno,i)  ;setup local variable, testnumber and i - record index.
@if(MAXINDX.out == 0)   ;if maxindx is 0, there is either no setup file
                        ;or a setup file with no tests in it.  So exit.
   @exit(current)
@endif
@if(access("userset.tdf"==1))
   @open(f1,"userset.tdf",ASCII,INPUT,80)  ;if test setup file exists, open it.
   @set(i,NEXTINDX.out)  ;get the index for the current test.
   @if(i==0)             ;are we at the beginning?
      @set(i,MAXINDX.out) ;if so, set the new index to the end of the file
      if(i<>0)
         @set(i,i-1)
      @endif
   @else
      @set(i,NEXTINDX.out-1)  ;we're not at the beginning, so decrement index pointer
   @endif
   @set(testno,f1[i,0]:4)  ;read the next number for the 'previous' record.
   @if(testno <> 999)  ;is it the end of file pointer?
      @set(UTESTNO.out,testno)    ;if not, set the test value blocks in the strategy
      @set(UANGLE.out,f1[i,5]:5)
```

```
      @set(USPEED.out,fl[i,11]:6)
      @set(UHRS.out,fl[i,19]:2)
      @set(UMINS.out,fl[i,23]:2)
      @set(USECS.out,fl[i,27]:2)
      @set(UACCEL.out,fl[i,30]:5)
      @set(UDECEL.out,fl[i,36]:5)
      @msg(UDESC.desc,fl[i,42]:21)
      @set(URESTIM.out,fl[i,75]:1)
      @set(INFILE.out,1)    ;set the in-file indication
      @set(NEXTINDX.out,i) ;set the new current record index
      @load("setold")       ;queue setold.rps
      @close(fl)            ;close and exit
      @exit(current)
   @endif
@endif
@exit(current)
```

R&R file: SAVE.RPS
REPORT block: TESTSET (File 1)
Controlled by: SAVEPUL.DOUT Set high by the operator to save a test setup.
Purpose: This saves a test setup in the test setup file.
Related task: Test setup.

```
;_____________________________________________________________________
; Code:          "SAVE.RPS"
;
; Purpose:       This code inserts the current user defined test setup
;                parameters into the test setup file, USERSET.TDF.
;
; Related task: Test Setup
;
; Author:        Jeff Martin     6/7/94
; Modifications:
;
; This code is executed by:
;                SAVEPUL.DOUT is set high by the operator selecting SAVE TEST
;                       in the TESTSET display.
;
; Reference:     TESTSET???.SOURCE_FILE1
;_____________________________________________________________________

@FLOAT(testno,i)  ;setup local variables, testno is the test number read from
                  ; the file.  i is the record index into the file
@DIGITAL(flg)     ;setup local variable. flg is set when the new test setup
                  ;has been inserted.
@msg(umsgl.desc,"Saving...")  ;write message to display
@if(access("userset.tdf"==1)) ;check to see if USERSET.TDF exists
```

```
@open(f1,"userset.tdf",ASCII,INPUT,80) ;USERSET exists so open it.
@open(f2,"newset.tdf",ASCII,OUTPUT,80) ;open NEWSET.TDF
@set(i,0)       ;initialize flg and record pointer
@set(flg,0)
@set(testno,f1[i,0]:4)   ;read the first test number from file
@loop((not(eof(f1))) and (testno <> 999))  ;loop until end of file is reached
   @if(flg==0) ;have we inserted the new test setup yet?
               ;if not, check to see if this is the place (the file is
               ;maintained so that the tests are in ascending numeric order).
      @if(((testno-0.05) < UTESTNO.out) and ((testno+0.05) > UTESTNO.out))
               ;new test number is equal to a test number already in the file
               ;so replace the old setup with the new one.
         @msg(UMSG1.desc,"Replacing previous test setup.")
         @wait(10)
         @out(f2,UTESTNO.out:2:1," ",UANGLE.out:3:1," ",USPEED.out:4:1)
         @out(f2,"  ",UHRS.out:2:0,"  ",UMINS.out:2:0,"  ",USECS.out:2:0)
         @out(f2," ",UACCEL.out:3:1," ",UDECEL.out:3:1)
         @out(f2," ",UDESC.desc,"  ",tab(80*i+75),URESTIM.out)
         @out(f2,"  ",nl)
         @set(NEXTINDX.out,i)  ;set index variable to point to the new test setup
         @set(flg,1) ;set flag indicating we have inserted new test setup.
      @else
         @if(UTESTNO.out < testno-0.05) ;are we in the right spot?
            ;yes. So insert new test setup.  We didn't find a test with the same
            ;number already in the file.
            @out(f2,UTESTNO.out:2:1," ",UANGLE.out:3:1," ",USPEED.out:4:1)
            @out(f2,"  ",UHRS.out:2:0,"  ",UMINS.out:2:0,"  ",USECS.out:2:0)
            @out(f2," ",UACCEL.out:3:1," ",UDECEL.out:3:1)
            @out(f2," ",UDESC.desc,"  ",tab(80*i+75),URESTIM.out)
            @out(f2,"  ",nl)
            @set(flg,1)          ;Set flag indicating we have inserted the new test.
            @set(NEXTINDX.out,i) ;set index variable to point to the new test.
            @set(MAXINDX.out,MAXINDX.out+1) ;increment the maximum test pointer since
                                            ; we have added a new one.
            @out(f2,f1[i,0]:80)  ; now copy the current test setup in the old file to
                                 ; the new one.
         @else
            @out(f2,f1[i,0]:80)  ;not the right place to insert the new test setup.
                                 ;copy test setup from the old file to the new one.
         @endif
      @endif
   @else  ;we have already inserted the new test set up so...
      @out(f2,f1[i,0]:80)  ;copy the remainder of the records from
                           ;the old file to the new file.
   @endif
   @set(i,i+1)  ;increment index pointer
   @set(testno,f1[i,0]:4) ;read next test number
@endloop
@if(flg==0)
   ;we've reached the end of file and still not inserted the new
```

```
      ;test.  That means we need to insert it at the end.
      @out(f2,UTESTNO.out:2:1," ",UANGLE.out:3:1," ",USPEED.out:4:1)
      @out(f2,"  ",UHRS.out:2:0,"  ",UMINS.out:2:0,"  ",USECS.out:2:0)
      @out(f2," ",UACCEL.out:3:1," ",UDECEL.out:3:1)
      @out(f2," ",UDESC.desc,"  ",tab(80*i+75),URESTIM.out)
      @out(f2,"  ",nl)
      @set(NEXTINDX.out,i)  ;set index pointer variable to new test
      @set(MAXINDX.out,MAXINDX.out+1) ;increment maximum test index pointer
   @endif
   @out(f2," 999",nl) ;append the end of file marker to the new test file
   @close(f1)         ;close the old setup file and the new setup file
   @close(f2)
   @wait(20)          ;wait 2 seconds then copy the new file to the old file.
   @COPY("newset.tdf","userset.tdf")
   @set(INFILE.out,1)  ;set the infile flag indicating that the current test
                       ;is in the setup file.
   @load("setold")     ;queue setold.rps
@else
   ;the file didn't exist, so create it and add the current setup.
   @open(f2,"userset.tdf",ASCII,OUTPUT,80)
   @out(f2,UTESTNO.out:2:1," ",UANGLE.out:3:1," ",USPEED.out:4:1)
   @out(f2,"  ",UHRS.out:2:0,"  ",UMINS.out:2:0,"  ",USECS.out:2:0)
   @out(f2," ",UACCEL.out:3:1," ",UDECEL.out:3:1)
   @out(f2," ",UDESC.desc,"  ",tab(75),URESTIM.out)
   @out(f2,"  ",nl)
   @out(f2," 999",nl)
   @set(MAXINDX.out,1)   ;set the maximum index to 1
   @set(NEXTINDX.out,0)  ;set the current index to 0 - the first record.
   @close(f2)
   @set(INFILE.out,1)  ;the current test is in the file.
   @load("setold")     ;queue setold.rps
@endif
@msg(umsg1.desc,"Save Complete.")  ;write message to display.
@set(clrmess.out,1)   ;run clrmess.rps after a delay.
@exit(current)   ;exit
```

R&R file: SETOLD.RPS
REPORT block: None
Controlled by: Called from the reports SAVE, ILOAD, ULOAD, NEXT, PREV.
Purpose: This code updates the old test parameter blocks whenever a new test becomes the active test. This allows the check for test parameter changes to be made.
Related task: Test setup.

```
@SET(OANGLE.OUT,UANGLE.OUT)
@SET(OSPEED.OUT,USPEED.OUT)
@SET(OHRS.OUT,UHRS.OUT)
```

```
@SET(OMINS.OUT,UMINS.OUT)
@SET(OSECS.OUT,USECS.OUT)
@SET(OACCEL.OUT,UACCEL.OUT)
@SET(ODECEL.OUT,UDECEL.OUT)
@SET(ORESTIM.OUT,URESTIM.OUT)
```

R&R file: ULOAD.RPS
REPORT block: TESTSET2 (File 1)
Controlled by: TSTCHG.DOUT Set whenever a new test number is entered by the operator on the TESTSET, PBTESTS or PUMPRUN screens.
Purpose: This loads the selected test from the test setup file..
Related task: Test setup.

```
@FLOAT(testno,i)
@set(OLDTEST.out,UTESTNO.out)
@if(access("userset.tdf"==1))
   @open(f1,"userset.tdf",ASCII,INPUT,80)
   @set(i,0)
   @set(testno,f1[i,0]:4)
   @loop((testno<>999) and (not(eof(f1))))
      @if(((testno-0.05) < UTESTNO.out) and ((testno+0.05) > UTESTNO.out))
         @set(UANGLE.out,f1[i,5]:5)
         @set(USPEED.out,f1[i,11]:6)
         @set(UHRS.out,f1[i,19]:2)
         @set(UMINS.out,f1[i,23]:2)
         @set(USECS.out,f1[i,27]:2)
         @set(UACCEL.out,f1[i,30]:5)
         @set(UDECEL.out,f1[i,36]:5)
         @msg(UDESC.desc,f1[i,42]:21)
         @set(URESTIM.out,f1[i,75]:1)
         @set(INFILE.out,1)
         @set(NEXTINDX.out,i)
         @close(f1)
         @load("setold")
         @exit(current)
      @else
         @if(UTESTNO.out < testno-0.05)
            @set(INFILE.out,0)
            @close(f1)
            @exit(current)
         @endif
      @endif
      @set(i,i+1)
      @set(testno,f1[i,0]:4)
   @endloop
   @set(INFILE.out,0)
   @set(NEXTINDX.out,0)
```

```
   @close(f1)
   exit(current)
@else
   @open(f1,"userset.tdf",ASCII,OUTPUT,80)
   @out(f1," 999",nl)
   @close(f1)
   @set(MAXINDX.out,0)
   @set(NEXTINDX.out,0)
   @set(INFILE.out,0)
@endif
@exit(current)
```

TEST Strategy Reports

R&R file: HISTCOPY.RPS
REPORT **block:** FILENAME (File 4)
Controlled by: D1MIN.DOUT Set high one minute after the beginning of a history file changeover.
Purpose: This copies history files from the TEST directory to the HIST and HIST2 directories.
Related task: History file transfer.

```
;_____________________________________________________
;Filename:  HISTCOPY.RPS
;
;Purpose:  Copies the history files to HIST and HIST2
;          directories and then deletes them from the test
;          directory.
;          Copies "NEW72" file to "OLD72" file and then
;          deletes "NEW72" so can start another set
;_____________________________________________________

@STRING(SNAM[4])
@IF(ACCESS("C:\TEST\OLDFN.DT")==1)
    @OPEN(F1,"OLDFN.DT",FREE,INPUT,0)
    @SET(SNAM,F1[])
    @COPY("C:\TEST\" + SNAM + "PUMP.PRN","C:\HIST\" + SNAM + "PUMP.PRN")
    @COPY("C:\TEST\" + SNAM + "RGA5.PRN","C:\HIST\" + SNAM + "RGA5.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT2.PRN","C:\HIST\" + SNAM + "MIT2.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT1.PRN","C:\HIST\" + SNAM + "MIT1.PRN")
    @COPY("C:\TEST\" + SNAM + "TBS1.PRN","C:\HIST\" + SNAM + "TBS1.PRN")
    @COPY("C:\TEST\" + SNAM + "VOL.PRN","C:\HIST\" + SNAM + "VOL.PRN")
    @COPY("C:\TEST\" + SNAM + "TBS2.PRN","C:\HIST\" + SNAM + "TBS2.PRN")
    @COPY("C:\TEST\" + SNAM + "GAS.PRN","C:\HIST\" + SNAM + "GAS.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT3.PRN","C:\HIST\" + SNAM + "MIT3.PRN")
```

```
    @COPY("C:\TEST\" + SNAM + "STRN.PRN","C:\HIST\" + SNAM + "STRN.PRN")
    @COPY("C:\TEST\" + SNAM + "PUMP.PRN","C:\HIST2\" + SNAM + "PUMP.PRN")
    @COPY("C:\TEST\" + SNAM + "RGA5.PRN","C:\HIST2\" + SNAM + "RGA5.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT2.PRN","C:\HIST2\" + SNAM + "MIT2.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT1.PRN","C:\HIST2\" + SNAM + "MIT1.PRN")
    @COPY("C:\TEST\" + SNAM + "TBS1.PRN","C:\HIST2\" + SNAM + "TBS1.PRN")
    @COPY("C:\TEST\" + SNAM + "VOL.PRN","C:\HIST2\" + SNAM + "VOL.PRN")
    @COPY("C:\TEST\" + SNAM + "TBS2.PRN","C:\HIST2\" + SNAM + "TBS2.PRN")
    @COPY("C:\TEST\" + SNAM + "GAS.PRN","C:\HIST2\" + SNAM + "GAS.PRN")
    @COPY("C:\TEST\" + SNAM + "MIT3.PRN","C:\HIST2\" + SNAM + "MIT3.PRN")
    @COPY("C:\TEST\" + SNAM + "STRN.PRN","C:\HIST2\" + SNAM + "STRN.PRN")
    @DELETE("C:\TEST\" + SNAM + "PUMP.PRN")
    @DELETE("C:\TEST\" + SNAM + "RGA5.PRN")
    @DELETE("C:\TEST\" + SNAM + "MIT2.PRN")
    @DELETE("C:\TEST\" + SNAM + "MIT1.PRN")
    @DELETE("C:\TEST\" + SNAM + "TBS1.PRN")
    @DELETE("C:\TEST\" + SNAM + "VOL.PRN")
    @DELETE("C:\TEST\" + SNAM + "TBS2.PRN")
    @DELETE("C:\TEST\" + SNAM + "GAS.PRN")
    @DELETE("C:\TEST\" + SNAM + "MIT3.PRN")
    @DELETE("C:\TEST\" + SNAM + "STRN.PRN")
    @CLOSE(F1)
@ENDIF

@IF(ACCESS("C:\TEST\FN.DT")==1)
    @COPY("C:\TEST\FN.DT","C:\TEST\OLDFN.DT")
@ENDIF

@IF(CHNG72OK.OUT == 1)
    @IF(ACCESS("NEW72.PRN")==1)
        @COPY("NEW72.PRN","OLD72.PRN")
        @DELETE("NEW72.PRN")
    @ENDIF
@ENDIF
```

R&R file: HISTFNAM.RPS
REPORT block: FILENAME (File 1,2 and 3)
Controlled by: #435.DOUT, #474.DOUT, PULON.DOUT Set high at two-hour history file changeover or at startup.
Purpose: This renames the history files based upon the time and date.
Related task: History file transfer.

```
@FLOAT(MNUM,DNUM,HNUM)
@STRING(YSTR[1],MSTR[1],DSTR[1],HSTR[1])
@OPEN(F1,"STORE.DT",ASCII,OUTPUT,8)
@OUT(F1,DATE(YY),DATE(MM),DATE(DD),DATE(HH))
@CLOSE(F1)
@OPEN(F1,"STORE.DT",ASCII,INPUT,8)
```

```
@SET(YSTR,F1[0,1]:1)
@SET(MNUM,F1[0,2]:2)
@SET(DNUM,F1[0,4]:2)
@SET(HNUM,F1[0,6]:2)
@CLOSE(F1)
@IF(MNUM > 9)
        @SET(MSTR,CHAR(MNUM + 55))
@ELSE
        @SET(MSTR,CHAR(MNUM + 48))
@ENDIF

@IF(DNUM > 9)
        @SET(DSTR,CHAR(DNUM + 55))
@ELSE
        @SET(DSTR,CHAR(DNUM + 48))
@ENDIF

@IF(HNUM > 9)
        @SET(HSTR,CHAR(HNUM + 55))
@ELSE
        @SET(HSTR,CHAR(HNUM + 48))
@ENDIF

@SET(PUMP.FNAM,YSTR + MSTR + DSTR + HSTR + "PUMP")
@SET(RGA5.FNAM,YSTR + MSTR + DSTR + HSTR + "RGA5")
@SET(MIT2.FNAM,YSTR + MSTR + DSTR + HSTR + "MIT2")
@SET(MIT1.FNAM,YSTR + MSTR + DSTR + HSTR + "MIT1")
@SET(TBS1.FNAM,YSTR + MSTR + DSTR + HSTR + "TBS1")
@SET(VOL.FNAM,YSTR + MSTR + DSTR + HSTR + "VOL")
@SET(TBS2.FNAM,YSTR + MSTR + DSTR + HSTR + "TBS2")
@SET(GAS.FNAM,YSTR + MSTR + DSTR + HSTR + "GAS")
@SET(MIT3.FNAM,YSTR + MSTR + DSTR + HSTR + "MIT3")
@SET(STRN.FNAM,YSTR + MSTR + DSTR + HSTR + "STRN")
@SET(GO.OUT,1)
@OPEN(F1,"FN.DT",FREE,OUTPUT,0)
@OUT(F1,YSTR + MSTR + DSTR + HSTR)
@CLOSE(F1)
```

R&R file: SETLIMS.RPS
REPORT block: DOWNLIMS (File 1)
Controlled by: DELAYON.DOUT Set 45 seconds after startup.
Purpose: Downloads abort limits to the PLC.
Related task: Abort limits.

```
;_____________________________________________________________________
; Code:            "SETLIMS.RPS"
;
; Purpose:         This code is executed to set the abort limits in the PLC.
;
; Related task:    Abort.
;
; Author:          Jeff Martin
; Modifications:   Sam Smith/Ross Truitt 04-03-96
:                    Changes in support of TEST version 3.04
; This code is executed by:
;                  DELAYON.DOUT
;
; Reference:       DOWNLIMS.SOURCE_FILE1
;_____________________________________________________________________
;
       ;SETS LIMITS IN AIN BLOCKS FOR OUTPUT THROUGH THE PAOT
       ;
@SET(HILIM.EVAL,4095)
       ;
@SET(HH2LIM.EVAL,0.75)
@SET(HTEMPLIM.EVAL,135)
;@SET(HPDPLIM.EVAL,21)
;@SET(HPMCLIM.EVAL,140)
@SET(HPMOTLIM.EVAL,225)
@SET(LVFLLIM.EVAL,400)
@SET(HTDPLIM.EVAL,-1)
@SET(HPCSLIM.EVAL,194)
;@SET(HMITSLIM.EVAL,357) Removed per SCR#384
@SET(H17CSLIM.EVAL,357)
;@SET(HPSPDLIM.EVAL,400)
@SET(HH2LIM2.EVAL,0.75)
@SET(HTMPLIM3.EVAL,135)
@SET(LVFLLIM2.EVAL,400)
@SET(HTDPLIM2.EVAL,-1)
@SET(H1BCSLIM.EVAL,546)
@SET(LPCSLIM.EVAL,-194)
;@SET(LMITSLIM.EVAL,-357) Removed per SCR#384
@SET(L17CSLIM.EVAL,-357)
@SET(L1BCSLIM.EVAL,-546)
@SET(LPCGPLIM.EVAL,7)
@SET(HVFLLIM1.EVAL,700)
@SET(HVFLLIM2.EVAL,700)
@SET(H1BSAL.EVAL,317)
@SET(L1BSAL.EVAL,-317)
@SET(H12ASAL.EVAL,145.5)
@SET(L12ASAL.EVAL,-145.5)
@SET(H17BCSAL.EVAL,267)
@SET(L17BCSAL.EVAL,-267)
```

Appendix D: Key Macros and State Fields

GENESIS software has several predefined key macros. The TEST and MOTOR strategies utilize a key macro definition file to redefine one of the key macros, the auto/manual function. RSS stations outside the DACS trailer do not have the redefined auto/manual function enabled. The text file EXTRA.KMS was created to disable <F9> and add <Alt-F9> as the auto/manual macro. The text file is compiled with KEYMAC EXTRA to create EXTRA.KML. This compiled file must be contained in the MOTOR and TEST directories where the respective strategies are contained. The following is the listing of EXTRA.KMS:

```
DISMOV1=ON(DISMOVE.OUT).
DISMOV0=OFF(DISMOVE.OUT).
CHKCLR1=ON(CHKCOLOR.OUT).
SETTES1=ON(SETTEST.OUT).
RESPTIM=DOWNLD_PT(0,PBHR.OUT) DOWNLD_PT(0,PBMIN.OUT)
DOWNLD_PT(0,PBSEC.OUT)
DOWNLD_PT(0,PMPSPDAL.OUT) DOWNLD_PT(0,PMPSPDAB.OUT)
DOWNLD_PT(0,DO5DEV.SETP) DOWNLD_PT(0,DO6DEV.SETP).

