GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION
SPONSORED PROJECT INITIATION

Date: 2/7/80

Project Title: Distributed Energy Management System (DEMS)

Project No: A-2563

Project Director: Mr. W. A. Baird

Sponsor: Megaplex Networks, Inc.; Atlanta, GA 30342

Agreement Period: From 1/24/80 Until 8/23/80

Type Agreement: Research Project and License Agreement dated 1/24/80

Amount: $20,000 A-2563-000 (Phase I)
22,000 A-2563-001 (Phase II)
$42,000 TOTAL

Reports Required: Monthly Progress Reports; Final Technical Report

Sponsor Contact Person (s):

Technical Matters

Contractual Matters
(thru OCA)

Megaplex Networks, Inc.
5188 Roswell Road, NE
Atlanta, GA 30342

Attn: Mr. John T. Stewart, Jr.

Defense Priority Rating: None

Assigned to: CSTL/SAD (School/Laboratory)

COPIES TO:

Project Director
Division Chief (EES)
School/Laboratory Director
Dean/Director—EES
Accounting Office
Procurement Office
Security Coordinator (OCA)
Reports Coordinator (OCA)

Library, Technical Reports Section
EES Information Office
EES Reports & Procedures
Project File (OCA)
Project Code (GTRI)
Other

SR-143

SPONSORED PROJECT TERMINATION SHEET

Date 1/7/82

Project Title: Distributed Energy Management System

Project No: A-2563

Project Director: W. A. Baird

Sponsor: Megaplex Networks, Inc.

Effective Termination Date: ______________________

Clearance of Accounting Charges: ______________________

Grant/Contract Closeout Actions Remaining:

- [ ] Final Invoice and Closing Documents
- [ ] Final Fiscal Report
- [ ] Final Report of Inventions
- [ ] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other ______________________

REVISED: To correct Project Director name and Project title.

This action also terminates all sub-budgets.

Assigned to: ECSL/CCB ~~(School~~/Laboratory)

COPIES TO:

Administrative Coordinator
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
~~Reports Coordinator (OCA)~~
Legal Services (OCA)
Library
EES Public Relations (2)
Computer Input
Project File
Other Bennett Teates

FORM OCA 10:781

DISTRIBUTED ENERGY MANAGEMENT SYSTEM FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
February 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird

for

Megaplex Networks, Inc.
5188 Roswell Road, N.E.
Atlanta, GA 30342

SOFTWARE

February through March 7

An initial investigation into how much of the FMS software can be transported to DEMS has resulted in two basic conclusions which support our initial estimate of the situation. First, since DEMS is different in concept from FMS, the microprocessor will take on more responsibilities than the central site minicomputer (CSM). Secondly, although the CSM software will be carried over to DEMS from FMS, it is expected that only a subset of this software will satisfy the requirements of DEMS. Much of the software is superfluous to DEMS, in particular, much of the real-time capabilities of the system.

Personnel have been assigned to DEMS in the following capacities:

Clyde G. Roby, Jr. - Technical Director

Mike Rowan - CSM modifications

Chuck Batterman - Microprocessor modifications

Enclosed is a copy of documentation relating to the microprocessor, as the sponsor has requested. Documentation of FMS is nearing completion thus allowing DEMS software development to proceed at a faster pace. The sponsor has requested that a short course, lasting no longer than one week, be taught to his personnel, concerning the schedules and modules to be put into the system, and in particular, the microprocessor. FMS development and testing is proceeding well, thus DEMS development is proceeding smoothly. The sponsor has been notified of potential problems concerning building retro-fit and some of the potential problems of sensor calibration. The sponsor has been cautioned that the design and testing of algorithms is still an "unknown" and not fully understood, art or science. Therefore, he may expect to experience a slow learning curve until he has achieved sufficient expertise.

The sponsor has been notified that exceptions in the operation of the system will have to be handled in the scope of FMS as it is now implemented. Although this will mean that operator communications with DEMS will be far from optimal, the capability to handle these exceptions will exist in DEMS as it currently exists in FMS. Exceptions, as it is used here, indicates those periods of time when a sponsor's customer may desire to come in on Sunday, for example, and a new schedule must be entered into DEMS for that customer (or potentially, many customers) possibly Friday evening. This has been noted as a shortcoming in FMS but will be carried over to DEMS as per contractual requirements.

March 10 through April 15

It is expected that DEMS will generate a sorted report of customer usage by account number. This automated report will serve as a basis for a manually generated detailed customer report. In this regard, towards the end of March, a report format should be available for the sponsor's examination. Documentation will continue in this time period with the emphasis shifting from FMS to DEMS in particular. Clyde Roby will be leading this effort. The seminar that has been requested by the sponsor is tentatively scheduled for the week of April 14. Chuck Batterman will probably be the instructor in this course. If time permits, the project director and technical director will sit in on this course.

Further work on FMS and DEMS will continue with Mike Rowan leading the work on DEMS. Doug Wrege will be working closely with Mike since Doug will be working in a parallel manner on FMS. Microprocessor software will continue with the following tentative schedule (Chuck Batterman assigned the task):

1. Implementation of central computer emulation locally in microprocessor - 2 weeks
2. Provision for the capability of reading returned data from the power prediction function - 1 week
3. Instruction of short course - 1 week

It is anticipated that Clyde Roby will visit the sponsor's site when he has installed his central site minicomputer (CSM). This is tentatively the week of March 17. Revision 5 of OS/32, the operating system for the CSM, may have corrected some problems that FMS has to work around now. Mike Rowan will gather information concerning what is currently available in the system that can be logged and determine how to log this data. In his modifications of the microprocessor software, Chuck Batterman will end up with a document describing changes and additions made. Georgia Tech will also come up with a definition of what we perceive as the sponsor's "standard" module definition. This will then be discussed with the sponsor so that he may gain some knowledge from it. In order to implement the microprocessor independence as described by the microprocessor modifications above, Carol Agee will modify TOKEN for the DEMS project.

HARDWARE

Procurement and fabrication of the Intel-based development system, two building-level energy management controllers, and two remote data transceivers have begun. All necessary hardware has been ordered and we are awaiting delivery. Building controller I/O card wire-wrap has begun. About 70% of P-C artwork for the RDT boards is completed and submitted for manufacture. The mechanical redesign of the building controller power supply for rack-mounting has started.

DISTRIBUTED ENERGY MANAGEMENT SYSTEM
FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
March, 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird
and
C. G. Roby, Jr.

for

Megaplex Networks, Inc.
5188 Roswell Road, N. E.
Atlanta, GA 30342

March 10 to April 1

Documentation has proceeded with the emphasis on user documentation rather than technical documentation. However, only microprocessor documentation has been delivered to the sponsor.

Mike Rowan has made several modifications to FMS sources for the Real-time portion of DEMS that will be run when the system is delivered to the sponsor. Carol Agee has begun modifications of the TOKEN program to implement the microprocessor independence commands. As mentioned in a previous report, Doug Wrege has been brought on board FMS and has gotten the real-time portions of FMS to properly synchronize with one another. He has also modified the communications routine so that fewer errors are made due to high activity. Microprocessor software has been modified by Chuck Batterman to include the DEMS-required changes as well as some improvements for FMS.

Clyde Roby visited the Megaplex site during the week of 19 March and all seems ready there for software delivery. The sponsor knows that a printer is not needed upon delivery but should be acquired soon. Also, the sponsor's configuration is a console terminal and one other CRT.

April

DEMS will generate a sorted report of customer usage by account number. This automated report can then serve as a basis for a manually-generated detailed customer report. The report format has not yet been designed.

It is still anticipated that the sponsor-requested seminar will be taught by Chuck Batterman. However, the time for this seminar will be closer to the end of the project period instead of in mid-April.

Mike Rowan will continue to gather information concerning what data is available in DEMS which is useful and how it should be logged.

Further documentation is forthcoming from Chuck Batterman concerning the changes made in the microprocessor.

Georgia Tech will also come up with a definition of what we perceive as the sponsor's "standard" module definition. When this is done, a meeting with the sponsor is expected so that everyone will be knowledgeable about this.

EXPENDITURES

A report of March expenditures will be made in mid-April.

DISTRIBUTED ENERGY MANAGEMENT SYSTEM
FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
April, 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird
and
C. O. Roby, Jr.

for

Megaplex Networks, Inc.
5180 Roswell Road, N. E.
Atlanta, GA 30342

SOFTWARE

April

Documentation has proceeded with emphasis continuing on user-oriented software and on the microprocessor documentation changes which reflect an evolution of FMS functionality towards DEMS specific functions. Since these modifications were made by Chuck Batterman he is coninuing the documentation effort.

The modifications in the TOKEN program of the cross-compiler for the microcomputer module generation have been completed by Carol Agee.

A pre-release version of the DEMS software was installed in the sponsor's mini-computer in mid-April. Appropriate modifications were made to vender software to activate a Command Processor talking to an Intecolor terminal. Problems were detected in communicating with the microcomputer from the Perkin-Elmer 3220. During the last two weeks of April, extensive research and investigation into probable causes finally resulted in the conclusion that ITAM under Revision 5.1 of OS/32 cannot communicate properly with the microcomputers, so a Revision 4.3 operating system was generated. Communications with the microprocessor were then established without errors.

May

We anticipate that in May the original contract requirements will be met. Both phases should be complete by mid-May, with possible extensions going beyond that date.

In early May, Version 1.0 of DEMS will be delivered for the mini-computer. By the second or third week of May, the new micro-computer software in the PROMs should be delivered.

CSTL is targeting a 16 May date to finish up contractual requirements for this project and in particular will have done the following:

1. Generated a report of customer usage by account number.
2. Begun to teach a short course to sponsor personnel about DEMS.
3. Completed documentation of Version 1.0 of DEMS as described in the contract.

We anticipate that in the first or second week of May Megaplex and CSTL will confer to establish goals for future Distributed Energy Management Systems work.

