
by
Edgar S. Neely
Robert D. Neathammer
Bill Wang

Maintenance Resource Prediction Models (MRPMs) are a set of models that run on various computer systems to assist Army managers to plan and program maintenance resources, based on the anticipated resource requirements of actual installation facilities, for prediction periods of 1 to 10 years.

This manual provides system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system.

This data base and computer system is presently used by U.S. Army Corps of Engineers (USACE) designers at district and installation levels, and by resource programmers at the USACE Headquarters, Army Major Command, and installation levels. MRPMs may also prove useful to other Government agencies and to the private sector.

Approved for public release; distribution is unlimited.
The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official indorsement or approval of the use of such commercial products. The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED

DO NOT RETURN IT TO THE ORIGINATOR
Maintenance Resource Prediction Models (MRPMs) are a set of models that run on various computer systems to assist Army managers to plan and program maintenance resources, based on the anticipated resource requirements of actual installation facilities, for prediction periods of 1 to 10 years.

This manual provides system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system.

This data base and computer system is presently used by U.S. Army Corps of Engineers (USACE) designers at district and installation levels, and by resource programmers at the USACE Headquarters, Army Major Command, and installation levels. MRPMs may also prove useful to other Government agencies and to the private sector.
FOREWORD

This research was conducted for the Office of the Chief of Engineers (OCE), under various RDTE and FAD funding documents. Work began under RDTE in 1980 and continued in reimbursable projects from 1984 through 1989. The technical monitors were, for the direct-funded part, Dr. Larry Schindler, CEMP-ECE-G, and at the termination of the project, Mr. Ed Davis, CEHSC-FM-R; and for the reimbursable part, Ms. Val Corbridge, DAEN-ZCP-B.

The work was performed by the Facility Systems Division (FS), U.S. Army Construction Engineering Research Laboratory (USACERL). Dr. Edgar S. Neely was the USACERL principal investigator. Dr. Michael J. O'Connor is Chief, USACERL-FS. The USACERL technical editor was Mr. William J. Wune, Information Management Office.

COL Everett R. Thomas is Commander and Director of USACERL, and Dr. L.R. Shaffer is Technical Director.
1 INTRODUCTION

The primary purpose of this manual is to provide the system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system. Chapter 2 describes the most efficient method for learning the functions and organization of the MRPMSS system. Chapter 3 defines the program flow from subroutine to subroutine. Chapter 4 contains the description of all standard or common subroutines that must be used by all programmers when writing new code or modifying existing code. Chapter 5 describes all standard common blocks that must be used when programming. Chapter 6 contains a list of standard programming packages used in the MRPMSS system. Chapter 7 describes the procedures to be followed during system maintenance and operation. Chapter 8 describes the overall management procedures required to operate the system. Chapter 9 describes resources required to support the MRPMSS system.
2 LEARNING THE FUNCTIONS

The first and most important step is to train the maintainer as a functional user of the system. Give the new person a user's manual and access to the MRPMSS system on a personal computer (PC). Have the person read the manual, learn the system, and write down all questions. Do not give the person any verbal description of the system. All information should be contained in the user's manual.

Revise the user's manual as needed using the programmer's questions. If the new person had the question, it is probable others will also. This method constantly improves both user and system documentation.

The user's manual is a self-teach document. The learning process takes approximately 1 week and should be very smooth and efficient. Once the maintainer knows the functions, this programmer's manual can be used.
3 PROGRAM FLOW

This chapter presents the flow of the program by functional use. Table 1 contains the functional use as displayed on the screen, the program named, and the files as tables accessed. Table 2 contains the same information ordered by program name. Table 3 contains the information ordered by file name.

Table 1

MRPM Files
Ordered as Shown on the Screen

<table>
<thead>
<tr>
<th>Function</th>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSTMENU MENU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INSTMENU</td>
<td>. INSTMENU.EXE</td>
<td>INSTMENU.TB1</td>
</tr>
<tr>
<td>MAIN MENU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MRPM</td>
<td>. MRPMSSS.EXE</td>
<td>MRPMTMP:DAT</td>
</tr>
<tr>
<td></td>
<td>. MRPMSSS.EXE</td>
<td>MRPMTMP:BAT</td>
</tr>
<tr>
<td>BASIC INFORMATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GENERAL INFORMATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Organization Chart</td>
<td>. ORGCHRT.EXE</td>
<td>ORGFGC:XDB</td>
</tr>
<tr>
<td>RMF Factors</td>
<td>. RMF-FACT.EXE</td>
<td>RMF-FACT:XDB</td>
</tr>
<tr>
<td></td>
<td>. RMF-FACT.EXE</td>
<td>INSTINFO:DAT</td>
</tr>
<tr>
<td>F4C Conversion Codes</td>
<td>. F4CAMS.EXE</td>
<td>AMSF4C:XDB</td>
</tr>
<tr>
<td>Report Periods</td>
<td>. RP-DATES.EXE</td>
<td>INSTINFO:DAT</td>
</tr>
<tr>
<td></td>
<td>. RP-DATES.EXE</td>
<td>ORGFGC:XDB</td>
</tr>
<tr>
<td>Unit Cost Factors</td>
<td>. UNC-FACT.EXE</td>
<td>UNC-FACT:XDB</td>
</tr>
<tr>
<td></td>
<td>. UNC-FACT.EXE</td>
<td>UNCDSC:XDB</td>
</tr>
<tr>
<td>U.S. Factor Description Editor</td>
<td>. COST-DES.EXE</td>
<td>UNCDSC:XDB</td>
</tr>
<tr>
<td>AMS Description Editor</td>
<td>. AMS-EDIT.EXE</td>
<td>AMSDSC:XDB</td>
</tr>
<tr>
<td>Area Identification</td>
<td>. AREA_TAB.EXE</td>
<td>AREA_TAB:XDB</td>
</tr>
<tr>
<td></td>
<td>. AREA_TAB.EXE</td>
<td>INSTINFO:DAT</td>
</tr>
<tr>
<td></td>
<td>. AREA_TAB.EXE</td>
<td>SUB-TAB:XDB</td>
</tr>
<tr>
<td>Subinstallation Identification</td>
<td>. SUB-TAB.EXE</td>
<td>INSTINFO:DAT</td>
</tr>
<tr>
<td></td>
<td>. SUB-TAB.EXE</td>
<td>SUB-TAB:XDB</td>
</tr>
</tbody>
</table>

The listing of programs is in menu order. For each MRPM function there is a corresponding program file name and data files accessed.
<table>
<thead>
<tr>
<th>Function</th>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lan unit cost graph</td>
<td>CHART.EXE</td>
<td></td>
</tr>
<tr>
<td>FACILITY RESOURCE DATA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F4C Resource Description Table</td>
<td>EDF4CS.EXE</td>
<td>DES-BTSM.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-TASK.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-TRWD.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td>Trade and Costs</td>
<td>TRD-EDIT.EXE</td>
<td>BT-PASS.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRDCOSTS.DAT</td>
</tr>
<tr>
<td>Total/Partial Summary Tasks</td>
<td>DES-EDT3.EXE</td>
<td>BTSMtdpd.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLASLST.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-BTSM.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRDCOSTS.DAT</td>
</tr>
<tr>
<td>F4C Description Editor</td>
<td>F4C-EDIT.EXE</td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td>FACILITY INFORMATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESOURCE CALCULATION</td>
<td>FACASS.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BTSMtdpd.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CTOseq.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RMF-FACT.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMYsecq.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCMDid.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TASKtigi.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRAVTxy.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VALLIST.DAT</td>
</tr>
<tr>
<td>DISPLAY RESOURCES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display Facility Totals</td>
<td>RSMTDPS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SYLCHART.ASC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GENERAL INFORMATION</td>
<td>FA-XEDIS.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Program</td>
<td>Data Files</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA_TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FFPROF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCMDEF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUB_TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WP-DESC.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA_TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FFPROF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCMDEF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUB_TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WP-DESC.DAT</td>
</tr>
<tr>
<td></td>
<td>AMSF4CRP.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td>RSMTREPS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td>URREDIT.EXE</td>
<td>AMSDSC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.URR</td>
</tr>
<tr>
<td></td>
<td>URAPR.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNCFACT.XDB</td>
</tr>
<tr>
<td></td>
<td>URRCOP.EXE</td>
<td>AMSDSC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URAPR.XDB</td>
</tr>
<tr>
<td></td>
<td>VIEWRP.EXE</td>
<td>ZZZZZZZZ.FTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.OSR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.UCR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.URA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.URC</td>
</tr>
<tr>
<td></td>
<td>DATFAC.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td>NGDFAC.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td>NOAFAC.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td>VCORPS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td>DRESTOS.EXE</td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td>COM-FAC.EXE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHART.EXE</td>
<td></td>
</tr>
</tbody>
</table>

**FACILITY REPORTS**

F4C/AMS Organizational Summary

Facility Totals Report

Unconstrained Reqrmnts Reports

URR Editor

Escalation Factor Editor

Produce Reports

View and Print Report Files

**MODEL FACITILITY**

Standard

National Guard

Non Army Organizations

V Corps

DELETE RESOURCE TOTAL FILE

COMBINE FACILITY TOTALS FILES

LAN DISPLAY GRAPH
Table 2

MRPM Files
Ordered by Program Name

<table>
<thead>
<tr>
<th>Function</th>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMS Description Editor</td>
<td>AMS-EDIT.EXE</td>
<td>AMSDSC.XDB</td>
</tr>
<tr>
<td>F4C/AMS Organizational Summary</td>
<td>AMSF4CRP.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>Area Identification</td>
<td>AREA_TAB.EXE</td>
<td>AREA_TAB.XDB</td>
</tr>
<tr>
<td>LAN Unit Cost/Display Graph</td>
<td>CHART.EXE</td>
<td></td>
</tr>
<tr>
<td>COMBINE FACILITY TOTALS FILES</td>
<td>COM-FAC.EXE</td>
<td>UNCDSC.XDB</td>
</tr>
<tr>
<td>U.S. Factor Description Editor</td>
<td>COST-DES.EXE</td>
<td></td>
</tr>
<tr>
<td>Standard</td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>Total/Partial Summary Tasks</td>
<td>DES-EDT3.EXE</td>
<td>BTSmtdpd.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BT-PASS.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLASLST.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-BTSM.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRDCOSTS.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRWDLy--.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VALLIST.DAT</td>
</tr>
<tr>
<td>DELETE RESOURCE TOTAL FILE</td>
<td>DRESTOS.EXE</td>
<td>RSMTTOTL.XDB</td>
</tr>
<tr>
<td>F4C Resource Description Table</td>
<td>EDF4CS.EXE</td>
<td>DES-BTSM.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-TASK.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DES-TRWD.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td>F4C Conversion Codes</td>
<td>F4CAM5.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td>F4C Description Editor</td>
<td>F4C-EDIT.EXE</td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td>RESOURCE CALCULATION</td>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B1SMtdpd.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CTODseq.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RMF-FACT.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMYseqq.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SC MIDid.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TASKtigl.XDB</td>
</tr>
</tbody>
</table>

The listing of programs is in alphabetical order. For each program there is a corresponding MRPM function and data files accessed.
<table>
<thead>
<tr>
<th>Function</th>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENERAL INFORMATION</td>
<td>FA-XEDIS.EXE</td>
<td>TRAVTIME.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRDCOSTS.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TREETseca.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNC-FACT.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AREA_TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F4CPROF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FDCDESC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMCDEF.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUB-TAB.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WP-DESC.XDB</td>
</tr>
<tr>
<td>INSTMENU</td>
<td>INSTMENU.EXE</td>
<td>INSTMENU.TBL</td>
</tr>
<tr>
<td>MRPM</td>
<td>MRPMESS.EXE</td>
<td>MRPTEMP.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MRPTEMP.BAT</td>
</tr>
<tr>
<td>National Guard</td>
<td>NGDFAC.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>Non Army Organizations</td>
<td>NOAFAC.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>Organization Chart</td>
<td>ORGCHRT.EXE</td>
<td>ORGFGC.XDB</td>
</tr>
<tr>
<td>RMF Factors</td>
<td>RMF-FACT.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RMF-FACT.XDB</td>
</tr>
<tr>
<td>Report Periods</td>
<td>RP-DATES.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ORGCHRT.XDB</td>
</tr>
<tr>
<td>Display Facility Totals</td>
<td>RSMTPDS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SYLCHART.ASC</td>
</tr>
<tr>
<td>facility Totals Report</td>
<td>RSMTREPS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td>Subinstallation Identification</td>
<td>SUB-TAB.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUB-TAB.XDB</td>
</tr>
<tr>
<td>Trade and Costs</td>
<td>TRD-EDIT.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRDCOSTS.DAT</td>
</tr>
<tr>
<td>Unit Cost Factors</td>
<td>UNC-FACT.EXE</td>
<td>UNC-FACT.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNCDS.C.XDB</td>
</tr>
<tr>
<td>Escalation Factor Editor</td>
<td>URRAPR.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNC-FACT.XDB</td>
</tr>
<tr>
<td>Produce Reports</td>
<td>URRCOP.EXE</td>
<td>AMSDS.C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSMTOTL.XDB</td>
</tr>
<tr>
<td>Function</td>
<td>Program</td>
<td>Data Files</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>URR Editor</td>
<td>URREDIT.EXE</td>
<td>AMSDSC.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>URR.XDB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZ.URR</td>
</tr>
<tr>
<td>V Corps</td>
<td>VCORPS.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>View and Print Report Files</td>
<td>VIEWRP.EXE</td>
<td>ZZZZZZZZ.FTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.OSR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.UCR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.URA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZZZZZZZ.URC</td>
</tr>
</tbody>
</table>
Table 3
MRPM Program Files Ordered by Data File Name