[F9]=DISABLE.
[Alt-F9]=A/M().
```

State fields are useful to tie words to analog or discrete values. The text file STATES.SFS was modified to create a state field for the displays linked to the MOTOR strategy:

- <Enable-Disable Move> is used to indicate and control the enabling of the position motor.

If a state field is revised (by editing STATES.SFS and compiling it with the command STATECMP STATES), the display connection in the display builder must be redone in order for the changes to come into effect.

Appendix E: MOTOR Strategy Display List

The following list of displays are currently implemented on STATION8 and are dynamically linked to the MOTOR strategy. (See Section 4.1.2 for a list and discussion of the TEST strategy displays).

Table F-1. Station Displays

Display	Functions
DMPROB	Help screen - positioning problem display.
MAINMENU	Menu display to select other displays.
PBENPROB	Help screen - Test enable problem display.
PSPROB	Help screen - Test start problem display.
PUMPRUN	Combined directional/run pump operation display.
SETPROB	Help screen - Test values set problem display.
TESTSET	User test setup display.
EXIT	Runtime exit display

Appendix F: Hardware Configurations

The following is a list of all cards and their settings on all computer stations in the DACS trailer. The addresses below ("Addr" column) are in hexadecimal. The printer ports are multiplexed; the port on the multiplexer is noted in the "Destination" column; the ARCNET star configuration ports are also noted here.

There are other standard IRQs that are being used by the system in addition to those noted in the tables: IRQ 0 (system timer), IRQ 1 (keyboard), IRQ 2 (cascade from higher IRQs), IRQ 6 (floppy disk), IRQ 8 (clock), IRQ 13 (80486DX), and IRQ 14 (hard disk).

Computer #1 (Texas Microsystems, in the DACS trailer)

running ModSoft; directly connected to the PLC

Slot	Card	Port(s)	IRQ	Addr	Destination
1	Kensington	9-pin ♂	10	338	Bus mouse
2	(empty)				
3	SA-85	9-pin ♀		C000	Modbus Plus, 984E
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	4 7	3F8 378	not used LaserJet printer (port 6)
9	I/O jumper	9-pin ♂ 25-pin ♀	3	2F8	PLC A/B switch (rack 13)[1] not used
10	(empty)				
11	(empty)				
12	(empty)				
13	SCSI controller	37-pin ♀	11	330	not used
14	(empty)				

[1] Connected through T-connector going to Computer #4 and to rack 8, where the cable is curled up near Computer #9. The card in not being used.

Computer #2 (Texas Microsystem, in the DACS trailer)

used for gas monitoring

Slot	Card	Port(s)	IRQ	Addr	Destination
1	Kensington bus mouse	9-pin ♂	10	338	not used
2	(empty)				
3	3Com Ethernet	15-pin ♀ BNC ♀			TLC fanout buffered repeater not used
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀	4 7	3F8 378	mouse LaserJet printer (port 2)
8	I/O jumper	9-pin	3	2F8	not used
9	(empty)				
10	3Com Ethernet	15-pin ♀ BNC ♀	5	300	not used Rack 10 GMS computer
11	(empty)				
12	(empty)				
13	SCSI controller	37-pin ♀	11	330	not used
14	(empty)				

Computer #3 (Texas Microsystem, in the DACS trailer)

Utility station; HLAN drop; communicates with ASCII/BASIC module

Slot	Card	Port(s)	IRQ	Addr	Destination
1	Kensington bus mouse	9-pin ♂	12	338	bus mouse
2	Iomega host adaptor	37-pin ♂	5		Switch box; branch 2 (Bernoulli)
3	3Com Ethernet II	RF connector RF connector	5	300	terminated (jumpered to slot 11) Hanford LAN
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	4 7	3F8 378	not used LaserJet printer (port 4)
9	I/O jumper	9-pin ♂ 25-pin ♀	3	2F8	ASCII module downloading not used
10	(empty)				
11	3Com Ethernet II	15-pin ♀			not used (jumpered to slot 3)
12	(empty)				
13	SCSI controller	37-pin ♀	11	330	not used
14	(empty)				

Computer #4 (Texas Microsystem, in the DACS trailer)

High speed strain gauge data (Nicolet system)

Slot	Card	Port(s)	IRQ	Addr	Destination
1	Kensington bus mouse	9-pin ♂	12	338	bus mouse
2	Iomega host adaptor	37-pin ♂	5		Switch box; branch 1 (Bernoulli)
3	GPIB-AT	IEEE-488	10	280	Nicolet system
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	4 7	3F8 378	not used LaserJet printer (port 3)
9	I/O jumper	9-pin ♂ 25-pin ♀	3	2F8	not used not used
10	(empty)				
11	(empty)				
12	(empty)				
13	SCSI controller	37-pin ♀	11	330	not used
14	(empty)				

Computer #5 (Texas Microsystem, in the DACS trailer)

STATION5, running the Genesis TEST strategy

Slot	Card	Port(s)	IRQ	Addr	Destination
1	(empty)				
2	Kensington bus mouse	9-pin ♂	12	338	bus mouse
3	SA-85	9-pin ♀		D800	Modbus Plus, 984E
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	7	378	not used Genesis C key; Black Box data switch; Proprinter #3 (port 1)
9	I/O jumper	9-pin ♂ 25-pin ♀	4 3	3F8 2F8	Dual ABC box (A) mouse
10	SMC ARCNET	BNC	5	2E0	ARCNET hub (port 5)
11	(empty)				
12	(empty)				
13	(empty)				
14	(empty)				

Computer #6 (Texas Microsystem, in the DACS trailer)

STATION6, Remote Supervisory Station for the TEST strategy

Slot	Card	Port(s)	IRQ	Addr	Destination
1	(empty)				
2	(empty)				
3	(empty)				
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀	4 7	3F8 378	mouse Genesis RSS key; Black Box data switch; Proprinter #3 (port 1)
8	(empty)				
9	SMC ARCNET	BNC	5	2E0	ARCNET hub (port 6)
10	(empty)				
11	(empty)				
12	(empty)				
13	(empty)				
14	(empty)				

Computer #7 (Texas Microsystem, in the DACS trailer)

STATION7, Backup for STATION5 or STATION8

Slot	Card	Port(s)	IRQ	Addr	Destination
1	(empty)				
2	Kensington bus mouse	9-pin ♂	12	338	bus mouse
3	SA-85	9-pin ♀		D800	Modbus Plus, 984E
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	7	378	not used Genesis RT key; Black Box data switch; Proprinter #3 (port 2)
9	I/O jumper	9-pin ♂ 25-pin ♀	4 3	3F8 2F8	mouse Dual ABC box (C)
10	SMC ARCNET	BNC	5	2E0	ARCNET hub (port 7)
11	(empty)				
12	(empty)				
13	(empty)				
14	(empty)				

Computer #8 (Texas Microsystem, in the DACS trailer)

STATION8, running the Genesis MOTOR strategy

Slot	Card	Port(s)	IRQ	Addr	Destination
1	(empty)				
2	Kensington bus mouse	9-pin ♂	12	338	bus mouse
3	SA-85	9-pin ♀		D800	Modbus Plus, 984E
4	Everex I/O	9-pin ♂ 25-pin ♀	 7	 278	not used Proprinter #8
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀			not used not used
8	Magic I/O	9-pin ♂ 25-pin ♀	 7	 378	not used Genesis C key; LaserJet printer multiplx. (port 8)
9	I/O jumper	9-pin ♂ 25-pin ♀	4 3	3F8 2F8	Dual ABC box (B) mouse
10	(empty)				
11	SMC ARCNET	BNC	5	2E0	not used
12	(empty)				
13	(empty)				
14	(empty)				

Computer #9 (Texas Microsystem, in the DACS trailer)

STATION9, stores data files and moves them to a HLAN file server

Slot	Card	Port(s)	IRQ	Addr	Destination
1	Kensington bus mouse	9-pin ♂	10	338	bus mouse
2	(empty)				
3	3Com Ethernet II	RF connector RF connector	 5	 300	terminated (jumpered to slot 11) Hanford LAN
4	(empty)				
5	Texas Micro	15-pin ♀ 9-pin ♀			V2VGA Video; Shinko multiplx. not used
6	(empty)				
7	CPU (80486)	9-pin ♂ 25-pin ♀	4 7	3F8 378	not used Genesis RT key
8	I/O jumper	9-pin ♂ 25-pin ♀			not used not used
9	Iomega host adaptor	37-pin ♂	5		Bernoulli drive
10	SMC ARCNET	BNC	3	2E0	ARCNET hub (port 4)
11	3Com Ethernet II	15-pin ♀			not used (jumpered to slot 3)
12	(empty)				
13	(empty)				
14	(empty)				

Appendix G: Network Configuration Settings

Station name: STATION5
Function: Runtime Master Station
Network settings
- Memory Buffer Address: D980
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 5
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 1
- Message retry count: 3
- Transmit Timeout: 8
- Primary ACK setting: 80
- Net Alarm Squelch After Startup: 0
- Net Alarm Squelch During Startup: 10
- Net Alarm Buffer size: 4096
- Transmit Buffer Full Level: 80
- RX Buffers: 32,32,16,16,32
- Rename Duplicates on file transfer: Yes
- Alarm broadcasts: ENABLE

Station name: STATION6
Function: Local RSS
Network settings
- Memory Buffer Address: D980
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 6
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 1
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- Network point database size: 300
- RSS master identification: MANUAL
- RSS auto reconfiguration: ENABLED
- RSS Master Node Name: STATION5

Station name: STATION7
Function: Local RSS
Network settings
- Memory Buffer Address: D980
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 7
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 1
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- Network point database size: 300
- RSS master identification: MANUAL
- RSS auto reconfiguration: ENABLED
- RSS Master Node Name: STATION5

Station name: STATION8
Function: Runtime Master
Network settings
- Memory Buffer Address: D980
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 8
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 1
- Message retry count: 3
- Transmit Timeout: 8
- Primary ACK setting: 80
- Net Alarm Squelch After Startup: 0
- Net Alarm Squelch During Startup: 10
- Net Alarm Buffer size: 512
- Transmit Buffer Full Level: 10
- RX Buffers: 8,8,4,4,8
- Rename Duplicates on file transfer: Yes
- Alarm broadcasts: DISABLE

Station name: STATION9
Function: DOS file server
Network settings
- Memory Buffer Address: D980
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 9
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 1
- Transmit Timeout: 8
- Primary ACK setting: 80
- Rename Duplicates on file transfer: Yes

Station name: STATION11
Function: Remote RSS
Network settings
- Memory Buffer Address: CC00
- I/O Port Address: 300
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 11
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 4
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- RSS master identification: MANUAL
- RSS auto reconfiguration: DISABLED
- RSS Master Node Name: STATION5

Station name: STATION13
Function: Remote RSS
Network settings
- Memory Buffer Address: CC00
- I/O Port Address: 300
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 13
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 6
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- RSS master identification: MANUAL
- RSS auto reconfiguration: DISABLED
- RSS Master Node Name: STATION5

Station name: STATION15
Function: Remote RSS
Network settings
- Memory Buffer Address: CC00
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 15
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 4
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- RSS master identification: MANUAL
- RSS auto reconfiguration: DISABLED
- RSS Master Node Name: STATION5

Station name: STATION17
Function: Remote RSS
Network settings
- Memory Buffer Address: CC00
- I/O Port Address: 2e0
- IRQ: 5
- Data Check Mode: CRC-16
- Network address: 17
- Node fail time: 20
- Watchdog message rate: 8
- Performance monitor rate: 4
- Network scan time: 4
- Message retry count: 3
- Transmit timeout: 8
- Primary ACK Setting: 80
- RSS master identification: MANUAL
- RSS auto reconfiguration: DISABLED
- RSS Master Node Name: STATION5

Appendix H: Computer Configurations and Software Versions

See Appendix G for a general description of the functions and network setup parameters (ARCNET VO Port, Memory Address, and IRQ) of the stations in the network.

All stations are running DOS, site standard version. The versions of Iconics software that are in use are:

- Genesis (GEN-6C and GEN-RT) 3.72
- Remote Supervisory System (RSS) 3.72
- GEN-NET 3.64
- Report & Recipe option (RRCOMBO) 3.62

The following table gives the parameters for the setup of the ARCNET cards and the Iconics software packages that are on the stations. GEN-NET and RSS serial numbers are provided as well, for those stations that have that software.

	ARCNET Hub Port	ARCNET I/O Port Address	ARCNET Board IRQ	Iconics Software
STATION5	5	2E0	5	GEN-6C, GEN-NET, R&R
STATION6	6	2E0	5	RSS, R&R
STATION7	7	2E0	5	GEN-6RT, GEN-NET, R&R
STATION8	N/A	2E0	5	GEN-6C, GEN-NET, R&R
STATION9	4	2E0	3	GEN-NET
STATION11	2	300	5	RSS, R&R
STATION13	8	300	5	RSS, R&R
STATION15	1	2E0	5	RSS, R&R
STATION17	3	2E0	5	RSS, R&R

The directory organization on the computers is the same for the following directories:

- \(Root directory) - AUTOEXEC.BAT and CONFIG.SYS system boot files
- \DOS - MS-DOS system files
- \MOUSE - Mouse driver
- \NETROOM3 - Expanded Memory Manager

Where directories differ between stations, the following table shows what those directories contain.

Station Name	Directory	Contents
STATION5	\GEN372	Genesis system, GEN-NET, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION6	\GEN372	RSS software, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION7	\GEN372	RSS software, GEN-NET, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION8	\GEN372	Genesis system, GEN-NET, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\MOTOR	MOTOR strategy files, related displays, R&R code, key macro files, state field files, startup batch files

STATION9	\GEN372	GEN-NET
	\NETWARE	Novell networking software (???)
STATION11	\GEN372	RSS software, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION13	\GEN372	RSS software, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION15	\GEN372	RSS software, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files
STATION17	\GEN372	RSS software, AF5000+/Genesis driver, Modicon/Genesis driver, RRCOMBO option, Help system files
	\TEST	TEST strategy files, related displays, R&R code, key macro files, state field files, startup batch files

Appendix I: Directory and File Listings

DACS STATION 5 DIRECTORY LISTINGS

```
 Volume in drive C is MS-DOS_5
 Volume Serial Number is 1C9B-5752
                  175841280 bytes free


 Directory of C:\*.BAT.*.SYS

AUTOEXEC BAT       230 07-25-95   2:29p
ALMCOPY  BAT        87 11-06-96  12:29a
CONFIG   SYS       440 01-26-95  10:02a
RENAME72 BAT       411 03-08-94   1:35p
RUNTIME  BAT      1155 05-25-95   2:12p