HARDWARE

March

Two building microprocessors and two RDTs were fabricated. The microprocessor and RDT for the Megaplex building were readied for delivery and held for minicomputer software installation. The microprocessor development system is complete except for disk drives.

April

The building microprocessor and RDT were installed at Megaplex. The design of battery backup and power-fail hardware were completed and the artwork finished.

May

The development system will be completed and the fabrication of the prototype battery backup chassis. Initial DEMS testing will be done at Megaplex.

EXPENDITURES

A report of April expenditures will be made in mid-May.

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
ATLANTA, GEORGIA 30332

15 May 1980

Mr. John Stewart
Megaplex Networks, Inc.
5188 Roswell Rd. NE
Atlanta, GA 30342

Dear John,

Included herewith is the budget breakdown for DEMS to supplement the April Monthly Status Report being sent to you.

If you have any questions, please advise me.

	MONTH	FISCAL YEAR	TOTAL CONTRACT
PERSONAL SERVICES			
Budget			14,155.00
Expended	6,262.57	11,572.20	11,572.20
Free Balance			2,582.80
RETIREMENT			
Budget			1,396.00
Expended	546.72	1,085.90	1,085.90
Encumbered	.00	.00	.00
Free Balance			310.10
MATERIALS AND SUPPLIES			
Budget			15,291.00
Expended	2,471.33	5,882,03	5,882.03
Encumbered	-1,375.15	4,621.50	4,621.50
Free Balance			4,787.47
TRAVEL			
Budget			400.00
Expended	7.80	7.80	7.80
Encumbered	.00	.00	.00
Free Balance			392.20

TOTAL DIRECT CHARGES			
Budget			31,242.00
Expended	9,288.42	18,547.93	18,547.93
Encumbered	-1,375.15	4,621.50	4,621.50
Free Balance			8.072.57
OVERHEAD			
Budget			10,758.00
Expended	4,759.55	8,794.87	8,794.87
Free Balance			1,963.13
TOTAL			
Budget			42,000.00
Expended	14,047.97	27,342.80	27,342.80
Encumbered	-1,375.15	4,621.50	4,621.50
Free Balance			10,035.70

Yours truly,

W. A. Baird
Project Director

DISTRIBUTED ENERGY MANAGEMENT SYSTEM
FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
May, 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird
and
C. G. Roby, Jr.

for

Megaplex Networks, Inc.
5188 Roswell Road, N. E.
Atlanta, GA 30342

May

Version 1.1 of DEMS was delivered to the sponsor. New PROMs should have been delivered in the microcomputer, since the new programs for these have been "burned".

The short course which was to have been taught to sponsor personnel under this contract has been postponed. The sponsor may prefer that it be taught under another project currently being proposed by CSTL.

Documentation is almost complete. Final program listings have, for the most part, been generated. Copies must now be printed to satisfy contractual requirements.

The DEMS system was brought up on the sponsor's computer system. A problem was discovered in DEMS, version 1.0, and was corrected with the delivery of version 1.1. Several modules were written and tested for the sponsor's building. A thorough test of the system was conducted.

June

Contractual requirements for DEMS should be completed in June. The sponsor will have received the latest version of DEMS and its software documentation. This will include program listings.

The anticipated short course will probably not be taught under this contract but will probably be postponed until a later date, or under a different contract.

EXPENDITURES

A report of May expenditures will be made in mid-June.

DISTRIBUTED ENERGY MANAGEMENT SYSTEM FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
June, 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird
and
C. G. Roby, Jr.

for

Megaplex Networks, Inc.
5188 Roswell Road, N. E.
Atlanta, GA 30342

June

The level of the June DEMS effort was low, and mainly confined to wrapping-up activities anticipating project completion.

July

We expect to make the final transfer of deliverables in July, and to finish this first Megaplex distributed-energy management project.

EXPENDITURES

A report of June expenditures will be made in mid-July.

DISTRIBUTED ENERGY MANAGEMENT SYSTEM
FOR MEGAPLEX NETWORKS, INC.

Monthly Status Report
July, 1980

EES/GIT Project A-2563

Prepared by

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

W. A. Baird
and
C. G. Roby, Jr.

for

Megaplex Networks, Inc.
5188 Roswell Road, N. E.
Atlanta, GA 30342

A final report for DEMS was drafted, comprising a detailed description of the DEMS software. It will be reviewed and completed during August.

The engineering drawings of the DEMS hardware are being gathered and blueprinted for delivery to the Sponsor. This effort will be completed during August.

Administrative review and approval of the final report will probably not be completed during August, but the level of chargeable effort will be virtually nil.

EXPENDITURE

	MONTH	FISCAL YEAR	TOTAL CONTRACT
PERSONAL SERVICES			
Budget			14,155.00
Expended	1,373.19	1,373.19	21,138.81
Free Balance			-6,983.81
RETIREMENT			
Budget			1,396.00
Expended	94.86	94.86	1,899.87
Encumbered	.00	.00	.00
Free Balance			-503.87
MATERIALS AND SUPPLIES			
Budget			15,291.00
Expended	33.30	33.30	10,878.73
Encumbered	100.00	100.00	100.00
Free Balance			4,312.27

TRAVEL			
Budget			400.00
Expended	.00	.00	46.80
Encumbered	.00	.00	.00
Free Balance			353.20
TOTAL DIRECT CHARGES			
Budget			31,242.00
Expended	1,501.35	1,501.35	33,964.21
Encumbered	100.00	100.00	100.00
Free Balance			-2,822.21
OVERHEAD			
Budget			10,758.00
Expended	1,000.31	1,000.31	16,022.18
Free Balance			-5,264.18
TOTAL			
Budget			42,000.00
Expended	2,501.66	2,501.66	49,986.39
Encumbered	100.00	100.00	100.00
Free Balance			-8,086.39

This represents virtually all the costs which will be incurred in implementing DEMS. However, in order for the remainder of the reporting and delivery work to be completed the sponsor must confirm to the Office of Contracts Administration his verbal orders of June and July that work should continue despite the expenditure of the original estimate of development costs.

GEORGIA INSTITUTE OF TECHNOLOGY
Engineering Experiment Station
Atlanta, Georgia 30332

DISTRIBUTED ENERGY MANAGEMENT SYSTEM SOFTWARE DESCRIPTION

Final Technical Report

by

Michael J. Rowan
Clyde G. Roby

of the

Computer Science and Technology Laboratory
GIT/EES Project A-2563
Software Technology Division

October 1980

Prepared for

Megaplex Networks, Inc.
5188 Roswell Road, N.W.
Atlanta, Georgia 30342

TABLE OF CONTENTS

PREFACE

This report provides a detailed description of the DEMS software and its differences and likenesses with the FMS software. As such it constitutes one of the several required deliverables to fulfill the contractual agreements between Megaplex Networks, Inc. and the Georgia Institute of Technology. Taken together with the Minicomputer Command Processor Users Manual and the Microcomputer Processor Users Manual (both provided under separate cover), this report provides the requisite information needed for operation of DEMS software.

1.0 INTRODUCTION

1.1 The Georgia Tech Facilities Management System

Rising energy costs on the Georgia Tech campus provided the original impetus for the development of the prototype Facilities Management System (FMS). Design efforts began in 1977 at the Computer Science and Technology Lab (CSTL) under the sponsorship of the Physical Plant Department (PPD) to develop a computer controlled system for reducing campus-wide energy consumption. The system has evolved to the point where numerous other functions are included in its capabilities. The system has become, in essence, an integrated set of generic software tools to process user-defined data. The data can be represented for example, by user-written programs, device control equations with corresponding constraints, or time dependent schedules. Moreover, the FMS capabilities are being continually upgraded[1].

1.2 Differences in the FMS/DEMS Operating Environment

Megaplex, Inc. of Atlanta contracted with GTRI to have the FMS adapted to permit them to offer an energy management service to the public. The software generated to provide this service is known as the Distributed Energy Management Systems (DEMS). Essentially, the nature of the proposed Megaplex operation, as compared to the one on the Georgia Tech campus, necessitated modifications to the FMS computer code. There are three key differences between the Georgia Tech FMS and the Megaplex DEMS which necessitated these

---

[1] This evolution is reflected in the report by addressing software components that will be added or modified. For example, the EVENTSUB component planned to be in the Command Processor and listed in Appendix A is now planned to be called EVENTMGR and to be a component for the Real-Time Processor.

software modifications. The basic entity in the Megaplex operation is the individual account, with the typical physical configuration being a single building to control, using a building microcomputer, Remote Data Transmitter (RDT's), and associated sensors and controllers. These individual accounts do not have or require a functional relationship with each other. On the other hand, the basic entity in the Georgia Tech operation is the zone, typically comprised of dozens of buildings with a common metered power feeder line, and hence is processed as a single billing unit.

The central site minicomputer in the DEMS operation functions for efficiency in the provision of the service. The customer (individual account) need not purchase mass storage devices or hire specialists to prepare control modules. The system, however, does not take actions in one building (account) because of current conditions in another, a typical situation in the FMS.

A second difference between the operating environments of DEMS and FMS is the geographic area covered and its impact on communications. The campus area is relatively small, so that communications between the control microcomputer and various campus buildings is over hard-wired lines at 9600 baud. Megaplex is trying to serve a market, dispersed throughout the Atlanta region, and therefore, will utilize common carrier lines at potentially slower speeds.

Finally, FMS was developed on a Perkin-Elmer Interdata 8/32 under the operating system OS32 version 3.2. Megaplex purchased a Perkin-Elmer 3220 with OS32 version 5.0. Although the machines are compatible, the different operating system versions necessitated some FMS software modifications for DEMS.

