ANL-7408
RETURN TO ANL (IDAHO) LIBRARY.

7408

ANL-7408

Argonne National Laboratory

COMPUTER ENVIRONMENT REPORT

by

M. K. Butler and A. L. Rago

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona
Carnegie-Mellon University
Case Western Reserve University
The University of Chicago
University of Cincinnati
Illinois Institute of Technology
University of Illinois
Indiana University
Iowa State University
The University of Iowa
Kansas State University
The University of Kansas
Loyola University
Marquette University
Michigan State University
The University of Michigan
University of Minnesota
University of Missouri
Northwestern University
University of Notre Dame
The Ohio State University
Ohio University
The Pennsylvania State University
Purdue University
Saint Louis University
Southern Illinois University
University of Texas
Washington University
Wayne State University
The University of Wisconsin

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Printed in the United States of America
Available from
Clearinghouse for Federal Scientific and Technical Information
National Bureau of Standards, U. S. Department of Commerce
Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.65

ANL-7408
Mathematics and
Computers (TID-4500)
AEC Research and
Development Report

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

COMPUTER ENVIRONMENT REPORT

by

M. K. Butler and A. L. Rago

Applied Mathematics Division

with Appendices by

George Concaildi, Donald Jordan, Allen S. Kennedy,
Ronald F. Krupp, Clifford LeVee, John Ohde,
Roger Rempert, and George A. Robinson

February 1968

TABLE OF CONTENTS

	Page
I. INTRODUCTION AND ACKNOWLEDGMENTS	11
II. IBM 360 COMPUTER ENVIRONMENT	12
A. Software Systems	12
1. Resident System	12
2. Catalogued Procedures	18
B. Hardware Description	50
1. Equipment List	50
2. Configuration Chart	52
C. Bibliography	53
III. CDC 3600 COMPUTER ENVIRONMENT	55
A. Software Systems	55
1. Resident System	55
2. Software Systems Available	56
3. ANL-SCOPE/SATCOPS-II Control Cards	56
B. Hardware Description	59
1. Equipment List	59
2. Configuration Chart	60
C. Bibliography	60
1. 3600	60
2. 160A	61
IV. PERIPHERAL AND SPECIAL-PURPOSE EQUIPMENT	62
A. Peripheral Equipment Listing	62
B. Peripheral Equipment Bibliography	63
1. Control Data Corporation	63
2. California Computer Products	63
C. Special-purpose Equipment	63

TABLE OF CONTENTS

Page

APPENDIXES

A. POINTR, A Dynamic Storage Allocation Program 66

1. Introduction 66
2. Machine 66
3. Usage 67
   a. Initialization 67
   b. Requesting Storage 67
   c. Requesting Pointers 68
   d. Array Purging 69
   e. Obtaining Available Storage 69
   f. Array Clearing 69
   g. Dynamic Dumps 69
   h. Error Procedure 70
4. Restrictions 71
5. Sample Program 71
   a. Description of Sample Program by Steps 71
   b. Output from Sample Program 74
6. POINTR FORTRAN Listings 79

B. A Disk-File Control System for the Control Data 3600 89

1. Introduction 89
2. Organization of User Data on the Disk 90
3. Using the Disk File 92
   a. Defining a Logical Unit on the Disk File 92
   b. Defining and Creating a File Complex Control Array 92
   c. Changing Pointer Positions 98
   d. Data Transmission 99
   e. Marking an End-of-File 101
   f. Checking Status 102

TABLE OF CONTENTS

Page

4. Disk-File Driver Specifications 103

a. SCOPE Calls and Driver Specifications 103
b. Control Words 108
c. Referencing a Systems Disk 109
d. Write Lockout 109

5. Diagnostics 109

a. SCOPE Recovery Dump 109
b. Q8QERROR 110

C. S/360 Programming Techniques for the CalComp 780 111

1. Introduction 111

2. General Discussion of Plotter Programming 112

3. Detailed Descriptions 113

a. Sign Conventions 113
b. Plotting Area 114
c. CalComp Subroutines 114

4. Sample Problem 122

5. Acknowledgments for Appendix C 130

D. Graphic Output 131

1. Introduction 131

2. Advantages and Disadvantages of Microfilm Output 131

a. Advantages 131
b. Disadvantages 131

3. How to Use the New Package 132

4. Correspondence of Plotting Area 132

5. Restrictions on Microfilm Output 133

a. Stacking Procedure 133
b. Axis Plotting 133
c. Partial Symbol Drawing 133
d. Data Outside the Available Area 134