<table>
<thead>
<tr>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMS-EDIT.EXE</td>
<td>AMSDSC.XDB</td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td></td>
</tr>
<tr>
<td>URREDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>AMSF4CRP.EXE</td>
<td>AMSF4C.XDB</td>
</tr>
<tr>
<td>F4CAM.S.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td></td>
</tr>
<tr>
<td>AREA_TAB.EXE</td>
<td>AREA-TAB.XDB</td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td></td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>BTSMdtpd.XDB</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>BT-PASS.DAT</td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>CLASLST.DAT</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>CTODsecq.XDB</td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>DES-BTSM.XDB</td>
</tr>
<tr>
<td>EDF4C.EXE</td>
<td></td>
</tr>
<tr>
<td>EDF4CS.EXE</td>
<td>DES-TASK.XDB</td>
</tr>
<tr>
<td>EDF4CS.EXE</td>
<td>DES-TRWD.XDB</td>
</tr>
<tr>
<td>EDF4CS.EXE</td>
<td>F4CDESC.XDB</td>
</tr>
<tr>
<td>F4C-EDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td></td>
</tr>
<tr>
<td>EDF4CS.EXE</td>
<td>F4C-YEAR.XDB</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>AMSF4CRP.EXE</td>
<td>FACILITY.XDB</td>
</tr>
<tr>
<td>DATFAC.EXE</td>
<td></td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>NGDFAC.EXE</td>
<td></td>
</tr>
<tr>
<td>NOAFAC.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTPS.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTPREPS.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td></td>
</tr>
<tr>
<td>VCORPS.EXE</td>
<td></td>
</tr>
</tbody>
</table>

The listing of data file name is in alphabetical order. Each data file has corresponding execut files.
<table>
<thead>
<tr>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA-XEDIS.EXE</td>
<td>FFPREF.XDB</td>
</tr>
<tr>
<td>AMSF4CRP.EXE</td>
<td>INSTINFO.DAT</td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td></td>
</tr>
<tr>
<td>EDF4CS.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td></td>
</tr>
<tr>
<td>RMF-FACT.EXE</td>
<td></td>
</tr>
<tr>
<td>RP-DATES.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTDPS.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTREPS.EXE</td>
<td></td>
</tr>
<tr>
<td>SUB-TAB.EXE</td>
<td></td>
</tr>
<tr>
<td>TRD-EDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>URRAPR.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td></td>
</tr>
<tr>
<td>URREDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>INSTMEMU.EXE</td>
<td>INSTMEMU.TB</td>
</tr>
<tr>
<td>MRPMSSS.EXE</td>
<td>MRPMTEMP.BAT</td>
</tr>
<tr>
<td>MRPMSSS.EXE</td>
<td>MRPMTEMP.DAT</td>
</tr>
<tr>
<td>ORGCHRT.EXE</td>
<td>ORGFGC.DEB</td>
</tr>
<tr>
<td>RP-DATES.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>RMF-FACT.XDB</td>
</tr>
<tr>
<td>RMF-FACT.EXE</td>
<td></td>
</tr>
<tr>
<td>AMSF4CRP.EXE</td>
<td>RSMTTOTAL.XDB</td>
</tr>
<tr>
<td>DRESTOS.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTDPS.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTREPS.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>RSMYSEQ.XDB</td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td>SCMDEF.XDB</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>SCMDID.DAT</td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td>SUB-TAB.XDB</td>
</tr>
<tr>
<td>SUB-TAB.EXE</td>
<td></td>
</tr>
<tr>
<td>RSMTDPS.EXE</td>
<td>SYLCHART.ASC</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>TASKTIGI.XDB</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>TRAVTIME.DAT</td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>TRMCOSTS.DAT</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td></td>
</tr>
<tr>
<td>TRD-EDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>TREESEQ.DAT</td>
</tr>
</tbody>
</table>
Table 3 (Cont'd)

<table>
<thead>
<tr>
<th>Program</th>
<th>Data Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES-EDT3.EXE</td>
<td>TRWDTi--.XDB</td>
</tr>
<tr>
<td>COST-DES.EXE</td>
<td>UNCDSC.XDB</td>
</tr>
<tr>
<td>UNC-FACT.EXE</td>
<td></td>
</tr>
<tr>
<td>UNC-COMP.EXE</td>
<td>UNCDSC.XDB</td>
</tr>
<tr>
<td>UNC-FACT.EXE</td>
<td>UNC-FACT.XDB</td>
</tr>
<tr>
<td>FACASS.EXE</td>
<td>UNC-FACT.XDB</td>
</tr>
<tr>
<td>UNC-FACT.EXE</td>
<td></td>
</tr>
<tr>
<td>URRAPR.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td>URR.XDB</td>
</tr>
<tr>
<td>URREDIT.EXE</td>
<td></td>
</tr>
<tr>
<td>URRCOP.EXE</td>
<td>URRAPR.XDB</td>
</tr>
<tr>
<td>DES-EDT3.EXE</td>
<td>VALLIST.DAT</td>
</tr>
<tr>
<td>FA-XEDIS.EXE</td>
<td>WP-DESC.DAT</td>
</tr>
</tbody>
</table>
The following is an alphabetical list of all standard subroutines with detailed descriptions.

<table>
<thead>
<tr>
<th>Name</th>
<th>ansist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>Bob Adams</td>
</tr>
<tr>
<td>Function</td>
<td>To perform various ansi-standard routines by sending the appropriate sequence of commands to the terminal. Note, the command parameter (com) can be in upper or lower case.</td>
</tr>
<tr>
<td>Usage</td>
<td>character com*3</td>
</tr>
<tr>
<td></td>
<td>integer*2 prml, prm2</td>
</tr>
<tr>
<td></td>
<td>call ansist (com, prml, prm2)</td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
</tr>
<tr>
<td>com</td>
<td>SGR (select graphic rendition)</td>
</tr>
<tr>
<td>prml</td>
<td>0 (turn all attributes off)</td>
</tr>
<tr>
<td></td>
<td>1 (increase screen intensity)</td>
</tr>
<tr>
<td></td>
<td>4 (dim screen intensity)</td>
</tr>
<tr>
<td></td>
<td>5 (blinking)</td>
</tr>
<tr>
<td></td>
<td>7 (reverse video)</td>
</tr>
<tr>
<td>prm2</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>HVP (horizontal and vertical position)</td>
</tr>
<tr>
<td>prml</td>
<td>1-24 (row in which to place cursor)</td>
</tr>
<tr>
<td>prml</td>
<td>1-80 (column in which to place cursor)</td>
</tr>
<tr>
<td>com</td>
<td>CUU (cursor up, doesn’t change column position)</td>
</tr>
<tr>
<td>prml</td>
<td>number of rows to move up (won’t move above top margin)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>CUD (cursor down, doesn’t change column position)</td>
</tr>
<tr>
<td>prml</td>
<td>number of rows to move down (won’t go below bottom margin)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>CUF (cursor forward, doesn’t change row position)</td>
</tr>
<tr>
<td>prml</td>
<td>number of cols to move right (won’t go past right margin)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>CUB (cursor backward, doesn’t change row position)</td>
</tr>
<tr>
<td>prml</td>
<td>number of cols to move left (won’t go past left margin)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>ED (erase in display, clear screen)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>prml</td>
<td>not used (use 0 for consistency)</td>
</tr>
<tr>
<td>com</td>
<td>EL (erase in line)</td>
</tr>
<tr>
<td>prml</td>
<td>0 (erase from cursor position to the end of line, inclusive)</td>
</tr>
<tr>
<td></td>
<td>2 (erase the entire line)</td>
</tr>
<tr>
<td>prm2</td>
<td>line number to erase</td>
</tr>
<tr>
<td>Returns</td>
<td>None.</td>
</tr>
<tr>
<td>Calls</td>
<td>fsc.lib: upper compac</td>
</tr>
<tr>
<td>Commons</td>
<td>None.</td>
</tr>
</tbody>
</table>
**Name** = box (assembler routine)

**Author** = Russ Hougland

**Function** = Draws a box with an upper left corner at \((x_1, y_1)\) and a lower right corner at \((x_2, y_2)\). The sides of the box are either single lines (type = 1) or double lines (type = 2). A box can also be erased (type = 0).

**Usage** = integer*2 \(x_1, y_1, x_2, y_2, \text{type, att}\)

  call box \((x_1, y_1, x_2, y_2, \text{type, att})\)

**Parameters** =

- \(x_1\) - row of the upper left hand corner
- \(y_1\) - column of the upper left hand corner
- \(x_2\) - row of the lower right hand corner (\(x_2 \text{ must be} > x_1\))
- \(y_2\) - column of the lower right hand corner (\(y_2 \text{ must be} > y_1\))
- \(\text{type}\) - 0 (blank lines, erase a box)
  - 1 (single line box)
  - 2 (double line box)
- \(\text{att}\) - the screen attribute to use on the box (see wt)

**Returns** = None.

**Calls** = None.

**Commons** = None.

*******************************************************************************

**Name** = cd (convert date)

**Author** = Russ Hougland

**Function** = Converts a date from the internal storage format to the format specified.

**Usage** = character fmt*01, cd*11, date*11, newdate*11

  newdate = cd \((\text{date, fmt})\)

**Parameters** =

- \(\text{date}\) - the date to be converted (must be in the internal format of 'YYYY/MMM/DD' i.e., '1985/001/01')
- \(\text{fmt}\) - the date format to convert to (can be lower or upper case)
  - 'E' (English format of 'MMM/DD/YYYY' i.e., 'JAN/01/1985')
  - 'M' (metric format of 'DD/MMM/YYYY' i.e., '01/JAN/1985')

**Returns** = The converted date stored in cd. If the date passed is empty or in error, cd returns an empty date (i.e., '/ / ')

**Calls** = fortran: ichar

**Commons** = None.

*******************************************************************************

17
**cdat**

* Name  =  cdat
* Author =  Kevin Stewart
* Function =  Converts a date from the internal database storage format to a real number (YYYY. + MM/12) that is to the nearest month.
* Usage  =  real*4  num, cdat
    character date*ll
    num = cdat (date)
* Parameters =
    date -  the date to be converted (in internal format 'Yyyyy/MMM/DD')
* Returns =  The date as a real number (YYYY. + MM/12).
* Calls =  fsc.lib: iconv
* Commons =  None.

---

**chkbit (assembler)**

* Name  =  chkbit (assembler)
* Author =  Russ Hougland
* Function =  Checks whether a bit is set or reset.
* Usage  =  integer*2  code, bit
    logical*2  i, chkbit
    i = chkbit (code, bit)
* Parameters =
    code -  variable to check
    bit -  the bit in variable (code) to check (0 - 15)
* Returns =  True if the bit is set.
* Calls =  None.
* Commons =  None.
clrbox (clear box)

Author = Russ Hougland

Function = Clears the interior area of a box with the upper left-hand corner at (x1,y1) and the lower right-hand corner at (x2,y2).
The attribute of the interior is determined by attr.

Usage = integer*2 x1, y1, x2, y2, attr
call clrbox (x1, y1, x2, y2, attr)

Parameters =
  * x1. - row of the upper left hand corner
  * y1 - column of the upper left hand corner
  * x2 - row of the lower right hand corner (x2 must be > x1)
  * y2 - column of the lower right hand corner (y2 must be > y1)
  * attr - attribute of the interior of the box (see wt)

Returns = None.

Calls = fsc.lib: scroll

Commons = None.

clrmod (color mode - assembler)

Author = Russ Hougland

Function = To determine if the user’s computer is in the color mode.

Usage = logical*2 clrmod, i
call clrmod ()

Parameters = None.

Returns = clrmod

Calls = None.

Commons = None.
**Name** = cmp2sd (compare to system date)

**Author** = Russ Hougland

**Function** = To compare a date to the system date.

**Usage**
- **dat** - the date to be compared (must be in internal format i.e., 'YYYY/MMM/DD')
- **flag** - indicates results of the comparison
  - -1 (if the date is before the system date)
  - 0 (if the two dates are equal)
  - +1 (if the date is after the system date)

**Returns** = flag (indicates results of comparison).

**Calls** = fsc.lib: sysdat

**Commons** = None.

---

**Name** = command

**Author** = Russ Hougland

**Function** = Execute a DOS command from within a program.

**Usage**
- logical*2 error
- character cmnd(*)
  - call command (cmnd, error) or
  - call command ('dir *.forlc,error)

**Parameters**
- **error** - true if an error occurred during execution.
- **cmnd** - character variable containing the DOS command to execute (if a variable is used, the string contained by it must be terminated with a null).
- **literal** - if the DOS command to be executed is contained in a literal string, the letter c must follow immediately after the string.

**Returns** = if error is true, a message will have been written to the file error.dat

**Calls** = fsc.lib: rpterr

**Commons** = None.
* Name = compac
* Author = Bobby Adams
* Function = Moves all blank characters in a string to the end of the
  string and returns the number of characters in the string.
* Usage = integer*2 num
  character str*(*)
  call compac (str, num)
* Parameters =
  str - the string variable of any length
  num - the number of non-blank characters in the string
* Returns = It returns num, the length of character string stored in str.
* Calls = fortran: len
* Commons = None.