1.3 Software System Attributes

The system has the following attributes:

- Distributed processing
- Real time
- Multi-tasking
- Data acquisition and process control
- Integrated hardware and software system

The distributed nature of the system is represented by a central site minicomputer which communicates with numerous micro-computers, generally one to a building. The building micro-computers, in turn, are connected to remote data transceivers (RDT's), microprocessor based devices, that communicate with sensors and control points within a building. Each level in the hierarchy has specific processing functions and maintains a data base to accomplish its tasks.

Each level in the hierarchy contains tasks which execute at a given time interval or specified time-of-day, and other tasks which continuously monitor the state of the system and make appropriate adjustments whenever significant changes or events are detected.

The central site minicomputer software executes under the control of a multi-tasking operating system (OS32). Additionally, a master control program controls the execution of tasks according to time-of-day, events, or requests from other tasks. At the microcomputer level, a multitasking executive controls resource sharing among tasks.

1.4 Scope

This report focuses on the central site minicomputer software. It addresses the major components and their functions, and then discusses specific tasks. Finally, a comparison of FMS and DEMS is presented.

2.0 SOFTWARE DESCRIPTION

2.1 Minicomputer Software

The minicomputer software can be subdivided into two major modules composed of several tasks each; the real-time module and the command processor module. Table 2.1 and 2.2 list the functions accomplished by the modules with examples. In order to successfully process these functions, a series of components were developed. In some cases, a component can be represented by a single task or a portion of a task; whereas in others, the component resides as a portion of several tasks. These major components are listed in Tables 2.3 and 2.4.

Table 2.1

COMMAND PROCESSOR

FUNCTION	EXAMPLE
Data preparation	. Schedule input and editing . Control equation input and compiling
Status-checking/monitoring	. Access to power consumption history . Listing alarm and error files . Access to equipment table variables in micros
Fine-tuning	. On-line editing, recompiling, and downloading of control equations . Modifying control variables, i.e., desired never exceed power level
Manual intervention when appropriate	. Shedding or restoring equipment . Modifying control point status
Data transmission	. Download schedules . Download modules
System access control	. Verifying operators . Logging all commands

Table 2.2

REAL-TIME MODULE

FUNCTION	EXAMPLE
Task-control	• MASTER starts up system • MASTER releases ALARM at requested time • EVTQMAN requests MASTER to start BOY after receipt of alarm from ALARM
Status-checking	• Alarm polling
Data collecting/evaluation/ response	• Power use evaluation/shed or restore • Activating user defined event-initiated procedures
Data transmission	• Shipping next hour schedules to all building micro computers
Logging	• Alarms • System errors • Printing log files • Writting system common to disk

Table 2.3

COMMAND PROCESSOR MODULE

Major Components:

- Access control (security)
- Command interpreter
- High and low level command processor
- Control language (FORCE)
- Editors
- Compiler
- Data base management capabilities
- Micro communications handler
- User I/O handler
- Task common data

Organization:

- Core resident kernel & I/O handler
- Non-core resident tasks
- Overlays
- Interactive

Table 2.4

REAL-TIME MODULE

Major components:

- Master control program.
- Function specific tasks i.e. ALARM, SCHED.
- Inter-task communication
- Task common data
- Communications with micros
- Logging & file control

Organization:

- Core resident tasks
- Non-core resident tasks (initiated by MASTER)
- Task priority scheme
- No user intervention

The two major modules are not discrete units, rather they interact at several points and utilize common subroutines and/or components. Real-time tasks, for example, need to output information to an operator's terminal. If the terminal is currently assigned to a command processor task, the real-time task sends a message to the command processor's I/O task, whereupon I/O is cancelled, permitting the real-time task to output the information. Upon completion, another message is sent, telling the I/O task to re-initiate I/O operations.

Common files are also processed by the two major modules. For example, schedule files are created utilizing the command processor and the real-time task, SCHED, extracts the next hour schedule for each building micro and transmits the data. Moreover, task common data are shared by the two modules. Finally, the modules have parallel capabilities. The command processor, for example, permits the operator to transmit schedule data.

2.2 Real-Time Tasks

2.2.1 MASTER

The MASTER program is a core resident task that oversees the timing and execution of all other tasks in the real-time module, and it assures the initialization and maintenance of important system and task common data items during initial start-ups or power-fail restarts. Moreover, it updates a disc file of task common variables whenever another task requests the action of MASTER. The controlling of task execution is attained both by logic embedded in MASTER and by requests from other tasks. Since the requests can be random, MASTER also contains logic to assure that the task due to be started has finished a previous execution. Subroutines QUEUE and Send Message (SNDMSG) from the FORTRAN seven run-time library are utilized for intertask communication.

The difference in the MASTER task in DEMS as compared to FMS is in the tasks which are controlled. In both systems, MASTER processes the tasks ALARM; BOY; EVTQMAN; SCHED; PMDAP; and MASLOG. FMS has additional control for FIVMIN, a zone power consumption forecasting and control task, and TMPO, a task to distribute temperature information to all microcomputers. These tasks are not included in DEMS, hence the code has been removed.

2.2.2 ALARM

The function of the core resident ALARM task is to poll the building microcomputers for information on any fire, security, or equipment alarms. If any alarms are encountered, messages containing the type of alarm, building number (account number for DEMS), and location in the building, are sent to the operator's console. Alarms are logged in the file ALRMLOG for access by other tasks and the alarms are sent to the event queue manager (EVTQMAN) which transfers them to the event-initiated processor (BOY). Any communication errors or micro software errors resulting from the polling are also logged. The task calculates its next execution time, informs MASTER, and suspends itself. It remains inactive but core resident until MASTER releases it at the appropriate time, and it executes again.

DEMS has an additional capability in its ALARM task which is not found in the corresponding FMS task, that of polling for temperature alarms. Every tenth polling iteration of the ALARM task, it requests information on temperature alarms as well as the other alarms, logs the information, informs the operator, and passes the alarms to the event queue manager task. FMS has these capabilities in the FIVMIN task, which is not included in DEMS.

2.2.3. SCHED

The scheduler task (SCHED) downloads the schedules for the next hour to each building microcomputer that is on-line. The task reads previously established files and transmits the next hour's schedule to the building microcomputers between 5 minutes and 59 minutes after the hour. This timing permits the program to run at a low priority and not interfere with critical programs such as the alarm processor, while permitting the building microcomputers to switch to the desired schedule on the hour. This method also permits the program to more efficiently perform its secondary function, which is to determine if a microcomputer is missing necessary control modules. If the control modules necessary to execute schedules are missing, due to an initial start condition or long power outage, the task downloads those modules. The control modules are stored in the building equipment/sensor files maintained on disc by the minicomputer. The program will download only those control modules not in the microcomputer.

The scheduler task is identical in FMS and DEMS.

2.2.4 EVTQMAN

The function of the event queue manager task (EVTQMAN) is to assure the orderly transfer of events (detected by real-time tasks) to the event-initiated processor task (BOY). EVTQMAN is core resident and remains in a trap wait until it detects receipt of a message. If the message contains an event, the task determines if BOY is active and if not requests MASTER to start it. The event is put in a first-in/first-out queue and the task waits for BOY to request events.

EVTQMAN is similar in both FMS and DEMS.

2.2.5 MASLOG

The purpose of the MASLOG task is to print out the contents of the alarm, micro software error, communication error, temperature alarm, and operator log files. The task is normally executed once each day by the MASTER task. The task prints out, on the line printer, the files for the past day, deletes them, and prepares the files for the current day's logging operations. Upon completion of the printing functions, the program sends a message to MASTER so that MASLOG is not executed again until the next day.

The FMS and DEMS versions of this task differ in two of the five subroutines because of the individual account nature of the Megaplex operation. The routines which print the alarm log (fire, security, and equipment) and the temperature alarm log sort the output by account rather than sequentially listing the file according to the order of event occurrence as is done with communication and micro software errors. The latter two files contain information on system reliability and are utilized internally, but the DEMS presents the other alarm information for customers.

2.2.6 PMDAP

PMDAP is the preventative maintenance data acquisition program. It determines which buildings (microcomputers) are on line, the modules executing in those microcomputers, and then polls the micro for data from its equipment tables for the total number of starts and total run-time. The task then updates the information in the building equipment files. PMDAP is executed by MASTER once each day.

There are no differences between the FMS and DEMS versions of PMDAP.

2.2.7 EVENTMGR

The purpose of this task is to assure that the creation of event modules in the command processor task COMPREV and the execution of event modules by the event-initiated processor (BOY) do not attempt simultaneous processes with regard to a module, thereby negatively impacting the system. EVENTMGR manages the event module file and controls which event modules are known to the system.

DEMS and FMS versions will be identical.

2.2.8 BOY

This task is the event-initiated-processor. It processes user-defined modules which are to be executed at specified times of the day or if an event occurs, such as a fire alarm. BOY is activated at regular time intervals to check for time-based modules, or the EVTQMAN task can request MASTER to activate it randomly whenever events occur. When BOY is activated, it sends a message to EVTQMAN asking for outstanding alarms. It then communicates with EVENTMGR to determine if a corresponding module exists or if a time-based module is due to be processed. If the module exists, EVENTMGR grants access to the module and BOY interprets the compiled instructions and executes the module. After processing all requisite modules it terminates. FMS and DEMS are identical.

2.2.9 SYSREAD

The purpose of the task is to perform console input on behalf of the command processor with interruption capabilities by real-time tasks. Whenever a command processor task needs to do input, it sends a message to SYSREAD containing a unique key. SYSREAD gets the input and places it in a task common location, identifiable by the unique key; and then releases the command processor task, which has been in a HOLD state. This configuration permits a real-time task to output information to the terminal. The real-time task sends a message to SYSREAD which frees the terminal. Upon completion of the output, another message is sent to SYSREAD which reassigns the terminal for its needs. It is a core resident task.

The task does not differ in the DEMS and FMS versions.