TABLE OF CONTENTS

Page

E. CalComp Plotting on the CDC 3600 135

1. Subroutine Package to Prepare Input to CalComp 580 and 780 Plotter 135

a. General Information 135
b. Specific Description of the Subroutines 135

2. CDC 3600 FORTRAN Programming for the CalComp 580 and 780 Plotters 140

a. Introduction 140
b. Deck Arrangement 140
c. CALL PLOTS(ARRAY,LENGTH,LUN) 141
d. CALL PLOT(X,Y,IND) 142
e. CALL LINE(X,Y,N,K) 143
f. CALL SCALE(X,N,S,XMIN,DX,K) 143
g. CALL AXIS(X,Y,BCD,N,S,THETA,XMIN,DX) 145
h. CALL SYMBOL(X,Y,H,BCD,THETA,N) 146
i. Subroutines LINE, AXIS, and SCALE 148
j. Program GRAPH 152
k. Program TEST 1 154
l. Program TEST 3 156
m. Program TRANSGRF 158
n. Program TRLINE (Test 2) 159
o. Program COMBGRAF 160

F. Specifications for the Input/Output Drivers for the SCOPE Monitor on the CONTROL DATA 3600 164

1. Introduction 164

2. Driver Specifications 166

a. Initializing 168
b. Connecting the Equipment 168
c. Processing the User's Request 169
d. Translating the Data for Core-to-Device Transmission 169
e. Checking the I/O Control Word 170
f. Modifying the Interrupt Table (ITAB) 170
g. Readying the Equipment 170

TABLE OF CONTENTS

Page

h. Initiating the I/O Operation 171
i. Returning Control to SCOPE 171
j. Reject Processing 172
k. Error Processing 173
l. External Equipment Interrupt Processing 174
m. Aborting a Job 174

3. Changes to SCOPE 175

a. Hardware Mnemonics and Hardware Codes 175
b. SCOPE EQAIOC Table Entries. 176
c. Placing the Driver in the System 180

4. Loading a Driver at Run Time. 180

5. Bibliography for Appendix F 180

LIST OF FIGURES

No.	Title	Page
1.	Configuration Chart of IBM System/360 System	52
2.	Configuration Chart of CDC 3600 Computer System	60
C.1.	Sign Conventions, Shown on CalComp Model 565 Digital Incremental Plotter	113
C.2.	Planning Layout	114
C.3.	Characters Available in Symbol Routine (IBM 360)	119
E.1.	Normal Flow Diagram	135
E.2.	Figure Generated for Example 1	142
E.3.	Figure Generated for Example 2	143
E.4.	Figure Generated for Example 5	144
E.5.	Figure Generated for Example 6	145
E.6.	Figure Generated for Example 7	145
E.7.	Regular Character Set; Bioctal Equivalence	146
E.8.	Special Character Set; Integer Equivalence	147
E.9.	Figure Generated for Example 8	147
E.10.	Figure Generated for Example 9	148
E.11.	Plot Generated by Program GRAPH	153
E.12.	Plot Generated by Program TEST 1	155
E.13.	Plot Generated by Program TEST 3	157
E.14.	Plot Generated by Program TRANSGRF	158
E.15.	Plot Generated by Program TRLINE	159
E.16.	Plot Generated by Program COMBGRAF for Y = SINX	161
E.17.	Plot Generated by Program COMBGRAF for Y = X	162
E.18.	Plot Generated by Program COMBGRAF for Y = SINX Rotated 45°	163
F.1.	Diagram of SCOPE I/O Processing	165

LIST OF TABLES

No.	Title	Page
A.I.	Contents of Container Array A at Various Steps in Sample Program	73
E.I.	Table of Subroutine Entry Points and Size	136

COMPUTER ENVIRONMENT REPORT

by

M. K. Butler and A. L. Rago

I. INTRODUCTION AND ACKNOWLEDGMENTS

Preparation of this report stems from the authors' interests and activities over the past years in the area of computer-program documentation, transfer, and exchange. What documentation of a computer program should be provided to ensure that:

a. Another person can most effectively use the program?

b. Another programmer can most readily modify or extend the program?

c. The program can most easily be moved to another computer configuration?