* Name = contain (logical function)
* Author = Russ Hougland
* Function = Determines if a substring (sub) is contained within another
  string (str).
* Usage = logical*2 contain, i
  character sub*(*) , str*(*)
  i = contain (str, sub)
* Parameters =
  str - the string to search
  sub - the substring to search for
* Returns = contain is true if the substring is found within the target
  string.
* Calls = fortran: len
* Commons = None.
Name = csort

Author = Russ Hougland

Function = This is an interface to a sort routine (written in c by Steve Dorner and also contained in fsc.lib). It will sort a file using multiple keys in either ascending or descending order.

Usage = integer*4 addr1, addr2, locfar
        character control*(*), filen*(*)

addr1 = locfar (control)
addr2 = locfar (filen)
call csort (addr1, addr2)

Parameters =
addr1  - the address of the control string
addr2  - the address of the name of the file to be sorted
filen  - character string containing the name of the file to be sorted.
control - a character variable containing the command string for the sort. the command string must be terminated with a null.

the format of control is as follows:
if first character is 's', second character specifies the field separator character.
followed by zero or more words of the form:
[b][c|n][r][<m>][.<n>][-[<m2>][.<n2>]]
b ignore leading blanks
 c ignore case
 n numeric comparison
 r reverse order
 <m> field number
.<n> begin column
.<m2> end field number
.<n2> end column

Returns = None.
Calls = fsc.lib: sort
Commons = None.
**csrpos (assembler routine)**

**Author**
Dan Weidenfeld

**Function**
To position the cursor.

**Usage**
integer*2 row, col
call csrpos (row, col)

**Parameters**
- row - the row to position to - 1 (0-24)
- col - the column to position to - 1 (0-79)

**Returns**
None.

**Calls**
None.

**Commons**
None.

---

**cursor (assembler routine)**

**Author**
Russ Hougland

**Function**
Change the type of cursor used.

**Usage**
integer*2 type
call cursor (type)

**Parameters**
- type - 0 (no cursor)
- 1 (underline cursor)
- 2 (dash cursor)
- 3 (overscore cursor)
- 4 (block cursor)
- 5 (bottom-half block cursor)
- 6 (top-half block cursor)

**Returns**
None.

**Calls**
None.

**Commons**
None.
**gettime**

* Name = dantim (assembler routine)
* Author = Dan Weidenfeld
* Function = To get the system time.
* Usage = integer*2 hours, min, secs
  call gettim (hours, min, secs)
* Parameters =
  hours - the hour of the day in military form (i.e., 1 pm is 13)
  min - the minutes
  secs - the seconds
* Returns = the time of day.
* Calls = None.
* Commons = None.

**daydif**

* Name = daydif (integer function)
* Author = Russ Hougland
* Function = Returns the number of days between two dates.
* Usage = character sdate*11, edate*11
  integer*4 days, daydif
  days = daydif (sdate, edate)
* Parameters =
  sdate - the starting date (in the internal format)
  edate - the ending date (in the internal format)
* Returns = The number of days between the two dates. If an invalid date
  is passed, the function returns the value -9,999.
* Calls = fsc.lib: iconv
* Commons = None.
* Name = delay
* Author = Russ Hougland
* Function = Delays program action for a specified number of seconds.
* Usage = integer*2 secs
  call delay (secs)
* Parameters =
  secs - the number of seconds to delay
* Returns = None.
* Calls = fortran: gettim
* Commons = None.

* Name = dskcnf (assembler routine)
* Author = Paul Shih
* Function = find out the disks type
* Usage = integer*2 info(26)
  call dskcnf(info)
* Returns =
  Entry 1 is drive A
  2 is drive B and so on.
* Commons =
  INVALID_DSK 0
  REMOTE_DSK -1
  LOCAL_RAM_DSK 1
  LOCAL_FIXED_DSK 2
  LOCAL_FLOPPY_DSK 3
**Name** = erase

**Author** = Russ Hougland

**Function** = Erases the lines between two rows (linel and line2), inclusive.

**Usage** = integer*2 linel, line2, att

* call erase (linel, line2, att)

**Parameters** =
- linel - the erasing starts with this line (linel must be <= line2)
- line2 - the erasing ends with this line (line2 must be >= linel)
- att - the attribute to fill the erased field with

**Returns** = None.

**Calls** = fsc.lib: scroll

**Commons** = None.

******************************************************************************

**Name** = execute

**Author** = Russ Hougland

**Function** = Suspends the program currently running and activates the new program (filnam). When the new program (filnam) terminates, the original program is awakened and resumes its execution with the next instruction after the call to this subroutine.

**Usage** = logical*2 error
- character filnam*(*)
- :call execute (filnam, error) or
call execute ('prog2.exe'c, error)

**Parameters** =
- error - true if an error occurred during execution
- filnam - character variable containing the program name to execute (if a variable is used, the string contained by it must be terminated with a null).
- literal - if the program name to be executed is contained in a literal string, the letter c must follow immediately after the string.

**Returns** = if error is true, a message will have been written to the file error.dat

**Calls** = fortran: spawnlp
- fsc.lib: rpterr

**Commons** = None.

******************************************************************************

26
**Name** = fkeys

**Author** = Russ Hougland

**Function** = Turns on/off the function display fields on line 25 (should be used in conjunction with fline).

**Usage** = integer*2 f1, f2, f3, f4, f5, f6, f7, f8, f9, f10

  call fkeys (f1, f2, f3, f4, f5, f6, f7, f8, f9, f10)

**Parameters** =

  f1 - f10 = 0 (no change)
  1 (turn display field off)
  2 (turn display field on) and displays (by default):
    f1 - Help
    f2 - Keys
    f3 - Add
    f4 - Delete
    f5 - Edit
    f6 - Find
    f7 - List
    f8 -
    f9 - Next
    f10 - Exit

**Returns** = None.

**Calls** = fsc.lib: wt

**Commons** = None.

fflds (array 10x2 of character*6), if needed, the field to be displayed can be changed by modifying the contents of fflds (i.e., fflds(function key,2) = 'Change').

**Name** = fline

**Author** = Russ Hougland

**Function** = Initially writes the function keys' display line.

**Usage** = call fline

**Parameters** = None.

**Returns** = None.

**Calls** = fsc.lib: wt

**Commons** = None.
********** frames (assembler routine) **********

* Name = frames (assembler routine)
* Author = Russ Hougland
* Function = To save up to five video screen windows into memory for later restoration. When saving a window, the row and column parameters specify the area to be saved. When restoring a window, the row and column parameters specify the location the window should be restored to.
* Usage = integer*2 action, frame, r1, c1, r2, c2
  call frames (action, frame, r1, c1, r2, c2)
* Parameters =
  action - the action to take
    .1 (save the window to memory)
    anything else (restore the window from memory)
  frame - memory frame to save/restore the window to/from
    1 - 5 (possible frames to use)
  r1 - row of the upper left hand corner of the window (1-25)
  c1 - column of the upper left hand corner of the window (1-80)
  r2 - row of the lower right hand corner of the window (1-25)
  c2 - column of the lower right hand corner of the window (1-80)
* Returns = None.
* Calls = None.
* Commons = None.

******************************************************************************

********** gtinsk (get instruction key) **********

* Name = gtinsk (get instruction key)
* Author = Russ Hougland
* Function = Returns the ASCII integer value for the nonalphanumeric keys
  (e.g., function keys, Esc, Ins, etc.). Alphanumeric keys are ignored. Those keys that send two codes (i.e., a nul and then another number) will only return to the calling program the second number.
* Usage = integer*2 key
  call gtinsk (key)
* Parameters =
  key = ASCII integer value of the depressed key
* Returns = key (the ASCII integer value of the depressed nonalphanumeric key)
* Calls = fsc.lib: inkey
* Commons = None.

******************************************************************************
* Name = iconv (convert character string to integer)
* Author = Bobby Adams
* Function = To convert a character string of numbers to its integer equivalent.
* Usage = character str*(*)
  integer*4 num, iconv
  logical*2 err
  num = iconv (str, err)
* Parameters =
  str = the character string containing the number to convert
  err = true if an error occurs
* Returns = iconv and err (if err, then iconv equals 0)
* Calls = fortran: ichar
* Commons = None.

* Name = inkey (assembler routine)
* Author = Dan Weidenfeld
* Function = To perform a single character read from the keyboard.
* Usage = integer*2 i, inkey, echo
  i = inkey (echo)
* Parameters =
  echo = if equal to zero the character is not echoed to the screen.
* Returns = the ascii value of the character entered from the keyboard.
* Calls = None.
* Commons = None.
Name = inout (assembler routine)

Author = Dan Weidenfeld

Function = To perform a single character write to the screen.

Usage = character char*01
        call inout (char)

Parameters =
        char - the character to be displayed on the screen

Returns = None.

Calls = None.

Commons = None.

Name = isfile (assembler routine)

Author = Paul Shih

Function = To check file or subdirectory status

Usage = integer*2 i
        i = isfile('test.dat'c)

Returns = -1 : if file not found

Bit 0 on : Read-Only file
Bit 1 on : Hidden file
Bit 2 on : System file
Bit 3 on : Read-Only file
Bit 4 on : Read-Only file

Commons = None.
Name = line (assembler routine)

Author = Russ Hougland

Function = To draw a line (horizontal or vertical) directly into the video memory. It will not draw a diagonal line. This routine allows the user to specify the beginning and ending characters of the line (in case it intercepts a box).

Usage = integer*2 r1, c1, r2, c2, begchr, endchr, type, attr
call line (r1, c1, r2, c2, begchr, endchr, type, attr)

Parameters =

 rl - starting row on the screen (1-25)
 c1 - starting column on the screen (1-80)
 r2 - ending row on the screen (1-25)
 c2 - ending column on the screen (1-80)
 begchr - the beginning character of the line (the decimal ascii value)
 endchr - the ending character of the line (the decimal ascii value)
 type - the type of line to draw
 1 - single line
 2 - double line
 attr - attribute in which to display the line (see wt)

Returns = None.

Calls = None.

Commons = None.

*******************************************************************************
**Name** = pi (prompt for integer)  

**Author** = Kevin Stewart  

**Function** = To write out a prompt and accept an integer response. After accepting the response, the prompt and reply are erased. The routine places one space between the end of the prompt and the start of the reply.  

**Usage** = integer*2 row, col, rlen, code, patt, ratt  

integer*4 reply  

character prompt(*)  

call pi (row, col, prompt, reply, rlen, code, patt, ratt)  

**Parameters** =  

row - the row to display the prompt on  

col - the column to begin displaying the prompt on  

0 (center the prompt and reply)  

prompt - the text to be displayed as a prompt  

reply - the integer reply of the user  

rlen - the maximum length of the user's reply  

code - code can give special instructions for the reply entry. more than one can be passed by summing the options desired.  

4 (user must enter a carriage return to enter a value)  

8 (the number will not be written upon entry and exit from ri)  

16 (the prompt will not be erased upon exit from pi)  

patt - the attribute to display the prompt in (see wt)  

ratt - the attribute to display the reply in (see wt)  

**Returns** = reply  

**Calls** = fsc.lib: chkbit ri window wt  

fortran: len  

**Commons** = None.
* Name = pt (prompt for text)
* 
* Author = Kevin Stewart
* 
* Function = To write out a prompt and accept a text response. After accepting the response, the prompt and reply are erased.
* The routine places one space between the end of the prompt and the start of the reply.
* 
* Usage = integer*2 row, col, code, patt, ratt
* character prompt*(*), reply*(*)
* call pt (row, col, prompt, reply, code, patt, ratt)
* 
* Parameters =
* row - the row to display the prompt on
* col - the column to begin displaying the prompt on.
* 0 (center the prompt and reply)
* prompt - the text to be displayed as a prompt
* reply - the text reply of the user
* code - code can give special instructions for the reply entry.
* more than one can be passed by summing the options desired.
* 1 (the string is made upper case)
* 2 (the string is packed -- no blanks)
* 4 (user must enter a carriage return to enter a value)
* 8 (the text will not be written upon entry and exit from rt)
* 16 (the prompt will not be erased upon exit from pi)
* patt - the attribute to display the prompt in (see wt)
* ratt - the attribute to display the reply in (see wt)
* 
* Returns = reply
* 
* Calls = fsc.lib: chkbit rt window wt
* fortran: len
* 
* Commons = None.
* Name = rconv (convert character string to real) *
* Author = Kevin Stewart *
* Function = To convert a character string of numbers to its real *
* equivalent. The largest number that can be converted can *
* have up to 10 digits and 10 decimal places. *
* Usage = character str(*)
* real*8 num, rconv
* logical*2 err
* num = rconv (str, err)
* Parameters = *
* str = the character string containing the number to convert *
* err = true if an error occurs *
* Returns = rconv and err (if err, then rconv equals 0) *
* Calls = fsc.lib: iconv compac *
* Commons = None. 

******************************************************************************
Name = rd (read date)

Author = Russ Hougland

Function = Reads a user's reply (date field) at a specified location (row, col). Before the reply is read, the field is written in the attribute specified (attin); upon termination, the reply is rewritten in the attribute specified (attout). The date is always returned in internal format ('YYYY/MM/DD'). This routine returns either a blank date ('0000/000/00') or a valid date, but never an invalid date.