 Directory of C:\GEN372

.            <DIR>      08-03-94   9:21a
..           <DIR>      08-03-94   9:21a
GENCFG   NCF        276 05-24-95   9:12a
GENNET   DNL        442 05-24-95   9:12a
GENNET   NDB        171 05-24-95   9:12a
CONFIG   EXE     637000 02-24-94   3:27a
GENCFG   NBC       1015 08-27-92   5:02p
AINDET   GSP       2995 02-24-94   3:27a
GENESIS  GSP       1109 02-24-94   3:27a
ALM      GSP       2649 02-24-94   3:27a
GENESIS  BAT         19 08-03-94  10:09a
ALMSUM   GSP       2244 02-24-94   3:27a
AMBDET   GSP       2449 02-24-94   3:27a
ANALOG   MNU       3377 02-24-94   3:27a
AOUTDET  GSP       2113 02-24-94   3:27a
BOOT     GSP       3450 02-15-95  11:29a
BUTTONS  SMB       1320 02-24-94   3:27a
CALC     GSP       3633 02-24-94   3:27a
CALC2    GSP       4408 02-24-94   3:27a
CALC3    GSP       2502 02-24-94   3:27a
CART     SMB        245 02-24-94   3:27a
CIRCUITS SMB      44898 02-24-94   3:27a
CNT      GSP       2703 02-24-94   3:27a
CONTROL  MNU      10065 02-24-94   3:27a
CTLDET2  GSP       3044 02-24-94   3:27a
CTLDET3  GSP       2580 02-24-94   3:27a
DGAP     GSP       3003 02-24-94   3:27a
```

DGAP2	GSP	2593	02-24-94	3:27a
DIGDET	GSP	2946	02-24-94	3:27a
DIGITAL	MNU	4434	02-24-94	3:27a
DIR	GSP	4139	02-24-94	3:27a
DISPDIR	GSP	4234	02-24-94	3:27a
DISPSAV	GSP	1001	02-24-94	3:27a
DISPSEC	GSP	980	02-24-94	3:27a
DOTDET	GSP	2064	02-24-94	3:27a
DTIMDET	GSP	2588	02-24-94	3:27a
DYNAMIC	MNU	14019	02-24-94	3:27a
EMPTY	GSP	845	02-24-94	3:27a
EPSON_MX	PRP	36	02-24-94	3:27a
EPSON_MX	PTR	2048	02-24-94	3:27a
EXIT	GSP	1021	02-24-94	3:27a
EXTASCII	FNT	1536	02-24-94	3:27a
FF1	GSP	2190	02-24-94	3:27a
FF2	GSP	2335	02-24-94	3:27a
FILEUTIL	GSP	1296	02-24-94	3:27a
GNET_OLD	KML	510	02-24-94	3:27a
GRAPHIC	MNU	2627	02-24-94	3:27a
HCURS	GSP	415	02-24-94	3:27a
HDWDET	GSP	1954	02-24-94	3:27a
HELP	GSP	1217	02-24-94	3:27a
HELP1	GSP	1217	02-16-95	3:01p
HELP2	GSP	1217	02-16-95	3:01p
HELP3	GSP	1217	02-16-95	3:01p
HELP4	GSP	1217	02-16-95	3:01p
HELP5	GSP	1210	02-24-94	3:27a
HELP6	GSP	1156	02-24-94	3:27a
HGLSTMEN	GSP	1344	02-24-94	3:27a
HGREP	GSP	781	02-24-94	3:27a
HGREPLST	GSP	1688	02-24-94	3:27a
HGREPMEN	GSP	1408	02-24-94	3:27a
HISMON	GSP	2463	02-24-94	3:27a
HISREP	GSP	1588	02-24-94	3:27a
HIST	MNU	15746	02-24-94	3:27a
HISTB	GSP	1862	02-24-94	3:27a
HISTLST	GSP	2038	02-24-94	3:27a
HISTMEN1	GSP	2029	02-24-94	3:27a
HTBREP	GSP	1226	02-24-94	3:27a
ICONS	ICN	66267	02-24-94	3:27a
IDB	GSP	2674	02-24-94	3:27a
IDBAIN	GSP	2739	02-24-94	3:27a
IDBAIO	GSP	3137	02-24-94	3:27a
IDBAO	GSP	2070	02-24-94	3:27a
IDBDIN	GSP	4381	02-24-94	3:27a
IDBDINB	GSP	1574	02-24-94	3:27a
IDBDIO	GSP	4831	02-24-94	3:27a
IDBDIOB	GSP	1644	02-24-94	3:27a
IDBDOT	GSP	2809	02-24-94	3:27a

IDBDOTB	GSP	1325	02-24-94	3:27a
IDBENTRY	GSP	663	02-24-94	3:27a
IDBSIO	GSP	1480	02-24-94	3:27a
IDBSTG	GSP	1271	02-24-94	3:27a
IDBSTGO	GSP	1309	02-24-94	3:27a
KB	TBL	3392	02-24-94	3:27a
KEYHELP	EXE	39261	02-24-94	3:27a
KEYMAC	EXE	34201	02-24-94	3:27a
LIBRARY	MNU	3471	02-24-94	3:27a
LISTMAIN	GSP	363	02-24-94	3:27a
LISTMENU	GSP	1657	02-24-94	3:27a
LOGIC	GSP	2843	02-24-94	3:27a
LOGO	DAT	9	02-24-94	3:27a
MACROS	MNU	2282	02-24-94	3:27a
MATH	GSP	2602	02-24-94	3:27a
MATH2	GSP	2506	02-24-94	3:27a
MEMSYS	MNU	863	02-24-94	3:27a
METACONF	EXE	69599	02-24-94	3:27a
METACONF	MNU	3859	02-24-94	3:27a
MNEMONIC	MNU	8335	02-24-94	3:27a
MOUSE	OPT	4118	02-24-94	3:27a
PAIO1	GSP	3453	02-24-94	3:27a
PAIO2	GSP	3453	02-24-94	3:27a
PAIO3	GSP	3888	02-24-94	3:27a
PAOUT	GSP	3143	02-24-94	3:27a
PASSWORD	GSP	1794	02-24-94	3:27a
PDIN	GSP	5661	02-24-94	3:27a
PDOUT	GSP	4171	02-24-94	3:27a
PLOT	GSP	465	02-24-94	3:27a
PLOTMEN	GSP	2405	02-24-94	3:27a
PULNOT	GSP	2322	02-24-94	3:27a
RAMP	GSP	3525	02-24-94	3:27a
README	354	7611	06-17-92	3:54a
RUNTIME	EXE	401424	02-24-94	3:27a
SEL	GSP	3718	02-24-94	3:27a
SEQ	GSP	2131	02-24-94	3:27a
SET_EMS	EXE	12857	06-17-92	3:54a
SIM	GSP	3038	02-24-94	3:27a
SLOGIC	GSP	2360	02-24-94	3:27a
SORTIDBM	GSP	1976	02-24-94	3:27a
SORTMAIN	GSP	2164	02-24-94	3:27a
SORTMENU	GSP	1128	02-24-94	3:27a
SPULNOT	GSP	1503	02-24-94	3:27a
STANDARD	CFN	18899	02-24-94	3:27a
STANDARD	HLS	1908	02-24-94	3:27a
STANDARD	KML	544	02-24-94	3:27a
STAT1	GSP	2271	02-24-94	3:27a
STAT2	GSP	2053	02-24-94	3:27a
STATECMP	EXE	28561	02-24-94	3:27a
STRING	MNU	1027	02-24-94	3:27a

SUPMAC	MNU	2461	02-24-94	3:27a
SWCH	GSP	2925	02-24-94	3:27a
SYSCONF	GSP	3011	02-24-94	3:27a
SYSCONF	MNU	15280	02-24-94	3:27a
SYSCONF2	GSP	2891	02-24-94	3:27a
SYSPERF	GSP	2325	02-24-94	3:27a
TD	GSP	3086	02-24-94	3:27a
TESTHDW	EXE	11969	06-17-92	3:54a
TIMERS	GSP	2363	02-24-94	3:27a
TOT	GSP	3285	02-24-94	3:27a
TPO	GSP	3658	02-24-94	3:27a
TREND	GSP	467	02-24-94	3:27a
TRENDLST	GSP	2042	02-24-94	3:27a
TRENDMEN	GSP	1990	02-24-94	3:27a
TRENDREP	GSP	3400	02-24-94	3:27a
TUNE	GSP	3152	02-24-94	3:27a
TUNE2	GSP	3156	02-24-94	3:27a
TUNE3	GSP	2935	02-24-94	3:27a
TUNE4	GSP	2432	02-24-94	3:27a
USER	GSP	3967	02-24-94	3:27a
USER2	GSP	3090	02-24-94	3:27a
XC	GSP	3543	02-24-94	3:27a
XLATE	EXE	78445	02-24-94	3:27a
CTLDET	GSP	2896	02-24-94	3:27a
ELECTRIC	SMB	2837	02-24-94	3:27a
FACEPLAT	SMB	15558	02-24-94	3:27a
HPPJET	PRP	36	05-08-92	3:53a
HPPJET	PTR	2560	05-08-92	3:53a
IBMPRO	PRP	36	09-17-90	3:32a
IBMPRO	PTR	2048	09-17-90	3:32a
NEWJET	PRP	82	05-08-92	3:53a
NEWJET	PTR	2560	05-08-92	3:53a
README	353	6763	05-08-92	3:53a
READPTR	ME	562	05-08-92	3:53a
XROX4020	PRP	82	05-08-92	3:53a
XROX4020	PTR	2560	05-08-92	3:53a
HEATEXCH	SMB	846	02-24-94	3:27a
LOGIC	SMB	1353	02-24-94	3:27a
MODICON	350	282	04-03-92	3:50a
MODICON	DES	2418	04-03-92	3:50a
MODICON	DRV	7038	04-03-92	3:50a
MODICON	MNU	14762	04-03-92	3:50a
MACHINE	SMB	1841	02-24-94	3:27a
MISC	SMB	1680	02-24-94	3:27a
MOTORS	SMB	396	02-24-94	3:27a
PRESS	SMB	1755	02-24-94	3:27a
STATES	SFL	642	02-24-94	3:27a
DOSNODE	PWD	226	02-18-94	5:01p
DOSPWD	EXE	113849	02-18-94	5:01p
DOSPWD	MNU	959	02-18-94	5:01p

EGA	DEV	13450	02-18-94	5:01p
GCOPY	BAT	33	01-15-92	3:52a
GCOPY	EXE	145747	02-18-94	5:01p
GEN_NET	OPT	61551	01-15-92	3:52a
GENCEXIT	GSP	928	02-18-94	5:02p
GENCFG	GNF	265	02-18-94	1:36a
GENCFG	MNU	33934	02-18-94	5:01p
GENCFGB	MNU	24190	01-15-92	3:52a
GENCMDR	BAT	46	01-15-92	3:52a
GENCMDR	DOC	5912	02-18-94	5:02p
GENCMDR	EXE	260257	02-18-94	5:02p
GENCMDR	GSP	948	02-18-94	5:02p
GENCMDR	HLP	23280	02-18-94	5:02p
GENCMDR	MNU	7682	02-18-94	5:02p
GENCMDR	NDX	625	02-18-94	5:02p
GENCMDR	OPT	43487	02-18-94	5:02p
GENCMDR2	GSP	860	02-18-94	5:02p
GENCOMP	EXE	44069	02-18-94	5:01p
GENCONF	EXE	91957	02-18-94	5:01p
GENMON	EXE	16487	02-18-94	5:01p
GN_CA	OPT	66311	01-15-92	3:52a
GN_CB	OPT	91089	01-15-92	3:52a
GN_CFA	OPT	99255	01-15-92	3:52a
GN_CFB	OPT	124033	01-15-92	3:52a
GN_COMM	OPT	35453	02-18-94	5:02p
GN_DB	OPT	86329	01-15-92	3:52a
GN_DCA	OPT	77441	01-15-92	3:52a
GN_DCB	OPT	102235	01-15-92	3:52a
GN_DCFA	OPT	110897	01-15-92	3:52a
GN_DCFB	OPT	135675	01-15-92	3:52a
GN_DFA	OPT	94991	01-15-92	3:52a
GN_DFB	OPT	119273	01-15-92	3:52a
GN_DRVR	OPT	67307	02-18-94	5:02p
GN_FTP	OPT	52219	02-18-94	5:02p
GN_INTF	OPT	32731	02-18-94	5:02p
GN_INTFB	OPT	56305	01-15-92	3:52a
GN_TIMER	EXE	13851	02-18-94	5:01p
GN_TIMER	SYS	1856	02-18-94	5:01p
GNETDIR	GSP	676	02-18-94	5:01p
GNETMON	GSP	2073	02-18-94	5:01p
GNETMON2	GSP	2045	02-18-94	5:01p
GNFXEXIT	GSP	972	02-18-94	5:01p
GNFXR1	GSP	1053	02-18-94	5:01p
GNFXR2	GSP	1043	02-18-94	5:01p
GNFXR3	GSP	1024	02-18-94	5:01p
GNFXR4	GSP	827	02-18-94	5:01p
LOGO_NET	DAT	9	02-18-94	1:36a
PS110RST	EXE	15259	02-18-94	5:01p
README	DOC	5912	01-15-92	3:52a
SHOW_OPT	EXE	15931	02-18-94	5:01p

UNPACK	EXE	29378	02-18-94	1:36a
STATES	SFS	913	02-24-94	3:27a
DXFTOGRP	EXE	58777	10-31-91	1:31a
GRPTODXF	EXE	39665	10-31-91	1:31a
RRCOMBO	OPT	73309	04-28-93	3:26a
TANKS	SMB	1890	02-24-94	3:27a
VALVES	SMB	2364	02-24-94	3:27a
VATS	SMB	3741	02-24-94	3:27a
HPLSRJII	PRP	36	09-17-90	3:32a
HPLSRJII	PTR	2560	09-17-90	3:32a
CONFMSMI	COM	1487	02-18-94	5:01p
CONF104	FNT	7168	02-18-94	5:01p
BOOT	BAT	79	02-18-94	5:01p
CONFIG	PRM	407	08-03-94	10:09a
AUTOCAD	DOC	9984	10-31-91	1:31a
SHUTTLE	SMB	48419	10-31-91	1:31a
REC	GSP	2266	10-07-91	3:41a
RRTEST	EXE	68794	10-07-91	3:41a
README	341	2434	10-07-91	3:41a
TEST	GIF	22019	01-12-93	11:25p
RSSBOOT	GSP	3450	02-15-95	11:29a
STANDARD	HLP	1373	01-14-94	4:16p
AF5000	DRV	35066	04-29-94	3:52a
AF5000	AIF	16959	05-12-92	3:52a
AF5000	DES	448	05-06-92	3:52a
AF5000	DOC	31716	05-12-92	3:52a
AF5000	MNU	812	05-06-92	3:52a
MODPLUS	DES	448	02-09-94	3:38a
MODPLUS	DRV	19766	08-24-94	9:55p
MODPLUS	MNU	12100	08-24-94	3:38a
MODPLUS	DOC	23524	03-29-93	3:38a
MODPLUS	335	579	06-03-93	3:36a
MODPLUS	336	747	06-03-93	3:36a
GNETMON3	GSP	1916	02-18-94	5:01p
DANISH	FNT	1536	02-24-94	3:27a
CONFIG	SYS	26	02-24-94	3:27a
README	372	8135	02-24-94	3:27a
ALARM	GSP	2219	02-24-94	3:27a
BLANK	GSP	243	02-24-94	3:27a
EVENT	GSP	2212	02-24-94	3:27a
RRCOMBO	GSP	2450	02-24-94	3:27a
USER2_2	GSP	3090	02-24-94	3:27a
USER2_3	GSP	3090	02-24-94	3:27a
USER2_4	GSP	3090	02-24-94	3:27a
USER2_5	GSP	3090	02-24-94	3:27a
USER2_6	GSP	3090	02-24-94	3:27a
USER2_7	GSP	3090	02-24-94	3:27a
USER3	GSP	1855	02-24-94	3:27a
USER3_2	GSP	1855	02-24-94	3:27a
USER3_3	GSP	1855	02-24-94	3:27a

```
USER3_4   GSP      1855 02-24-94   3:27a
USER3_5   GSP      1855 02-24-94   3:27a
USER3_6   GSP      1855 02-24-94   3:27a
USER3_7   GSP      1855 02-24-94   3:27a
USER_2    GSP      3967 02-24-94   3:27a
USER_3    GSP      3967 02-24-94   3:27a
USER_4    GSP      3967 02-24-94   3:27a
USER_5    GSP      3967 02-24-94   3:27a
USER_6    GSP      3967 02-24-94   3:27a
USER_7    GSP      3967 02-24-94   3:27a
SHADOW    GSP      1912 02-18-94   5:01p
NETALMS   GSP      1825 02-18-94   5:01p
EXPORT    OPT     18307 02-18-94   5:01p
MODPLUS   338        60 09-23-94   3:38a
HPLSR4    PRP        36 03-31-94   5:12p
HPLSR4    PTR      2560 09-17-90   3:32a
HELP      HLS      1481 02-16-95   3:01p
HELP      HLP       998 02-16-95   3:01p

 Directory of C:\TEST

.             <DIR>      03-09-94   9:39a
..            <DIR>      03-09-94   9:39a
LCL_HELP      <DIR>      03-09-94   9:40a
RMT_HELP      <DIR>      03-09-94   9:40a
LCL_KEY       <DIR>      03-09-94   9:40a
RMT_KEY       <DIR>      03-09-94   9:40a
KEYMACS       <DIR>      03-09-94   9:40a
ALRMCOPY      <DIR>      03-09-94   9:40a
C_TEST        <DIR>      04-12-94   1:11p
TIME          <DIR>      10-31-94   5:41a
ALMRENAM EXE     11602 07-04-93  12:26a
TEST     CA      19228 11-08-96  10:17a
TEST     CFG      1752 11-08-96  10:17a
TEST     CI      20560 11-08-96  10:17a
TEST     DB     134274 11-08-96  10:17a
TEST     KML       103 05-31-95   2:20p
TEST     KMS       155 05-31-95   2:20p
TEST     MDB        12 11-08-96  10:17a
TEST     SFL       138 07-07-93   7:13p
TEST     SFS       457 07-07-93   7:10p
TEST     XDB      1892 11-08-96  10:17a
RUNTEST  CA      19228 11-08-96  10:17a
RUNTEST  CFG      1752 11-08-96  10:17a
RUNTEST  CI      20560 11-08-96  10:17a
RUNTEST  DB     134274 11-08-96  10:17a
RUNTEST  KML       103 05-31-95   2:20p
RUNTEST  KMS       155 05-31-95   2:20p
RUNTEST  MDB        12 11-08-96  10:17a
```

RUNTEST	XDB	1892	11-08-96	10:17a
ROLLOVER	GRP	13928	04-04-96	4:51p
ABRTENAB	GRP	44959	04-11-96	11:13a
ABRTNAME	GRP	28563	04-11-96	11:14a
ASMAIN	GRP	2827	07-28-94	5:47p
CSMAIN	GRP	9058	04-24-96	2:33p
DACS	GRP	12755	03-11-94	10:19a
GASSUM	GRP	9140	04-08-96	10:04a
HVTALARM	GRP	10289	04-24-96	3:03p
IOSTATUS	GRP	13096	06-08-95	2:04p
MANABRT	GRP	4199	04-16-96	8:22a
MAP	GRP	4087	03-22-95	1:55p
MININ1	GRP	15878	04-12-96	12:28p
MININ2	GRP	12135	09-04-96	1:04p
MIT17B	GRP	13092	04-03-96	12:38p
MIT17C	GRP	11621	03-10-94	8:14a
MSMAIN	GRP	5643	04-24-96	2:53p
NEW72HR	GRP	5626	04-05-96	9:43a
OLD72HR	GRP	5631	04-05-96	9:46a
PUMP	GRP	9579	04-04-96	5:17p
PUMPALRM	GRP	8695	02-15-95	11:27a
PUMPOPS	GRP	5126	03-29-96	4:15p
ABRTCHEK	GRP	10487	04-24-96	3:08p
STRNALM	GRP	11330	04-04-96	5:21p
SUMMARY	GRP	11579	03-03-95	9:53a
SY101-1	GRP	7092	01-28-94	12:27p
TBSTC	GRP	7392	10-26-94	11:08a
TEMPALM	GRP	10475	03-29-96	4:24p
TEMPRFL	GRP	13453	03-10-94	8:24a
TESTEXIT	GRP	2574	06-05-95	10:54a
WELCOME	GRP	3450	02-15-95	11:29a
BOOT	GSP	3450	02-15-95	11:29a
HELP	GSP	1217	06-17-92	3:54a
HELP1	GSP	1217	03-29-94	10:40a
HELP2	GSP	1217	03-29-94	10:40a
HELP3	GSP	1217	03-29-94	10:40a
HELP4	GSP	1217	03-29-94	10:40a
HELP5	GSP	1217	03-29-94	10:40a
HELP6	GSP	1217	03-29-94	10:40a
RSSBOOT	GSP	3450	02-15-95	11:29a
HISTCOPY	RPS	2712	08-12-94	3:24p
HISTFNAM	RPS	1243	08-12-94	3:23p
SETLIMS	RPS	1423	04-03-96	3:17p
ALARM00	TXT	9108	03-12-97	11:03a
STORE	DT	8	03-12-97	10:00a
FN	DT	7	03-12-97	10:00a
73CAPUMP	PRN	1616	03-12-97	11:03a
NEW72	PRN	52625	03-12-97	11:03a
TEST	LST	0	03-12-97	11:06a
OLDFN	DT	7	03-12-97	10:02a

```
TEST       NEW    134274 02-11-97   1:37p
RSS        BAT       927 05-25-95   2:42p
XXX        XXX        26 03-10-97   7:08p
OLD72      PRN     93773 03-10-97   7:08p
73CAMIT2   PRN     39963 03-12-97  11:03a
73CATBS1   PRN      2236 03-12-97  11:03a
73CAMIT1   PRN     44981 03-12-97  11:03a
73CAGAS    PRN      4003 03-12-97  11:03a
73CARGA5   PRN      2855 03-12-97  11:03a
73CAMIT3   PRN     39715 03-12-97  11:03a
73CATBS2   PRN      2236 03-12-97  11:03a
73CASTRN   PRN      2708 03-12-97  11:03a
73CAVOL    PRN     54921 03-12-97  11:03a

 Directory of C:\TEST\LCL_HELP

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
STANDARD HLS         1981 01-14-94   4:12p
STANDARD HLP         1373 01-14-94   4:16p
HELP       GSP       1217 06-17-92   3:54a
HELP1      GSP       1217 01-14-94   4:16p
HELP2      GSP       1217 01-14-94   4:16p
HELP3      GSP       1217 01-14-94   4:16p
HELP4      GSP       1217 01-14-94   4:16p
HELP5      GSP       1217 01-14-94   4:16p
HELP6      GSP       1217 01-14-94   4:16p

 Directory of C:\TEST\RMT_HELP

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
STANDARD HLS         2162 01-14-94   4:09p
HELP       GSP       1217 06-17-92   3:54a
HELP1      GSP       1217 01-14-94   4:17p
HELP2      GSP       1217 01-14-94   4:17p
HELP3      GSP       1217 01-14-94   4:17p
HELP4      GSP       1217 01-14-94   4:17p
HELP5      GSP       1217 01-14-94   4:17p
HELP6      GSP       1217 01-14-94   4:17p
STANDARD HLP         1366 01-14-94   4:17p
```

```
 Directory of C:\TEST\LCL_KEY

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
TEST     KML            55 04-24-93   1:40p
TEST     KMS            32 04-23-93   5:16p

 Directory of C:\TEST\RMT_KEY

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
TEST     KML          1437 01-22-94   1:57p
TEST     KMS          9293 01-22-94   1:57p

 Directory of C:\TEST\KEYMACS

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
TEST     KML           103 05-31-95   2:20p
TEST     KMS           155 05-31-95   2:20p
HELP     GSP          1217 06-17-92   3:54a
HELP1    GSP          1217 04-24-93   1:43p
HELP2    GSP          1217 04-24-93   1:43p
HELP3    GSP          1217 04-24-93   1:43p
HELP4    GSP          1217 04-24-93   1:43p
HELP5    GSP          1217 04-24-93   1:43p
HELP6    GSP          1217 04-24-93   1:43p

 Directory of C:\TEST\ALRMCOPY

.            <DIR>      03-09-94   9:40a
..           <DIR>      03-09-94   9:40a
RUNTIME  BAT           187 11-14-94   2:36p
ALEXP    EXE         11602 07-04-93  12:26a
RUNTIME  OLD           186 07-08-93  11:50p
ALMRENAM EXE         11602 07-04-93  12:26a

 Directory of C:\TEST\C_TEST

.            <DIR>      04-12-94   1:11p
..           <DIR>      04-12-94   1:11p
ALMRENAM CPP          1760 04-11-94  12:55p
ALMCOPY  BAT            87 12-10-93   3:46p
ALMRENAM OBJ          1603 07-04-93  12:26a
ALMRENAM EXE         11602 07-04-93  12:26a
```

DACS STATION 5 FILE LISTINGS

"ALMCOPY.BAT" on Station 5

```
copy \test\alarm00.txt \hist\6B6029AL.TXT
copy \test\alarm00.txt \hist2\6B6029AL.TXT
```

"AUTOEXEC.BAT" on Station 5

```
@ECHO OFF
PROMPT $p$g
C:\NETROOM\XLOAD.EXE C:\MOUSE\MOUSE
goto %config%

:GEN354
PATH C:\DOS;C:\;C:\GENESIS;C:\NETROOM;C:\EXPERT
goto end

:GEN372
PATH C:\DOS;C:\;C:\GEN372;C:\NETROOM;C:\EXPERT
SHARE
goto end

:end
```

"CONFIG.SYS" on Station 5

```
[MENU]
menuitem=GEN354,Genesis system version 3.54
menuitem=GEN372,Genesis system version 3.72
menudefault=GEN372[,0]

[COMMON]
DEVICE=C:\NETROOM\RM386.EXE AUTO X=D900-DDFF
DEVICE=C:\NETROOM\SYSCLOAK.EXE
DEVICE=C:\NETROOM\XLOAD.SYS -O
BUFFERS=20,0
FILES=55
LASTDRIVE=E
FCBS=4,0
STACKS=0,0
SHELL=C:\NETROOM\XLOAD.EXE -SDE01 -M57021 -E C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICE=C:\NETROOM\STACKS.EXE 9,256

[GEN354]

[GEN372]
```

"RUNTIME.BAT" on Station 5

```
:_____________________________________________________
:Filename:  RUNTIME.BAT rev.1   05-25-95 sos/rwt
:
:Purpose:  This batch file creates runtime files
:          for TEST and MOTOR strategies on Station 7.
:          Format of command:
:          runtime %1
:             where %1 is the strategy name to be used runtime
:
:_____________________________________________________

ECHO
CLS
@ECHO OFF

IF "%1"=="test" GOTO TEST
IF "%1"=="TEST" GOTO TEST
IF "%1"=="motor" GOTO MOTOR
IF "%1"=="MOTOR" GOTO MOTOR

REM will run other strategies
C:\GEN372\RUNTIME %1
GOTO END

:TEST
COPY C:\TEST\RUNTEST.DB C:\TEST\TEST.DB > XXX.XXX
COPY C:\TEST\RUNTEST.MDB C:\TEST\TEST.MDB > XXX.XXX
COPY C:\TEST\RUNTEST.XDB C:\TEST\TEST.XDB > XXX.XXX
COPY C:\TEST\RUNTEST.CI C:\TEST\TEST.CI > XXX.XXX
COPY C:\TEST\RUNTEST.CA C:\TEST\TEST.CA > XXX.XXX
COPY C:\TEST\RUNTEST.CFG C:\TEST\TEST.CFG > XXX.XXX
COPY C:\TEST\WELCOME.GRP C:\GEN372\BOOT.GSP
C:\TEST\ALMRENAM
CALL C:\ALMCOPY
DEL ALARM*.*
C:\GEN372\RUNTIME %1
GOTO END

:MOTOR
COPY C:\MOTOR\MAINMENU.GRP C:\GEN372\BOOT.GSP
C:\GEN372\RUNTIME %1

:END
```

"RENAME72.BAT" on Station 5

```
:_____________________________________________________
:Filename:  RENAME72.BAT
:
:Purpose:  This batch file deletes OLD72.PRN and renames
:          NEW72.PRN to OLD72.PRN to help minimize the
:          file size build up of NEW72.PRN
:          Executed manually at DOS prompt when desired.
:_____________________________________________________

DEL OLD72.PRN
REN NEW72.PRN OLD72.PRN
```

DACS STATION 6 DIRECTORY LISTINGS

```
 Volume in drive C is MS-DOS_5
 Volume Serial Number is 1C28-514E
                 188358656 bytes free