These questions are not unique to the Laboratory, and answers are being sought in most computer installations and by many computer user groups and professional societies. The American Nuclear Society is one of these. In 1966 a Society Subcommittee formulated "A Code of Good Practices for the Documentation of Digital Computer Programs," which was published as Nuclear Engineering Bulletin 4-1. Currently, this Committee, of which the authors are members, is preparing a description of recommended programming practices and program-exchange pitfalls.

Although a computer program may be completely written in a machine-independent language, such as FORTRAN, it requires a certain minimum environment to function properly. This environment consists of the software systems and hardware devices and equipment available at the originating installation. Consequently, to be complete, documentation of the program must include a description of the program's operating environment as well as a description of the program. This can be accomplished by providing a general document describing the establishment's computer facilities and service routines in addition to the individual reports prepared for each program. This document is intended as an initial version of such an environment report for the Laboratory.

Copies of internal publications describing ANL-produced utility routines currently used by many of the computer programs and program systems prepared or developed at Argonne are included as appendices.

This installation environment report has been compiled with the assistance and cooperation of many members of the Applied Mathematics Division. It will require correction, review, extension, and modification by many others to ensure its continued usefulness and effectiveness.

II. IBM 360 COMPUTER ENVIRONMENT

In June 1966, Argonne signed a contract with the IBM Corporation for purchase of an IBM 360 System. As of September 1967 all equipment components specified by that contract were on site. This system is described most effectively for our purposes under the separate categories of software systems and hardware description with references to manufacturer-supplied and other relevant documentation included as the third and final category, bibliography.

A. Software Systems

The Laboratory's 360 Models 50 and 75 operate under the IBM Attached Support Processor (ASP) program, which provides a multiprocessor operating system as an extension of the IBM System/360 Operating System (OS/360). In our configuration, the Model 50 acts as a Support Processor providing peripheral functions, while the Model 75 Main Processor operates under OS/360 executing the applications programs. The system input and output devices for the Model 75 are replaced by the channel-to-channel adapter connection.