Usage = integer*2 row, col, code, attin, attout
        character fmt*01, date*11
        call rd (row, col, date, fmt, code, attin, attout)

Parameters =

row - the row on which to read the date
col - the column on which to read the date
date - the date to be read
fmt - the format of date for display and entry purposes (any case)
'M' (metric DD/MMM/YYYY)
'E' (English MMM/DD/YYYY)
code - code can give special instructions for the date entry.
more than one can be passed by summing the options desired.
4 (user must enter a carriage return to enter a value)
8 (the date will not be written upon entry and exit from rd)
attin - the attribute to display the date in upon entry (see wt)
attout - the attribute to display the date in upon exit (see wt)

Returns = code - indicates how rd was terminated (e.g., carriage return, up arrow, etc.)
date - the date entered by the user

Calls = fsc.lib: chkbit window ri wi rt wt wd
fortran: ichar mod

Commons = None.
Name = rf (read fixed real)

Author = Bobby Adams

Function = Reads a user's reply (real field) at a specified location (row, col). Before the reply is read, the field can be written in a specified attribute (attin); upon termination, the reply can be rewritten in a specified attribute (attout). Note the real number read is double precision. Further, the total length of the number is len1 + len2 + 1 (decimal point).

Usage = integer*2 row, col, len1, len2, code, attin, attout

real*8 num

call rf (row, col, num, len1, len2, code, attin, attout)

Parameters =
row - the row on which to read the number
col - the column on which to read the number
num - the number to be read
len1 - the maximum length of the left side of the number
len2 - the maximum length of the right side of the number
code - code can give special instructions for the number entry. more than one can be passed by summing the options desired.
4 (user must enter a carriage return to enter a value)
8 (the number will not be written upon entry and exit from rf)
attin - the attribute to display the number in upon entry (see wt)
attout - the attribute to display the number in upon exit (see wt)

Returns = code - indicates how rf was terminated (e.g., carriage return, up arrow, etc.)
num - the number entered by the user

Calls = fsc.lib: chkbit window cursor csrpos inkey wt wf

Commons = None.
** ri (read integer)

** Author = Bobby Adams

** Function = Reads a user's reply (integer field) at a specified location
(row,col). Before the reply is read, the field can be
written in a specified attribute (attin); upon termination,
the reply can be rewritten in a specified attribute (attout).

** Usage = integer*2 row, col, len, code, attin, attout
integer*4 num
call ri(row, col, num, len, code, attin, attout)

** Parameters =
* row - the row on which to read the number
* col - the column on which to read the number
* num - the number to be read
* len - the maximum length of the number
* code - code can give special instructions for the number entry.
  more than one can be passed by summing the options desired.
  4 (user must enter a carriage return to enter a value)
  8 (the number will not be written upon entry and exit from ri)
* attin - the attribute to display the number in upon entry (see wt)
* attout - the attribute to display the number in upon exit (see wt)

** Returns = code - indicates how ri was terminated (e.g., carriage return,
  up arrow, etc.)
* num - the number entered by the user

** Calls = fsc.lib: chkbit cursor window csrpos inkey wi wt

** Commons = None.
**Name**
= rr (read real)

**Author**
= Bobby Adams

**Function**
= Reads a user's reply (real field) at a specified location (row, col). Before the reply is read, the field can be written in a specified attribute (attin); upon termination, the reply can be rewritten in a specified attribute (attout).

**Usage**
= integer*2 row, col, len, code, attin, attout
  real*4 num
call rr(row, col, num, len, code, attin, attout)

**Parameters**
  row - the row on which to read the number
  col - the column on which to read the number
  num - the number to be read
  len - the maximum length of the number
  code - code can give special instructions for the number entry. more than one can be passed by summing the options desired.
    4 (user must enter a carriage return to enter a value)
    8 (the number will not be written upon entry and exit from rr)
  attin - the attribute to display the number in upon entry (see wt)
  attout - the attribute to display the number in upon exit (see wt)

**Returns**
  code - indicates how rr was terminated (e.g., carriage return, up arrow, etc.)
  num - the number entered by the user

**Calls**
= fsc.lib: chkbit window cursor csrpos inkey wt wr

**Commons**
= None.

*******************************************************************************

**Name**
= rsscrn (assembler routine)

**Author**
= Russ Hougland

**Function**
= Restores the screen display previously saved by svscrn.

**Usage**
= call rsscrn

**Parameters**
= None

**Returns**
= None.

**Calls**
= None.

**Commons**
= None.

*******************************************************************************
*Name* = rt (read text)
*Author* = Russ Hougland
*Function* = Reads a user's reply (text field) at a specified location (row, col). Before the reply is read, the field can be written in a specified attribute (attin); upon termination, the reply can be rewritten in a specified attribute (attout).
*Usage* = integer*2 row, col, code, attin, attout
*Parameters* =
row - the row on which to read the string
col - the column on which to read the string
str - the string to be read
code - code can give special instructions for the text entry. more than one can be passed by summing the options desired.
1 (the string is made upper case)
2 (the string is packed -- no blanks)
4 (user must enter a carriage return to enter a value)
8 (the number will not be written upon entry and exit from rt)
attin - the attribute to display the string in upon entry (see wt)
attout - the attribute to display the string in upon exit (see wt)
*Returns* = code - indicates how rt was terminated (e.g., carriage return, up arrow, etc.)
str - the string entered by the user
*Calls* = fsc.lib: chkbit cursor csrpos compac inkey wt
fortran: ichar
*Commons* = None.

******************************************************************************

*Name* = rv (assembler routine)
*Author* = Russ Hougland
*Function* = To convert an attribute to reverse video. It reverses the foreground and background RGB bits while leaving the intensity and blinking bits unchanged.
*Usage* = integer*2 newatt, oldatt, rv
newatt = rv (oldatt)
*Parameters* =
oldatt - the old attribute that needs reversing
*Returns* = rv (the new attribute after reversing).
*Calls* = None.
*Commons* = None.
* Name = scroll (assembler routine)
* Author = Dan Weidenfeld
* Function = To scroll a window on the screen.
* Usage = integer*2 rl, cl, r2, c2, action, att
  call scroll (rl, cl, r2, c2, action, att)
* Parameters =
  rl - row of the upper left hand corner of the window to be
       scrolled - 1 (0-24)
  cl - column of the upper left hand corner of the window to be
       scrolled - 1 (0-79)
  r2 - row of the lower right hand corner of the window to be
       scrolled - 1 (0-24)
  c2 - column of the lower right hand corner of the window to be
       scrolled - 1 (0-79)
  action - the action to perform
    *n (scroll up n lines)
    *-n (scroll down n lines)
    0 (scroll the entire window)
  att - the attribute of blank lines added after scrolling (see wt)
* Returns = None.
* Calls = None.
* Commons = None.

* Name = setatt
* Author = Bobby Adams
* Function = To initialize the values of the attribute variables used by
            the screen read and write routines. This routine must be
            called once at the beginning of any program that uses the
            attribute variables stored in attrib.inc. The color values of
            the variables can be changed by changing the file, color.dat.
* Usage = call setatt
* Parameters = None.
* Returns = None.
* Calls = fsc.lib: clrmod
* Commons = attrib (attrib.inc) - contains the values to initialize
**Name** = slen (integer function)

**Author** = Kevin Stewart

**Function** = Find the length of a character string within a character variable (i.e., the position of the last nonblank character).

**Usage** = integer*2 i, slen
character str*(*)
i = slen (str)

**Parameters =**
str - character variable containing the string being passed

**Returns =** slen - the length of the character string (0 if all blanks)

**Calls =** None.

**Commons =** None.

---

**Name** = svscrn (assembler routine)

**Author** = Russ Hougland

**Function** = Saves the screen display in a buffer area for later restoration by rsscrn.

**Usage** = call svscrn.

**Parameters =** None

**Returns =** None.

**Calls =** None.

**Commons =** None.

---

**Name** = sysdat (system date)

**Author** = Kevin Stewart

**Function** = To return the system date in the internal storage format.

**Usage** = character*11 dat
call sysdat (dat)

**Parameters =**
dat - the system date in internal format ('YYYY/MMM/DD')

**Returns =** the system date (dat).

**Calls =** fortran: getdat

**Commons =** None.
************

Name = today (assembler routine)

Author = Dan Weidenfeld

Function = To get the system date.

Usage = integer*2 year, month, day
    call today (year, month, day)

Parameters =
    year - the year of the system date (e.g., 1986)
    month - the month of the system date (e.g., 12)
    day - the day of the system date (e.g., 25)

Returns = the system date (year, month, day).

Calls = None.

Commons = None.

************

Name = upper

Author = Russ Hougland

Function = Converts to upper case all letters in the parameter.

Usage = character*(*) str
    call upper (str)

Parameters =
    str - the character string to be converted to upper case

Returns = str (all letters in upper case)

Calls = fortran: ichar char

Commons = None.

************
**Name** = wd (write date)

**Author** = Russ Hougland

**Function** = Write a date at a specified location (row,col) in the form specified. The date should be passed in the internal format ('YYYY/MMM/DD').

**Usage** = integer*2 row, col, attr
character fmt*01, date*11
call wd (row, col, date, fmt, attr)

**Parameters** =
row - the row on which to write the date
col - the column on which to write the date
0 (to center the date on the line)
date - the date to be written
fmt - the format of date (can be lower or upper case)
'M' (metric DD/MMM/YYYY)
'E' (English MMM/DD/YYYY)
attr - the attribute to display the date in (see wt)

**Returns** = None.

**Calls** = fsc.lib: cd wt

**Commons** = None.

---

**Name** = wf (write fixed real)

**Author** = Bobby Adams

**Function** = Write a real at a specified location (row,col) using the desired attribute. Note the number written is double precision. Further, the total length of the number is \( len1 + len2 + 1 \) (decimal point).

**Usage** = integer*2 row, col, attr, len1, len2
real*8 num
call wf (row, col, num, len1, len2, attr)

**Parameters** =
row - the row on which to write the number
col - the column on which to write the number
0 (to center the number on the line)
num - the number to be written
len1 - the length of the left side of the number in
len2 - the length of the right side of the number in
attr - the attribute to display the date in (see wt)

**Returns** = None.

**Calls** = fsc.lib: compac wt

**Commons** = None.
Name = wi (write integer)

Author = Russ Hougland

Function = Write an integer at a specified location (row, col) using the desired attribute.

Usage = integer*2 row, col, attr, len
        integer*4 num
        call wi (row, col, num, len, attr)

Parameters =
row - the row on which to write the number
col - the column on which to write the number
0 (to center the number on the line)
num - the number to be written
len - the length of the field to display the number in
attr - the attribute to display the number in (see wt)

Returns = None.

Calls = fsc.lib: wi

Commons = None.

Name = window (assembler routine)

Author = Russ Hougland

Function = Set the video attribute for a window directly into video memory.

Usage = integer*2 rl, cl, r2, c2, attr
        call window (rl, cl, r2, c2, attr)

Parameters =
rl - row of the upper left hand corner
cl - column of the upper left hand corner
r2 - row of the lower right hand corner (x2 must be > x1)
c2 - column of the lower right hand corner (y2 must be > y1)
attr - the attribute of the window (see wt)

Returns = None.

Calls = None.

Commons = None.
* Name = wr (write real)
* Author = Bobby Adams
* Function = Write a real at a specified location (row,col) using the desired attribute.
* Usage = integer*2 row, col, attr, len
          real*4 num
          call wr (row, col, num, len, attr)
* Parameters =
  * row - the row on which to write the number
  * col - the column on which to write the number
  * 0 (to center the number on the line)
  * num - the number to be written
  * len - the length of the field to display the number in
  * attr - the attribute to display the date in (see wt)
* Returns = None.
* Calls = fsc.lib: wt
* Commons = None.

* Name = wsd (write screen date)
* Author = Kevin Stewart
* Function = To write a prompt and a variable to the screen.
* Usage = integer*2 row, col, patt, vatt
          character prompt*, dat*11, fmt*01
          call wsd (row, col, prompt, dat, fmt, patt, vatt)
* Parameters =
  * row - the row position of the date
  * col - the column position of the date
  * 0 (center the prompt and the date in the row)
  * prompt - the prompt (displayed at row,col-plen-1)
  * dat - the date variable to display
  * fmt - the format to display the date in (English or metric)
  * patt - the attribute to use when displaying the prompt
  * vatt - the attribute to use when displaying the date
* Returns = None.
* Calls = fsc.lib: wd wt
* Commons = None.
**Name** = wsf (write screen fixed real)

**Author** = Kevin Stewart

**Function** = To write a prompt and a variable to the screen.

**Usage** = integer*2 row, col, len1, len2, patt, vatt

character prompt*(*)

real*8 var

call wsf (row, col, prompt, var, len1, len2, patt, vatt)

**Parameters** =

row - the row position of the variable
col - the column position of the variable
0 (center the prompt and the variable in the row)
prompt - the prompt (displayed at row, col-plen-1)
var - the variable to display
len1 - the maximum length of the left side of the number
len2 - the maximum length of the right side of the number
patt - the attribute to use when displaying the prompt
vatt - the attribute to use when displaying the variable

**Returns** = None.

**Calls** = fsc.lib: wf wt

**Commons** = None.

**Name** = wsi (write screen integer)

**Author** = Kevin Stewart

**Function** = To write a prompt and a variable to the screen.

**Usage** = integer*2 row, col, vlen, patt, vatt

character prompt*(*)

integer*4 var

call wsi (row, col, prompt, var, vlen, patt, vatt)

**Parameters** =

row - the row position of the variable
col - the column position of the variable
0 (center the prompt and the variable in the row)
prompt - the prompt (displayed at row, col-plen-1)
var - the variable to display
vlen - the length of the variable
patt - the attribute to use when displaying the prompt
vatt - the attribute to use when displaying the variable

**Returns** = None.

**Calls** = fsc.lib: wi wt

**Commons** = None.
* ** wsr (write screen real)**
* Author = Kevin Stewart
* Function = To write a prompt and a variable to the screen.
* Usage = integer*2 row, col, vlen, patt, vatt
  character prompt*, var*
  call wsr (row, col, prompt, var, vlen, patt, vatt)
  Parameters =
    row - the row position of the variable
    col - the column position of the variable
    0 (center the prompt and the variable in the row)
    prompt - the prompt (displayed at row,col-plen-1)
    var - the variable to display
    vlen - the length of the variable
    patt - the attribute to use when displaying the prompt
    vatt - the attribute to use when displaying the variable
* Returns = None.
* Calls = fsc.lib: wr wt
* Commons = None.