 Directory of C:\*.BAT,*.SYS

AUTOEXEC BAT       268 11-27-96  12:02p
CONFIG   SYS       404 12-02-94   9:10a
RSS      BAT       927 05-25-95   2:42p


 Directory of C:\GEN372

.            <DIR>      04-07-94  12:20p
..           <DIR>      04-07-94  12:20p
REPORT   OPT     57547 05-01-90   3:21a
IBMPRO   PRP        36 09-17-90   3:32a
HPLSRJII PRP        36 09-17-90   3:32a
IBMPRO   PTR      2048 09-17-90   3:32a
HPLSRJII PTR      2560 09-17-90   3:32a
RRCOMBO  OPT     73309 04-28-93   3:26a
COUNTRY  FR          4 10-31-91   3:52a
COUNTRY  GR          5 10-31-91   3:52a
LOGO     DAT         9 02-24-94   3:27a
EPSON_MX PRP        36 02-24-94   3:27a
LISTMAIN GSP       363 02-24-94   3:27a
HCURS    GSP       415 02-24-94   3:27a
PLOT     GSP       465 02-24-94   3:27a
TREND    GSP       467 02-24-94   3:27a
STANDARD KML       511 02-24-94   3:27a
GNET_OLD KML       510 02-24-94   3:27a
IDBENTRY GSP       663 02-24-94   3:27a
HGREP    GSP       781 02-24-94   3:27a
HTBREP   GSP      1226 02-24-94   3:27a
EMPTY    GSP       845 02-24-94   3:27a
DISPSEC  GSP       980 02-24-94   3:27a
DISPSAV  GSP      1001 02-24-94   3:27a
EXIT     GSP      1021 02-24-94   3:27a
SORTMENU GSP      1128 02-24-94   3:27a
LISTMENU GSP      1657 02-24-94   3:27a
IDBSTG   GSP      1271 02-24-94   3:27a
IDBSTGO  GSP      1309 02-24-94   3:27a
HGLSTMEN GSP      1344 02-24-94   3:27a
IDBDOTB  GSP      1325 02-24-94   3:27a
FILEUTIL GSP      1296 02-24-94   3:27a
HISREP   GSP      1588 02-24-94   3:27a
```

HGREPMEN	GSP	1408	02-24-94	3:27a
IDBSIO	GSP	1480	02-24-94	3:27a
SPULNOT	GSP	1503	02-24-94	3:27a
IDBDINB	GSP	1574	02-24-94	3:27a
IDBDIOB	GSP	1644	02-24-94	3:27a
HGREPLST	GSP	1688	02-24-94	3:27a
ALMSUM	GSP	2244	02-24-94	3:27a
PASSWORD	GSP	1794	02-24-94	3:27a
HISTB	GSP	1862	02-24-94	3:27a
HDWDET	GSP	1954	02-24-94	3:27a
SORTIDBM	GSP	1976	02-24-94	3:27a
TRENDMEN	GSP	1990	02-24-94	3:27a
HISTMEN1	GSP	2029	02-24-94	3:27a
HISTLST	GSP	2038	02-24-94	3:27a
TRENDLST	GSP	2042	02-24-94	3:27a
EPSON_MX	PTR	2048	02-24-94	3:27a
STAT2	GSP	2053	02-24-94	3:27a
DOTDET	GSP	2064	02-24-94	3:27a
IDBAO	GSP	2070	02-24-94	3:27a
AOUTDET	GSP	2113	02-24-94	3:27a
KB	TBL	3392	02-24-94	3:27a
SEQ	GSP	2131	02-24-94	3:27a
SORTMAIN	GSP	2164	02-24-94	3:27a
FF1	GSP	2190	02-24-94	3:27a
STAT1	GSP	2271	02-24-94	3:27a
PULNOT	GSP	2322	02-24-94	3:27a
SYSPERF	GSP	2325	02-24-94	3:27a
RSYSPERF	GSP	2325	02-24-94	3:27a
REMPERF	GSP	2332	02-24-94	3:27a
FF2	GSP	2335	02-24-94	3:27a
SLOGIC	GSP	2360	02-24-94	3:27a
TIMERS	GSP	2363	02-24-94	3:27a
PLOTMEN	GSP	2405	02-24-94	3:27a
AMBDET	GSP	2449	02-24-94	3:27a
TUNE4	GSP	2432	02-24-94	3:27a
HISMON	GSP	2463	02-24-94	3:27a
CALC3	GSP	2502	02-24-94	3:27a
MATH2	GSP	2506	02-24-94	3:27a
SYSCONF	GSP	3011	02-24-94	3:27a
CTLDET3	GSP	2580	02-24-94	3:27a
DTIMDET	GSP	2588	02-24-94	3:27a
DGAP2	GSP	2593	02-24-94	3:27a
MATH	GSP	2602	02-24-94	3:27a
ALM	GSP	2649	02-24-94	3:27a
IDB	GSP	2674	02-24-94	3:27a
CNT	GSP	2703	02-24-94	3:27a
IDBAIN	GSP	2739	02-24-94	3:27a
IDBDOT	GSP	2809	02-24-94	3:27a
LOGIC	GSP	2843	02-24-94	3:27a
SYSCONF2	GSP	2891	02-24-94	3:27a

CTLDET	GSP	2896	02-24-94	3:27a
SWCH	GSP	2925	02-24-94	3:27a
TUNE3	GSP	2935	02-24-94	3:27a
DIGDET	GSP	2946	02-24-94	3:27a
AINDET	GSP	2995	02-24-94	3:27a
DGAP	GSP	3003	02-24-94	3:27a
SIM	GSP	3038	02-24-94	3:27a
CTLDET2	GSP	3044	02-24-94	3:27a
TD	GSP	3086	02-24-94	3:27a
USER2	GSP	3090	02-24-94	3:27a
IDBAIO	GSP	3137	02-24-94	3:27a
PAOUT	GSP	3143	02-24-94	3:27a
TUNE	GSP	3152	02-24-94	3:27a
TUNE2	GSP	3156	02-24-94	3:27a
TOT	GSP	3285	02-24-94	3:27a
TRENDREP	GSP	3400	02-24-94	3:27a
PAIO1	GSP	3453	02-24-94	3:27a
PAIO2	GSP	3453	02-24-94	3:27a
RAMP	GSP	3525	02-24-94	3:27a
XC	GSP	3543	02-24-94	3:27a
CALC	GSP	3633	02-24-94	3:27a
TPO	GSP	3658	02-24-94	3:27a
SEL	GSP	3718	02-24-94	3:27a
GENESIS	GSP	3776	02-24-94	3:27a
PAIO3	GSP	3888	02-24-94	3:27a
USER	GSP	3967	02-24-94	3:27a
DIR	GSP	4139	02-24-94	3:27a
PDOUT	GSP	4171	02-24-94	3:27a
DISPDIR	GSP	4234	02-24-94	3:27a
IDBDIN	GSP	4381	02-24-94	3:27a
CALC2	GSP	4408	02-24-94	3:27a
IDBDIO	GSP	4831	02-24-94	3:27a
PDIN	GSP	5661	02-24-94	3:27a
MOUSE	OPT	4118	02-24-94	3:27a
RSS	EXE	343638	02-24-94	3:27a
LOGO_NET	DAT	9	02-18-94	1:36a
GCOPY	BAT	33	01-15-92	3:52a
GENCMDR	BAT	46	01-15-92	3:52a
BOOT	BAT	79	02-18-94	5:01p
DOSNODE	PWD	226	02-18-94	5:01p
GENCFG	GNF	265	02-18-94	1:36a
GENCMDR	NDX	625	02-18-94	5:02p
GNETDIR	GSP	676	02-18-94	5:01p
GNFXR4	GSP	827	02-18-94	5:01p
GENCMDR2	GSP	860	02-18-94	5:02p
GENCEXIT	GSP	928	02-18-94	5:02p
GENCMDR	GSP	948	02-18-94	5:02p
DOSPWD	MNU	959	02-18-94	5:01p
GNFXEXIT	GSP	972	02-18-94	5:01p
GNFXR3	GSP	1024	02-18-94	5:01p

GNFXR2	GSP	1043	02-18-94	5:01p
GNFXR1	GSP	1053	02-18-94	5:01p
CONFMSMI	COM	1487	02-18-94	5:01p
GN_TIMER	SYS	1856	02-18-94	5:01p
GNETMON	GSP	2073	02-18-94	5:01p
GNETMON2	GSP	2045	02-18-94	5:01p
GENCMDR	DOC	5912	02-18-94	5:02p
README	DOC	5912	01-15-92	3:52a
CONF104	FNT	7168	02-18-94	5:01p
GENCMDR	MNU	7682	02-18-94	5:02p
GN_TIMER	EXE	13851	02-18-94	5:01p
PS110RST	EXE	15259	02-18-94	5:01p
EGA	DEV	13450	02-18-94	5:01p
GENMON	EXE	16487	02-18-94	5:01p
SHOW_OPT	EXE	15931	02-18-94	5:01p
GENCFG	MNU	33934	02-18-94	5:01p
GENCMDR	HLP	23280	02-18-94	5:02p
UNPACK	EXE	29378	02-18-94	1:36a
GENCFGB	MNU	24190	01-15-92	3:52a
GN_INTF	OPT	32731	02-18-94	5:02p
GN_COMM	OPT	35453	02-18-94	5:02p
GENCMDR	OPT	43487	02-18-94	5:02p
GENCOMP	EXE	44069	02-18-94	5:01p
GN_FTP	OPT	52219	02-18-94	5:02p
GN_DRVR	OPT	67307	02-18-94	5:02p
GN_INTFB	OPT	56305	01-15-92	3:52a
GEN_NET	OPT	61551	01-15-92	3:52a
GN_CA	OPT	66311	01-15-92	3:52a
DOSPWD	EXE	113849	02-18-94	5:01p
GN_DCA	OPT	77441	01-15-92	3:52a
GN_DB	OPT	86329	01-15-92	3:52a
GN_CB	OPT	91089	01-15-92	3:52a
GENCONF	EXE	91957	02-18-94	5:01p
GN_DFA	OPT	94991	01-15-92	3:52a
GN_CFA	OPT	99255	01-15-92	3:52a
GN_DCB	OPT	102235	01-15-92	3:52a
GN_DCFA	OPT	110897	01-15-92	3:52a
GN_DFB	OPT	119273	01-15-92	3:52a
GN_CFB	OPT	124033	01-15-92	3:52a
GN_DCFB	OPT	135675	01-15-92	3:52a
GCOPY	EXE	145747	02-18-94	5:01p
BCOPY	EXE	165287	01-15-92	3:52a
GENCMDR	EXE	260257	02-18-94	5:02p
BENCMDR	EXE	279813	01-15-92	3:52a
MODICON	DES	2418	04-03-92	3:50a
MODICON	DRV	7038	04-03-92	3:50a
AF5000	DES	448	05-06-92	3:52a
AF5000	DRV	35066	04-29-94	3:52a
HELP	GSP	1217	06-17-92	3:54a
GENCFG	NBC	1015	09-03-92	6:43p

$$REMOTE	DIR	9591	09-10-92.	9:12a
BOOT	GSP	3450	02-15-95	11:29a
RSSBOOT	GSP	3450	02-15-95	11:29a
HELP3	GSP	1217	02-16-95	3:01p
HELP2	GSP	1217	02-16-95	3:01p
HELP1	GSP	1217	02-16-95	3:01p
HELP5	GSP	1217	03-29-94	10:40a
HELP6	GSP	1217	03-29-94	10:40a
HELP4	GSP	1217	02-16-95	3:01p
GENNET	NDB	171	05-09-95	10:34a
GENCFG	NCF	276	05-09-95	10:34a
GENNET	DNL	442	05-09-95	10:34a
GENCMDR0	HLP	23280	01-15-92	3:52a
STANDARD	HLP	1373	01-14-94	4:16p
STANDARD	HLS	1981	01-14-94	4:12p
GNETMON3	GSP	1916	02-18-94	5:01p
AF5000	AIF	16959	05-12-92	3:52a
SHADOW	GSP	1912	02-18-94	5:01p
NETALMS	GSP	1825	02-18-94	5:01p
AF5000	DOC	31716	05-12-92	3:52a
DANISH	FNT	1536	02-24-94	3:27a
EXTASCII	FNT	1536	02-24-94	3:27a
ALARM	GSP	2219	02-24-94	3:27a
BLANK	GSP	243	02-24-94	3:27a
EVENT	GSP	2212	02-24-94	3:27a
RRCOMBO	GSP	2450	02-24-94	3:27a
USER2_2	GSP	3090	02-24-94	3:27a
USER2_3	GSP	3090	02-24-94	3:27a
USER2_4	GSP	3090	02-24-94	3:27a
USER2_5	GSP	3090	02-24-94	3:27a
USER2_6	GSP	3090	02-24-94	3:27a
USER2_7	GSP	3090	02-24-94	3:27a
USER3	GSP	1855	02-24-94	3:27a
USER3_2	GSP	1855	02-24-94	3:27a
USER3_3	GSP	1855	02-24-94	3:27a
USER3_4	GSP	1855	02-24-94	3:27a
USER3_5	GSP	1855	02-24-94	3:27a
USER3_6	GSP	1855	02-24-94	3:27a
USER3_7	GSP	1855	02-24-94	3:27a
USER_2	GSP	3967	02-24-94	3:27a
USER_3	GSP	3967	02-24-94	3:27a
USER_4	GSP	3967	02-24-94	3:27a
USER_5	GSP	3967	02-24-94	3:27a
USER_6	GSP	3967	02-24-94	3:27a
USER_7	GSP	3967	02-24-94	3:27a
EXPORT	OPT	18307	02-18-94	5:01p
AF5000	MNU	812	05-06-92	3:52a
MODPLUS	DES	448	06-03-93	3:36a
MODPLUS	DRV	19130	06-03-93	3:36a
MODPLUS	MNU	12100	06-03-93	3:36a

```
MODPLUS   DOC     24188 06-03-93   3:36a
MODPLUS   335       579 06-03-93   3:36a
MODPLUS   336       747 06-03-93   3:36a
HPLSR4    PRP        36 03-31-94   5:12p
HPLSR4    PTR      2560 09-17-90   3:32a
HELP      HLS      1481 02-16-95   3:01p
HELP      HLP       998 02-16-95   3:01p


 Directory of C:\TEST

.            <DIR>      12-07-93   2:14p
..           <DIR>      12-07-93   2:14p
KEYMACS      <DIR>      12-07-93   2:15p
BATFILES     <DIR>      12-07-93   2:15p
LCL_HELP     <DIR>      12-07-93   2:15p
LCL_KEY      <DIR>      12-07-93   2:15p
TEST      CA     19228 09-04-96   1:15p
TEST      CFG     1752 09-04-96   1:15p
TEST      CI     20560 09-04-96   1:15p
TEST      DB    134274 09-04-96   1:15p
TEST      MDB       12 09-04-96   1:15p
TEST      XDB     1892 09-04-96   1:15p
RSSTEST   CA     19228 09-04-96   1:15p
RSSTEST   CFG     1752 09-04-96   1:15p
RSSTEST   CI     20560 09-04-96   1:15p
RSSTEST   DB    134274 09-04-96   1:15p
RSSTEST   MDB       12 09-04-96   1:15p
RSSTEST   XDB     1892 09-04-96   1:15p
ROLLOVER  GRP    13928 04-04-96   4:51p
ABRTENAB  GRP    44959 04-11-96  11:13a
ABRTNAME  GRP    28563 04-11-96  11:14a
ASMAIN    GRP     2827 07-28-94   5:47p
CSMAIN    GRP     9058 04-24-96   2:33p
DACS      GRP    12755 03-11-94  10:19a
GASSUM    GRP     9140 04-08-96  10:04a
HVTALARM  GRP    10289 04-24-96   3:03p
IOSTATUS  GRP    13096 06-08-95   2:04p
MANABRT   GRP     4199 04-16-96   8:22a
MAP       GRP     4087 03-22-95   1:55p
MININ1    GRP    15878 04-12-96  12:28p
MININ2    GRP    12135 09-04-96   1:04p
MIT17B    GRP    13092 04-03-96  12:38p
MIT17C    GRP    11621 03-10-94   8:14a
MSMAIN    GRP     5643 04-24-96   2:53p
NEW72HR   GRP     5626 04-05-96   9:43a
OLD72HR   GRP     5631 04-05-96   9:46a
PUMP      GRP     9579 04-04-96   5:17p
PUMPALRM  GRP     8695 02-15-95  11:27a
PUMPOPS   GRP     5126 03-29-96   4:15p
```

```
ABRTCHEK GRP      10487 04-24-96   3:08p
STRNALM  GRP      11330 04-04-96   5:21p
SUMMARY  GRP      11579 03-03-95   9:53a
SY101-1  GRP       7092 01-28-94  12:27p
TBSTC    GRP       7392 10-26-94  11:08a
TEMPALM  GRP      10475 03-29-96   4:24p
TEMPRFL  GRP      13453 03-10-94   8:24a
TESTEXIT GRP       2574 06-05-95  10:54a
WELCOME  GRP       3450 02-15-95  11:29a
BOOT     GSP       3450 02-15-95  11:29a
HELP     GSP       1217 06-17-92   3:54a
HELP1    GSP       1217 03-29-94  10:40a
HELP2    GSP       1217 03-29-94  10:40a
HELP3    GSP       1217 03-29-94  10:40a
HELP4    GSP       1217 03-29-94  10:40a
HELP5    GSP       1217 03-29-94  10:40a
HELP6    GSP       1217 03-29-94  10:40a
RSSBOOT  GSP       3450 02-15-95  11:29a
HISTCOPY RPS       2712 08-12-94   3:24p
HISTFNAM RPS       1243 08-12-94   3:23p
SETLIMS  RPS       1423 04-03-96   3:17p
TEST     KML        103 05-31-95   2:20p
TEST     KMS        155 05-31-95   2:20p
RSSTEST  KML        103 05-31-95   2:20p
RSSTEST  KMS        155 05-31-95   2:20p

 Directory of C:\TEST\KEYMACS

.            <DIR>      12-07-93   2:15p
..           <DIR>      12-07-93   2:15p
TEST     KMS         32 04-23-93   5:16p
TEST     KML         55 04-24-93   1:40p


 Directory of C:\TEST\BATFILES

.            <DIR>      12-07-93   2:15p
..           <DIR>      12-07-93   2:15p
SETUP6   BAT         31 06-18-93   4:39p
```

```
 Directory of C:\TEST\LCL_HELP

.            <DIR>      12-07-93   2:15p
..           <DIR>      12-07-93   2:15p
STANDARD HLS       1981 01-14-94   4:12p
STANDARD HLP       1373 01-14-94   4:16p
HELP     GSP       1217 06-17-92   3:54a
HELP1    GSP       1217 01-14-94   4:16p
HELP2    GSP       1217 01-14-94   4:16p
HELP3    GSP       1217 01-14-94   4:16p
HELP4    GSP       1217 01-14-94   4:16p
HELP5    GSP       1217 01-14-94   4:16p
HELP6    GSP       1217 01-14-94   4:16p


 Directory of C:\TEST\LCL_KEY

.            <DIR>      12-07-93   2:15p
..           <DIR>      12-07-93   2:15p
TEST     KML         55 04-24-93   1:40p
TEST     KMS         32 04-23-93   5:16p
```

DACS STATION 6 FILE LISTINGS

"AUTOEXEC.BAT" on Station 6

```
@ECHO OFF
PROMPT $P$G
C:\NETROOM\XLOAD.EXE -SB001 -M23248 C:\EXPERT\MOUSE
goto %config%

:GEN354
PATH C:\DOS;C:\;C:\GENESIS;C:\NETROOM;C:\EXPERT
goto end

:GEN372
PATH C:\DOS;C:\;C:\GEN372;C:\NETROOM;C:\EXPERT
SHARE
goto end

:end

CD \TEST
RSS TEST
```

"CONFIG.SYS" on Station 6

```
[MENU]
menuitem=GEN354,Genesis system version 3.54
menuitem=GEN372,Genesis system version 3.72
menudefault=GEN372[,0]

[COMMON]
DEVICE=C:\NETROOM\RM386.EXE AUTO X=D900-DDFF
DEVICE=C:\NETROOM\SYSCLOAK.EXE
DEVICE=C:\NETROOM\XLOAD.SYS -O
BUFFERS=20,0
FILES=55
LASTDRIVE=E
FCBS=4,0
SHELL=C:\NETROOM\XLOAD.EXE -SDE01 -M57021 -E C:\DOS\COMMAND.COM C:\DOS\  /p
STACKS=0,0

[GEN354]

[GEN372]
```

"RSS.BAT" on Station 6

```
:_____________________________________________________________
:Filename:  RSS.BAT rev.2   05-25-95 sos/rwt
:
:Purpose:  This batch file creates RSS files for all RSS stations
:          Format of command:
:          rss %1
:             where %1 is the strategy name (typically TEST)
:
:_____________________________________________________________

ECHO
CLS
@ECHO OFF

IF "%1"=="test" GOTO TEST
IF "%1"=="TEST" GOTO TEST
f:\GEN372\RSS %1
GOTO END

:TEST
COPY f:\TEST\RSSTEST.DB f:\TEST\TEST.DB > XXX.XXX
COPY f:\TEST\RSSTEST.CI f:\TEST\TEST.CI > XXX.XXX
COPY f:\TEST\RSSTEST.XDB f:\TEST\TEST.XDB > XXX.XXX
COPY f:\TEST\RSSTEST.CA f:\TEST\TEST.CA > XXX.XXX
COPY f:\TEST\RSSTEST.CFG f:\TEST\TEST.CFG > XXX.XXX
COPY f:\TEST\RSSTEST.MDB f:\TEST\TEST.MDB > XXX.XXX
COPY WELCOME.GRP f:\GEN372\BOOT.GSP
COPY WELCOME.GRP f:\GEN372\RSSBOOT.GSP
COPY WELCOME.GRP RSSBOOT.GSP
f:\GEN372\RSS %1

:END
```

DACS STATION 7 DIRECTORY LISTINGS

```
 Volume in drive C has no label
 Volume Serial Number is 1C49-3EBC
                  167182336 bytes free


 Directory of C:\*.BAT,*.SYS

AUTOEXEC BAT       285 11-27-96  12:02p
RUNTIME  BAT      1357 04-26-96  10:26a
CONFIG   SYS       404 12-03-94   8:47a
RSS      BAT       927 05-25-95   2:42p