2.3 Command Processor Module

2.3.1 COMPR

The Command Processor (COMPR) is currently composed of a memory-resident task which performs the following functions:

1. Displays a sign-on message
2. Asks for an operator identification
3. Verifies operator identification
4. Prompts for commands
5. Checks operator security level vs. command

6. Loads in one of the sub-command processors
7. Executes the sub-command
8. Logs the command in operator log file
9. When operator logs off, displays logout message

It is also composed of several other sub-command tasks which perform various functions on different portions of the database. Currently, there are five such sub-command processors:

1. "Low-level" commands
2. "High-level" commands
3. Commands to manipulate the schedule database
4. Commands to mainipulate the equipment and sensor database
5. Commands to manipulate event modules

These sub-command tasks are described in more detail below. Certain commands, because of their ease of implementation, are contained in the memory-resident task COMPR. These commands are: DATE, TIME, OFF, and BYE.

2.3.2 COMPRLO

The "low-level" command processor executes commands which can be executed by an operator. These commands are broken into several functional areas, each one executed by a different overlay segment of the low-level command processor.

ADD — Add a building to the system. This effectively allows the real-time programs to communicate with the particular building.

DELETE — Remove a building from the real-time system. No more communications will be allowed with the building that has just been deleted. The building is still in the database.

CHANGE — Allows modification of many portions of the database, as well as direct communication with any of the building micros to modify a device state. Portions of the database that can be changed are: the 7-day default schedule file, the 30-day schedule file, the state of a building (see ADD and DELETE). Through direct communication with a building micro, an Equipment Table value or a device state can be modified, to ON, OFF, or a particular value.

DISPLAY — An operator can display on his console any of the following log files: communication errors; micro-processor software errors; temperature alarms; or fire, security, and equipment alarms.

STATUS — Via access to portions of the database or via communications with particular building microprocessors, the status of the following can be determined: whether a building is on-line or off-line to the real-time system, whether a piece of equipment is ON or OFF, the value of a sensor (including temperature, humidity, or pressure sensors), the value of an equipment variable in a module, or the value of all sensors for a particular module.

LIST — The list command can list those buildings which are on-line to

LIST — The list command can list those buildings which are on-line to the real-time system; or can examine portions of the database, in particular, any schedule in any of the schedule files, and any segment of a control module in any equipment file. More details are given on the latter two in explanations of the SCHED and EQUIP sub-command processors, respectively.

HELP — Displays information about the commands that are allowed to be executed by a "low-level" operator. In addition, the operator can request help for a particular command.

SHED/RESTORE — This command is executed to shed and/or restore certain pieces of operating equipment via their controlling modules in building microprocessors. In addition, this command allows temperatures to be floated and/or restored in buildings by similar means.

SEND — Allows an operator to send a schedule down to a building microprocessor for a particular module, or for all modules in that building.

MODULE — The MODULE sub-command processor allows one who is intimately knowledgeable of the microprocessor and its environment to control what is going on in a microprocessor. This includes single module capabilities such as: IGNORE a module, turn a module OFF and IGNORE it, make a module a candidate for SHEDDING, SHED a module, restore a SHEDDED module, disable module SHEDDING, enable floating a control variable CTHC in a module, delete a module, download a module, check existence of a module in a building microprocessor, and send the current temperature to a module in equipment variable TMPO. Multi-module or microprocessor commands are also available: start control equations, delete all modules, download all modules, start all modules, send the current temperature to all modules via the equipment variable TMPO, and send the current time to a microprocessor. In addition, the temperature can be sent to all building microprocessors which are currently on-line.

2.3.3 COMPRHI

The "high-level" command processor executes those commands which have been labeled as "dangerous" or "secure." Only operators with a "high-level" access can execute these commands, which are broken down into functional areas, each one of which is executed by a different overlay segment.

MANAGE — Manipulates the HELP file so that it can be used by the Command Processor.

SECURE — Allows additions, deletions, and modifications to the security file. Operator numbers, passwords, security levels, and names of operators are kept in the security file.

INSPECT — Only an operator with an intimate knowledge of the entire overall system should have access to this command. The INSPECT command allows access to a global common area in memory which affects the proper running of the entire real-time system.

2.3.4 COMPREQ

The EQUIP commands are implemented as various overlays in the COMPREQ sub-command processor. The commands that are available are ADD, EDIT, CROSS, and INSERT.

ADD — Allows an operator to add a new control module to the equipment file for a particular building. The information needed includes such items as: equipment constraints and power, any sensors and their definitions, and the actual control equations written in FORCE.

EDIT — With the EDIT command, an operator can edit any of the control module segments: equipment, sensors, or control equations, as well as add or delete sensors. A major subroutine (KONED) is used to edit the FORCE control equations.

CROSS — Is used to cross-assemble a completed control equation segment by invoking the FORCE cross-assembler. In the process of compiling the control equations, the equipment and sensor data portions of the control module are accessed so that the final pseudo-codes generated will contain all the information needed by the microprocessor.

INSERT Allows an operator to insert a new control module in the equipment file for a particular building. This command is similar to the ADD command.

For each of these commands, the operator will be prompted for any data that the system may require.

2.3.5 COMPRSC

Like the EQUIP commands, the SCHED commands are implemented as various overlays in the COMPRSC sub-command processor. The commands that are available are: ADD, EDIT, INSERT, and DELETE.

ADD Is used to build a daily schedule for a control module and to add that schedule to a daily schedule file for a building which is not yet in that schedule file.

EDIT Allows an operator to edit individual hourly schedules within a daily schedule file. Any number of hourly schedules may be edited in a session.

INSERT Is used to insert schedules into a daily schedule file.

DELETE Allows the operator to delete a daily schedule from a daily schedule file.

The operator is prompted with self-explanatory messages so that he may enter exactly what the program needs.

2.3.6 COMPREV

The EVENT sub-command processor consists of several overlays where each one implements the various EVENT commands. Commands to be implemented are: EDIT, COMPILE and INSTALL.

EDIT	Allows the operator to create an Event Module. In addition, after the initial creation of an Event Module, it can be edited with the Event Module Editor.
COMPILE	Compiles an Event Module into a form that can be used by the Event Initiated Program.
INSTALL	Properly places some control information about a compiled Event Module into the database so that the Event Initiated Program can access it as a result of an event.

In all cases, the operator is prompted by the system for the necessary information that it wants.

2.3.7 DEMS and FMS Comparison

The command processor modules in the FMS and DEMS versions are nearly identical. The areas of difference include the file containing the signon and signoff messages; the temperature units utilized by the system; and the CPRINT statement requested by Megaplex. CPRINT is a statement which permits the micro to simulate any valid central site minicomputer request or instruction. The information is returned to the requesting module for subsequent processing. Changes in the command processor module are in the COMPREQ task which contains the equipment module compiler. The compiler was modified to accept CPRINT as a valid statement, and the corresponding files of tokens or op codes were similarly modified. Megaplex also wished to use Fahrenheit degrees instead of Centigrade, so the appropriate modifications were made in the conversion routines in task COMPRSC.

Appendix B shows the structure of the real-time and command processor tasks. It includes the calling sequence of both user-written and system subroutines along with the file names where the code resides. Descriptions of the system library subroutines can be obtained from Perkin-Elmer supplied manuals. Appendix C includes descriptions of the purpose of user-written subroutines.

3.0 SUMMARY

It is appropriate to emphasize the functional nature of the FMS/DEMS systems. It is a data collection, evaluation, and control system which has been flexibly designed to interface with numerous types of equipment, sensors, and controllers and to permit the user to define the control strategy. The important implication of these attributes is that the user defines the overall functional nature of the system (i.e. energy management).

Most of the tasks in the minicomputer and microcomputer are generic processes designed to implement the flexible control strategy. Examples of such tasks include editors, compilers, a control language communications handlers, task controllers, and a pseudo-code interpreter to execute compiled control modules. Other tasks, however, are specific to the energy management function of the system such as the five minute tasks in both the mini and microcomputers which retrieve and evaluate power consumption information. Finally, the system is designed for operating in a real-time, stand-alone mode once the necessary control data have been entered; however, for additional flexibility, parallel capabilities for on-line intervention by operators are included.

As a result of the general nature of the system, the conversion of FMS to the DEMS operating environment was not a major task. It was facilitated by both being implemented on Perkin-Elmer equipment, in the same language, and with similar operating systems. Some of the conversion work involved removing FMS specific file volume names and making the code reflect the desired DEMS peripheral device configuration. The latter was modified to allow for some expansion in the future.

The most prominent functional difference is the absence of some FMS tasks from the DEMS version (see Appendix A). These tasks relate to the Georgia Tech campus configuration needs, specifically the central control of power consumption. DEMS utilizes control module capabilities to achieve energy savings in each individual building microcomputer.

The other modifications mentioned in this report do not affect the functional similarity of the two systems. In one case, data were reordered to facilitate providing information to individual customers versus one report for the Georgia Tech campus, but both systems are logging information to evaluate system reliability and to record specific events. In other cases, functions found in an FMS task were moved into another task in DEMS, because the original task was not transferred to DEMS, but the component was desired. DEMS is an commercial, time-sharing application of FMS.

Future modifications are planned for the event-initiated portions of the system, and these changes impact both the command processor and real-time tasks due to the interactions. Briefly, events are user defined and can be time based or represent a condition (i.e., fire alarm). The user creates modules (programmed actions to be taken if and when the event occurs) using a control language (FORCE-B) and the event module editor. The modules are then compiled and placed in a file of executable modules. A real-time task, EVENTMGR, controls access to the file for both the command processor task placing the module in the file and the real-time task BOY which executes the module. The master controlling program loads and starts BOY at specificed intervals to check for time-based modules, or when requested to start it by EVTQMAN when a condition has been detected. EVENTMGR has yet to be implemented. When it is completed, BOY and the command processor task COMPREV must be modified to coordinate the access to event module files. It has also been proposed that the control language FORCE-B be enhanced thereby adding capabilities. If these changes are made, the compiler and BOY need corresponding modifications to process the instructions. Finally, BOY currently needs to be enhanced.