* ** wst (write screen text)**
* Author = Kevin Stewart
* Function = To write a prompt and a variable to the screen.
* Usage = integer*2 row, col, patt, vatt
  character prompt*, var*
  call wst (row, col, prompt, var, patt, vatt)
  Parameters =
    row - the row position of the variable
    col - the column position of the variable
    0 (center the prompt and the variable in the row)
    prompt - the prompt (displayed at row,col-plen-1)
    var - the variable to display
    patt - the attribute to use when displaying the prompt
    vatt - the attribute to use when displaying the variable
* Returns = None.
* Calls = fsc.lib: wt
* Commons = None.

47
**Name** = wt (write text -- assembler routine)

**Author** = Russ Hougland

**Function** = Write a string at a specified location (row, col) using the desired attribute.

**Usage** = integer*2 row, col, attr, len

character str*(*)
call wt (row, col, str, len, attr)

**Parameters** =

row - the row on which to write the string

col - the column on which to write the string

0 (center the field including trailing blanks)

str - the string to be written

len - the length of the field to display the string in

attr - the attribute to display the string in
to determine the number for a color use the following:

foreground color: blue 1

green 2

red 4

intensity : 8

background color: blue 16

green 32

red 64

blinking : 128

to get the desired color, simply sum the numbers of the desired attributes, for example: to get a blinking white text on a black background attr = 1 + 2 + 4 (white foreground) + 0 (black background) + 128 (blinking)

**Returns** = None.

**Calls** = None.

**Commons** = None.

*******************************************************************************
*FUNCTION : INTERFACE WITH THE BTRIEVE TO ACCESS A SPECIFIED FILE*

*USAGE : INTEGER*2 STS,FILBLK(64),REC(N),LREC,LKEY,OP,KEYNUM
* CHARACTER KEY*N
* CALL BTREE(FILBLK,OP,KEYNUM,KEY,LKEY,REC,LREC,STS)

*PARAMETERS: FILBLK - STORES THE FILE NAME AND PATH, CURRENT RECORD POINTER
* OP - THE TYPE OF OPERATION TO BE EXECUTED (i.e., OPEN, UPDATE, etc.)
* KEYNUM - THE KEY PATH USED IN ALL FILE OPERATIONS
* ** NOTE **
* THE KEYNUM USED MUST BE A VALID KEY SPECIFIED DURING THE FILES INITIAL OPERATION.
* KEY - CONTAINS THE KEY FIELD OF THE RECORD
* LKEY - LENGTH OF KEY FIELD IN BYTES
* REC - INTEGER*2 ARRAY CONTAINING RECORD DATA
* LREC - LENGTH OF RECORD IN BYTES
* STS - CONTAINS THE STATUS OF THE OPERATION PERFORMED
* RETURN ZERO IF NORMAL OPERATION

*NOTE : WHEN CREATING AND OPENING FILES, THE FILE NAME IS IN THE 'KEY' FIELD AND THE FILE NAME LENGTH IS IN THE 'LKEY' FILED, example:
* CALL BTREE (FILBLK,0,0,'FILE.XDB',8,REC,LREC,STS)

* BTRIEVE OPERATIONS

* (0) - OPEN
* PURPOSE - TO OPEN A BTRIEVE FILE
* (1) - CLOSE
* PURPOSE - TO CLOSE A BTRIEVE FILE
* (2) - INSERT
* PURPOSE - TO INSERT A RECORD IN A FILE
* (3) - UPDATE
* PURPOSE - TO UPDATE AN EXISTING RECORD IN A BTRIEVE FILE
* (4) - DELETE
* PURPOSE - TO DELETE AN EXISTING RECORD IN A BTRIEVE FILE
* (5) - GET EQUAL
* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING TO A SPECIFIC KEY VALUE
* (6) - GET NEXT
* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE THAT FOLLOWEDS  
* THE "CURRENT RECORD"

* (7) - GET PREVIOUS

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE THAT PRECEDES  
* THE "CURRENT RECORD"

* (8) - GET GREATER

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE KEY VALUE THAT IS GREATER THAN A SPECIFIC KEY VALUE

* (9) - GET GREATER OR EQUAL

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE KEY VALUE THAT IS GREATER OR EQUAL TO A SPECIFIC  
* KEY VALUE

* (10) - GET LESS THAN

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE KEY VALUE WHICH IS GREATER THAN A SPECIFIC KEY VALUE

* (11) - GET LESS THAN OR EQUAL

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE KEY VALUE WHICH IS LESS THAN OR EQUAL TO A SPECIFIC  
* KEY VALUE

* (12) - GET LOWEST

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE LOWEST KEY VALUE FOR A SPECIFIED ACCESS PATH.

* (13) - GET HIGHEST

* PURPOSE - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING  
* TO THE HIGHEST KEY VALUE FOR A SPECIFIED ACCESS PATH.

* (14) - CREATE

* PURPOSE - TO CREATE A BTRIEVE FILE WITH THE SPECIFIED SET OF  
* CHARACTERISTICS

* (15) - STAT

* PURPOSE - TO RETRIEVE A SPECIFIED FILE'S CHARACTERISTICS

* (16) - EXTEND

* PURPOSE - TO EXTEND A BTRIEVE FILE TO A SECOND LOGICAL DISK DRIVE

* (17) - SET DIRECTORY

* PURPOSE - TO SET THE CURRENT DIRECTORY TO A SPECIFIED VALUE

* (18) - GET DIRECTORY

* PURPOSE - TO RETRIEVE THE "CURRENT" DIRECTORY

* (19) - BEGIN TRANSACTION

* PURPOSE - TO MARK THE BEGINNING OF A SET OF LOGICALLY RELATED BTRIEVE  
* OPERATIONS.

* (20) - END TRANSACTION

* PURPOSE - TO COMPLETE A TRANSACTION AND COMMIT THE OPERATIONS PERFORMED  
* SINCE THE TRANSACTION BEGAN

* (21) - ABORT TRANSACTION

* PURPOSE - TO REMOVE ALL OPERATIONS PERFORMED SINCE BEGINNING OF  
* AN ACTIVE TRANSACTION, AND TO TERMINATE THE TRANSACTION.

* (22) - GET POSITION

* PURPOSE - TO RETURN THE PHYSICAL POSITION OF THE RECORD IN THE BTRIEVE
FILE THAT HAS BEEN ESTABLISHED AS THE "CURRENT RECORD"

(23) - GET DIRECT
* PURPOSE - TO RETRIEVE THE DATA RECORD POSITIONED AT A SPECIFIED
* ADDRESS IN THE BTREEVE FILE.

(24) - STEP DIRECT
* PURPOSE - TO RETRIEVE THE DATA RECORD IN THE LOCATION PHYSICALLY
* FOLLOWING THE CURRENT RECORD IN THE BTREEVE FILE.

(25) - STOP
* PURPOSE - TO TERMINATE THE RECORD MANAGER AND REMOVE IT FROM MEMORY.

************************************************************************************

************************************************************************************

* BTREEVE ERROR CODES
* (For detailed information see Btrieve menu Appendix B ERROR CONDITIONS)
*
* A utility program (STS.EXE) is available to prompt Btrieve Error
* Condition and Fortran Runtime Error.
* *
* 01 INVALID OPERATION
* 02 I/O ERROR
* 03 NO OPEN
* 04 KEY NOT FOUND
* 05 DUPLICATES ERROR
* 06 INVALID KEY NUMBER
* 07 DIFFERENT KEY NUMBER
* 08 INVALID POSITIONING
* 09 END OF FILE
* 10 MODIFIABLE ERROR
* 11 INVALID FILE NAME
* 12 FILE NOT FOUND
* 13 EXTENSION ERROR
* 14 PRE-OPEN ERROR
* 15 PRE-IMAGE ERROR
* 16 EXPANSION ERROR
* 17 CLOSE ERROR
* 18 DISK FULL
* 19 UNRECOVERABLE ERROR
* 20 RECORD MANAGER INACTIVE
* 21 KEY BUFFER ERROR
* 22 RECORD BUFFER (DATA BUFFER NOT LONG ENOUGH)
* 23 POSITION BLOCK (MUST BE 128 BYTES)
* 24 PAGE SIZE (MUST BE MULTIPLE OF 512)
* 25 CREATE I/O ERROR
* 26 NUMBER OF KEYS (PAGE SIZE AND NUM OF KEYS DO NOT MATCH)
* 27 KEY POSITION
* 28 RECORD LENGTH (NO GREATER THAN PAGE SIZE - 6)
* 29 KEY LENGTH (1-255)
* 30 BTREEVE FILE NAME (INVALID BTREEVE FILE)
* 31 EXTEND ERROR (ALREADY EXTENDED)
* 32 EXTEND I/O ERROR
* 33 EXTEND DRIVE ERROR
* 34 EXTEND NAME
* 35 DIRECTORY ERROR
* 36 TRANSACTION (I/T OPTION NOT SPECIFIED WHEN REC-MAN LOADED)
* 37 BEGIN TRANSACTION (TRANSACTION ALREADY ACTIVE)
* 38 TRANSACTION CONTROL FILE
* 39 END/ABORT ERROR
* 40 TRANSACTION MAX FILES (UP TO 8 FILES MAY BE UPDATED DURING TRANS)

51
* 41 TRANSACTION OPEN/CLOSE
* 42 INCOMPLETE ACCELERATED ACCESS
* 43 INVALID DATA RECORD ADDRESS
* 44 NULL KEY PATH
* 45 INCONSISTENT KEY FLAGS
* 46 ACCESS DENIED
* 47 MAXIMUM OPEN FILES
* 48 INVALID ALTERNATE SEQUENCE DEFINITION
* 49 KEY TYPE ERRCR
* 50 OWNER ALREADY SET
* 51 INVALID OWNER
* 52 ERROR WRITING CACHE
* 53 INVALID INTERFACE
* 54 VARIABLE PAGE UNREADABLE
* 55 CONFLICT
* 56 LOCK FULL
* 57 LOST POSITION
* 58 READ OUTSIDE TRANSACTION
* 59 RECORD IN USE
* 60 FILE IN USE
* 61 FILE FULL
* 62 HANDLE FULL
* 63 MODE ERROR
* 64 NAME ERROR
* 65 DEVICE FULL
* 66 SERVER ERROR
* 67 TRANSACTION FULL
* 68 DEMO ERROR

*****************************************************************************

52
5 STANDARD COMMON BLOCKS

The following is an alphabetical list of the standard common blocks and a short description of each block.

<table>
<thead>
<tr>
<th>FILE NAME: AMSCOD.XDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE TYPE: BTRIEVE</td>
</tr>
<tr>
<td>INCLUDE FILE: AMSCOD.INC</td>
</tr>
<tr>
<td>NUMBER OF KEYS: 2</td>
</tr>
<tr>
<td>RECORD SIZE : 52</td>
</tr>
<tr>
<td>PAGE SIZE : 512</td>
</tr>
<tr>
<td>RECORD NAME : AMSREC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>**** KEY INFORMATION ****</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYNO</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>**** RECORD INFORMATION *****</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIELD</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>AMSKEY</td>
</tr>
<tr>
<td>AMSCOD</td>
</tr>
<tr>
<td>AMSDSC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FILE NAME: AMSDSC.XDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE TYPE: BTRIEVE</td>
</tr>
<tr>
<td>INCLUDE FILE: AMSDSC.INC</td>
</tr>
<tr>
<td>NUMBER OF KEYS: 2</td>
</tr>
<tr>
<td>RECORD SIZE : 50</td>
</tr>
<tr>
<td>PAGE SIZE : 512</td>
</tr>
<tr>
<td>RECORD NAME : AMSRC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>**** KEY INFORMATION ****</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYNO</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>**** RECORD INFORMATION *****</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIELD</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>AMSID</td>
</tr>
<tr>
<td>AMSDSC</td>
</tr>
</tbody>
</table>
### AMSF4C.XDB

**FILE NAME:** AMSF4C.XDB  
**FILE TYPE:** BTRIEVE  
**INCLUDE FILE:** AMSF4C.INC  
**NUMBER OF KEYS:** 2  
**RECORD SIZE:** 18  
**PAGE SIZE:** 512  
**RECORD NAME:** F4CREC

**KEY INFORMATION**

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>F4CCOD</td>
<td>1</td>
<td>7</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>AMSEQV</td>
<td>9</td>
<td>10</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

**RECORD INFORMATION**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE LENGTH POSITION DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>F4CCOD</td>
<td>S 7 1 F4C CODE</td>
</tr>
<tr>
<td>AMSEQV</td>
<td>S 10 9 AMS NUMBER</td>
</tr>
</tbody>
</table>

### APRCOD.XDB

**FILE NAME:** APRCOD.XDB  
**FILE TYPE:** BTRIEVE  
**INCLUDE FILE:** APRCOD.INC  
**NUMBER OF KEYS:** 2  
**RECORD SIZE:** 52  
**PAGE SIZE:** 512  
**RECORD NAME:** APRREC