 Directory of C:\GEN372

.            <DIR>      12-03-94   9:02a
..           <DIR>      12-03-94   9:02a
REPORT   OPT     57547 05-01-90   3:21a
HPLSRJII PRP        36 09-17-90   3:32a
IBMPRO   PRP        36 09-17-90   3:32a
IBMPRO   PTR      2048 09-17-90   3:32a
HPLSRJII PTR      2560 09-17-90   3:32a
RRCOMBO  OPT     73309 04-28-93   3:26a
COUNTRY  FR          4 10-31-91   3:52a
COUNTRY  GR          5 10-31-91   3:52a
LOGO     DAT         9 02-24-94   3:27a
EPSON_MX PRP        36 02-24-94   3:27a
LISTMAIN GSP       363 02-24-94   3:27a
HCURS    GSP       415 02-24-94   3:27a
PLOT     GSP       465 02-24-94   3:27a
TREND    GSP       467 02-24-94   3:27a
STANDARD KML       511 02-24-94   3:27a
GNET_OLD KML       510 02-24-94   3:27a
IDBENTRY GSP       663 02-24-94   3:27a
HGREP    GSP       781 02-24-94   3:27a
HTBREP   GSP      1226 02-24-94   3:27a
EMPTY    GSP       845 02-24-94   3:27a
DISPSEC  GSP       980 02-24-94   3:27a
DISPSAV  GSP      1001 02-24-94   3:27a
EXIT     GSP      1021 02-24-94   3:27a
SORTMENU GSP      1128 02-24-94   3:27a
LISTMENU GSP      1657 02-24-94   3:27a
IDBSTG   GSP      1271 02-24-94   3:27a
IDBSTGO  GSP      1309 02-24-94   3:27a
HGLSTMEN GSP      1344 02-24-94   3:27a
IDBDOTB  GSP      1325 02-24-94   3:27a
FILEUTIL GSP      1296 02-24-94   3:27a
HISREP   GSP      1588 02-24-94   3:27a
```

HGREPMEN	GSP	1408	02-24-94	3:27a
IDBSIO	GSP	1480	02-24-94	3:27a
SPULNOT	GSP	1503	02-24-94	3:27a
IDBDINB	GSP	1574	02-24-94	3:27a
IDBDIOB	GSP	1644	02-24-94	3:27a
HGREPLST	GSP	1688	02-24-94	3:27a
ALMSUM	GSP	2244	02-24-94	3:27a
PASSWORD	GSP	1794	02-24-94	3:27a
HISTB	GSP	1862	02-24-94	3:27a
HDWDET	GSP	1954	02-24-94	3:27a
SORTIDBM	GSP	1976	02-24-94	3:27a
TRENDMEN	GSP	1990	02-24-94	3:27a
HISTMEN1	GSP	2029	02-24-94	3:27a
HISTLST	GSP	2038	02-24-94	3:27a
TRENDLST	GSP	2042	02-24-94	3:27a
EPSON_MX	PTR	2048	02-24-94	3:27a
STAT2	GSP	2053	02-24-94	3:27a
DOTDET	GSP	2064	02-24-94	3:27a
IDBAO	GSP	2070	02-24-94	3:27a
AOUTDET	GSP	2113	02-24-94	3:27a
KB	TBL	3392	02-24-94	3:27a
SEQ	GSP	2131	02-24-94	3:27a
SORTMAIN	GSP	2164	02-24-94	3:27a
FF1	GSP	2190	02-24-94	3:27a
STAT1	GSP	2271	02-24-94	3:27a
PULNOT	GSP	2322	02-24-94	3:27a
RSYSPERF	GSP	2325	02-24-94	3:27a
SYSPERF	GSP	2325	02-24-94	3:27a
REMPERF	GSP	2332	02-24-94	3:27a
FF2	GSP	2335	02-24-94	3:27a
SLOGIC	GSP	2360	02-24-94	3:27a
TIMERS	GSP	2363	02-24-94	3:27a
PLOTMEN	GSP	2405	02-24-94	3:27a
AMBDET	GSP	2449	02-24-94	3:27a
TUNE4	GSP	2432	02-24-94	3:27a
HISMON	GSP	2463	02-24-94	3:27a
CALC3	GSP	2502	02-24-94	3:27a
MATH2	GSP	2506	02-24-94	3:27a
SYSCONF	GSP	3011	02-24-94	3:27a
CTLDET3	GSP	2580	02-24-94	3:27a
DTIMDET	GSP	2588	02-24-94	3:27a
DGAP2	GSP	2593	02-24-94	3:27a
MATH	GSP	2602	02-24-94	3:27a
ALM	GSP	2649	02-24-94	3:27a
IDB	GSP	2674	02-24-94	3:27a
CNT	GSP	2703	02-24-94	3:27a
IDBAIN	GSP	2739	02-24-94	3:27a
IDBDOT	GSP	2809	02-24-94	3:27a
LOGIC	GSP	2843	02-24-94	3:27a
SYSCONF2	GSP	2891	02-24-94	3:27a

CTLDET	GSP	2896	02-24-94	3:27a
SWCH	GSP	2925	02-24-94	3:27a
TUNE3	GSP	2935	02-24-94	3:27a
DIGDET	GSP	2946	02-24-94	3:27a
AINDET	GSP	2995	02-24-94	3:27a
DGAP	GSP	3003	02-24-94	3:27a
SIM	GSP	3038	02-24-94	3:27a
CTLDET2	GSP	3044	02-24-94	3:27a
TD	GSP	3086	02-24-94	3:27a
USER2	GSP	3090	02-24-94	3:27a
IDBAIO	GSP	3137	02-24-94	3:27a
PAOUT	GSP	3143	02-24-94	3:27a
TUNE	GSP	3152	02-24-94	3:27a
TUNE2	GSP	3156	02-24-94	3:27a
TOT	GSP	3285	02-24-94	3:27a
TRENDREP	GSP	3400	02-24-94	3:27a
PAIO2	GSP	3453	02-24-94	3:27a
PAIO1	GSP	3453	02-24-94	3:27a
RAMP	GSP	3525	02-24-94	3:27a
XC	GSP	3543	02-24-94	3:27a
CALC	GSP	3633	02-24-94	3:27a
TPO	GSP	3658	02-24-94	3:27a
SEL	GSP	3718	02-24-94	3:27a
GENESIS	GSP	3776	02-24-94	3:27a
PAIO3	GSP	3888	02-24-94	3:27a
USER	GSP	3967	02-24-94	3:27a
DIR	GSP	4139	02-24-94	3:27a
PDOUT	GSP	4171	02-24-94	3:27a
DISPDIR	GSP	4234	02-24-94	3:27a
IDBDIN	GSP	4381	02-24-94	3:27a
CALC2	GSP	4408	02-24-94	3:27a
IDBDIO	GSP	4831	02-24-94	3:27a
PDIN	GSP	5661	02-24-94	3:27a
MOUSE	OPT	4118	02-24-94	3:27a
RSS	EXE	343638	02-24-94	3:27a
LOGO_NET	DAT	9	02-18-94	1:36a
GCOPY	BAT	33	01-15-92	3:52a
GENCMDR	BAT	46	01-15-92	3:52a
BOOT	BAT	79	02-18-94	5:01p
DOSNODE	PWD	226	02-18-94	5:01p
GENCFG	GNF	265	02-18-94	1:36a
GENCMDR	NDX	625	02-18-94	5:02p
GNETDIR	GSP	676	02-18-94	5:01p
GNFXR4	GSP	827	02-18-94	5:01p
GENCMDR2	GSP	860	02-18-94	5:02p
GENCEXIT	GSP	928	02-18-94	5:02p
GENCMDR	GSP	948	02-18-94	5:02p
DOSPWD	MNU	959	02-18-94	5:01p
GNFXEXIT	GSP	972	02-18-94	5:01p
GNFXR3	GSP	1024	02-18-94	5:01p

GNFXR2	GSP	1043	02-18-94	5:01p
GNFXR1	GSP	1053	02-18-94	5:01p
CONFMSMI	COM	1487	02-18-94	5:01p
GN_TIMER	SYS	1856	02-18-94	5:01p
GNETMON	GSP	2073	02-18-94	5:01p
GNETMON2	GSP	2045	02-18-94	5:01p
README	DOC	5912	01-15-92	3:52a
GENCMDR	DOC	5912	02-18-94	5:02p
CONF104	FNT	7168	02-18-94	5:01p
GENCMDR	MNU	7682	02-18-94	5:02p
GN_TIMER	EXE	13851	02-18-94	5:01p
PS110RST	EXE	15259	02-18-94	5:01p
EGA	DEV	13450	02-18-94	5:01p
GENMON	EXE	16487	02-18-94	5:01p
SHOW_OPT	EXE	15931	02-18-94	5:01p
GENCFG	MNU	33934	02-18-94	5:01p
GENCMDR	HLP	23280	02-18-94	5:02p
UNPACK	EXE	29378	02-18-94	1:36a
GENCFGB	MNU	24190	01-15-92	3:52a
GN_INTF	OPT	32731	02-18-94	5:02p
GN_COMM	OPT	35453	02-18-94	5:02p
GENCMDR	OPT	43487	02-18-94	5:02p
GENCOMP	EXE	44069	02-18-94	5:01p
GN_FTP	OPT	52219	02-18-94	5:02p
GN_DRVR	OPT	67307	02-18-94	5:02p
GN_INTFB	OPT	56305	01-15-92	3:52a
GEN_NET	OPT	61551	01-15-92	3:52a
GN_CA	OPT	66311	01-15-92	3:52a
DOSPWD	EXE	113849	02-18-94	5:01p
GN_DCA	OPT	77441	01-15-92	3:52a
GN_DB	OPT	86329	01-15-92	3:52a
GN_CB	OPT	91089	01-15-92	3:52a
GENCONF	EXE	91957	02-18-94	5:01p
GN_DFA	OPT	94991	01-15-92	3:52a
GN_CFA	OPT	99255	01-15-92	3:52a
GN_DCB	OPT	102235	01-15-92	3:52a
GN_DCFA	OPT	110897	01-15-92	3:52a
GN_DFB	OPT	119273	01-15-92	3:52a
GN_CFB	OPT	124033	01-15-92	3:52a
GN_DCFB	OPT	135675	01-15-92	3:52a
GCOPY	EXE	145747	02-18-94	5:01p
BCOPY	EXE	165287	01-15-92	3:52a
GENCMDR	EXE	260257	02-18-94	5:02p
BENCMDR	EXE	279813	01-15-92	3:52a
MODICON	DES	2418	04-03-92	3:50a
MODICON	DRV	7038	04-03-92	3:50a
AF5000	DES	448	05-06-92	3:53a
AF5000	DRV	35386	09-16-94	3:53a
INCR	RPS	46	06-08-92	2:43p
OVERLIM	RPS	141	06-09-92	10:30a

NEXT	RPS	115	06-10-92	7:44p
PREV	RPS	108	06-10-92	7:47p
DISMOVE	RPS	21	06-11-92	3:26p
RESMOVE	RPS	32	06-12-92	7:47a
HELP	GSP	1217	06-17-92	3:54a
CTREPT	RPS	7628	07-29-92	9:54a
EXTRA	KMS	62	07-29-92	10:33a
EXTRA	KML	84	07-29-92	10:33a
BACKUP1	BAT	121	07-29-92	3:35p
UNLOAD1	BAT	96	07-29-92	3:37p
BACKUP2	BAT	57	07-29-92	3:38p
TEST	GIF	24682	06-05-93	12:23p
GENCFG	NBC	1015	01-26-93	1:29a
CONFIG	SYS	160	01-26-93	2:09p
AUTOEXEC	BAT	159	01-31-93	4:31p
RSSBOOT	GSP	3450	02-15-95	11:29a
BOOT	GSP	3450	02-15-95	11:29a
HELP4	GSP	1217	02-16-95	3:01p
HELP6	GSP	1217	03-29-94	10:40a
HELP3	GSP	1217	02-16-95	3:01p
HELP1	GSP	1217	02-16-95	3:01p
HELP2	GSP	1217	02-16-95	3:01p
HELP5	GSP	1217	03-29-94	10:40a
GENNET	NDB	171	05-09-95	10:33a
GENCFG	NCF	276	05-09-95	10:33a
GENNET	DNL	442	05-09-95	10:33a
GENCMDR0	HLP	23280	01-15-92	3:52a
STANDARD	HLP	1373	01-14-94	4:16p
STANDARD	HLS	1981	01-14-94	4:12p
AF5000	AIF	17103	09-16-94	3:53a
AF5000	DOC	31716	05-12-92	3:53a
AF5000	MNU	912	09-16-94	3:53a
MODPLUS	DES	448	02-09-94	3:38a
MODPLUS	DRV	19750	09-29-94	4:59p
MODPLUS	MNU	12100	08-24-94	3:38a
MODPLUS	DOC	23524	03-29-93	3:38a
MODPLUS	335	579	06-03-93	3:36a
MODPLUS	336	747	06-03-93	3:36a
GNETMON3	GSP	1916	02-18-94	5:01p
SHADOW	GSP	1912	02-18-94	5:01p
NETALMS	GSP	1825	02-18-94	5:01p
DANISH	FNT	1536	02-24-94	3:27a
EXTASCII	FNT	1536	02-24-94	3:27a
ALARM	GSP	2219	02-24-94	3:27a
BLANK	GSP	243	02-24-94	3:27a
EVENT	GSP	2212	02-24-94	3:27a
RRCOMBO	GSP	2450	02-24-94	3:27a
USER2_2	GSP	3090	02-24-94	3:27a
USER2_3	GSP	3090	02-24-94	3:27a
USER2_4	GSP	3090	02-24-94	3:27a

USER2_5	GSP	3090	02-24-94	3:27a
USER2_6	GSP	3090	02-24-94	3:27a
USER2_7	GSP	3090	02-24-94	3:27a
USER3	GSP	1855	02-24-94	3:27a
USER3_2	GSP	1855	02-24-94	3:27a
USER3_3	GSP	1855	02-24-94	3:27a
USER3_4	GSP	1855	02-24-94	3:27a
USER3_5	GSP	1855	02-24-94	3:27a
USER3_6	GSP	1855	02-24-94	3:27a
USER3_7	GSP	1855	02-24-94	3:27a
USER_2	GSP	3967	02-24-94	3:27a
USER_3	GSP	3967	02-24-94	3:27a
USER_4	GSP	3967	02-24-94	3:27a
USER_5	GSP	3967	02-24-94	3:27a
USER_6	GSP	3967	02-24-94	3:27a
USER_7	GSP	3967	02-24-94	3:27a
EXPORT	OPT	18307	02-18-94	5:01p
MODPLUS	338	60	09-23-94	3:38a
RUNTIME	EXE	401424	02-24-94	3:27a
HPLSR4	PRP	36	03-31-94	5:12p
HPLSR4	PTR	2560	09-17-90	3:32a
HELP	HLS	1481	02-16-95	3:01p
HELP	HLP	998	02-16-95	3:01p

Directory of C:\TEST

.		<DIR>	10-05-93	9:56a
..		<DIR>	10-05-93	9:56a
LCL_KEY		<DIR>	12-14-93	11:56a
KEYMACS		<DIR>	10-05-93	9:58a
BATFILES		<DIR>	10-05-93	9:58a
TEST	CA	19228	09-04-96	1:15p
TEST	CFG	1752	09-04-96	1:15p
TEST	CI	20560	09-04-96	1:15p
TEST	DB	134274	09-04-96	1:15p
TEST	MDB	12	09-04-96	1:15p
TEST	XDB	1892	09-04-96	1:15p
RSSTEST	CA	19228	09-04-96	1:15p
RSSTEST	CFG	1752	09-04-96	1:15p
RSSTEST	CI	20560	09-04-96	1:15p
RSSTEST	DB	134274	09-04-96	1:15p
RSSTEST	MDB	12	09-04-96	1:15p
RSSTEST	XDB	1892	09-04-96	1:15p
ROLLOVER	GRP	13928	04-04-96	4:51p
ABRTENAB	GRP	44959	04-11-96	11:13a
ABRTNAME	GRP	28563	04-11-96	11:14a
ASMAIN	GRP	2827	07-28-94	5:47p
CSMAIN	GRP	9058	04-24-96	2:33p
DACS	GRP	12755	03-11-94	10:19a

```
GASSUM    GRP        9140 04-08-96   10:04a
HVTALARM  GRP       10289 04-24-96    3:03p
IOSTATUS  GRP       13096 06-08-95    2:04p
MANABRT   GRP        4199 04-16-96    8:22a
MAP       GRP        4087 03-22-95    1:55p
MININ1    GRP       15878 04-12-96   12:28p
MININ2    GRP       12135 09-04-96    1:04p
MIT17B    GRP       13092 04-03-96   12:38p
MIT17C    GRP       11621 03-10-94    8:14a
MSMAIN    GRP        5643 04-24-96    2:53p
NEW72HR   GRP        5626 04-05-96    9:43a
OLD72HR   GRP        5631 04-05-96    9:46a
PUMP      GRP        9579 04-04-96    5:17p
PUMPALRM  GRP        8695 02-15-95   11:27a
PUMPOPS   GRP        5126 03-29-96    4:15p
ABRTCHEK  GRP       10487 04-24-96    3:08p
STRNALM   GRP       11330 04-04-96    5:21p
SUMMARY   GRP       11579 03-03-95    9:53a
SY101-1   GRP        7092 01-28-94   12:27p
TBSTC     GRP        7392 10-26-94   11:08a
TEMPALM   GRP       10475 03-29-96    4:24p
TEMPRFL   GRP       13453 03-10-94    8:24a
TESTEXIT  GRP        2574 06-05-95   10:54a
WELCOME   GRP        3450 02-15-95   11:29a
BOOT      GSP        3450 02-15-95   11:29a
HELP      GSP        1217 06-17-92    3:54a
HELP1     GSP        1217 03-29-94   10:40a
HELP2     GSP        1217 03-29-94   10:40a
HELP3     GSP        1217 03-29-94   10:40a
HELP4     GSP        1217 03-29-94   10:40a
HELP5     GSP        1217 03-29-94   10:40a
HELP6     GSP        1217 03-29-94   10:40a
RSSBOOT   GSP        3450 02-15-95   11:29a
HISTCOPY  RPS        2712 08-12-94    3:24p
HISTFNAM  RPS        1243 08-12-94    3:23p
SETLIMS   RPS        1423 04-03-96    3:17p
TEST      KML         103 05-31-95    2:20p
TEST      KMS         155 05-31-95    2:20p
RSSTEST   KML         103 05-31-95    2:20p
RSSTEST   KMS         155 05-31-95    2:20p

 Directory of C:\TEST\LCL_KEY

.             <DIR>      12-14-93   11:56a
..            <DIR>      12-14-93   11:56a
TEST      KML            55 04-24-93    1:40p
TEST      KMS            32 04-23-93    5:16p
```

```
 Directory of C:\TEST\KEYMACS

.            <DIR>      10-05-93   9:58a
..           <DIR>      10-05-93   9:58a
HELP    GSP      1217 06-17-92   3:54a
TEST    KMS        32 04-23-93   5:16p
TEST    KML        55 04-24-93   1:40p
HELP1   GSP      1217 04-24-93   1:43p
HELP2   GSP      1217 04-24-93   1:43p
HELP3   GSP      1217 04-24-93   1:43p
HELP4   GSP      1217 04-24-93   1:43p
HELP5   GSP      1217 04-24-93   1:43p
HELP6   GSP      1217 04-24-93   1:43p

 Directory of C:\TEST\BATFILES

.            <DIR>      10-05-93   9:58a
..           <DIR>      10-05-93   9:58a
RSS     BAT       348 06-16-93   4:11p
RUNTIME BAT       405 02-09-94   8:51a
SETUP7  BAT       123 06-19-93   5:08p
```

DACS STATION 7 FILE LISTINGS

"AUTOEXEC.BAT" on Station 7

```
@ECHO OFF
PROMPT $p$g
C:\EXPERT\MOUSE
goto %config%

:GEN354
PATH C:\DOS;C:\;C:\RUNTIME;C:\GENESIS;C:\NETROOM
goto end

:GEN372
REM PATH C:\DOS;C:\;C:\RUN372;C:\GEN372;C:\NETROOM
PATH C:\DOS;C:\;C:\GEN372;C:\NETROOM;C:\EXPERT
SHARE
goto end

:end

CD \TEST
RSS TEST
```

"CONFIG.SYS" on Station 7

```
[MENU]
menuitem=GEN354,Genesis system version 3.54
menuitem=GEN372,Genesis system version 3.72
menudefault=GEN372[,0]

[COMMON]
DEVICE=C:\NETROOM\RM386.EXE AUTO X=D900-DDFF
DEVICE=C:\NETROOM\SYSCLOAK.EXE
DEVICE=C:\NETROOM\XLOAD.SYS -O
BUFFERS=20,0
FILES=55
LASTDRIVE=E
FCBS=4,0
SHELL=C:\NETROOM\XLOAD.EXE -SDE01 -M57021 -E C:\DOS\COMMAND.COM C:\DOS\  /p
STACKS=0,0

[GEN354]

[GEN372]
```

"RUNTIME.BAT" on Station 7

```
:_____________________________________________________
:Filename:  RUNTIME.BAT rev.1   05-25-95 sos/rwt
:
:Modified:  04-26-96 sos/rwt to disable runtime capability.
:
:Purpose:  This batch file creates runtime files
:          for TEST and MOTOR strategies on Station 7.
:          Format of command:
:          runtime %1
:             where %1 is the strategy name to be used runtime
:
:_____________________________________________________