APPENDIX A

FMS/DEMS DIFFERENCES

Real-Time Module Tasks

FMS	DEMS VERSION
MASTER	17% of the code removed other portions identical
ALARM	code added for temperature alarm processing 5% more code
MASLOG	code added for sorting by account 13% more code
SCHED	no difference
PMDAP	no difference
EVTQMAN	no difference
BOY	no difference
TMPØ	not a DEMS task
FIVMIN	not a DEMS task

TOTAL - DEMS version consists of approximately 80% of the FMS code.

Command Processor Module Tasks

FMS	DEMS VERSION
SYSREAD	no difference
COMPR	no difference
COMPRLO	utilizes different temperature units (F^o/C^o)
COMPRHI	no difference
COMPREQ	added capability to compiler to accept and process CPRINT command as an addition to the FORCE language. token files also reflect CPRINT addition. additional code $<$ 1%
COMPRSC	utilizes different temperature units
COMPREV	no difference
EVENTSUB	no difference

TOTAL - with the exeption of CPRINT, temperature units, and the log-on files, the versions are identical.

FMS/DEMS DIFFERENCES
(continued)

Micro Computer Tasks

FMS	DEMS VERSION
System Clock	no difference
Central Site Communications	no difference
Central Site Request Decode	no difference
Pseudo Code	modified to accept CPRINT command. takes message generated by PRINT command and reroutes to central site request decode
Five Minute Information	no difference
Local Operator Communication	no difference
Local Monitor	no difference
Sensor Interface	no difference
RDT Communication	no difference

TOTAL - Less than 1% of the code differs.

FMS/DEMS DIFFERENCES
(continued)

Micro Computer Tasks

FMS	DEMS VERSION
System Clock	no difference
Central Site Communications	no difference
Central Site Request Decode	no difference
Pseudo Code	modified to accept CPRINT command. takes message generated by PRINT command and reroutes to central site request decode
Five Minute Information	no difference
Local Operator Communication	no difference
Local Monitor	no difference
Sensor Interface	no difference
RDT Communication	no difference

TOTAL - Less than 1% of the code differs.