**KEY INFORMATION**

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>APRKEY</td>
<td>1</td>
<td>2</td>
<td>N</td>
<td>Y</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>APRCOD</td>
<td>3</td>
<td>10</td>
<td>N</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

**RECORD INFORMATION**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE LENGTH POSITION DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>APRKEY</td>
<td>S 2 1 APPROPRIATION ID</td>
</tr>
<tr>
<td>APRCOD</td>
<td>S 10 3 APP. CODE</td>
</tr>
<tr>
<td>APRDSC</td>
<td>S 40 13 APP. DESCRIPTION</td>
</tr>
</tbody>
</table>

54
FILE NAME: AREA TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 34
PAGE SIZE : 512
RECORD NAME : ARDAT

***** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ********* ***** *** *** *****
0 CODE 1 2 N Y S

***** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME
***** **** ********* ******* **********
CODE S 2 1 AREA ID
DEF S 30 3 AREA DESCRIPTION

FILE NAME: BT-PASS.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 1

***** FILE INFORMATION ****
LINE FORMAT FIELDS
***** ********** **********
1 A4 TTGG

***** FIELD INFORMATION *****
FIELD TYPE LENGTH DESCRIPTION
NAME (BYTES)
***** **** ********* **********
TTGG S 4 BASIC TASK SUMMARY FILE NAME SPECIFIER

***** NOTES *****
THIS VARIABLE REPLACES POSITIONS 5 to 8 IN THE FILE NAME 'BTSMttgg.XDB'

FILE NAME: CLASLST.DAT
FILE TYPE: TEXT
NUMBER OF KEYS: 1
NUMBER OF LINES: 1

***** FILE INFORMATION ****
LINE FORMAT FIELDS
***** ********** **********
1 12,A30 INDEX,DESCR

***** FIELD INFORMATION *****
FIELD TYPE LENGTH DESCRIPTION
NAME (BYTES)
***** **** ********* **********
INDEX S 2 CLASSIFICATION ID ** KEY **
DESCR S 30 CLASSIFICATION DESCRIPTION
FILE NAME: DES-BTSM.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE: 34
PAGE SIZE: 512
RECORD NAME: DESDAT

*** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 KEY 1 4 Y N S

***** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
***** ******* ******** ********
TREEID S 2 1 TREE TABLE ID
GRPID S 2 3 BASIC TASK TABLE SUMMARY ID
DESC S 30 5 BASIC TASK SUMMARY TABLE NAME

*** NOTES ****
The fields 'TREEID' and 'GRPID' are combined to form the key field.

FILE NAME: DES-TASK.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE: 34
PAGE SIZE: 512
RECORD NAME: DESDAT

*** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 KEY 1 4 Y N S

***** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
***** ******* ******** ********
TREEID S 2 1 TREE TABLE ID
GRPID S 2 3 BASIC TASK TABLE SUMMARY ID
DESC S 30 5 BASIC TASK SUMMARY TABLE NAME

*** NOTES ****
The fields 'TREEID' and 'GRPID' are combined to form the key field.
FILE NAME: DES-TRWD.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 34
PAGE SIZE : 512
RECORD NAME : DESDAT

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 KEY 1 4 Y N S

***** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES) ********
TREEID S 2 1 TREE TABLE ID
GRPID S 2 3 BASIC TASK TABLE ID
DESC S 30 5 TREE WIDE TABLE NAME

**** NOTES ****
The fields 'TREEID' and 'GRPID' are combined to form the key field

FILE NAME: EQC-TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: EQCREC.INC
NUMBER OF KEYS: 1
RECORD SIZE : 50
PAGE SIZE : 512
RECORD NAME : EQCREC

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 EQCCDE 1 2 N N S

***** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES) ********
EQCCDE S 2 1 EQUIPMENT ID
EQCDES S 40 3 EQUIPMENT DESCRIPTION
EQCCOST R 4 43 DOLLERS PER HOUR
**FILE NAME:** F4C-YEAR.XDB
**FILE TYPE:** BTRIEVE
**INCLUDE FILE:** FYREC.INC
**NUMBER OF KEYS:** 1
**RECORD SIZE:** 34
**PAGE SIZE:** 512
**RECORD NAME:** FXREC

****** KEY INFORMATION ******

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FXF4CB</td>
<td>1</td>
<td>7</td>
<td>Y</td>
<td>N</td>
<td>S</td>
</tr>
</tbody>
</table>

****** RECORD INFORMATION ******

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE LENGTH POSITION (BYTES)</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FXF4CB</td>
<td>S 7 1</td>
<td>BEGINNING F4C CODE</td>
</tr>
<tr>
<td>FXF4CE</td>
<td>S 7 9</td>
<td>ENDING F4C CODE</td>
</tr>
<tr>
<td>FXTEID</td>
<td>S 2 17</td>
<td>TREE ID TABLE</td>
</tr>
<tr>
<td>FXBTDID</td>
<td>S 2 19</td>
<td>PERM. UNIT COST ID TABLE</td>
</tr>
<tr>
<td>FXSMID</td>
<td>S 2 21</td>
<td>SUMMARY ID TABLE</td>
</tr>
<tr>
<td>FXPTID</td>
<td>S 2 23</td>
<td>TEMP. UNIT COST ID TABLE</td>
</tr>
<tr>
<td>FXUNID</td>
<td>S 2 25</td>
<td>TOTAL UNIT COST ID TABLE</td>
</tr>
<tr>
<td>FXB4C</td>
<td>S 27</td>
<td>BEGINNING YEAR</td>
</tr>
<tr>
<td>FXEYER</td>
<td>S 31</td>
<td>ENDING YEAR</td>
</tr>
</tbody>
</table>

**FILE NAME:** FACTAB.DAT
**FILE TYPE:** TEXT
**NUMBER OF LINES:** 1

**FORMAT USED:** A80

****** RECORD INFORMATION ******

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE LENGTH POSITION (BYTES)</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FGID</td>
<td>S 9 1</td>
<td>FACILITY ID</td>
</tr>
<tr>
<td>FGSUBI</td>
<td>S 2 10</td>
<td>SUBINSTALLATION ID</td>
</tr>
<tr>
<td>FGAREA</td>
<td>S 2 12</td>
<td>AREA ID</td>
</tr>
<tr>
<td>FGDESC</td>
<td>S 30 14</td>
<td>FACILITY DESCRIPTION</td>
</tr>
<tr>
<td>FGF4C</td>
<td>S 7 29</td>
<td>F4C CODE</td>
</tr>
<tr>
<td>FGNUM</td>
<td>I 2 36</td>
<td>NUMBER OF FACILITIES</td>
</tr>
<tr>
<td>FGZONE</td>
<td>I 2 39</td>
<td>TRAVEL ZONE</td>
</tr>
<tr>
<td>FGMTH</td>
<td>I 2 41</td>
<td>WORK PERFORMANCE METHOD</td>
</tr>
<tr>
<td>FGSCLM</td>
<td>I 2 43</td>
<td>SPECIAL CONDITION MULTIPLIER ID</td>
</tr>
<tr>
<td>FGCHNG</td>
<td>S 8 45</td>
<td>LAST CHANGED DATE</td>
</tr>
<tr>
<td>FGLCAL</td>
<td>S 8 53</td>
<td>LAST CALCULATION DATE</td>
</tr>
<tr>
<td>FGFUND</td>
<td>S 10 61</td>
<td>FACILITY FUNDING PROFILE</td>
</tr>
<tr>
<td>FGCAL</td>
<td>I 2 63</td>
<td>CALCULATION MODELING ID</td>
</tr>
<tr>
<td>FGSOFT</td>
<td>I 4 64</td>
<td>FLOOR AREA (SQ FT)</td>
</tr>
<tr>
<td>FGCVR</td>
<td>I 2 73</td>
<td>CONSTRUCTION YEAR</td>
</tr>
<tr>
<td>FGSDSP</td>
<td>S 8 77</td>
<td>SCHEDULED DISPOSAL DATE</td>
</tr>
</tbody>
</table>

****** NOTES ******

USED TO BUILD THE 'FACILITY.XDB' FILE IN BATCH MODE.
FILE NAME: FACILITY.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: FAREC.INC
NUMBER OF KEYS: 3
RECORD SIZE : 120
PAGE SIZE : 1024
RECORD NAME : FGREC

**** KEY INFORMATION ****

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FGID</td>
<td>1</td>
<td>9</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>FGXSEQ</td>
<td>11</td>
<td>4</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>2</td>
<td>FGF4C</td>
<td>49</td>
<td>7</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

**** RECORD INFORMATION *****

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FGID</td>
<td>S</td>
<td>9</td>
<td>1</td>
<td>FACILITY ID</td>
</tr>
<tr>
<td>FGXSEQ</td>
<td>S</td>
<td>4</td>
<td>11</td>
<td>FACILITY SEQUENCE NUMBER</td>
</tr>
<tr>
<td>FGDESC</td>
<td>S</td>
<td>30</td>
<td>15</td>
<td>FACILITY DESCRIPTION</td>
</tr>
<tr>
<td>FGSUBI</td>
<td>S</td>
<td>2</td>
<td>45</td>
<td>SUBINSTALLATION ID</td>
</tr>
<tr>
<td>FGAREA</td>
<td>S</td>
<td>2</td>
<td>47</td>
<td>AREA ID</td>
</tr>
<tr>
<td>FGF4C</td>
<td>S</td>
<td>7</td>
<td>49</td>
<td>F4C CODE</td>
</tr>
<tr>
<td>FGNUM</td>
<td>I</td>
<td>2</td>
<td>57</td>
<td>NUMBER OF FACILITIES</td>
</tr>
<tr>
<td>FGZONE</td>
<td>I</td>
<td>2</td>
<td>59</td>
<td>TRAVEL ZONE</td>
</tr>
<tr>
<td>FGSQFT</td>
<td>I</td>
<td>4</td>
<td>61</td>
<td>FLOOR AREA (SQ FT)</td>
</tr>
<tr>
<td>FGCYR</td>
<td>I</td>
<td>2</td>
<td>65</td>
<td>CONSTRUCTION YEAR</td>
</tr>
<tr>
<td>FGDSP</td>
<td>S</td>
<td>8</td>
<td>67</td>
<td>SCHEDULED DISPOSAL DATE</td>
</tr>
<tr>
<td>FGFUNDF</td>
<td>S</td>
<td>2</td>
<td>75</td>
<td>FACILITY FUNDING PROFILE</td>
</tr>
<tr>
<td>MODSYS</td>
<td>S</td>
<td>1</td>
<td>77</td>
<td>COMPONENTS ENTERED</td>
</tr>
<tr>
<td>BFAPCT</td>
<td>I</td>
<td>2</td>
<td>79</td>
<td>PERCENTAGE OF BASE FACILITY</td>
</tr>
<tr>
<td>FGSCM</td>
<td>I</td>
<td>2</td>
<td>85</td>
<td>SPECIAL CONDITION MULTIPLIER ID</td>
</tr>
<tr>
<td>FGMTH</td>
<td>I</td>
<td>2</td>
<td>87</td>
<td>WORK PERFORMANCE METHOD</td>
</tr>
<tr>
<td>FGCAL</td>
<td>I</td>
<td>2</td>
<td>89</td>
<td>CALCULATION MODELING ID</td>
</tr>
<tr>
<td>FGCHNG</td>
<td>S</td>
<td>8</td>
<td>91</td>
<td>LAST CHANGED DATE</td>
</tr>
<tr>
<td>FGLCAL</td>
<td>S</td>
<td>8</td>
<td>99</td>
<td>LAST CALCULATION DATE</td>
</tr>
</tbody>
</table>
FILE NAME: FFPROF.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: FFPROF.INC
NUMBER OF KEYS: 1
RECORD SIZE : 90
PAGE SIZE : 512
RECORD NAME : FFREC

**** KEY INFORMATION ****
KEYNO  FIELD  POSITION  LENGTH  DUP  MOD  TYPE
*****  *****  ********  *****  ***  ***  ****
0      FFKEY   1        2      N    N    S

**** RECORD INFORMATION ****
FIELD  TYPE  LENGTH  POSITION  DESCRIPTION
NAME   (BYTES)
*****  *****  ********  ********
FFKEY   S    2        1      FACILITY FUNDING PROFILE ID
FFDSCR  S    40       3      PROFILE DESCRIPTION
FFLAP   S    (5)2    43     IN-HOUSE LABOR APPROPRIATION ID
FFLPR   I    (5)2    53     IN-HOUSE LABOR PERCENTAGE
FFEAP   S    (5)2    63     IN-HOUSE EQUIPMENT APP. ID
FFEPR   I    (5)2    73     IN-HOUSE EQUIPMENT PERCENTAGE
FFLNO   I    4        83     NUMBER OF LABOR ACCOUNTS
FFENO   I    4        87     NUMBER OF EQUIPMENT ACCOUNTS

FILE NAME: INSTINFO.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 6

**** FILE INFORMATION ****
LINE  FORMAT  FIELDS
*****  ***********  ********
1      A30      INSTNAM
2      1X,A4,1X,A4 BEGYR,ENDYR
3      3F5.2   MATADJ,MATTAF,RMFTAF
4      A2      ORGID
5      A2      MAXLIN
6      A1      VDRIVE