ECHO
CLS
@ECHO OFF

IF "%1"=="test" GOTO TEST
IF "%1"=="TEST" GOTO TEST
IF "%1"=="motor" GOTO MOTOR
IF "%1"=="MOTOR" GOTO MOTOR

REM will run other strategies
C:\GEN372\RUNTIME %1
GOTO END

:TEST
REM COPY C:\TEST\RUNTEST.DB C:\TEST\TEST.DB > XXX.XXX
REM COPY C:\TEST\RUNTEST.MDB C:\TEST\TEST.MDB > XXX.XXX
REM COPY C:\TEST\RUNTEST.XDB C:\TEST\TEST.XDB > XXX.XXX
REM COPY C:\TEST\RUNTEST.CI C:\TEST\TEST.CI > XXX.XXX
REM COPY C:\TEST\RUNTEST.CA C:\TEST\TEST.CA > XXX.XXX
REM COPY C:\TEST\RUNTEST.CFG C:\TEST\TEST.CFG > XXX.XXX
REM COPY C:\TEST\WELCOME.GRP C:\GEN372\BOOT.GSP
REM C:\TEST\ALMRENAM
REM CALL C:\ALMCOPY
REM DEL ALARM*.*
REM C:\GEN372\RUNTIME %1
ECHO RUNTIME IS NOT AVAILABLE ON STATION 7
GOTO END

:MOTOR
REM COPY C:\MOTOR\MAINMENU.GRP C:\GEN372\BOOT.GSP
REM C:\GEN372\RUNTIME %1
ECHO RUNTIME IS NOT AVAILABLE ON STATION 7
:END
```

"RSS.BAT" on Station 7

```
:_____________________________________________________
:Filename:  RSS.BAT rev.2   05-25-95 sos/rwt
:
:Purpose:  This batch file creates RSS files for all RSS stations
:          Format of command:
:          rss %1
:             where %1 is the strategy name (typically TEST)
:
:_____________________________________________________

ECHO
CLS
@ECHO OFF

IF "%1"=="test" GOTO TEST
IF "%1"=="TEST" GOTO TEST
f:\GEN372\RSS %1
GOTO END

:TEST
COPY f:\TEST\RSSTEST.DB f:\TEST\TEST.DB > XXX.XXX
COPY f:\TEST\RSSTEST.CI f:\TEST\TEST.CI > XXX.XXX
COPY f:\TEST\RSSTEST.XDB f:\TEST\TEST.XDB > XXX.XXX
COPY f:\TEST\RSSTEST.CA f:\TEST\TEST.CA > XXX.XXX
COPY f:\TEST\RSSTEST.CFG f:\TEST\TEST.CFG > XXX.XXX
COPY f:\TEST\RSSTEST.MDB f:\TEST\TEST.MDB > XXX.XXX
COPY WELCOME.GRP f:\GEN372\BOOT.GSP
COPY WELCOME.GRP f:\GEN372\RSSBOOT.GSP
COPY WELCOME.GRP RSSBOOT.GSP
f:\GEN372\RSS %1

:END
```

DACS STATION 8 DIRECTORY LISTINGS

```
 Volume in drive C has no label
 Volume Serial Number is 1C9D-4760
                 186703872 bytes free


 Directory of C:\*.BAT,*.SYS

AUTOEXEC BAT       190 11-28-94   1:03p
CONFIG   SYS       390 12-13-94   8:41a
RUNTIME  BAT      1155 05-25-95   2:12p


 Directory of C:\GEN372

.            <DIR>      08-01-94   5:37p
..           <DIR>      08-01-94   5:37p
LOGO_NET DAT         9 02-18-94   1:36a
UNPACK   EXE     29378 02-18-94   1:36a
GENNET   DNL       442 08-03-94  12:50p
CONF104  FNT      7168 02-18-94   5:01p
EXTASCII FNT      1536 02-24-94   3:27a
CONFIG   PRM       407 08-01-94   5:49p
EGA      DEV     13450 02-18-94   5:01p
XLATE    EXE     78445 02-24-94   3:27a
OLDBOOT  GSP      4245 12-06-93  10:16a
GENESIS  GSP      1109 02-24-94   3:27a
LOGO     DAT         9 02-24-94   3:27a
EPSON_MX PRP        36 02-24-94   3:27a
EPSON_MX PTR      2048 02-24-94   3:27a
GENESIS  BAT        19 08-01-94   5:49p
CONFMSMI COM      1487 02-18-94   5:01p
ICONS    ICN     66267 02-24-94   3:27a
README   354      7611 06-17-92   3:54a
ELECTRIC SMB      2837 02-24-94   3:27a
HEATEXCH SMB       846 02-24-94   3:27a
LOGIC    SMB      1353 02-24-94   3:27a
MISC     SMB      1680 02-24-94   3:27a
MOTORS   SMB       396 02-24-94   3:27a
TANKS    SMB      1890 02-24-94   3:27a
VALVES   SMB      2364 02-24-94   3:27a
KEYHELP  EXE     39261 02-24-94   3:27a
STANDARD HLS      1908 02-24-94   3:27a
KEYMAC   EXE     34201 02-24-94   3:27a
MOUSE    OPT      4118 02-24-94   3:27a
STATECMP EXE     28561 02-24-94   3:27a
STATES   SFL       642 02-24-94   3:27a
STATES   SFS       913 02-24-94   3:27a
```

BUTTONS	SMB	1320	02-24-94	3:27a
CART	SMB	245	02-24-94	3:27a
CIRCUITS	SMB	44898	02-24-94	3:27a
FACEPLAT	SMB	15558	02-24-94	3:27a
MACHINE	SMB	1841	02-24-94	3:27a
PRESS	SMB	1755	02-24-94	3:27a
VATS	SMB	3741	02-24-94	3:27a
AUTOCAD	DOC	9984	10-31-91	1:31a
DXFTOGRP	EXE	58777	10-31-91	1:31a
GRPTODXF	EXE	39665	10-31-91	1:31a
SPACE	GRP	33622	10-31-91	1:31a
STAIR	DXF	46932	10-31-91	1:31a
PUMP	GRP	12703	10-31-91	1:31a
STAIR	GRP	20625	10-31-91	1:31a
SHUTTLE	SMB	48419	10-31-91	1:31a
CONFIG	EXE	637000	02-24-94	3:27a
METACONF	EXE	69599	02-24-94	3:27a
METACONF	MNU	3859	02-24-94	3:27a
STANDARD	CFN	18899	02-24-94	3:27a
ANALOG	MNU	3377	02-24-94	3:27a
CONTROL	MNU	10065	02-24-94	3:27a
DIGITAL	MNU	4434	02-24-94	3:27a
DYNAMIC	MNU	14019	02-24-94	3:27a
GRAPHIC	MNU	2627	02-24-94	3:27a
HIST	MNU	15746	02-24-94	3:27a
LIBRARY	MNU	3471	02-24-94	3:27a
MACROS	MNU	2282	02-24-94	3:27a
MEMSYS	MNU	863	02-24-94	3:27a
MNEMONIC	MNU	8335	02-24-94	3:27a
STRING	MNU	1027	02-24-94	3:27a
SUPMAC	MNU	2461	02-24-94	3:27a
SYSCONF	MNU	15280	02-24-94	3:27a
RUNTIME	EXE	401424	02-24-94	3:27a
AINDET	GSP	2995	02-24-94	3:27a
ALM	GSP	2649	02-24-94	3:27a
ALMSUM	GSP	2244	02-24-94	3:27a
AMBDET	GSP	2449	02-24-94	3:27a
AOUTDET	GSP	2113	02-24-94	3:27a
CALC	GSP	3633	02-24-94	3:27a
CALC2	GSP	4408	02-24-94	3:27a
CALC3	GSP	2502	02-24-94	3:27a
CNT	GSP	2703	02-24-94	3:27a
CTLDET	GSP	2896	02-24-94	3:27a
CTLDET2	GSP	3044	02-24-94	3:27a
CTLDET3	GSP	2580	02-24-94	3:27a
DGAP	GSP	3003	02-24-94	3:27a
DGAP2	GSP	2593	02-24-94	3:27a
DIGDET	GSP	2946	02-24-94	3:27a
DIR	GSP	4139	02-24-94	3:27a
DISPDIR	GSP	4234	02-24-94	3:27a

DISPSAV	GSP	1001	02-24-94	3:27a
DISPSEC	GSP	980	02-24-94	3:27a
DOTDET	GSP	2064	02-24-94	3:27a
DTIMDET	GSP	2588	02-24-94	3:27a
EMPTY	GSP	845	02-24-94	3:27a
EXIT	GSP	1021	02-24-94	3:27a
FF1	GSP	2190	02-24-94	3:27a
FF2	GSP	2335	02-24-94	3:27a
FILEUTIL	GSP	1296	02-24-94	3:27a
HCURS	GSP	415	02-24-94	3:27a
HDWDET	GSP	1954	02-24-94	3:27a
HELP	GSP	1217	02-24-94	3:27a
HELP1	GSP	1124	02-24-94	3:27a
HELP2	GSP	1217	02-24-94	3:27a
HELP3	GSP	1217	02-24-94	3:27a
HELP4	GSP	1214	02-24-94	3:27a
HELP5	GSP	1210	02-24-94	3:27a
HELP6	GSP	1156	02-24-94	3:27a
HGLSTMEN	GSP	1344	02-24-94	3:27a
HGREP	GSP	781	02-24-94	3:27a
HGREPLST	GSP	1688	02-24-94	3:27a
HGREPMEN	GSP	1408	02-24-94	3:27a
HISMON	GSP	2463	02-24-94	3:27a
HISREP	GSP	1588	02-24-94	3:27a
HISTB	GSP	1862	02-24-94	3:27a
HISTLST	GSP	2038	02-24-94	3:27a
HISTMEN1	GSP	2029	02-24-94	3:27a
HTBREP	GSP	1226	02-24-94	3:27a
IDB	GSP	2674	02-24-94	3:27a
IDBAIN	GSP	2739	02-24-94	3:27a
IDBAIO	GSP	3137	02-24-94	3:27a
IDBAO	GSP	2070	02-24-94	3:27a
IDBDIN	GSP	4381	02-24-94	3:27a
IDBDINB	GSP	1574	02-24-94	3:27a
IDBDIO	GSP	4831	02-24-94	3:27a
IDBDIOB	GSP	1644	02-24-94	3:27a
IDBDOT	GSP	2809	02-24-94	3:27a
IDBDOTB	GSP	1325	02-24-94	3:27a
IDBENTRY	GSP	663	02-24-94	3:27a
IDBSIO	GSP	1480	02-24-94	3:27a
IDBSTG	GSP	1271	02-24-94	3:27a
IDBSTGO	GSP	1309	02-24-94	3:27a
LISTMAIN	GSP	363	02-24-94	3:27a
LISTMENU	GSP	1657	02-24-94	3:27a
LOGIC	GSP	2843	02-24-94	3:27a
MATH	GSP	2602	02-24-94	3:27a
MATH2	GSP	2506	02-24-94	3:27a
PAIO1	GSP	3453	02-24-94	3:27a
PAIO2	GSP	3453	02-24-94	3:27a
PAIO3	GSP	3888	02-24-94	3:27a

PAOUT	GSP	3143	02-24-94	3:27a
PASSWORD	GSP	1794	02-24-94	3:27a
PDIN	GSP	5661	02-24-94	3:27a
PDOUT	GSP	4171	02-24-94	3:27a
PLOT	GSP	465	02-24-94	3:27a
PLOTMEN	GSP	2405	02-24-94	3:27a
PULNOT	GSP	2322	02-24-94	3:27a
RAMP	GSP	3525	02-24-94	3:27a
SEL	GSP	3718	02-24-94	3:27a
SEQ	GSP	2131	02-24-94	3:27a
SIM	GSP	3038	02-24-94	3:27a
SLOGIC	GSP	2360	02-24-94	3:27a
SORTIDBM	GSP	1976	02-24-94	3:27a
SORTMAIN	GSP	2164	02-24-94	3:27a
SORTMENU	GSP	1128	02-24-94	3:27a
SPULNOT	GSP	1503	02-24-94	3:27a
STAT1	GSP	2271	02-24-94	3:27a
STAT2	GSP	2053	02-24-94	3:27a
SWCH	GSP	2925	02-24-94	3:27a
SYSCONF	GSP	3011	02-24-94	3:27a
SYSCONF2	GSP	2891	02-24-94	3:27a
SYSPERF	GSP	2325	02-24-94	3:27a
TD	GSP	3086	02-24-94	3:27a
TIMERS	GSP	2363	02-24-94	3:27a
TOT	GSP	3285	02-24-94	3:27a
TPO	GSP	3658	02-24-94	3:27a
TREND	GSP	467	02-24-94	3:27a
TRENDLST	GSP	2042	02-24-94	3:27a
TRENDMEN	GSP	1990	02-24-94	3:27a
TRENDREP	GSP	3400	02-24-94	3:27a
TUNE	GSP	3152	02-24-94	3:27a
TUNE2	GSP	3156	02-24-94	3:27a
TUNE3	GSP	2935	02-24-94	3:27a
TUNE4	GSP	2432	02-24-94	3:27a
USER	GSP	3967	02-24-94	3:27a
USER2	GSP	3090	02-24-94	3:27a
XC	GSP	3543	02-24-94	3:27a
KB	TBL	3392	02-24-94	3:27a
STANDARD	KML	544	02-24-94	3:27a
GNET_OLD	KML	510	02-24-94	3:27a
GENCFG	GNF	265	02-18-94	1:36a
GENCFG	NCF	276	08-03-94	12:50p
GENNET	NDB	171	08-03-94	12:50p
GENCMDR	DOC	5912	02-18-94	5:02p
HPLSRJII	PRP	36	09-17-90	3:32a
GN_COMM	OPT	35453	02-18-94	5:02p
DOSPWD	EXE	113849	02-18-94	5:01p
GEN_NET	OPT	61551	01-15-92	3:52a
GN_DFA	OPT	94991	01-15-92	3:52a
GN_CA	OPT	66311	01-15-92	3:52a

GN_CFA	OPT	99255	01-15-92	3:52a
GN_DCA	OPT	77441	01-15-92	3:52a
GN_DCFA	OPT	110897	01-15-92	3:52a
GN_DRVR	OPT	67307	02-18-94	5:02p
GN_FTP	OPT	52219	02-18-94	5:02p
GN_INTF	OPT	32731	02-18-94	5:02p
GCOPY	EXE	145747	02-18-94	5:01p
GENCOMP	EXE	44069	02-18-94	5:01p
GENCONF	EXE	91957	02-18-94	5:01p
GENMON	EXE	16487	02-18-94	5:01p
GN_TIMER	EXE	13851	02-18-94	5:01p
PSI10RST	EXE	15259	02-18-94	5:01p
SHOW_OPT	EXE	15931	02-18-94	5:01p
DOSPWD	MNU	959	02-18-94	5:01p
GNETDIR	GSP	676	02-18-94	5:01p
GNETMON	GSP	2073	02-18-94	5:01p
GNETMON2	GSP	2045	02-18-94	5:01p
GNFXEXIT	GSP	972	02-18-94	5:01p
GNFXR1	GSP	1053	02-18-94	5:01p
GNFXR2	GSP	1043	02-18-94	5:01p
GNFXR3	GSP	1024	02-18-94	5:01p
GNFXR4	GSP	827	02-18-94	5:01p
GENCFG	MNU	33934	02-18-94	5:01p
GN_TIMER	SYS	1856	02-18-94	5:01p
DOSNODE	PWD	226	02-18-94	5:01p
BOOT	BAT	79	02-18-94	5:01p
GENCMDR	EXE	260257	02-18-94	5:02p
GENCEXIT	GSP	928	02-18-94	5:02p
GENCMDR	GSP	948	02-18-94	5:02p
GENCMDR2	GSP	860	02-18-94	5:02p
GENCMDR	MNU	7682	02-18-94	5:02p
GENCMDR	OPT	43487	02-18-94	5:02p
GENCMDR	HLP	23280	02-18-94	5:02p
GENCMDR	NDX	625	02-18-94	5:02p
HPLSRJII	PTR	2560	09-17-90	3:32a
RRCOMBO	OPT	73309	04-28-93	3:26a
AF5000	DES	448	05-06-92	3:53a
AF5000	DOC	31716	05-12-92	3:53a
AF5000	DRV	35386	09-16-94	3:53a
AF5000	AIF	17103	09-16-94	3:53a
AF5000	MNU	912	09-16-94	3:53a
TEST	GIF	24682	06-05-93	12:23p
STANDARD	HLP	1373	01-14-94	4:16p
MODICON	DES	2418	04-03-92	3:50a
RSSBOOT	GSP	4245	12-06-93	10:16a
MODICON	350	282	04-03-92	3:50a
MODICON	DRV	7038	04-03-92	3:50a
MODICON	MNU	14762	04-03-92	3:50a
BOOT	GSP	2505	02-15-95	11:07a
MODPLUS	DES	448	02-09-94	3:38a

MODPLUS	DRV	19750	09-29-94	4:59p
DANISH	FNT	1536	02-24-94	3:27a
CONFIG	SYS	26	02-24-94	3:27a
README	372	8135	02-24-94	3:27a
ALARM	GSP	2219	02-24-94	3:27a
BLANK	GSP	243	02-24-94	3:27a
EVENT	GSP	2212	02-24-94	3:27a
RRCOMBO	GSP	2450	02-24-94	3:27a
USER2_2	GSP	3090	02-24-94	3:27a
USER2_3	GSP	3090	02-24-94	3:27a
USER2_4	GSP	3090	02-24-94	3:27a
USER2_5	GSP	3090	02-24-94	3:27a
USER2_6	GSP	3090	02-24-94	3:27a
USER2_7	GSP	3090	02-24-94	3:27a
USER3	GSP	1855	02-24-94	3:27a
USER3_2	GSP	1855	02-24-94	3:27a
USER3_3	GSP	1855	02-24-94	3:27a
USER3_4	GSP	1855	02-24-94	3:27a
USER3_5	GSP	1855	02-24-94	3:27a
USER3_6	GSP	1855	02-24-94	3:27a
USER3_7	GSP	1855	02-24-94	3:27a
USER_2	GSP	3967	02-24-94	3:27a
USER_3	GSP	3967	02-24-94	3:27a
USER_4	GSP	3967	02-24-94	3:27a
USER_5	GSP	3967	02-24-94	3:27a
USER_6	GSP	3967	02-24-94	3:27a
USER_7	GSP	3967	02-24-94	3:27a
MODPLUS	MNU	12100	08-24-94	3:38a
MODPLUS	DOC	23524	03-29-93	3:38a
MODPLUS	335	579	06-03-93	3:36a
MODPLUS	336	747	06-03-93	3:36a
GNETMON3	GSP	1916	02-18-94	5:01p
SHADOW	GSP	1912	02-18-94	5:01p
NETALMS	GSP	1825	02-18-94	5:01p
EXPORT	OPT	18307	02-18-94	5:01p
HPLSR4	PRP	36	03-31-94	5:12p
MODPLUS	338	60	09-23-94	3:38a
HPLSR4	PTR	2560	09-17-90	3:32a
AF5000	RWT	35386	09-16-94	3:53a

Directory of C:\MOTOR

.		<DIR>	06-01-94	3:45p
..		<DIR>	06-01-94	3:45p
USERSET	TDF	2886	05-04-95	9:47a
NEWSET	TDF	2886	05-04-95	9:47a
KMLEXTRA	OLD	278	06-10-94	10:29a
KMSEXTRA	OLD	324	06-10-94	10:29a
MOTOR	DB	50363	11-06-96	9:05a
MOTOR	CI	13956	11-06-96	9:05a
MOTOR	XDB	676	11-06-96	9:05a
MOTOR	CA	19228	11-06-96	9:05a
MOTOR	CFG	1752	01-02-97	8:28p
MOTOR	MDB	10070	11-06-96	9:05a
MAINMENU	GRP	2505	02-15-95	11:07a
PBENPROB	GRP	3298	02-15-95	11:07a
POSPROB	GRP	2963	02-15-95	11:07a
PSPROB	GRP	3056	02-15-95	11:08a
PUMPRUN	GRP	9256	02-15-95	11:10a
SETPROB	GRP	2600	02-15-95	11:10a
TESTSET	GRP	6255	02-26-95	10:01a
EXIT	GRP	2674	02-15-95	11:06a
DMPROB	GRP	2921	02-15-95	11:06a
CLRMESS	RPS	38	05-26-94	1:34p
DELETE	RPS	2028	08-23-94	4:12p
ILOAD	RPS	1088	08-24-94	10:33a
NEXT	RPS	2845	08-24-94	10:33a
PBSETVAL	RPS	4857	08-24-94	10:06a
PREV	RPS	2263	08-24-94	10:34a
SAVE	RPS	5687	08-23-94	4:18p
SETOLD	RPS	226	06-13-94	10:33a
ULOAD	RPS	1260	08-24-94	10:34a
INITENAB	RPS	1260	07-12-95	9:57a
STATES	SFS	4231	12-06-93	10:07a
STATES	SFL	2771	12-06-93	10:08a
BOOT	GSP	2505	02-15-95	11:07a
EXTRA	KMS	659	10-16-95	10:23a
EXTRA	KML	246	10-17-95	7:36a
MOTOR	NEW	50363	01-02-97	8:28p

DACS STATION 8 FILE LISTINGS

"AUTOEXEC.BAT" on Station 8

```
@ECHO OFF
PROMPT $p$g
C:\EXPERT\MOUSE
goto %config%

:GEN354
PATH C:\DOS;C:\;C:\GENESIS;C:\NETROOM
goto end

:GEN372
PATH C:\DOS;C:\;C:\GEN372;C:\NETROOM
SHARE
goto end

:end
```

"CONFIG.SYS" on Station 8

```
[MENU]
menuitem=GEN354,Genesis version 3.54
menuitem=GEN372,Genesis version 3.72
menudefault=GEN372[,0]

[COMMON]
DEVICE=C:\NETROOM\RM386.EXE AUTO X=D900-DDFF
DEVICE=C:\NETROOM\SYSCLOAK.EXE
DEVICE=C:\NETROOM\XLOAD.SYS -O
BUFFERS=20,0
FILES=55
LASTDRIVE=E
FCBS=4,0
SHELL=C:\NETROOM\XLOAD.EXE -SDE01 -M57021 -E C:\DOS\COMMAND.COM C:\DOS\  /p
STACKS=0,0