Appendix B

CALLING SEQUENCES OF TASKS

```
MASTER    [ MASTER.FLX/2 ]
.   DEFLST    [ F7RTL ]
.   SETQUE    [ F7SVCS.CAL/9 ]
.   SVC9      [ F7SVCS.CAL/9 ]
.   OPENW     [ F7RTL ]
.   TIME      [ F7RTL ]
.   DATE      [ F7RTL ]
.   GBYTE     [ F7SVCS.CAL/9 ] (FUNCTION)
.   ICLOCK    [ F7RTL ]
.   SVC2      [ F7SVCS.CAL/9 ]
.   TRAPW     [ F7SVCS.CAL/9 ]
.   MERGE     [ F7SVCS.CAL/9 ]
.   ERRCOD    [ ERRCOD.FLX/2 ]
.   CLOSE     [ F7RTL ]
.   RTL       [ F7RTL ]
.   STTSK     [ F7RTL ]
.   RELSE     [ F7RTL ]
.   LOAD      [ F7RTL ]
.   START     [ F7RTL ]
.   CFILW     [ F7TRL ]
...FIN
```

```
ALARM      [ ALARMFMS.FLX/2 ]
.   OPENW      [ F7RTL ]
.   CLOSE      [ F7RTL ]
.   SNDRCV     [ SNDRCV3.FLX/2 ]
.   .   ASITAM     [ F7SVCS.CAL/9 ]
.   .   WAIT       [ F7RTL ]
.   .   SVC9       [ F7SVCS.CAL/9 ]
.   .   SVC15      [ F7SVCS.CAL/9 ]
.   .   MERGE      [ F7SVCS.CAL/9 ]
.   .   SVC2       [ F7SVCS.CAL/9 ]
.   .   MOVBYT     [ F7SVCS.CAL/9 ]
.   .   GCRC16     [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   GBYTE      [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   DEITAM     [ F7SVCS.CAL/9 ]
.   .   RTL        [ F7RTL ]
.   .   DEFLST     [ F7RTL ]
.   .   SETQUE     [ F7SVCS.CAL/9 ]
.   .   GADDR      [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   TRAPW      [ F7SVCS.CAL/9 ]
.   ...FIN
.   ERPROC     [ ALARMFMS.FLX/2 ]
.   MICERR     [ ALARMFMS.FLX/2 ]
.   ALARM      [ ALARMFMS.FLX/2 ]
.   .   QUEUE
.   ...FIN
.   DATE       [ F7RTL ]
.   TIME       [ F7RTL ]
.   PRINT      [ ALARMFMS.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   ERRCOD     [ ERRCOD.FLX/2 ]
.   .   CLOSE      [ F7RTL ]
.   .   QUEUE      [ F7RTL ]
.   ...FIN
.   LMICER     [ ALARMFMS.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   ERRCOD     [ ERRCOD.FLX/2 ]
.   .   CLOSE      [ F7RTL ]
.   .   QUEUE      [ F7RTL ]
.   ...FIN
.   ALRLOG     [ ALARMFMS.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   ERRCOD     [ ERRCOD.FLX/2 ]
.   .   TEMPF      [ ALARMFMS.FLX/2 ]
.   .   .   OPENW      [ F7RTL ]
.   .   .   ERRCOD     [ ERRCOD.FLX/2 ]
.   .   .   SNDMSG     [ F7RTL ]
.   .   .   WAIT       [ F7RTL ]
.   .   .   CLOSE      [ F7RTL ]
.   .   .   QUEUE      [ F7RTL ]
.   .   ...FIN
.   .   SNDMSG     [ F7RTL ]
.   .   WAIT       [ F7RTL ]
.   .   CLOSE      [ F7RTL ]
.   ...FIN
.   ICLOCK     [ F7RTL ]
.   QUEUE      [ F7RTL ]
.   HOLD       [ F7RTL ]
...FIN
```

```
SCHED        [ SCHED.FLX/2 ]
.   TIME         [ F7RTL ]
.   ONLINE       [ ONLINE.FLX/2 ]    (FUNCTION)
.   .   RTEST        [ F7RTL ]    (FUNCTION)
.   ...FIN
.   DATE         [ F7RTL ]
.   OPENW        [ F7TRL ]
.   QUEUE        [ F7TRL ]
.   CLOSE        [ F7TRL ]
.   JULIAN       [ JULIAN.FLX/2 ]    (FUNCTION)
.   WEEKDAY      [ JULIAN.FLX/2 ]    (FUNCTION)
.   ICLOCK       [ F7RTL ]
.   SNDRCV       [ SNDRCV3.FLX/2 ]
.   .   ASITAM       [ F7SVCS.CAL/9 ]
.   .   WAIT         [ F7RTL ]
.   .   SVC9         [ F7SVCS.CAL/9 ]
.   .   SVC15        [ F7SVCS.CAL/9 ]
.   .   MERGE        [ F7SVCS.CAL/9 ]
.   .   SVC2         [ F7SVCS.CAL/9 ]
.   .   MOVBYT       [ F7SVCS.CAL/9 ]
.   .   GCRC16       [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   CBYTE        [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   DEITAM       [ F7SVCS.CAL/9 ]
.   .   RTL          [ F7RTL ]
.   .   DEFLST       [ F7RTL ]
.   .   SETPRI       [ F7SVCS.CAL/9 ]
.   .   GANDR        [ F7SVCS.CAL/9 ]    (FUNCTION)
.   .   TRANS        [ F7SVCS.CAL/9 ]
.   ...FI
.   MODSUB       [ MODSUB.FLX/2 ]
.   .   OPENW        [ F7RTL ]
.   .   TIME         [ F7RTL ]
.   .   CLOSE        [ F7RTL ]
.   .   SNDRCV       [ SNDRCV3.FLX ]
.   .   WRITE        [ F7RTL ]
.   .   WAIT         [ F7RTL ]
.   ...FI
.   EXIT         [ F7RTL ]
...FI
```

```
MASLOG    [ MASLOG.FLX/2 ]
.   OPENW     [ F7RTL  ]
.   CARCON [ F7RTL ]
.   DATE      [ F7RTL  ]
.   PALOG     [ MASLOG.FLX/2  ]
.   .   OPENW     [ F7RTL  ]
.   .   WAIT      [ F7RTL  ]
.   .   CLOSE     [ F7RTL  ]
.   .   DFILW     [ F7RTL  ]
.   .   CFILW     [ F7RTL  ]
.   ...FIN
.   PTEMPL    [ MASLOG.FLX/2  ]
.   .   OPENW     [ F7RTL  ]
.   .   WAIT      [ F7RTL  ]
.   .   CLOSE     [ F7RTL  ]
.   .   DFILW     [ F7RTL  ]
.   .   CFILW     [ F7RTL  ]
.   ...FIN
.   PMIKEP    [ MASLOG.FLX/2  ]
.   .   OPENW     [ F7RTL  ]
.   .   WAIT      [ F7RTL  ]
.   .   CLOSE     [ F7RTL  ]
.   .   DFILW     [ F7RTL  ]
.   .   CFILW     [ F7RTL  ]
.   ...FIN
.   PCOLOG    [ MASLOG.FLX/2  ]
.   .   OPENW     [ F7RTL  ]
.   .   WAIT      [ F7RTL  ]
.   .   CLOSE     [ F7RTL  ]
.   .   DFILW     [ F7RTL  ]
.   .   CFILW     [ F7RTL  ]
.   ...FIN
.   LGPRNT    [ MLLGPRNT.FLX/2  ]
.   .   OPENW     [ F7RTL  ]
.   .   CLOSE     [ F7RTL  ]
.   .   DFILW     [ F7RTL  ]
.   ...FIN
.   CLOSE     [ R7RTL  ]
.   QUEUE     [ F7RTL  ]
...FIN
```

```
EVTQMAN     [ EVTQMAN.FLX/2 ]
.   OPENW     [ F7RTL ]
.   CFILW     [ F7RTL ]
.   DEFLST    [ F7RTL ]
.   SETQUE    [ F7SVCS.CAL/9 ]
.   MSGLNK    [ F7RTL ]
.   SVC9      [ F7SVCS.CAL/9 ]
.   SNDMSG    [ F7RTL ]
.   WAIT      [ F7RTL ]
.   TRAPW     [ F7SVCS.CAL/9 ]
.   RTL       [ F7RTL ]
.   GETMSG    [ F7RTL ]
.   MSGLKA    [ F7RTL ]
.   STTSK     [ F7RTL ]
.   QUEUE     [ F7TRL ]
...FIN
```

```
SYSREAD    [ SYSREAD.FLX/2 ]
.   TRAPW      [ F7SVCS.CAL/9  ]
.   RTL        [ F7RTL  ]
.   MOVBYT     [ F7SVCS.CAL/9  ]
.   RELSE      [ F7RTL  ]
.   CLOSE      [ F7RTL  ]
.   GETMSG     [ F7RTL  ]
.   MSGLKA     [ F7RTL  ]
.   OPENW      [ F7RTL  ]
.   SYSIO      [ F7RTL  ]
.   DEFLST     [ F7RTL  ]
.   SETQUE     [ F7SVCS.CAL/9  ]
.   MSGLNK     [ F7RTL  ]
.   SVC9       [ F7SVCS.CAL/9  ]
...FIN
```

```
COMPR    [ COMPR.FLX/2 ]
.   OPENW      [ F7RTL ]
.   CFILW      [ F7RTL ]
.   PROMPT     [ PROMPT.FLX/2 ]
.   .   SYSIO      [ F7RTL ]
.   .   SREAD      [ SYSREAD.FLX/2 ]
.   ...FIN
.   CLOSE      [ F7RTL ]
.   TRAPW      [ F7SVCS.CAL/9 ]
.   SVC9       [ F7SVCS.CAL/9 ]
.   RTL        [ F7RTL ]
.   LOAD       [ F7RTL ]
.   START      [ F7RTL ]
.   DEFLST     [ F7RTL ]
.   SETQUE     [ F7SVCS.CAL/9 ]
.   MSGLNK     [ F7RTL ]
.   SVC9       [ F7SVCS.CAL/9 ]
.   SNDMSG     [ F7RTL ]
.   WAIT       [ F7RTL ]
.   DATE       [ F7RTL ]
.   TIME       [ F7RTL ]
...FIN
```

```
COMPRLO    [ COMPRLO.FLX/2 ]
.  DEFLST    [ F7RTL ]
.  SETQUE    [ F7SVCS.CAL/9 ]
.  MSGLNK    [ F7RTL ]
.  SVC9      [ F7SVCS.CAL/9 ]
.  TRAPW     [ F7SVCS.CAL/9 ]
.  OPENW     [ F7RTL ]
.  IFETCH    [ F7RTL ]
.  CLOSE     [ F7RTL ]
.  CHANPR    [ CHAN1.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  HELPPR    [ HELPPR.FLX/2 ]
.  .  EQIDIN    [ EQID1.FLX/2 ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  CLOSE     [ F7RTL ]
.  .  D30ED     [ D30ED.FLX/2 ]
.  .  .  RDINPT    [ RDINPT.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  CFILW     [ F7RTL ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  .  DATE      [ F7RTL ]
.  .  ...FIN
.  .  D7EDIT    [ D7EDIT.FLX/2 ]
.  .  .  RDINPT    [ RDINPT.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CFILW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  ...FIN
.  ...FIN
.  DISPPR    [ DISP1.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  HELPPR    [ HELPPR.FLX/2 ]
.  .  LOGCOM    [ COML.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  .  LOGMIC    [ MODMICL.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  .  LOGTEM    [ TEML.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  .  LOGFSE    [ FSEL.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  ...FIN
.  ADDPR     [ ADD1.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  HELPPR    [ HELPPR.FLX/2 ]
.  .  OPENW     [ F7RTL ]
```

```
.  .  QUEUE      [ F7RTL ]
.  .  WAIT       [ F7RTL ]
.  .  CLOSE      [ F7RTL ]
.  ...FIN
.  DELEPR    [ DELE1.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  HELPPR     [ HELPPR.FLX/2 ]
.  .  OPENW      [ F7RTL ]
.  .  CLOSE      [ F7RTL ]
.  .  QUEUE      [ F7RTL ]
.  ...FIN
.  SHEDPR    [ SHEDNEW.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  HELPPR     [ HELPPR.FLX/2 ]
.  .  SNDRCV     [ SNDRCVT.FLX/2 ]
.  .  OPENW      [ F7RTL ]
.  .  CLOSE      [ F7RTL ]
.  ...FIN
.  STATPR    [ STAT1.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  HELPPR     [ HELPPR.FLX/2 ]
.  .  GETID      [ GETID1.FLX/2 ]
.  .  .  OPENW      [ F7RTL ]
.  .  .  ERRCOD     [ ERRCOD.FLX/2 ]
.  .  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  .  CLOSE      [ F7RTL ]
.  .  ...FIN
.  .  SNDRCV     [ SNDRCVT.FLX/2 ]
.  .  OPENW      [ F7RTL ]
.  .  ERRCOD     [ ERRCOD.FLX/2 ]
.  .  LOCATE     [ LOCATE.FLX/2 ]
.  .  CLOSE      [ F7RTL ]
.  ...FIN
.  LISTPR    [ LIST1.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  HELPPR     [ HELPPR.FLX/2 ]
.  .  SCREAD     [ SCREAD.FLX/2 ]
.  .  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  .  OPENW      [ F7RTL ]
.  .  .  SCDISP     [ SCDISP.FLX/2 ]
.  .  .  CLOSE      [ F7RTL ]
.  .  .  RDINPT     [ RDINPT.FLX/2 ]
.  .  .  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  .  ...FIN
.  .  .  EQIDIN     [ EQID1.FLX/2 ]
.  .  .  SCPRT      [ SCPRT.FLX/2 ]
.  .     ERRCOD     [ ERRCOD.FLX/2 ]
.  .  ...FIN
```

```
.  .  EQREAD    [ EQREAD.FLX/2 ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  .  REPORT    [ REPORT.FLX/2 ]
.  .  .  KONED     [ KON1.FLX/2 ]
.  .  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  .  .  FITTER    [ KONFIT.FLX/2 ]
.  .  .  .  CFILW     [ F7RTL ]
.  .  .  .  OPENW     [ F7RTL ]
.  .  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  .  CLOSE     [ F7RTL ]
.  .  .  .  DFILW     [ F7RTL ]
.  .  .  .  GETOK     [ KON2.FLX/2 ]
.  .  .  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  .  .  GETINC    [ KON2.FLX/2 ]
.  .  .  .  .  OPENW     [ F7RTL ]
.  .  .  .  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  .  .  .  CLOSE     [ F7RTL ]
.  .  .  .  ...FIN
.  .  .  ...FIN
.  .  .  RDINPT    [ RDINPT.FLX/2 ]
.  .  ...FIN
.  .  OPENW     [ F7RTL ]