**** FIELD INFORMATION ****
FIELD  TYPE  LENGTH  DESCRIPTION
NAME   (BYTES)
*****  *****  ********  ********
INSTNAM S    30      INSTALLATION NAME
BEGYR   S    4       BEGINNING REPORT YEAR
ENDYR   S    4       ENDING REPORT YEAR
MATADJ  R    4       MATERIAL LOCATION ADJUSTMENT FACTOR
MATTAF  R    4       MATERIAL TIME ADJUSTMENT FACTOR
RMFTAF  R    4       RMF TIME ADJUSTMENT FACTOR
ORGID   S    2       ORGANIZATION ID
MAXLIN  S    2       MAXIMUM NUMBER OF LINES PER PAGE
VDRIVE  S    1       DRIVE DESIGNATION OF VIRTUAL DISK, C=NONE
**FILE NAME:** ORGFGC.XDB  
**FILE TYPE:** BTRIEVE  
**INCLUDE FILE:** **NONE**  
**NUMBER OF KEYS:** 2  
**RECORD SIZE:** 52  
**PAGE SIZE:** 512  
**RECORD NAME:** ORGREC

#### KEY INFORMATION

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ORGKEY</td>
<td>1</td>
<td>4</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>MACOMID</td>
<td>43</td>
<td>2</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

#### RECORD INFORMATION

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORGKEY</td>
<td>S</td>
<td>4</td>
<td>1</td>
<td>ORGANIZATION ID</td>
</tr>
<tr>
<td>ORGCDE</td>
<td>S</td>
<td>2</td>
<td>5</td>
<td>ORGANIZATION CODE</td>
</tr>
<tr>
<td>INSTID</td>
<td>S</td>
<td>2</td>
<td>7</td>
<td>INSTALLATION ID</td>
</tr>
<tr>
<td>ORGDESC</td>
<td>S</td>
<td>30</td>
<td>13</td>
<td>ORG. DESCRIPTION</td>
</tr>
<tr>
<td>MACOMID</td>
<td>S</td>
<td>43</td>
<td></td>
<td>MACOM ID</td>
</tr>
<tr>
<td>RELCDE</td>
<td>S</td>
<td>6</td>
<td>45</td>
<td>RELATION CODE</td>
</tr>
<tr>
<td>SUBCDE</td>
<td>S</td>
<td>2</td>
<td>51</td>
<td>SUB-INSTALLATION CODE</td>
</tr>
</tbody>
</table>

---

**FILE NAME:** PMCOMP.XDB  
**FILE TYPE:** BTRIEVE  
**INCLUDE FILE:** **NONE**  
**NUMBER OF KEYS:** 3  
**RECORD SIZE:** 64  
**PAGE SIZE:** 512  
**RECORD NAME:** COMPDT

#### KEY INFORMATION

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>STF4C</td>
<td>1</td>
<td>8</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>CASCES</td>
<td>17</td>
<td>8</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
<tr>
<td>2</td>
<td>IFSNUM</td>
<td>25</td>
<td>8</td>
<td>Y</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

#### RECORD INFORMATION

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>STF4C</td>
<td>S</td>
<td>8</td>
<td>1</td>
<td>STARTING F4C CODE</td>
</tr>
<tr>
<td>END4C</td>
<td>S</td>
<td>8</td>
<td>9</td>
<td>ENDING F4C CODE</td>
</tr>
<tr>
<td>CASCES</td>
<td>S</td>
<td>8</td>
<td>17</td>
<td>TASK/COMPONENT ID</td>
</tr>
<tr>
<td>IFSNUM</td>
<td>S</td>
<td>8</td>
<td>25</td>
<td>IFS NUMBER</td>
</tr>
<tr>
<td>PMNUM</td>
<td>S</td>
<td>2</td>
<td>33</td>
<td>NUMBER OF THE CURRENT PREDICTION MODEL</td>
</tr>
<tr>
<td>LCALDTE</td>
<td>S</td>
<td>8</td>
<td>35</td>
<td>LAST CALCULATION DATE</td>
</tr>
<tr>
<td>LMRGDTM</td>
<td>S</td>
<td>8</td>
<td>43</td>
<td>LAST MERGE DATE</td>
</tr>
<tr>
<td>TOTPM</td>
<td>S (6)</td>
<td>2</td>
<td>51</td>
<td>TOTAL NUMBER OF ALLOWABLE PRED. MODELS</td>
</tr>
<tr>
<td>ALLOWPM</td>
<td>S</td>
<td>2</td>
<td>53</td>
<td>LIST OF THE ALLOWABLE PRED. MODELS</td>
</tr>
</tbody>
</table>
FILE NAME: PMDEF.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE: 42
PAGE SIZE: 512
RECORD NAME: PMDDAT

### KEY INFORMATION

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PMNUM</td>
<td>1</td>
<td>2</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
</tbody>
</table>

### RECORD INFORMATION

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMID</td>
<td>S</td>
<td>2</td>
<td>1</td>
<td>PREDICTION MODEL ID</td>
</tr>
<tr>
<td>PMDEF</td>
<td>S</td>
<td>40</td>
<td>3</td>
<td>PREDICTION MODEL DEFINITION</td>
</tr>
</tbody>
</table>

FILE NAME: PMF4C.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE: 48
PAGE SIZE: 512
RECORD NAME: F4CDAT

### KEY INFORMATION

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>STF4C</td>
<td>1</td>
<td>8</td>
<td>N</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

### RECORD INFORMATION

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>STF4C</td>
<td>S</td>
<td>8</td>
<td>1</td>
<td>STARTING F4C CODE</td>
</tr>
<tr>
<td>ENDF4C</td>
<td>S</td>
<td>8</td>
<td>9</td>
<td>ENDING F4C CODE</td>
</tr>
<tr>
<td>PMNUM</td>
<td>S</td>
<td>2</td>
<td>17</td>
<td>NUMBER OF THE CURRENT PREDICTION MODEL</td>
</tr>
<tr>
<td>LCALDTE</td>
<td>S</td>
<td>8</td>
<td>19</td>
<td>LAST CALCULATION DATE</td>
</tr>
<tr>
<td>LMGDTE</td>
<td>S</td>
<td>8</td>
<td>27</td>
<td>LAST MERGE DATE</td>
</tr>
<tr>
<td>TOTPM</td>
<td>S</td>
<td>2</td>
<td>35</td>
<td>TOTAL NUMBER OF ALLOWABLE PRED. MODELS</td>
</tr>
<tr>
<td>ALLOWPM</td>
<td>S</td>
<td>2(6)</td>
<td>37</td>
<td>LIST OF THE ALLOWABLE PRED. MODELS</td>
</tr>
</tbody>
</table>
FILE NAME: RMF-FACT.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: **NONE**
NUMBER OF KEYS: 1
RECORD SIZE : 336
PAGE SIZE : 512
RECORD NAME : RMFREC

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
**** **** ****** ****** *** *** ****
0 STAMS 1 7 N Y S

**** RECORD INFORMATION ****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
**** ****** ******** ******* ************
STAMS S 7 1 STARTING AMS CODE
ENDAMS S 7 9 ENDING AMS CODE
RMCOST R (80)4 17 ARRAY OF COST FACTORS (80 YEARS)

FILE NAME: RSMTTOTL.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: RSREC.INC
NUMBER OF KEYS: 1
RECORD SIZE : 340
PAGE SIZE : 1024
RECORD NAME : RSREC

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
**** **** ****** ****** *** *** ****
0 RSRTSK 1 7 N N S

**** RECORD INFORMATION ****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
**** ****** ******** ******* ************
RSRTSK S 8 1 COMPONENT ID
RSRTRD I 2 9 TRADE INDEX
RSRWPM I 2 11 WORK PERFORMANCE METHOD
RSRBFY I 2 13 BEGINNING YEAR
RSRFY I 2 15 NUMBER OF YEARS
RSROCC R (10)4 17 NUMBER OF OCCURRENCES (10 YEARS)
RSRTOT R (10)4 57 TOTAL COSTS (10 YEARS)
RSRHRS R (3,10) 97 ARRAY OF HOURS (10 YEARS)
RSRDLR R (3,10) 217 ARRAY OF COSTS (10 YEARS)

**** NOTES ****
Both two dimensional arrays (RSRHRS & RSRDLR) are divided as follows:
(1, I) = LABOR
(2, I) = EQUIPMENT
(3, I) = MATERIALS
FILE NAME: SCMDEF.XDB  
FILE TYPE: BTRIEVE  
INCLUDE FILE: ***NONE***  
NUMBER OF KEYS: 1  
RECORD SIZE: 32  
PAGE SIZE: 512  
RECORD NAME: SCMDAT

<table>
<thead>
<tr>
<th>KEYNO</th>
<th>FIELD</th>
<th>POSITION</th>
<th>LENGTH</th>
<th>DUP</th>
<th>MOD</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SCMID</td>
<td>1</td>
<td>2</td>
<td>N</td>
<td>Y</td>
<td>S</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCMID</td>
<td>S</td>
<td>2</td>
<td>1</td>
<td>SPECIAL CONDITION MULTIPLIER ID</td>
</tr>
<tr>
<td>SCMDEF</td>
<td>S</td>
<td>30</td>
<td>3</td>
<td>SCM DEFINITION</td>
</tr>
</tbody>
</table>

FILE NAME: SCMDEF.DAT  
FILE TYPE: TEXT  

FORMAT USED: A2,A30

<table>
<thead>
<tr>
<th>FIELD</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>POSITION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>S</td>
<td>2</td>
<td>1</td>
<td>SPECIAL CONDITION MULTIPLIER ID</td>
</tr>
<tr>
<td>DESC</td>
<td>S</td>
<td>30</td>
<td>3</td>
<td>SCM DEFINITION</td>
</tr>
</tbody>
</table>
FILE NAME: SCMIDxx.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 20

**** FILE INFORMATION ****
LINE FORMAT FIELDS
****** ********** ******
1  I5   NUMBER OF DATA LINES(19)
2-20 A7,3X,F10.2 COMPONENT ID, CALCULATION TOTALS

**** RECORD INFORMATION ****
LINE COMPONENT CALCULATION
****** ********** **********
 2  0000000  F(9)*F(13)
 3  0300000  F(1)*F(2)*F(5)*F(6)*F(7)*F(8)
 4  0400000  F(1)*F(2)*F(5)*F(6)*F(8)*F(11)
 5  0410000  F(15)
 6  0415400  F(4)
 7  0415000  F(4)
 8  0415800  F(4)
 9  0415900  F(4)
10  0415A00  F(4)
11  0415E00  F(4)
12  0415F00  F(4)
13  0415G00  F(14)
14  0415H00  F(14)
15  0420000  F(14)
16  0430000  F(14)
17  0500000  F(11)
18  0530000  F(12)
19  0540000  F(12)
20  0600000  F(11)

**** NOTES ****
The 'xx' IS REPLACES BY THE SPECIAL CONDITION MULTIPLIER ID

FILE NAME: STDREP.XDB
FILE TYPE: BTFILE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 4
PAGE SIZE : 512
RECORD NAME : STDREC

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
****** ***** ******** ** ** ** **
 0  STDKEY   1  4  N  N  N  S

**** RECORD INFORMATION ****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
******  **** ******** ******** ********
STDAPR  S  2  1  APPROPRIATION ID
STDAMS  S  2  3  AMS CODE

**** NOTES ****
The fields 'STDAPR' AND 'STDAMS' ARE COMBINED TO FORM THE KEY FIELD
FILE NAME: SUB_TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 32
PAGE SIZE : 512
RECORD NAME : ARDAT

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 CODE 1 2 N Y S

**** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
***** **** ******** ****** ************
CODE S 2 1 ** SUBINSTALLATION ID
DEF S 30 3 ** SUBINSTALLATION DESCRIPTION

FILE NAME: UNCDSC.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: UNCDSC.INC
NUMBER OF KEYS: 1
RECORD SIZE : 52
PAGE SIZE : 512
RECORD NAME : UNCRD

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** ****** *** *** ****
0 UNTID 1 2 N Y S

**** RECORD INFORMATION *****
FIELD TYPE LENGTH POSITION DESCRIPTION
NAME (BYTES)
***** **** ******** ****** ************
UNTID S 2 1 UNIT COST TABLE ID
DESP S 3 ** DESCRIPTION
FILE NAME: URRAPR.XDB
FILE '/PE: BTRIEVE
INCLUDE FILE: NONE
NUMBER OF KEYS: 1
RECORD SIZE : 50
PAGE SIZE : 512
RECORD NAME : APRMUL

**** KEY INFORMATION ****
KEYNO FIELD POSITION LENGTH DUP MOD TYPE
***** ***** ******** *** *** ****
0 AMSCOD 1 6 N Y S

**** RECORD INFORMATION ****
FIELD NAME (BYTES) TYPE LENGTH POSITION DESCRIPTION
***** **** ******** ******** ********
APRID S 6 1 APPROPRIATION ID
MULPLY R 4(10) 7 MULTIPLIER
6 STANDARD PROGRAMMING PACKAGES

These programming packages are used by the MRPM system:
1. MSFORTRAN 3.31
2. BTRIEVE
3. MS CHART

Each package has its own printed documentation.
7 REQUIRED PROGRAMMING PRACTICES

To ensure system uniformity, programmers should follow these instructions:

1. Always get source codes from and return them to the coordinator; do not update the execution file in the main machine. Only the coordinator is allowed to write into the main machine.

2. Test the program as you work on it, and make sure it is 100 percent error free. Write down all testing procedures for future reference.

3. The first line of each screen should be a header that contains the exact words of the selection menu.

4. Always show the computer status on the screen: 'COMMAND MODE', 'EDIT MODE', 'ADD MODE'; this helps users know where they are.