[GEN354]

[GEN372]
```

"RUNTIME.BAT" on Station 8

```
:_____________________________________________________
:Filename:  RUNTIME.BAT rev.1   05-25-95 sos/rwt
:
:Purpose:  This batch file creates runtime files
:          for TEST and MOTOR strategies on Station 7.
:          Format of command:
:          runtime %1
:             where %1 is the strategy name to be used runtime
:
:_____________________________________________________

ECHO
CLS
@ECHO OFF

IF "%1"=="test" GOTO TEST
IF "%1"=="TEST" GOTO TEST
IF "%1"=="motor" GOTO MOTOR
IF "%1"=="MOTOR" GOTO MOTOR

REM will run other strategies
C:\GEN372\RUNTIME %1
GOTO END

:TEST
COPY C:\TEST\RUNTEST.DB C:\TEST\TEST.DB > XXX.XXX
COPY C:\TEST\RUNTEST.MDB C:\TEST\TEST.MDB > XXX.XXX
COPY C:\TEST\RUNTEST.XDB C:\TEST\TEST.XDB > XXX.XXX
COPY C:\TEST\RUNTEST.CI C:\TEST\TEST.CI > XXX.XXX
COPY C:\TEST\RUNTEST.CA C:\TEST\TEST.CA > XXX.XXX
COPY C:\TEST\RUNTEST.CFG C:\TEST\TEST.CFG > XXX.XXX
COPY C:\TEST\WELCOME.GRP C:\GEN372\BOOT.GSP
C:\TEST\ALMRENAM
CALL C:\ALMCOPY
DEL ALARM*.*
C:\GEN372\RUNTIME %1
GOTO END

:MOTOR
COPY C:\MOTOR\MAINMENU.GRP C:\GEN372\BOOT.GSP
C:\GEN372\RUNTIME %1

:END
```

DACS STATION 9 DIRECTORY LISTINGS

```
 Volume in drive C is MS-DOS_5
 Volume Serial Number is 1C99-5306
                 175628288 bytes free


 Directory of C:\*.BAT,*.SYS

AUTOEXEC BAT       892 04-26-96  10:53a
CONFIG   SYS       760 04-26-96  10:51a


 Directory of C:\GEN372

.            <DIR>      08-03-94   9:52a
..           <DIR>      08-03-94   9:52a
MODICON  DES      2418 04-03-92   3:50a
UNPACK   EXE     29378 02-18-94   1:36a
MODICON  MNU     14762 04-03-92   3:50a
CONF104  FNT      7168 02-18-94   5:01p
EXTASCII FNT      1536 02-24-94   3:27a
CONFIG   SYS        26 02-24-94   3:27a
CONFIG   PRM       407 08-03-94   9:56a
EGA      DEV     13450 02-18-94   5:01p
XLATE    EXE     78445 02-24-94   3:27a
BOOT     GSP      1492 02-24-94   3:27a
GENESIS  GSP      1109 02-24-94   3:27a
LOGO     DAT         9 02-24-94   3:27a
EPSON_MX PRP        36 02-24-94   3:27a
EPSON_MX PTR      2048 02-24-94   3:27a
GENESIS  BAT        19 08-03-94   9:56a
CONFMSMI COM      1487 02-18-94   5:01p
ICONS    ICN     66267 02-24-94   3:27a
README   354      7611 06-17-92   3:54a
ELECTRIC SMB      2837 02-24-94   3:27a
HEATEXCH SMB       846 02-24-94   3:27a
LOGIC    SMB      1353 02-24-94   3:27a
MISC     SMB      1680 02-24-94   3:27a
MOTORS   SMB       396 02-24-94   3:27a
TANKS    SMB      1890 02-24-94   3:27a
VALVES   SMB      2364 02-24-94   3:27a
KEYHELP  EXE     39261 02-24-94   3:27a
STANDARD HLS      1908 02-24-94   3:27a
KEYMAC   EXE     34201 02-24-94   3:27a
MOUSE    OPT      4118 02-24-94   3:27a
HPPJET   PRP        36 06-17-92   3:54a
HPPJET   PTR      2560 06-17-92   3:54a
IBMPRO   PRP        36 06-17-92   3:54a
```

IBMPRO	PTR	2048	06-17-92	3:54a
NEWJET	PRP	82	06-17-92	3:54a
NEWJET	PTR	2560	06-17-92	3:54a
XROX4020	PRP	82	06-17-92	3:54a
XROX4020	PTR	2560	06-17-92	3:54a
READPTR	ME	562	06-17-92	3:54a
STATECMP	EXE	28561	02-24-94	3:27a
STATES	SFL	642	02-24-94	3:27a
STATES	SFS	913	02-24-94	3:27a
BUTTONS	SMB	1320	02-24-94	3:27a
CART	SMB	245	02-24-94	3:27a
CIRCUITS	SMB	44898	02-24-94	3:27a
FACEPLAT	SMB	15558	02-24-94	3:27a
MACHINE	SMB	1841	02-24-94	3:27a
PRESS	SMB	1755	02-24-94	3:27a
VATS	SMB	3741	02-24-94	3:27a
AUTOCAD	DOC	9984	10-31-91	1:31a
DXFTOGRP	EXE	58777	10-31-91	1:31a
GRPTODXF	EXE	39665	10-31-91	1:31a
SPACE	GRP	33622	10-31-91	1:31a
STAIR	DXF	46932	10-31-91	1:31a
PUMP	GRP	12703	10-31-91	1:31a
STAIR	GRP	20625	10-31-91	1:31a
SHUTTLE	SMB	48419	10-31-91	1:31a
CONFIG	EXE	637000	02-24-94	3:27a
METACONF	EXE	69599	02-24-94	3:27a
METACONF	MNU	3859	02-24-94	3:27a
STANDARD	CFN	18899	02-24-94	3:27a
ANALOG	MNU	3377	02-24-94	3:27a
CONTROL	MNU	10065	02-24-94	3:27a
DIGITAL	MNU	4434	02-24-94	3:27a
DYNAMIC	MNU	14019	02-24-94	3:27a
GRAPHIC	MNU	2627	02-24-94	3:27a
HIST	MNU	15746	02-24-94	3:27a
LIBRARY	MNU	3471	02-24-94	3:27a
MACROS	MNU	2282	02-24-94	3:27a
MEMSYS	MNU	863	02-24-94	3:27a
MNEMONIC	MNU	8335	02-24-94	3:27a
STRING	MNU	1027	02-24-94	3:27a
SUPMAC	MNU	2461	02-24-94	3:27a
SYSCONF	MNU	15280	02-24-94	3:27a
RUNTIME	EXE	401424	02-24-94	3:27a
AINDET	GSP	2995	02-24-94	3:27a
ALM	GSP	2649	02-24-94	3:27a
ALMSUM	GSP	2244	02-24-94	3:27a
AMBDET	GSP	2449	02-24-94	3:27a
AOUTDET	GSP	2113	02-24-94	3:27a
CALC	GSP	3633	02-24-94	3:27a
CALC2	GSP	4408	02-24-94	3:27a
CALC3	GSP	2502	02-24-94	3:27a

CNT	GSP	2703	02-24-94	3:27a
CTLDET	GSP	2896	02-24-94	3:27a
CTLDET2	GSP	3044	02-24-94	3:27a
CTLDET3	GSP	2580	02-24-94	3:27a
DGAP	GSP	3003	02-24-94	3:27a
DGAP2	GSP	2593	02-24-94	3:27a
DIGDET	GSP	2946	02-24-94	3:27a
DIR	GSP	4139	02-24-94	3:27a
DISPDIR	GSP	4234	02-24-94	3:27a
DISPSAV	GSP	1001	02-24-94	3:27a
DISPSEC	GSP	980	02-24-94	3:27a
DOTDET	GSP	2064	02-24-94	3:27a
DTIMDET	GSP	2588	02-24-94	3:27a
EMPTY	GSP	845	02-24-94	3:27a
EXIT	GSP	1021	02-24-94	3:27a
FF1	GSP	2190	02-24-94	3:27a
FF2	GSP	2335	02-24-94	3:27a
FILEUTIL	GSP	1296	02-24-94	3:27a
HCURS	GSP	415	02-24-94	3:27a
HDWDET	GSP	1954	02-24-94	3:27a
HELP	GSP	1217	02-24-94	3:27a
HELP1	GSP	1124	02-24-94	3:27a
HELP2	GSP	1217	02-24-94	3:27a
HELP3	GSP	1217	02-24-94	3:27a
HELP4	GSP	1214	02-24-94	3:27a
HELP5	GSP	1210	02-24-94	3:27a
HELP6	GSP	1156	02-24-94	3:27a
HGLSTMEN	GSP	1344	02-24-94	3:27a
HGREP	GSP	781	02-24-94	3:27a
HGREPLST	GSP	1688	02-24-94	3:27a
HGREPMEN	GSP	1408	02-24-94	3:27a
HISMON	GSP	2463	02-24-94	3:27a
HISREP	GSP	1588	02-24-94	3:27a
HISTB	GSP	1862	02-24-94	3:27a
HISTLST	GSP	2038	02-24-94	3:27a
HISTMEN1	GSP	2029	02-24-94	3:27a
HTBREP	GSP	1226	02-24-94	3:27a
IDB	GSP	2674	02-24-94	3:27a
IDBAIN	GSP	2739	02-24-94	3:27a
IDBAIO	GSP	3137	02-24-94	3:27a
IDBAO	GSP	2070	02-24-94	3:27a
IDBDIN	GSP	4381	02-24-94	3:27a
IDBDINB	GSP	1574	02-24-94	3:27a
IDBDIO	GSP	4831	02-24-94	3:27a
IDBDIOB	GSP	1644	02-24-94	3:27a
IDBDOT	GSP	2809	02-24-94	3:27a
IDBDOTB	GSP	1325	02-24-94	3:27a
IDBENTRY	GSP	663	02-24-94	3:27a
IDBSIO	GSP	1480	02-24-94	3:27a
IDBSTG	GSP	1271	02-24-94	3:27a

IDBSTGO	GSP	1309	02-24-94	3:27a
LISTMAIN	GSP	363	02-24-94	3:27a
LISTMENU	GSP	1657	02-24-94	3:27a
LOGIC	GSP	2843	02-24-94	3:27a
MATH	GSP	2602	02-24-94	3:27a
MATH2	GSP	2506	02-24-94	3:27a
PAIO1	GSP	3453	02-24-94	3:27a
PAIO2	GSP	3453	02-24-94	3:27a
PAIO3	GSP	3888	02-24-94	3:27a
PAOUT	GSP	3143	02-24-94	3:27a
PASSWORD	GSP	1794	02-24-94	3:27a
PDIN	GSP	5661	02-24-94	3:27a
PDOUT	GSP	4171	02-24-94	3:27a
PLOT	GSP	465	02-24-94	3:27a
PLOTMEN	GSP	2405	02-24-94	3:27a
PULNOT	GSP	2322	02-24-94	3:27a
RAMP	GSP	3525	02-24-94	3:27a
SEL	GSP	3718	02-24-94	3:27a
SEQ	GSP	2131	02-24-94	3:27a
SIM	GSP	3038	02-24-94	3:27a
SLOGIC	GSP	2360	02-24-94	3:27a
SORTIDBM	GSP	1976	02-24-94	3:27a
SORTMAIN	GSP	2164	02-24-94	3:27a
SORTMENU	GSP	1128	02-24-94	3:27a
SPULNOT	GSP	1503	02-24-94	3:27a
STAT1	GSP	2271	02-24-94	3:27a
STAT2	GSP	2053	02-24-94	3:27a
SWCH	GSP	2925	02-24-94	3:27a
SYSCONF	GSP	3011	02-24-94	3:27a
SYSCONF2	GSP	2891	02-24-94	3:27a
SYSPERF	GSP	2325	02-24-94	3:27a
TD	GSP	3086	02-24-94	3:27a
TIMERS	GSP	2363	02-24-94	3:27a
TOT	GSP	3285	02-24-94	3:27a
TPO	GSP	3658	02-24-94	3:27a
TREND	GSP	467	02-24-94	3:27a
TRENDLST	GSP	2042	02-24-94	3:27a
TRENDMEN	GSP	1990	02-24-94	3:27a
TRENDREP	GSP	3400	02-24-94	3:27a
TUNE	GSP	3152	02-24-94	3:27a
TUNE2	GSP	3156	02-24-94	3:27a
TUNE3	GSP	2935	02-24-94	3:27a
TUNE4	GSP	2432	02-24-94	3:27a
USER	GSP	3967	02-24-94	3:27a
USER2	GSP	3090	02-24-94	3:27a
XC	GSP	3543	02-24-94	3:27a
KB	TBL	3392	02-24-94	3:27a
STANDARD	KML	544	02-24-94	3:27a
MODICON	DRV	7038	04-03-92	3:50a
MODICON	350	282	04-03-92	3:50a

LOGO_NET	DAT	9	02-18-94	1:36a
GENNET	DNL	442	05-09-95	10:39a
GENCFG	GNF	265	02-18-94	1:36a
GENCFG	NCF	276	05-09-95	10:39a
GENNET	NDB	171	05-09-95	10:39a
GENCMDR	DOC	5912	02-18-94	5:02p
GN_INTF	OPT	32731	02-18-94	5:02p
DOSPWD	EXE	113849	02-18-94	5:01p
GEN_NET	OPT	61551	01-15-92	3:52a
GN_CA	OPT	66311	01-15-92	3:52a
GN_CFA	OPT	99255	01-15-92	3:52a
GN_DCA	OPT	77441	01-15-92	3:52a
GN_DCFA	OPT	110897	01-15-92	3:52a
GN_DFA	OPT	94991	01-15-92	3:52a
GN_DRVR	OPT	67307	02-18-94	5:02p
GN_COMM	OPT	35453	02-18-94	5:02p
GN_FTP	OPT	52219	02-18-94	5:02p
GCOPY	EXE	145747	02-18-94	5:01p
GENCOMP	EXE	44069	02-18-94	5:01p
GENCONF	EXE	91957	02-18-94	5:01p
GENMON	EXE	16487	02-18-94	5:01p
GN_TIMER	EXE	13851	02-18-94	5:01p
PS110RST	EXE	15259	02-18-94	5:01p
SHOW_OPT	EXE	15931	02-18-94	5:01p
DOSPWD	MNU	959	02-18-94	5:01p
GNETDIR	GSP	676	02-18-94	5:01p
GNETMON	GSP	2073	02-18-94	5:01p
GNETMON2	GSP	2045	02-18-94	5:01p
GNFXEXIT	GSP	972	02-18-94	5:01p
GNFXR1	GSP	1053	02-18-94	5:01p
GNFXR2	GSP	1043	02-18-94	5:01p
GNFXR3	GSP	1024	02-18-94	5:01p
GNFXR4	GSP	827	02-18-94	5:01p
GENCFG	MNU	33934	02-18-94	5:01p
GN_TIMER	SYS	1856	02-18-94	5:01p
DOSNODE	PWD	226	02-18-94	5:01p
BOOT	BAT	79	02-18-94	5:01p
GENCMDR	EXE	260257	02-18-94	5:02p
GENCEXIT	GSP	928	02-18-94	5:02p
GENCMDR	GSP	948	02-18-94	5:02p
GENCMDR2	GSP	860	02-18-94	5:02p
GENCMDR	MNU	7682	02-18-94	5:02p
GENCMDR	OPT	43487	02-18-94	5:02p
GENCMDR	HLP	23280	02-18-94	5:02p
GENCMDR	NDX	625	02-18-94	5:02p
RRCOMBO	OPT	65233	10-07-91	3:41a
RRTEST	EXE	68794	10-07-91	3:41a
FL		280918	08-06-93	7:02p
GNETMON3	GSP	1916	02-18-94	5:01p
DANISH	FNT	1536	02-24-94	3:27a

README	372	8135	02-24-94	3:27a
ALARM	GSP	2219	02-24-94	3:27a
BLANK	GSP	243	02-24-94	3:27a
EVENT	GSP	2212	02-24-94	3:27a
RRCOMBO	GSP	2450	02-24-94	3:27a
USER2_2	GSP	3090	02-24-94	3:27a
USER2_3	GSP	3090	02-24-94	3:27a
USER2_4	GSP	3090	02-24-94	3:27a
USER2_5	GSP	3090	02-24-94	3:27a
USER2_6	GSP	3090	02-24-94	3:27a
USER2_7	GSP	3090	02-24-94	3:27a
USER3	GSP	1855	02-24-94	3:27a
USER3_2	GSP	1855	02-24-94	3:27a
USER3_3	GSP	1855	02-24-94	3:27a
USER3_4	GSP	1855	02-24-94	3:27a
USER3_5	GSP	1855	02-24-94	3:27a
USER3_6	GSP	1855	02-24-94	3:27a
USER3_7	GSP	1855	02-24-94	3:27a
USER_2	GSP	3967	02-24-94	3:27a
USER_3	GSP	3967	02-24-94	3:27a
USER_4	GSP	3967	02-24-94	3:27a
USER_5	GSP	3967	02-24-94	3:27a
USER_6	GSP	3967	02-24-94	3:27a
USER_7	GSP	3967	02-24-94	3:27a
GNET_OLD	KML	510	02-24-94	3:27a
SHADOW	GSP	1912	02-18-94	5:01p
NETALMS	GSP	1825	02-18-94	5:01p
EXPORT	OPT	18307	02-18-94	5:01p

DACS STATION 9 FILE LISTINGS

"AUTOEXEC.BAT" on Station 9

```
C:\DOS\SETVER SCPLUS.EXE 5.00
REM @ECHO OFF
SET OAD_DRIVER=C:\OADDOS
C:\NETROOM\XLOAD.EXE -C -AB001,27664 -AC156,12496 -ACE01,47232 C:\OADDOS\SCPLUS\SCPLUS INSTALL
SET OAD_UTILITY=C:\OADDOS
PROMPT $p$g
goto %config%

:GEN354
PATH C:\DOS;C:\;C:\GENESIS;C:\MOUSE;C:\BAT;
goto finish

:GEN372
PATH C:\DOS;C:\;C:\GEN372;C:\MOUSE;C:\BAT;
SHARE
goto finish

:finish
C:\NETROOM\XLOAD.EXE -SCE01 -M23248 -X C:\MOUSE\MOUSE

@REM ==== LANMAN 2.1a === DO NOT MODIFY BETWEEN THESE LINES === LANMAN 2.1a ===
PATH C:\DOS;C:\LM\NETPROG;%PATH%
C:\NETROOM\XLOAD.EXE -C -AB001,4448 -AC156,13488 -AF001,35520 NET START WORKSTATION
NET LOGON DACCS9 DACCS9 /DOMAIN:STANDALONE
net use f: \\whc324\dpu strength
@REM ==== LANMAN 2.1a === DO NOT MODIFY BETWEEN THESE LINES === LANMAN 2.1a ===
guest
f:
cd \raw\transfer

REM ======GENESIS=========
GENCMDR
REM ======GENESIS=========
```

"CONFIG.SYS" on Station 9

```
[MENU]
menuitem=GEN354,Genesis system version 3.54
menuitem=GEN372,Genesis system version 3.72
menudefault=GEN372[,0]

[COMMON]
DEVICE=C:\NETROOM\RM386.EXE AUTO X=C800-CBFF X=D900-DDFF
rem DEVICE=C:\NETROOM\VIDCLOAK.EXE /PMBIOS
rem DEVICE=C:\NETROOM\SYSCLOAK.EXE
DEVICE=C:\NETROOM\XLOAD.SYS -O
REM  DOS=HIGH
BUFFERS=17,0
FILES=55
LASTDRIVE=P
FCBS=4,0
DEVICE=C:\NETROOM\XLOAD.SYS -SC156 -M13728 C:\DOS\SETVER.EXE
DEVICE=C:\OADDOS\DOSCFG.EXE /M1 /V
DEVICE=C:\NETROOM\XLOAD.SYS -SF001 -M51376 C:\OADDOS\DOSOAD.SYS

SHELL=C:\NETROOM\XLOAD.EXE -SCE01 -M57021 -E C:\DOS\COMMAND.COM C:\DOS\  /p
STACKS=0,0
DEVICE=C:\NETROOM\XLOAD.SYS -SF001 -M2832 C:\NETROOM\STACKS.EXE 9,256
device=c:\scsi\aspipc2.sys
[GEN354]

[GEN372]
```

Appendix J: ASCII/BASIC Listings

```
10 REM_____________________________________________________
15 REM  Filename:  GC3.BAS
20 REM
25 REM  Version:   2.0
30 REM
35 REM  Purpose:  GC3, GC2 and FTIR Data Collection
40 REM            Reads gas data and places it in registers for PLC access.
45 REM            Port 0 for terminal communication
50 REM            Port 1 for data from the gc3 computer
55 REM
60 REM  Modifications:
65 REM            V1.3 - GC2 H2 split into ms byte and ls byte
66 REM            V2.0 - Comments added
67 REM            V2.1 - GC2 removed, PHOTO NH3 added
70 REM_____________________________________________________
75 REM
100 PORT0
110 REM  SND(8) is read by the PLC.  Indicates that data is available.
120 REM
130 SND(8)=0
140 Z=-1
150 PRINT "GC-3 Data Collection"
160 PORT1
170 REM  Clear reg(30).  This register is set by the PLC when data
180 REM  has been captured.
190 REM
200 REG(30)=0
210 REM
220 REM Look for start of transmission (STX - ASCII 2)
230 DO
240 Q=GET
250 REM Strip off any parity bits.
260 REM
270 IF Q>127 THEN Q=Q-128
280 UNTIL Q=2
290 REM
300 REM  STX found.  Input 17 data values and place in registers 0-16.
310 FOR I=0 TO 13
320 PORT1
330 INPUT X
340 REG(I)=X
350 PORT0
360 PRINT REG(I)
370 NEXT I
380 REM Data is in registers.  Notify PLC by setting SND(8) to 1.
390 REM
400 SND(8)=1
410 REM Wait until PLC gets data, then exit.  PLC will automatically
```