1. Resident System

a. SYSGEN Deck

```
SYSG     TITLE   STAGE I MACROS FOR MODEL 50 SYSTEM - ARGONNE NATIONAL X
                LABORATORY  RELEASE 13
*
*                DESCRIPTION OF HARDWARE
*
CPU      CENPROCS MODEL=50,STORAGE=I,FEATURE=(TIMER,PROTECT)
CHO      CHANNEL    TYPE=MULTIPLEXOR,ADDRESS=0
URCTL1   IOCONTRL   UNIT=2821,MODEL=5,ADDRESS=00,FEATURE=COLBNRY,       X
                TRNMODE=BYTE
CR1      IODEVICE   UNIT=2540R,ADDRESS=00C,MODEL=1
PU1      IODEVICE   UNIT=2540P,ADDRESS=00D,MODEL=1
PR1      IODEVICE   UNIT=1403,MODEL=N1,ADDRESS=00E,FEATURE=UNVCHSET
PR2      IODEVICE   UNIT=1403,MODEL=N1,ADDRESS=00F,FEATURE=UNVCHSET
PR3      IODEVICE   UNIT=1403,MODEL=N1,ADDRESS=010,FEATURE=UNVCHSET
URCTL2   IOCONTRL   UNIT=2821,MODEL=1,ADDRESS=01,FEATURE=COLBNRY,       X
                TRNMODE=BYTE
CR2      IODEVICE   UNIT=2540R,ADDRESS=01C,MODEL=1
CP2      IODEVICE   UNIT=2540P,ADDRESS=01D,MODEL=1
PR4      IODEVICE   UNIT=1403,MODEL=N1,ADDRESS=01E,FEATURE=UNVCHSET
CN1      IODEVICE   UNIT=1052,MODEL=7,ADDRESS=01F
DSPCU    IOCONTRL   UNIT=2848,ADDRESS=02,MODEL=3
DISPA    IODEVICE   UNIT=2260,MODEL=1,ADDRESS=020,FEATURE=ALKYB2260
```

```
DISPB     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=021,FEATURE=ALKYB2260
DISPC     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=022,FEATURE=ALKYB2260
DISPD     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=023,FEATURE=ALKYB2260
DISPE     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=024,FEATURE=ALKYB2260
DISPF     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=025,FEATURE=ALKYB2260
DISPG     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=026,FEATURE=ALKYB2260
DISPH     IODEVICE   UNIT=2260,MODEL=1,ADDRESS=027,FEATURE=ALKYB2260
DISPPR    IODEVICE   UNIT=1053,ADDRESS=028,MODEL=4
TELECU1   IOCONTRL   UNIT=2701,ADDRESS=03
PDA1      IODEVICE   ADDRESS=030,UNIT=1050,ADAPTER=IBM1,DEVTYPE=51004072
PDA2      IODEVICE   ADDRESS=031,UNIT=1050,ADAPTER=IBM1,DEVTYPE=51004072
TELECU2   IOCONTRL   UNIT=2702,ADDRESS=04
TWX01     IODEVICE   UNIT=TWX,ADDRESS=040,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX02     IODEVICE   UNIT=TWX,ADDRESS=041,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX03     IODEVICE   UNIT=TWX,ADDRESS=042,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX04     IODEVICE   UNIT=TWX,ADDRESS=043,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TYPE01    IODEVICE   UNIT=2740,ADDRESS=044,ADAPTER=IBM1,SETADDR=2,      X
                FEATURE=(CHECKING,SCONTROL)
TYPE02    IODEVICE   UNIT=2740,ADDRESS=045,ADAPTER=IBM1,SETADDR=2,      X
                FEATURE=(CHECKING,SCONTROL)
TWX07     IODEVICE   UNIT=TWX,ADDRESS=046,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX08     IODEVICE   UNIT=TWX,ADDRESS=047,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX09     IODEVICE   UNIT=TWX,ADDRESS=048,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX10     IODEVICE   UNIT=TWX,ADDRESS=049,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX11     IODEVICE   UNIT=TWX,ADDRESS=04A,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX12     IODEVICE   UNIT=TWX,ADDRESS=04B,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX13     IODEVICE   UNIT=TWX,ADDRESS=04C,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX14     IODEVICE   UNIT=TWX,ADDRESS=04D,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX15     IODEVICE   UNIT=TWX,ADDRESS=04E,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX16     IODEVICE   UNIT=TWX,ADDRESS=04F,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TELECU3   IOCONTRL   UNIT=2702,ADDRESS=05