5. When the user presses 'F8' to delete records, always ask to make sure the data should be deleted.

6. Use the bottom line for standard functions. Show only functions that can be used in this mode.

7. On reports and lists to the printer, be sure to print the last page.

8. Use the 'F6 BEGIN' key to start the processing on all functions.

9. Programs must return to the previous screen after successful execution.

10. Use the standard error message subroutine "ERRMSG" to handle all the pop-up window style error messages.

11. Always make a source code backup before working on the source code. Always save the final program on a diskette in your diskette file. Write the date on the diskette.

12. There is no 'STOP' command in the program.

13. Increase labels in increments of 100.

Figure 1 shows an example of a well-documented program.
PROGRAM RECFACZ
* ==> this program recovers the general facility information
* stored in FACILITY.XDB from subdirectories
IMPLICIT INTEGER*2 (A-Z)
CHARACTER SDIR*2,EDIR*2,BUFFER*30
INTEGER*2 FILE1(64),FILE2(64),NSDIR,NEDIR,DIR
INTEGER*2 CODE,OP
CHARACTER KEY*9
CHARACTER PATH*18,SDRIVE*1,NEWDIR*2
LOGICAL*2 THERE,OPENED,SUCESS
COMMON /COLORS/ COLOR
* ==> include file for error message
$INCLUDE: 'BTEROR.INC'
* ==> include file for FACILITY.XDB
$INCLUDE: 'FACREC.INC'
* ==> include files for color table
$INCLUDE: 'COLOR1.INC'
$INCLUDE: 'COLOR2.INC'
ROUTIN = 'REC-FAC'
* ==> initialize the error message subroutine
CALL ERRMSG(-1,ROUTIN,'INITIAL')
* ==> install function keys
CALL INFKEY
* ==> Make sure Record Manager is loaded
IF (BTREEX(0).NE.0) THEN
  INFOE(1)='Record Manager is not Loaded'
  CALL ERRMSG(99,ROUTIN,"
GOTO 999
ENDIF
* ==> set up for screen
100 CALL SCROLL (00,00,24,80,00,COLOR(2))
CALL FLINE
CALL FKEYS (1,1,1,1,1,1,1,1,1,2)
CALL BOX(1,1,3,80,2,COLOR(2))
CALL BOX(4,1,24,80,2,COLOR(2))
CALL WT (2,25,'REBUILD FACILITY.XDB FROM SUBS',31,COLOR(14))
CALL WT (10,15,'Starting Subdirectory : ',23,COLOR(14))
CALL WT (12,15,'Ending Subdirectory : ',23,COLOR(14))
* ==> read screen
200 CODE = 5
CALL RT (10,39,SDIR,2,CODE,COLOR(8),COLOR(15))
IF (CODE.EQ.68) GOTO 999
IF (CODE.EQ.80.OR.COD_EQ.13) GOTO 300
GOTO 200
300 CODE = 5
CALL RT (12,39,EDIR,2,CODE,COLOR(8),COLOR(15))
IF (CODE.EQ.68) GOTO 999
IF (CODE.EQ.64) GOTO 400
IF (CODE.EQ.72) GOTO 200
IF (CODE.EQ.13) THEN
  CALL FLINE
  CALL FKEYS (1,1,1,1,2,1,1,1,2)
ENDIF
GOTO 300
400 CONTINUE
* ==> create FACTEMP.XDB in current directory
CALL CRFAC(FILE1,SUCESS)
* ==> if not created sucessfully terminate the program
IF (NOT.SUCESS) GOTO 999
* ==> open FACTEMP.XDB in current directory
CALL BTREE (FILE1,0,0,'FACTEMP.XDB',9,FGREC,120,STS)
IF (STS.NE.0) THEN
  INFOE(1) = 'ERROR OPEN'
  CALL ERRMSG(STS,ROUTIN,'DF..98')
GOTO 999
ENDIF

Figure 1. Example of a well-documented program.
READ(SDIR,'(I2)') NSDIR
READ(EDIR,'(I2)') NEDIR
DIR = NSDIR
500 CONTINUE
OPEND=.FALSE.
WRITE(NEWDIR,'(12)') DIR
CALL FINDDIR(NEWDIR,SDRIVE)
* => set up filename for subdirectory general facility file
PATH='E:\01\FACILITY.XDB'
PATH(1:1)=SDRIVE
PATH(4:5)=NEWDIR
* => open FACILITY.XDB in subdirectory
CALL BTREE (FILE2,0,0,PATH,18,FGREC,120,STS)
IF (STS.eq.12) THEN
INFOE(1) = 'FACILITY.XDB not found in Subdirectory'
INFOE(2) = PATH
INFOA(1) = 'Please check files in Subdirectory'
CALL ERRMSG(99,ROUTIN,' ')
GOTO 800
ENDIF
IF (STS.NE.0) THEN
FILNAM = 'FACILITY.XDB'
INFOE(1) = 'Error in opening file'
CALL ERRMSG(STS, ROUTIN, 'OPENO2')
GOTO 800
END IF
OPEND=.TRUE.
* => write processing information to screen
BUFFER = 'RECOVER FACILITY IN E:\'
BUFFER(22:22) = SDRIVE
WRITE(BUFFER(25:26),'(I2)') DIR
CALL WT (20,15,BUFFER,30,COLOR(14))
* => loop for copy facility records
OP = 12
600 CONTINUE
CALL BTREE (FILE2,OP,0,KEY,9,FGREC,120,STS)
IF (STS.EQ.9) GOTO 700
OP = 6
WRITE(FGDIR,'(12)') DIR
CALL BTREE (FILE1,2,0,KEY,9,FGREC,120,STS)
IF (STS.EQ.5) THEN
INFOE(1) = 'Facility ID = '
INFOE(1)(15:23) = FGID
INFOE(2) = 'Facility ID already exists'
INFOA(1) = 'Please check facility ID in subdirectory'
CALL ERRMSG(99,ROUTIN,' ')
GOTO 600
ENDIF
IF (STS.NE.0) THEN
INFOE(1) = 'ERROR INSERTING FILE'
CALL ERRMSG(STS, ROUTIN, 'D...HGL')
GOTO 990
ENDIF
GOTO 600.
700 CONTINUE
* => close FACILITY.XDB in subdirectory
CALL BTREE (FILE2,1,0,KEY,9,FGREC,120,STS)
IF (STS.EQ.0) OPEND=.FALSE.
800 CONTINUE
DIR = DIR + 1
IF (DIR.GT.NEDIR) GOTO 990
GOTO 500
990 CONTINUE

Figure 1. Cont'd.
IF (OPENP) CALL BTREE(FILE2, 1, 0, KEY, 9, FGREC, 120, STS)
CALL BTREE (FILE1, 1, 0, KEY, 9, FGREC, 120, STS)
* ===> overwrite FACILITY.XDB
CALL COMMAND ('COPY FACTEMP.XDB FACILITY.XDB >NUL'C, ERR)
CALL COMMAND ('DEL FACTEMP.XDB >NUL'C, ERR)
999 END
* ===> subroutine for creating general facility information
SUBROUTINE CRFAC(FILE1, SUCESS)
IMPLICIT INTEGER*2 (A-Z)
INTEGER*2 FILE1(64), MAKE(32)
LOGICAL SUCESS

INCLUDE 'BTERROR.INC'
* ===> file information for FACILITY.XDB
DATA MAKE/120, 1024, 3*5*0, 1, 9, 0, 5*0, 11, 4, 2, 5*0, 49, 7, 3, 5*0/
SUCESS=.TRUE.
ROUTIN = CRFAC
CALL BTREE (FILE1, 14, 0, 'FACTEMP.XDB', 10, MAKE, 64, STS)
IF (STS.NE.0) THEN
INFOE(1) = 'ERROR IN CREATE FILE'
CALL ERRMSG(STS, ROUTIN, 'X.987')
SUCESS=.FALSE.
ENDIF
RETURN
END

Figure 1. Cont'd.
A toll-free 800 number should be provided to all users. This hotline should be used to report possible improvements, questions, and system failures. The success of the MRPM system depends solely on how well system operators/programmers respond to the questions and problems that users bring to the telephone hotline. The following procedure will ensure that users receive a fast response to all problems and questions:

1. The hotline operator will note the problem and take immediate action to answer questions. The names of the MRPM user and the hotline operator, along with the message and any action taken, should be recorded on a sequentially numbered, three-part report log (Figure 2). When a specific function is being addressed, the function name should also be recorded. The log contains a white, yellow, and red copy of the report. The last (red) copy should be entered into the official logbook immediately. The white and yellow copies are given to the processor.

2. The processor should take action on the problem as soon as possible and record any further action taken on the report log. The processor keeps the yellow copy of the report for a record, and passes the white copy to the reviewer.

3. If a system command causes the question or problem, then a reviewer should reassess the complete command to ensure that the command works as intended. This second check is for the sake of quality assurance.

4. The problem log is returned to the hotline operator, who will call the users to report the action on their specific problems. It is important to maintain this direct contact with the user, and to keep an informal relationship between the system operator and the user. When the user's problem has been resolved, the completed white form should replace the red action form in the official report log book.

5. At any step in the process, emergencies should be referred to the supervisor for review. Periodically, a supervisor should review the accumulation of reports to prioritize the problems recorded in the log, to specify action to resolve specific problems, and to delegate the workload.

6. A regular newsletter should be mailed to each user, giving a short description of new changes to the system created in response to user requests. Users and their organizations should be credited for their suggestions.
<table>
<thead>
<tr>
<th>Report: Reporters Name/Org:</th>
<th>Tel. No.:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Date:</td>
<td>Time:</td>
<td>Received by:</td>
</tr>
<tr>
<td>Message:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action taken:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Related commands:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Review: Reviewed by:</td>
<td>Date:</td>
<td></td>
</tr>
<tr>
<td>Priority:</td>
<td>Sent to:</td>
<td>Date sent:</td>
</tr>
<tr>
<td>Comments:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process: Processed by:</td>
<td>Completion date:</td>
<td></td>
</tr>
<tr>
<td>Action taken:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Review: Reviewed by:</td>
<td>Date:</td>
<td></td>
</tr>
<tr>
<td>Comments:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Approval:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feedback: Date reporter called:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Callers name:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. MRPM report log.
There are five basic functions that must be maintained to provide full-service support:

1. Supervision
2. Training
3. Hotline
4. HQ-IFS System Maintenance
5. HQ-IFS Operations.

Supervision

The functions of the supervisor include scheduling training; review and assignment of report logs; management of all corrections, improvements, and problem identification. This function will probably consume 25 percent of a man-year at a GS-11 level, at a cost of approximately $12,800 per year.

Training

There are three types of training:

1. Self-teaching using the system manual
2. On-site training of the user
3. Centralized (group) training.

The self-teach method requires the user to have access to a resource person to answer questions at later stages of progress. This function is performed by the hotline operator.

On-site training entails sending one person to a site for a minimum of 3 days. The user provides equipment and training room. The cost for each such session would be:

GS-11 Trainer 5 days @ $200/day $1000
TDY, 4 days @$100/day, air fare and car 1000
Supplies, manuals, etc. 300
Total $2300

This training method is by far the best possible training situation for both the trainer and the students. Five classes per year at $2300 each would total $11,500.

Centralized training is the most expensive way to perform training in the Army. All students must travel to one central site. The central site must rent computer equipment to perform the training. (During this training each student should have a PC and no more than two students should be assigned to one PC.) This equipment may not be the same as that used by the trainees at their installations. A central Army training center must be paid to plan and conduct the training. Estimated costs will be:
Hotline

The hotline is a telephone number used to answer user questions, handle user problems, and report suggestions for improvement. This number should be given to all users. The person answering the hotline should be able to either answer all basic problems or refer the request to someone else for action. This activity would probably consume about 15 percent of one GS-9 or $8000 per year.

PC System Maintenance

One standard system will be required at an initial purchase cost of $35,000. Annual system hardware maintenance costs would run $2000/yr. The U.S. Army Construction Engineering Research Laboratory (USACERL) will transfer the system to the U.S. Army Engineering and Housing Support Center (USAESHC) at a cost of $20,000. Two GS-9 Fortran programmers must be trained on the use of the system. Training will take 6 months at a cost of $30,000. Normal annual requirement will be the equivalent of one half-time person at a cost of $15,000/yr.

Newsletter

A quarterly newsletter should inform users of updates and answers to common questions. The annual cost is estimated at $5000.

Cost Summary

<table>
<thead>
<tr>
<th>Initial Costs</th>
<th>Annual Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>USACERL transfer</td>
<td>$20K</td>
</tr>
<tr>
<td>Personal computer</td>
<td>$35K</td>
</tr>
<tr>
<td>Fortran program (2 @ 6 mo)</td>
<td>$30K</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>$85K</strong></td>
</tr>
<tr>
<td>Supervisor (1/4 time)</td>
<td>$12K</td>
</tr>
<tr>
<td>User training (five classes onsite)</td>
<td>$12K</td>
</tr>
<tr>
<td>Hotline (15 percent)</td>
<td>$8K</td>
</tr>
<tr>
<td>PC maintenance</td>
<td>$2K</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>$58K</strong></td>
</tr>
</tbody>
</table>

For a class of 20 students (10 computers) Total $27,300
USACERL DISTRIBUTION

Chief of Engineers
ATTN: CEHEC-IM-LH (2)
ATTN: CEHEC-IM-LP (2)
ATTN: CERD-L
ATTN: DAEN-ZCP-B

CEHSC 22060
ATTN: CEHSC-FM-R

Ft Belvoir, VA
ATTN: CECC-R 22060

Defense Technical Info. Center 22304
ATTN: DTIC-FAB (2)

10
8/91

This publication was reproduced on recycled paper.