```
420 REM Re-run program.
430 REM
440 IF(REG(30)=0)THEN 440
450 SND(8)=0
460 END
```

```
10 REM_____________________________________________________
15 REM   Filename: RGA5.BAS
20 REM   Author:   Jeff Martin
25 REM   Version:  1.0
30 REM
35 REM   Purpose:  RGA-5 Data Collection
40 REM             Extract Data from RGA-5 ASCII stream and send it to
45 REM             the PLC.
50 REM
55 REM   Notes:    Port 0 is for terminal communication.
60 REM             Port 1 receives the RGA-5 data stream.
65 REM
70 REM             The following ASCII formats are used by the program
75 REM             RMSG1: S0,D3
80 REM             RMSG2: S1,A1
85 REM             RMSG3: S2
90 REM             RMSG4: S2,D2,D2
95 REM             RMSG5: S4,F6.1
100 REM            RMSG6: S6,D1,D4,D4
105 REM            RMSG7: S9,D3
110 REM            RMSG8: S10
115 REM            RMSG9: S10,F6.1
120 REM            RMSG10:S12,D1,D4,D4
125 REM            RMSG11:S15,D3
130 REM            RMSG12:S16
135 REM
140 REM           These ASCII formats need to be loaded into the ASCII
145 REM           section of the ASCII/Basic module.
150 REM           In addition, the ASCII/Basic module must be set for no
155 REM           delimeter (DL# 0) and for a prefix string of "=" (PR# 3Dh).
160 REM           XON/XOFF should be enabled (XI# 1).  The communications
165 REM             parameters are 1200 baud, 8 bits, no parity, 1 stop bit.
170 REM
175 REM           The program should be loaded into RAM:2 of the Basic module.
180 REM
185 REM  Modifications:
190 REM
195 REM_____________________________________________________
280 ONERR 6000
285 PORT0
290 REM  SND(8) is sent when data has been collected.  PLC uses this
295 REM  to determine when to grab data.
300 REM
305 SND(8)=0
310 REM  "Z" is a constant used in Subroutine at line 1390
315 REM
320 Z=-1
325 PRINT "RGA-5 Data Collection"
330 REM  Search ASCII stream for the word "Run"
335 REM  Reg(30) is set to 1 by the PLC when it has read the data
340 REM
345 REG(30)=0
350 port1
355 DO
```

```
360 Q=GET
365 UNTIL Q=ASC("R")
370 REM  Keep reading characters until an "R" is found.
375 REM
380 GOSUB 1390
385 REM Subroutine 1390 will return the next character in Q
390 IF Q<>ASC("u")THEN 355
395 GOSUB 1390
400 IF Q<>ASC("n")THEN 355
405 PORT0
410 PRINT "Message detected"
415 PORT1
420 REM  Read data by issuing rmsg's to the ASCII processor
425 REM
430 FOR I = 1 TO 12
435   RMSG(I)
455 NEXT I
460 REM
465 REM Put data to be sent to PLC into registers 50-60.
470 REM Some data is just copied over, others are converted.
475 REM
480 REM Copy run number
485 REM
490 reg(50) = reg(0)
495 REM
500 REM
505 REM We are expecting stream to be "Tank", "Calibrate" or "Flush".
510 REM Check the first character, set status to 0 for tank, 1 for calibrate
515 REM 2 for flush and 3 for unknown.
520 REM
525 reg(51)=3
530 if reg(1) = asc("t") .OR. reg(1) = asc("T") then reg(51)=0
535 if reg(1) = asc("c") .OR. reg(1) = asc("C") then reg(51)=1
540 if reg(1) = asc("f") .OR. reg(1) = asc("F") then reg(51)=2
545 REM
550 REM
555 REM Store time as hhmm.
560 REM
565 reg(52)=reg(2)*100+reg(3)
570 REM
575 REM
580 REM Scale concentration to a 16 bit register
585 REM
590 reg(53) = int((reg(4)*1000.0 + reg(5)/10.0) * (65535.0/9999.9))
595 REM
600 REM Break the area down so that it fits into two registers.
605 REM To re-constitute: H2A Area = reg(54)*32000 + reg(55)
610 REM
615 A = (reg(6)*3125.0) + (reg(7)*0.3125) + (reg(8)/32000.0)
620 reg(54) = int(A)
625 reg(55) = int((A-int(A))*32000.0+0.5)
630 REM
635 REM
640 REM Copy H2A Retention time
```

```
645 REM
650 reg(56) = reg(9)
655 REM
660 REM
665 REM H2B Concentration scaled to a 16 bit register
670 REM
675 reg(57) = int((reg(10)*1000.0 + reg(11)/10.0) * (65535.0/9999.9))
680 REM
685 REM H2B Area
690 REM
695 A = (reg(12)*3125.0) + (reg(13)*0.3125) + (reg(14)/32000.0)
700 reg(58) = int(A)
705 reg(59) = int((A-int(A))*32000.0+0.5)
710 REM
715 REM H2B Retention Time
720 reg(60) = reg(15)
725 REM
730 REM  Print data to terminal
735 port0
740 print
745 print "Run =                    ",reg(50)
750 print "Status =                 ",reg(51)
755 print "Time =                   ",reg(52)
760 print
765 print "H2A Concentration =      ",(reg(53)/65535.0)*9999.9
770 print "H2A Area =               ",reg(54)*32000.0+reg(55)
775 print "H2A retention time =     ",reg(56)
780 print
785 print "H2B Concentration =      ",(reg(57)/65535.0)*9999.9
790 print "H2B Area =               ",reg(58)*32000.0+reg(59)
795 print "H2B retention time =     ",reg(60)
800 print
805 REM  Data is in the registers. Set SND(8) to notify PLC.
810 REM
815 SND(8)=1
820 REM Wait until PLC grabs data (it will set REG(30) to 1), then
825 REM exit.  PLC will automatically re-run the program.
830 REM
835 IF(REG(30)=0)THEN 835
840 SND(8)=0
845 END

1350 REM
1360 REM Subroutine for reading characters.  If no character is available,
1370 REM GET returns a -1.  This subroutine strips off these -1's until a
1380 REM character is available.
1390 DO : Q=GET : UNTIL Q<>Z
1400 RETURN

6000 port0
6005 print "Error ",ERRNO," Detected
6010 END
```

Appendix K: Software License Agreements

The following is a list of the commercial software that is used in the DACS system.

Company	Software	Version
Helix	Netroom3	3.02
Kensington Microware, Ltd.	Expert Mouse	n/a
Iconics, Inc.	Genesis Control Series	3.72
Iconics, Inc.	GEN-NET	3.64
Iconics, Inc.	Remote Supervisory Station	3.72
Modicon, Inc	Modsoft	2.1
Microsoft Corp.	MS-DOS	6.0
Microsoft Corp.	Windows	3.1
Nicolet Instrument Corp.	FAMOS	n/a
Symantec Corp.	Norton DiskLock	3.5

Appendix L: DeCipher Plus and DeTerminal Software

The DeCipher Plus software and DeTerminal software are used with the Datataker 50 data logger in support of the tank 241SY101 mixer pump removal. Descriptions of the software are summarized as follows:

DeCipher Plus Program Disk, version 1.1, July 1, 1992

```
 Volume in drive A is DCP11_1
 Directory of A:\

READ     ME         12,709 06-30-92   9:37a
README   BAT             22 06-26-92   1:01p
INSTALL  EXE        104,645 02-26-92  11:34p
DCP01100 001        448,830 06-30-92   9:51a
        4 file(s)        566,206 bytes
                         160,768 bytes free
```

"READ.ME" File Listing

```
DeCipher Plus Release 1.1 - Release Notes
-----------------------------------------

Contents
   1. How to install version 1.1
   2. Running the DeCipher Plus demonstration
   3. Vital hints for getting started
   4. Known software problems
   5. Known documentation problems
   6. What to do when you have problems
   7. Revision History


1. How to install version 1.1
-----------------------------

To install DeCipher Plus :-

   - the DeCipher Plus program disk contains the installation
     program:-

           INSTALL.EXE

   - execute this program either by
```

```
drive_name:INSTALL <Enter>
(e.g. >a:INSTALL)
```

or

```
drive_name: <Enter>
INSTALL <Enter>
(e.g. >a:)
(       >INSTALL)
```

where drive_name should be replaced by the letter denoting the drive in which the program disk now resides.

- as the INSTALL program executes, it

  * creates the DeCipher Plus directory structure
  * copies the DeCipher Plus files
  * manages any changes to CONFIG.SYS and AUTOEXEC.BAT which may be required (with your permission)

2. Running the DeCipher Plus Demonstration

The \DCP directory on the installation drive contains the demonstration program

```
DEMO.BAT
```

To run the demonstration, enter the DOS command

```
cd  \DCP<Enter>
DEMO<Enter>
```

The DeCipher Plus demonstration will execute various application displays using data previously acquired already installed onto your computer.

3. Vital Hints for Getting Started

* DeCipher Plus requires a very specific directory structure to be set up on disk in order for it to run. When this directory structure is not present DeCipher Plus will not run. This directory structure begins at the root directory level with the \DCP directory. The \DCP directory must always be at root directory level. By default the database is found on the drive on which DeCipher Plus is executing, although the DB command line parameter allows this to be modified.

* DeCipher Plus file connections can only be made to files with the extension DCP. These files are in a very special format and can only be generated through the File application of DeCipher Plus. The .DCP files are very different in format to any data saved in the top window of the Command application.

  IT IS NOT POSSIBLE to connect, plot or translate any data saved using Alt-S in the Command application, nor is it possible to connect to any data saved in the $Session.DAT file.

  The correct sequence of actions to save data from a logger into a disk file, which can then be used by DeCipher Plus is as follows

  - Connect to the logger
  - Select the desired retrieval mode (real time or logged) in the data window.
  - Select the desired channels in the File application
  - Select DECIPHER format in the File application (indicated by *)
  - Select a file name for the disk file (e.g. DATA)
  - Use the Run menu to select and run the File application

  As a result the file DATA.DCP will be created in the \DCP\DCP_DATA directory.

  To access this data file, we must now create a connection to the file as follows

  - Enter the Connect menu
  - Highlight Filedata
  - Press Alt N to create a new connection
  - Select the Datafile type
  - Tab to the Datafile tile
  - Ensure that the path is set for \DCP\DCP_DATA
  - Change the filename to DATA
  - Tab to the top menu
  - Press Alt-S to save
  - Name the connection DATA

  As a result a connection definition named DATA will now be listed in the connection list. Connect to the DATA connection as by pressing <Enter> or Alt C. The original data from the logger is now available for use within DeCipher Plus.

* Note that the files created by the File application are for IMPORTing into the spreadsheets only. They are NOT worksheet files.

* The inbuilt context sensitive help system is extensive and is available throughout DeCipher Plus. It may prove useful to browse the help files

before using the product. These files and other useful information can be accessed through the ? window. Simply highlight the desired subject and press <Enter>.

4. Known Software Problems with Version 1.1

* DeCipher Plus does not exit gracefully to DOS if it cannot find a database structure with which to execute (either by default or as specified on the command line). Reboot the machine to restart DOS. Investigate why no database (\DCP directory structure) exists where DeCipher Plus was executed.

* There are some memory problems associated with running DeCipher Plus under Windows as a DOS application. If the EMM386 expanded memory manager is being used with the NOEMS parameter, DeCipher Plus will crash on initialization. It is not compatible with this expanded memory arrangement. Run EMM386 without the NOEMS parameter.

* We have had reports of DeCipher Plus crashes in the plot application connected to use of the axis changing keys and non-standard graphics hardware. As yet no problems have been reported with VGA systems but we have received one report each for Hercules and EGA. It should be stressed that in general DeCipher Plus does work with these graphics adaptors. In cases where the axis change facility causes problems, it should not be used. Check that there are no TSRs loaded which may be making sophisticated or unusual use of the graphics adaptor.

* Some problems have been reported when a DeCipher Plus connection is made to a COM port on which a serial mouse driver is resident. Some mouse drivers have been found to interfere with DeCipher Plus, which causes various phenomena such as no communications when running applications. We have found this to be a problem especially with

  Genius Mouse driver
  Agiler Mouse driver

  The Microsoft mouse driver seems to function correctly. As a rule don't install mouse drivers installed on any COM port which you intend to use to connect to loggers.

* A small number of problems have been found with the saving of tasks

  - File application special date and time format are not saved (The first three for all file formats except DECIPHER). Use Date and Time from logger channels.
  - Tasks which include channels which have been deselected in the Data window will not load correctly. Ensure all deselection is managed within the individual application windows.

* The Lotus default file formats are incorrect for date and time. You should select .dddd format for time and ddd elapsed format for date manually. Once in Lotus it is necessary to import these as numbers and to convert them to TIME_VALUE and DATE_VALUE with a formula. The TIME_VALUE converts directly but the DATE_VALUE must be adjusted by adding 32509 to the imported number (converts to the Lotus date offset from 01\01\1900).

* It is possible to receive an alarm or an error message during the transmission of a response to DeCipher Plus program query. However this results in eventual loss of connection after a period displaying the Waiting window. Try using protocol mode if this is a problem. Otherwise just reconnect as this is only likely to be a problem in situations where large numbers of alarms or errors are being returned continuously or frequently.

* On entering the File application after a task load, the file destination may be selected but the filename may not be visible. To display the filename simply toggle the file destination selection.

* The Setup window always shows the default graphics printer after restart, even after another printer has been selected and saved. In fact the correct printer driver is active, the problem is that it has not been displayed correctly.

* Creation of a connection file with the name NEW_CON prevents creation of any other connection. The connections may be defined but they cannot be saved. The NEW_CON.CON should be deleted from \DCP\CON_DB\CCT. Unfortunately NEW_CON is the default connection name suggested on a new connection save.

* The available data range for a connection which includes an internal memory card (as displayed in the data window) is not visible. The data still exists on the card, but DeCipher Plus cannot access its range. Any applications executed can define any time\date range for retrieved data as required without problem.

* It is possible to select any format for date and time channels in the DECIPHER file format. However these have no effect, and will be forced to integer days and integer seconds due to the requirements for reconnection. This will be apparent by returning to the File application window.

* The units for date and time channels are inconsistent between different applications. However this does not affect running of the applications in any way.

* Y axis labeling on some plots is not quite correct. On occasion the largest Y axis label is not displayed.

* When connecting to a Datataker using an address not directly attached to the COM port of the computer, the 'Connected to' message is issued once DeCipher Plus has contacted the network and NOT the logger itself. If no logger with the specified address exists on the network, the first communication to that logger will be the first failure. This occurs on an attempt to agree on the program map with the Waiting window displayed. A network error is displayed on the Alarm line. At this point use Alt R to release the connection.

5. Omissions from the Manual in version 1.1

Three sections are missing from the manual.

- 6.3.8 Setup Module
- 6.3.9 Help overview
- A2 System messages

Exiting DeCipher Plus should be section 6.3.10

Synopsis

6.3.8 Setup Module
The setup module allows definition and saving of the contents of the DCP_CFG.CFG file. All overall system parameters can be set here.

It also allows loading and saving of task files. A task file is a complete copy of all DeCipher Plus menu definitions. Loading a task file results in DeCipher Plus menus being configured to the contents of the task file.

Examples are provided in the demonstration.

- DEMO_1.DDT
- DEMO_2.DDT
  etc...

6.3.9 Help overview
All DeCipher Plus help files (and other useful text files provided by Data Electronics) can be browsed within the help overview menu. Simply highlight the desired file and press <Enter>.

The filename is provided on the right hand side of the list to allow access to the text files for printing. The files can be found in \DCP\HELP.

The file KEY_HELP.HLP provides on line access to the manual section 6.2 Keyboard Usage.

6. What to do when you have Problems

* Please check both the manual appendix and help files for possible solutions to the problem.

* If that fails, fill out the problem report sheet as fully as possible. In particular describe the sequence of DeCipher Plus operations that highlighted the problem, and details of the Datataker program and configuration you were using. These are of paramount importance.

* Fax the information to us as follows
  Within Australia (03)-764-8997
  Outside Australia 61-3-764-8997

* For Telephone support you can contact us as follows
  Within Australia (03)-764-8600
  Outside Australia 61-3-764-8600

6. Revision History

24-April-1992	(JPM)	Original Version
01-July-1992	(JPM)	Upgrade Revision

Enjoy using DeCipher Plus!

DeTerminal Software, version 2., Science/Electronics Part #71025

```
 Directory of A:\

BLOKSCAN CMD         1,098  09-16-93  2:46p
DEPOLY   EXE        37,254  06-25-93 11:07a
DOWNTIME CMD         2,732  09-16-93  2:46p
DT       CFG           374  03-13-94 11:14p
DT       EXE       292,984  03-11-94  3:28p
DT       HLP        61,964  03-13-94 11:07p
EXAMPLE  CMD         2,283  01-20-94  5:01p
FUELFLOW CMD         1,767  09-16-93  2:46p
LOGSCAN  CMD         1,247  09-16-93  2:46p
LSQ      DOC         2,176  04-14-88  2:31p
LSQ      EXE        85,585  04-19-88  4:16p
PROGRAM  CMD         1,138  11-09-93  4:24p
READ     ME          4,958  03-16-94 12:19p
README   BAT             44  02-09-94 12:07a
STEPSCAN CMD         1,355  09-16-93  2:46p
TEMPLATE CMD         1,218  11-09-93  4:23p
UNLD_100 CMD         1,050  03-16-94 11:55a
UNLD_200 CMD         1,348  03-16-94 11:56a
WIND_SET CMD         3,632  09-16-93  2:46p
       19 file(s)        504,207 bytes
        0 dir(s)         947,200 bytes free
```

"READ.ME" File Listing

```
                        DeTerminal Command Files
                        ========================

Introduction
============
DeTerminal is a terminal type program which provides an easy to use interface
to the Datataker data loggers from IBM-PC and compatible computers.

DeTerminal is used to create and edit command files, to send commands and
command files to the Datataker, and to view and save data returned by the
Datataker.

This DeTerminal distribution disc contains the following files

        DT.EXE          - DeTerminal program file
        DT.HLP          - DeTerminal help file, accessed via F1
        DT.CFG          - Last used configuration of the DeTerminal system

        READ.ME         - This information file
```

PROGRAM.CMD - A suggested layout for Datataker programs
TEMPLAT.CMD - A template for Datataker programs, based on the suggested layout. A simple program can be quickly developed by entering details into the template.

Command Files
==========

The distribution disc also contains a number of Datataker command files which provide a guide to using DeTerminal to program the Datataker, and some useful applications programs, as follows

BLOKSCAN.CMD - Demonstrates programming of the Datataker to scan input channels for short periods of time, at longer regular intervals of time.

STEPSCAN.CMD - Demonstrates programming of the Datataker to scan input channels at increasing intervals as the data logging session proceeds. This technique can be used to collect data for a parameter which asymptotes with time, such that data is collected more slowly as the parameter changes more slowly.

LOGSCAN.CMD - Demonstrates programming of the Datataker to scan input channels at logarithmic intervals as the data logging session proceeds. This technique can also be used to collect data for a parameter which asymptotes with time.

DOWNTIME.CMD - Demonstrates programming of the Datataker to perform calculations based on time. Operator buttons are monitored to indicate reason for machine downtime, which provide triggers for measuring total elapsed time for each cause of downtime.

FUELFLOW.CMD - Demonstrates the use of Datataker expressions to combine data from two related measurements, to provide corrected results in real time.

WINDSET.CMD - Demonstrates use of Datataker expressions to calculate average wind direction and wind sigma from a wind direction and wind speed sensor.

UNLD_100.CMD - Provides assistance in unloading the DT100 family of loggers, by stepping the user through the decisions to be made about data delimiters, time formats, and filenames. The logger is then set as desired and unloaded automatically.

Note that UNLD_100.CMD is supplied for DT100 users as DeTerminal's Alt Unload feature is specifically for supporting DT500's and DT50's.

UNLD_200.CMD - Provides assistance in unloading the DT200 family of loggers, by stepping the user through the decisions to be made about data delimiters, time and date formats, and filenames. The logger is then set as desired and unloaded automatically.

Note that UNLD_200.CMD is supplied for DT200 users as DeTerminal's Alt Unload feature is specifically for supporting DT500's and DT50's.

Utilities
=========

There are also two utility programs on the distribution disc, for extracting polynomials from data sets of sensor calibrations. These polynomials can then be programmed into the Datataker to return data in engineering units from sensors with simple analog or digital outputs.

DEPOLY.EXE - Calculates polynomials mathematically from selected data points from a data set.

LSQ.EXE - Calculates polynomials for data sets using the least squares method.

DISTRIBUTION SHEET

To	From	Page 1 of 1
Distribution	Remote Sensing and Sampling Equipment Engineering	Date May 20, 1997
Project Title/Work Order		EDT No.
Computer System Design Description for SY-101 Hydrogen Mitigation Test Project Data Acquisition and Control System (DACS-1) - Rev. 3		ECN No. 637502

Name	MSIN	Text With All Attach.	Text Only	Attach./ Appendix Only	EDT/ECN Only
W. G. Brown	T4-07	X			
D. W. Crass	H6-11				X
J. A. Ellingsworth	R2-87	X			
A. M. Ermi (5)	L6-37	X			
G. J. Gauck	T4-07	X			
C. E. Hanson	S7-12				X
G. D. Johnson	S7-14				X
L. S. Krogsrud	T4-07	X			
D. C. Larsen	T4-08	X			
S. O. Smith	L6-37	X			
D. D. Tate	L6-37				X
R. W. Truitt	L6-37	X			
R. R. True	T4-07	X			
DACS Project File	L6-37	X			
Central Files (1)	A3-88	X			