.  .  CLOSE     [ F7RTL ]
.  ...FIN
.  MODUPR    [ MODU1.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  EQIDIN    [ EQID1.FLX/2 ]
.  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  MODDN     [ MODDN.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  ...FIN
.  .  MODALL    [ MODALL.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  TIME      [ F7RTL ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  ...FIN
.  .  TIME      [ F7RTL ]
.  .  STRALL    [ STRALL.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  ...FIN
```

```
.  .  TMPD       [ MODU1.FLX/2 ]
.  .  .  SNDRCV    [ SNDRCVT.FLX/2 ]
.  .  .  EQIDIN    [ EQID1.FLX/2 ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  ...FIN
.  .  OPENW      [ F7RTL ]
.  .  CLOSE      [ F7RTL ]
.  ...FIN
.  PRINPR     [ PRIN1.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  HELPPR     [ HELPPR.FLX/2 ]
.  ...FIN
.  SENDPR     [ SEND1.FLX/2 ]
.  .  SREAD      [ SYSREAD.FLX/2 ]
.  .  TIME       [ F7RTL ]
.  .  STTSK      [ F7RTL ]
.  .  EQIDIN     [ EQID1.FLX/2 ]
.  .  RDINPT     [ RDINPT.FLX/2 ]
.  .  DATE       [ F7RTL ]
.  .  OPENW      [ F7RTL ]
.  .  ERRCOD     [ ERRCOD.FLX/2 ]
.  .  CLOSE      [ F7RTL ]
.  .  SNDRCV     [ SNDRCVT.FLX/2 ]
.  ...FIN
.  HELPPR     [ HELPPR.FLX/2 ]
.  .  OPENW      [ F7RTL ]
.  .  CLOSE      [ F7RTL ]
.  ...FIN
.  SNDMSG     [ F7RTL ]
.  EXIT       [ F7RTL ]
...FIN
```

```
COMPRHI    [ COMPRHI.FLX/2 ]
.   DEFLST     [ F7RTL ]
.   SETQUE     [ F7SVCS.FLX/9 ]
.   MSGLNK     [ F7RTL ]
.   SVC9       [ F7SVCS.FLX/9 ]
.   TRAPW      [ F7SVCS.FLX/9 ]
.   OPENW      [ F7RTL ]
.   IFETCH     [ F7RTL ]
.   CLOSE      [ F7RTL ]
.   SECMAN     [ SECU1.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   CLOSE      [ F7RTL ]
.   .   RDINP      [ SECU1.FLX ]
.   .   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   ...FIN
.   ...FIN
.   INSPECT    [ CPINSP1.FLX/2 ]
.   .   I4VECT     [ CPINSP1.FLX/2 ]
.   .   .   GETSUB     [ CPINSP1.FLX/2 ]
.   .   .   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   .   .   PUTBYT     [ CPINSP1.FLX/2 ]
.   .   .   .   .   ISBYTE     [ F7RTL ]
.   .   .   .   ...FIN
.   .   .   ...FIN
.   .   .   BINVAL     [ CPINSP1.FLX/2 ]
.   .   .   .   PUTBYT     [ CPINSP1.FLX/2 ]
.   .   .   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   .   .   BSET       [ F7RTL ]
.   .   .   .   BCLR       [ F7RTL ]
.   .   .   ...FIN
.   .   ...FIN
.   .   I2VECT     [ CPINSP1.FLX/2 ]
.   .   .   GETSUB     [ CPINSP1.FLX/2 ]
.   .   .   BINVAL     [ CPINSP1.FLX/2 ]
.   .   ...FIN
.   .   KMRJH      [ CPINSP1.FLX/2 ]
.   .   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   .   I2VECT     [ CPINSP1.FLX/2 ]
.   .   ...FIN
.   .   STRING     [ CPINSP1.FLX/2 ]
.   .   .   PUTBYT     [ CPINSP1.FLX/2 ]
.   .   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   ...FIN
.   ...FIN
.   HLPMAN     [ HLPM1.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   CLOSE      [ F7RTL ]
.   ...FIN
.   SNDMSG     [ F7RTL ]
.   EXIT       [ F7RTL ]
...FIN
```

```
COMPREQ   [ COMPREQ.FLX/2 ]
.  DEFLST    [ F7RTL ]
.  SETQUE    [ F7SVCS.CAL/9 ]
.  MSGLNK    [ F7RTL ]
.  SVC9      [ F7SVCS.CAL/9 ]
.  TRAPW     [ F7SVCS.CAL/9 ]
.  OPENW     [ F7RTL ]
.  IFETCH    [ F7RTL ]
.  CLOSE     [ F7RTL ]
.  EGADD     [ EQADD.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  CFILW     [ F7RTL ]
.  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  CLOSE     [ F7RTL ]
.  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  REPORT    [ REPORT.FLX/2 ]
.  .  KONED     [ KON1.FLX/2 ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  .  FITTER    [ KONFIT.FLX/2 ]
.  .  .  CFILW     [ F7RTL ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  DFILW     [ F7RTL ]
.  .  .  GETOK     [ KON2.FLX/2 ]
.  .  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  .  GETINC    [ KON2.FLX/2 ]
.  .  .  .  OPENW     [ F7RTL ]
.  .  .  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  .  .  CLOSE     [ F7RTL ]
.  .  .  ...FIN
.  .  ...FIN
.  .  RDINPT    [ RDINPT.FLX/2 ]
.  .  PICKUP    [ INCLUDE.FLX/2 ]
.  .  .  OPENW    [ F7RTL ]
.  .  .  ERRCOD   [ ERRCOD.FLX/2 ]
.  .  .  CLOSE    [ F7RTL ]
.  .  ...FIN
.  ...FIN
.  EQEDIT    [ EQEDIT.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  CLOSE     [ F7RTL ]
.  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  REPORT    [ REPORT.FLX/2 ]
.  .  KONED     [ KON1.FLX/2 ]
.  .  RDINPT    [ RDINPT.FLX/2 ]
.  ...FIN
```

```
.  EQCROS    [ EQCROS.FLX/2 ]
.  .  CLOSE     [ F7RTL ]
.  .  RESET     [ EQXASM.FLX/2 ]
.  .  NIBTAB    [ EQXASM.FLX/2 ]
.  .  ERRMES    [ EQXASM.FLX/2 ]
.  .  .  RESET     [ EQXASM.FLX/2 ]
.  .  ...FIN
.  .  FITTER    [ KONFIT.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  DFILW     [ F7RTL ]
.  .  CFILW     [ F7RTL ]
.  ...FIN
.  EQINS    [ EQINS.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  CLOSE     [ F7RTL ]
.  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  REPORT    [ REPORT.FLX/2 ]
.  .  FITTER    [ KONFIT.FLX/2 ]
.  .  RDINPT    [ RDINPT.FLX/2 ]
.  .  PICKUP    [ PICKUP.FLX/2 ]
.  ...FIN
.  DELMOD    [ DELMOD.FLX/2 ]
.  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  LSTMOD    [ LSTMOD.FLX/2 ]
.  .  FITTER    [ KONFIT.FLX/2 ]
.  .  OPENW     [ F7RTL ]
.  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  DELBL     [ DELBL.FLX/2 ]
.  .  .  DFILW     [ F7RTL ]
.  .  .  CFILW     [ F7RTL ]
.  .  .  OPENW     [ F7RTL ]
.  .  .  ERRCOD    [ ERRCOD.FLX/2 ]
.  .  .  CLOSE     [ F7RTL ]
.  .  .  RENAME    [ F7RTL ]
.  .  .  LOCATE    [ LOCATE.FLX/2 ]
.  .  ...FIN
.  .  CLOSE     [ F7RTL ]
.  ...FIN
.  SNDMSG    [ F7RTL ]
.  EXIT      [ F7RTL ]
...FIN
```

```
COMPRSC   [ COMPRSC.FLX/2 ]
.   DEFLST    [ F7RTL ]
.   SETQUE    [ F7SVCS.CAL/9 ]
.   MSGLNK    [ F7RTL ]
.   SVC9      [ F7SVCS.CAL/9 ]
.   TRAPW     [ F7SVCS.CAL/9 ]
.   OPENW     [ F7RTL ]
.   IFETCH    [ F7RTL ]
.   CLOSE     [ F7RTL ]
.   SCADD     [ SCADD1.FLX ]
.   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   CFILW     [ F7RTL ]
.   .   ERRCOD    [ ERRCOD.FLX/2 ]
.   .   OPENW     [ F7RTL ]
.   .   CLOSE     [ F7RTL ]
.   .   RDINPT    [ RDINPT.FLX/2 ]
.   .   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   ...FIN
.   ...FIN
.   SCEDIT    [ SCEDIT1.FLX/2 ]
.   .   SCDISP    [ SCDISP.FLX/2 ]
.   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   OPENW     [ F7RTL ]
.   .   ERRCOD    [ ERRCOD.FLX/2 ]
.   .   RDINPT    [ RDINPT.FLX/2 ]
.   .   CLOSE     [ F7RTL ]
.   ...FIN
.   SCINS     [ SCINS1.FLX/2 ]
.   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   OPENW     [ F7RTL ]
.   .   ERRCOD    [ ERRCOD.FLX/2 ]
.   .   CLOSE     [ F7RTL ]
.   ...FIN
.   SCDELE    [ SCDELE1.FLX ]
.   .   SCDISP    [ SCDISP.FLX/2 ]
.   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   OPENW     [ F7RTL ]
.   .   ERRCOD    [ ERRCOD.FLX/2 ]
.   .   RDINPT    [ RDINPT.FLX/2 ]
.   .   EQIDIN    [ EQID1.FLX/2 ]
.   .   CLOSE     [ F7RTL ]
.   ...FIN
...FIN
```

```
COMPREV    [ COMPREV.FLX/2 ]
.  DEFLST    [ F7RTL ]
.  SETGUL    [ F7SVCS.CAL/9 ]
.  MSGLNK    [ F7RTL ]
.  SVC9      [ F7SVCS.CAL/9 ]
.  TRAPW     [ F7SVCS.CAL/9 ]
.  OPENW     [ F7RTL ]
.  IFETCH    [ F7RTL ]
.  EVEDIT    [ EVEDIT.FLX/2 ]
.  .  PROMPT    [ PROMPT.FLX/2 ]
.  .  .  SYSIO     [ F7RTL ]
.  .  .  SREAD     [ SYSREAD.FLX/2 ]
.  .  ...FIN
.  .  OPENW     [ F7RTL ]
.  .  OUTMSG    [ PROMPT.FLX/2 ]
.  .  CFILW     [ F7RTL ]
.  .  GETC      [ GETC.FLX/2 ]
.  .  .  ILBYTE    [ F7RTL ]
.  .  .  ISBYTE    [ F7RTL ]
.  .  ...FIN
.  .  PUTC      [ PUTC.FLX/2 ]
.  .  .  ILBYTE    [ F7RTL ]
.  .  .  ISBYTE    [ F7RTL ]
.  .  ...FIN
.  .  GETOK     [ EVTOKEN.FLX/2 ]
.  .  .  GETC    [ GETC.FLX/2 ]
.  .  .  PUTC    [ PUTC.FLX/2 ]
.  .  ...FIN
.  .  CLOSE     [ F7RTL ]
.  .  DFILW     [ F7RTL ]
.  ...FIN
.  BCCOMP    [ BCCOMP.FLX/2 ]
.  .  GETTOK    [ BCGETTOK.FLX/2 ]
.  .  .  PUTC      [ BCSUBS.FLX/2 ]
.  .  .  .  ILBYTE    [ F7RTL ]
.  .  .  .  ISBYTE    [ F7RTL ]
.  .  .  ...FIN
.  .  .  ERRCMP    [ BCSUBS.FLX/2 ]
.  .  .  GETC      [ BCSUBS.FLX/2 ]
.  .  .  .  ILBYTE    [ F7RTL ]
.  .  .  .  ISBYTE    [ F7RTL ]
.  .  .  ...FIN
.  .  ...FIN
```

```
.   .   CWHEN     [ BCCWHEN.FLX/2 ]
.   .   .   GETTOK    [ BCGETTOK.FLX/2 ]
.   .   .   ERRCMP    [ BCSUBS.FLX/2 ]
.   .   ...FIN
.   .   CDEFIN    [ BCCDEFIN.FLX/2 ]
.   .   .   GETTOK    [ BCGETTOK.FLX/2 ]
.   .   .   ERRCMP    [ BCSUBS.FLX/2 ]
.   .   .   GETEOL    [ BCGETTOK.FLX/2 ]
.   .   .   .   GETTOK    [ BCGETTOK.FLX/2 ]
.   .   .   ...FIN
.   .   .   ADDSYM
.   .   .   .   GETC      [ BCSUBS.FLX/2 ]
.   .   .   .   PUTC      [ BCSUBS.FLX/2 ]
.   .   .   .   ERRCMP    [ BCSUBS.FLX/2 ]
.   .   .   ...FIN
.   .   FIXSYM    [ BCFIXSYM.FLX/2 ]
.   .   .   GETC      [ BCSUBS.FLX/2 ]
.   .   .   PUTC      [ BCSUBS.FLX/2 ]
.   .   ...FIN
.   .   CEXEC     [ BCCEXEC.FLX/2 ]
.   .   .   GETTOK    [ BCGETTOK.FLX/2 ]
.   .   .   GETEOL    [ BCGETTOK.FLX/2 ]
.   .   .   GEXPR     [ BCGEXPR.FLX/2 ]
.   .   .   .   BCSUBS    [ BCSUBS.FLX/2 ]
.   .   .   .   GETTOK    [ BCGETTOK.FLX/2 ]
.   .   .   .   GETEOL    [ BCGETTOK.FLX/2 ]
.   .   .   .   GENCOD    [ BCSUBS.FLX/2 ]
.   .   .   ...FIN
.   .   .   GENCOD    [ BCSUBS.FLX/2 ]
.   .   .   .   GETC      [ BCSUBS.FLX/2 ]
.   .   .   .   PUTC      [ BCSUBS.FLX/2 ]
.   .   .   ...FIN
.   .   .   ERRCMP    [ BCSUBS.FLX/2 ]
.   .   .   ADDSYM    [ BCSUBS.FLX/2 ]
.   .   ...FIN
.   .   ERRCMP    [ BCSUBS.FLX/2 ]
.   .   SREAD     [ SYSREAD.FLX/2 ]
.   .   OPENW     [ F7RTL ]
.   .   CFILW     [ F7RTL ]
.   .   CLOSE     [ F7RTL ]
.   .   DFILW     [ F7RTL ]
.   ...FIN
```

```
.   BBUILD     [ BBUILD.FLX/2 ]
.   .   CLOSE      [ F7RTL ]
.   .   DFILW      [ F7RTL ]
.   .   SREAD      [ SYSREAD.FLX/2 ]
.   .   OPENW      [ F7RTL ]
.   .   RENAME     [ F7RTL ]
.   .   DATE       [ F7RTL ]
.   .   ICLOCK     [ F7RTL ]
.   ...FIN
.   SNDMSG     [ F7RTL ]
.   EXIT       [ F7RTL ]
...FIN
```