TWX17     IODEVICE   UNIT=TWX,ADDRESS=050,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX18     IODEVICE   UNIT=TWX,ADDRESS=051,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX19     IODEVICE   UNIT=TWX,ADDRESS=052,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX20     IODEVICE   UNIT=TWX,ADDRESS=053,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX21     IODEVICE   UNIT=TWX,ADDRESS=054,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX22     IODEVICE   UNIT=TWX,ADDRESS=055,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX23     IODEVICE   UNIT=TWX,ADDRESS=056,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX24     IODEVICE   UNIT=TWX,ADDRESS=057,ADAPTER=TELE2,SETADDR=1,      X
                FEATURE=(AUTOANSR,AUTOCALL)
```

```
TWX25     IODEVICE   UNIT=TWX,ADDRESS=058,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX26     IODEVICE   UNIT=TWX,ADDRESS=059,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX27     IODEVICE   UNIT=TWX,ADDRESS=05A,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX28     IODEVICE   UNIT=TWX,ADDRESS=05B,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX29     IODEVICE   UNIT=TWX,ADDRESS=05C,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX30     IODEVICE   UNIT=TWX,ADDRESS=05D,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX31     IODEVICE   UNIT=TWX,ADDRESS=05E,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
TWX32     IODEVICE   UNIT=TWX,ADDRESS=05F,ADAPTER=TELE2,SETADDR=1,        X
                FEATURE=(AUTOANSR,AUTOCALL)
CH1       CHANNEL    TYPE=SELECTOR,ADDRESS=1
DA01      IODEVICE   UNIT=2314,ADDRESS=130
CH2       CHANNEL    TYPE=SELECTOR,ADDRESS=2
TAPES     IOCONTRL   UNIT=2403,MODEL=3,ADDRESS=28,                        X
                FEATURE=(7-TRACK,DATACONV)
TAPE1     IODEVICE   UNIT=2403,ADDRESS=280,MODEL=3,FEATURE=7-TRACK
TAPE2     IODEVICE   UNIT=2403,ADDRESS=281,MODEL=3,FEATURE=7-TRACK
TAPE3     IODEVICE   UNIT=2402,ADDRESS=282,MODEL=3,FEATURE=9-TRACK
TAPE4     IODEVICE   UNIT=2402,ADDRESS=283,MODEL=3,FEATURE=9-TRACK
TAPE5     IODEVICE   UNIT=2402,ADDRESS=284,MODEL=3,FEATURE=9-TRACK
TAPE6     IODEVICE   UNIT=2402,ADDRESS=285,MODEL=3,FEATURE=9-TRACK
DCCU      IOCONTRL   UNIT=2841,ADDRESS=29,FEATURE=RECOFLO
DCELLA    IODEVICE   ADDRESS=293,UNIT=2321
DCELLB    IODEVICE   ADDRESS=294,UNIT=2321
          IOCONTRL   UNIT=2840,MODEL=2,ADDRESS=2E
FILM      IODEVICE   UNIT=2280,ADDRESS=2E0,NUMSECT=16
CRT       IODEVICE   UNIT=2250,MODEL=3,NUMSECT=64,FEATURE=ALKYB2250,     X
                ADDRESS=2E1
CH3       CHANNEL    TYPE=SELECTOR,ADDRESS=3
CTC       IODEVICE   UNIT=1052,MODEL=7,ADDRESS=370
*
*               END OF HARDWARE DESCRIPTION
*               GIVE ALTERNATE NAMES TO I/O DEVICES
*
          UNITNAME   UNIT=(130,131,132.133,134,135,136,137),              X
                NAME=SYSDA
          UNITNAME   UNIT=(130,131,132,133,134,135,136,137),              X
                NAME=SYSSQ
          UNITNAME   UNIT=(130,131,132,133,134,135,136,137),              X