Appendix C

SUBROUTINE DESCRIPTIONS

ADDPR - Executes the command processor "ADD" option to add a building on-line; sets the building's status in BLDGS; sets the building information in POWHDR1.

ALARM - Processes the return buffer from the micro on alarm polling; determines the individual types of alarms; informs the operator; and stores alarms in arrays.

ALRLOG - Writes the contents of the fire, security, and equipment alarm array to the file ALRMLOG, if available.

BINVAL - Accesses a string of bits from task common block/SYSTEM for the command processor command INSPECT; displays them; and modifies them if so requested.

CHANPR - Executes the command processor change option; changing an equipment table variable, device state, building state, power consumption level, 30 and 7 day schedule files.

CONEQ - The control equations along with corresponding sensor and equipment information are converted to a packed unit of op codes interpretable by the microcomputer software; it includes the capability to process the CPRINT command.

DELBL - Deletes blank lines in equipment files.

DELEPR - Executes the command processor "DELETE" option to delete a building from the system; changes a building's status in BLDGS and POWHDR1.

DELMOD - Deletes modules from a building equipment file and updates the pointer table.

DISPPR - Executes the "DISPLAY" option when requested by the operator; determines what information (communication errors, micro software errors, temperature alarms, or fire, security, and equipment alarms) is to be displayed.

D7EDIT - Allows an operator to change a schedule name in the 7 day default file.

D30ED - Creates a 30 day schedule file from the 7 day default file, if none exists; displays and allows the operator to change a schedule file name for any day in the 30 day file.

EQADD - Interactive routine for leading operator through input segments necessary to build new equipment files or modules; manages the location of information via the record pointer table.

EQCROS - Verifies that the module information is satisfactory and starts the cross assembler.

EQEDIT - Permits operator to change control modules contained within equipment files; can access all module segments except op codes; manages the location of information via the record pointer table.

EQIDIN - Requests and processes equipment identification code from the operator.

EQINS - Permits operator to add or delete sensors from a control module in an equipment file and updates the pointer table to reflect changes in the file.

EQREAD - Permits the operator to read and list information from equipment modules, specifically equipment identifiers, sensors, and control equations.

ERPROC - Builds an array of communication errors that occur during a polling of building microcomputers; stores date, time, building, and type of communication errors.

ERRCOD - Outputs to the operator any difficulties a task has using F7RTL file handling subroutines.

EVCOMP - Compiler routine for event modules (work to be done).

EVEDIT - A text editor for creating or editing event modules (work to be done).

EVINST - Installs compiled event modules into list of ready modules for the real-time tasks to execute.

FITTER - A file management procedure for inserting or deleting records in equipment files.

GETID - Queries operator for module or sensor ID information.

GETINC - Copies a requested line from a specified module in a specified file into a temporary file.

GETOK - A routine to pack a character string of up to 8 characters into a double precision variable.

GETSUB - Queries operator for an array variable subscript number for INSPECT.

HELPMAN - A routine to manage pointers for the help file.

HELPPR - Opens the help file, reads the information requested, and writes it out.

I2VECT - Accesses INTEGER*2 variables for INSPECT.

I4VECT - Accesses INTEGER*4 variables for INSPECT.

INSPECT - Reads and can change system common variables in task common block /SYSTEM/.

JULIAN - Calculates the julian date.

KMJRH - Processes data arrays (1, 2, or 3 dimensions) for INSPECT; prints out all non-zero elements.

KONED - Editor routine for preparing control modules utilizing FORCE.

LGPRNT - Prints the contents of the operator command log file on the line printer.

LISTPR - Executes the command processor "LIST" function to display schedules equipment, and buildings to the operator.

LMICER - Writes the contents of the micro software errors array to the disc file MIKERR.

LOCATE - Gets the row and column locator pointers from an equipment file pointer directory for a module in the equipment file.

LOGCOM - Displays the contents of the communication error file to the operator.

LOGFSE - Displays the contents of the fire, security, and equipment alarm file to the operator.

LOGMIC - Displays the contents of the micro software error file to the operator.

LOGTEM - Displays the contents of the temperature alarm file to the operator.

LSTMOD - Outputs the module information for EQREAD.

MICERR - Called when a building microcomputer returns a non-zero software error code during a polling; stores the date, time, building, error code, and the header that generated the error in an array.

MODDN - Sends all control modules to micros.

MODSUB - Downloads a specified module to a building when it is determined that module is missing in the microcomputer.

MODUPR - Executes module command processor options; queries for options and parameters; sends instructions to the micro.

NIBTAB - A cross assembler subroutine for placing op codes in a table for writing disc file.

ONLINE - Tests that a building ID matches names in the file BLDGS, and if valid, determines if it is on-line or not.

PALOG - Prints, on the line printer, the contents of the fire, security, and equipment alarm file ALRMLOG.

PCOLOG - Prints, on the line printer, the contents of the communications error file COMLOG.

PICKUP - Looks for a match between sensor ID elements given as parameters and sensor ID's contained in the analog sensor library file.

PMIKER - Prints, on the line printer, the contents of the micro software error file MIKERR.

PRINPR - Executes command processor report printer option (work to be done).

PRINT - Writes the contents of the communication error array to the file COMLOG.

PROMPT - Outputs a prompt for operator input; waits; and gets input via SYSIO and SREAD.

PTEMPL - Prints, on the line printer, the contents of the temperature alarm file TEMPLOG.

PUTBYT - Puts the rightmost byte of a word into a specified location of another specified word.

RDINPT - A free format numeric input routine processing floating point, hexadecimal, exponential, or integer data.

REPORT - Standardized report procedure for equipment file operator subroutines; gives status of module.

RESET - Resets control variables for cross assembler.

SCADD - Adds a new building and schedules to an existing schedule file, or builds a new file; queries operator for input and data verification.

SCDELE - File management procedure to delete a daily schedule from a file; updates the pointer directory.

SCDISP - Displays a selected daily schedule to an operator.

SCEDIT - Interactive editor for schedule files; operates on one-hour information.

SCINS - File management procedure to insert a schedule into a daily schedule file for a building that already has some schedules in the file.

SCPRT - Ouputs the schedule information for SCREAD to the line printer.

SCREAD - Queries operator for the appropriate schedule information; reads the schedule file and displays the information.

SECMAN - Manager of the authorized operator file; executes adding, deleting, changing attributes of operators, and listing the file.

SENDPR - Executes the command processor send schedule command; accesses and verifies schedule information; and warns the operator if the real-time scheduler has not executed in order to avoid overwrites.

SHEDPR - Executes the command processor "SHED" option permitting the operator to shed or restore equipment by priority in a building; float temperature control points by priority in a building; and set and clear the global ignore.

SNDRCV - The communications handler for minicomputer/microcomputer messages.

SREAD - Subroutine for command processor I/O; sends information to task SYSREAD and suspends itself.

STATPR - Executes the command processor "STATUS" option permitting the operator to get the status of a building, sensors, and equipment table variables.

STRALL - Starts all modules in a micro for a given building in the system.

STRING - Access string variables for INSPECT.

TEMPF - If the primary alarm log file ALRMLOG is unavailable, it writes the alarm array information to the temporary disc file TALRLOG.

TMPO - Retrieves temperature value from sensor and sends value to a single module or all modules.

WEEKDAY - Returns a code for the day of the week.