                NAME=DISK
          UNITNAME   UNIT=(282,283,284,285),NAME=TAPE
          UNITNAME   UNIT=(282,283,284,285),NAME=TAPE9TRK
          UNITNAME   UNIT=(280,281),NAME=TAPE7TRK
          UNITNAME   UNIT=(00E,00F,010,01E),NAME=PRINTER
          UNITNAME   UNIT=(00C,01C),NAME=READER
          UNITNAME   UNIT=(00D,01D),NAME=PUNCH
          UNITNAME   UNIT=(00D),NAME=SYSCP
          UNITNAME   UNIT=(020,021,022,023,024,025,026,027),NAME=DISPLAY
          UNITNAME   UNIT=(044,045),NAME=TYPER
          UNITNAME   UNIT=(293,294),NAME=CELL
          UNITNAME   UNIT=2E0,NAME=FILM
          UNITNAME   UNIT=2E1,NAME=CRT
          UNITNAME   UNIT=(040,041,042,043),NAME=LINEGRP1
          EJECT
```

```
        DESCRIPTION OF OS/360 CONFIGURATION
  SPACE 3
  CTRLPROG   TYPE=MFT,MAXIO=130,FETCH=PCI,OVERLAY=ADVANCED,      X
        HITASK=258048,LOWTASK=(9000,192000)
  SCHEDULR   TYPE=SEQUENTIAL,DESIGN=100K,CANCEL=NOACCNUM,        X
        CONSOLE=01F,WTOBFRS=10,REPLY=10,                         X
        ACCTRTN=SUPPLIED,OPTIONS=BYLABEL,RESJOBQ=40,             X
        STARTR=A-00C,STARTW=A-00F,                               X
        TSYSIN=556,TSYSOUT=556
  SUPRVSOR   WAIT=MULTIPLE,TIMER=INTERVAL,TRACE=512,SER=SER1,    X
        RESIDNT=(BLDLTAB,ACSMETH,SPIE,ATTACH,EXTRACT,IDENTIFY,   X
        TRSVC),                                                  X
        OPTIONS=(PROTECT,TRSVCTBL,ONLNTEST,COMM)
SVCTABLE SVC-255-T4-S6,SVC-254-T3-S6,SVC-253-T2-S6,SVC-252-T1-S0,X
        SVC-251-T4-S6,SVC-250-T3-S6,SVC-249-T2-S6,SVC-248-T1-S0,X
        SVC-247-T4-S6,SVC-246-T3-S6,SVC-245-T2-S6,SVC-244-T1-S0,X
        SVC-243-T4-S6,SVC-242-T3-S6,SVC-241-T2-S6,SVC-240-T1-S0,X
        SVC-239-T4-S6,SVC-238-T3-S6,SVC-237-T2-S6,SVC-236-T1-S0,X
        SVC-235-T4-S6,SVC-234-T3-S6,SVC-233-T2-S6,SVC-232-T1-S0,X
        SVC-231-T4-S6,SVC-230-T3-S6,SVC-229-T2-S6,SVC-228-T1-S0,X
        SVC-227-T4-S6,SVC-226-T3-S6,SVC-225-T2-S6,SVC-224-T1-S0,X
        SVC-223-T4-S6,SVC-222-T3-S6,SVC-221-T2-S6,SVC-220-T1-S0,X
        SVC-219-T4-S6,SVC-218-T3-S6,SVC-217-T2-S6,SVC-216-T1-S0,X
        SVC-215-T4-S6,SVC-214-T3-S6,SVC-213-T2-S6,SVC-212-T1-S0,X
        SVC-211-T4-S6,SVC-210-T3-S6,SVC-209-T2-S6,SVC-208-T1-S0,X
        SVC-207-T4-S6,SVC-206-T3-S6,SVC-205-T2-S6,SVC-204-T1-S0,X
        SVC-203-T4-S6,SVC-202-T3-S6,SVC-201-T2-S6,SVC-200-T1-S0
  RESMODS    PDS=SYS1.AMDSVCS,                                   X
        MEMBERS=(ANLTYPE1,ANLTYPE2)
  SVCLIB     PDS=SYS1.AMDSVCS,                                   X
        MEMBERS=(IGC0025D,IGC0025E)
  PROCLIB
  LINKLIB    PDS=SYS1.OBJMODS,MEMBERS=(COPYAGO,SUPERSCR)
  TELCMLIB
  GRAPHICS   PORRTNS=INCLUDE,GSP=INCLUDE
  DATAMGT    ACSMETH=(BDAM,ISAM,BTAM)
  SYSUTILS   SIZE=100K
  EDITOR     DESIGN=E18
  EDITOR     DESIGN=E44
  ASSEMBLR   DESIGN=F
  TESTRAN    PHASES=(INTER,EDITOR),MODE=TRACE,                   X
        PAGES=100,EXEC=256
  GRAPHICS   PORRTNS=INCLUDE,GSP=INCLUDE
  DATAMGT    ACSMETH=(BDAM,ISAM,BTAM)
  SYSUTILS   SIZE=100K
  EDITOR     DESIGN=E18
  EDITOR     DESIGN=E44
  ASSEMBLR   DESIGN=F
  TESTRAN    PHASES=(INTER,EDITOR),MODE=TRACE,                   X
        PAGES=100,EXEC=256
  MACLIB
  CHKPOINT
  SORTMERG   SIZE=200000,SORTOPT=FULLIB
  SORTLIB
  ALGOL      SIZE=200000
  ALGLIB
  COBOL      DESIGN=F,SIZE=200000,STORMAP=MAP
  COBLIB     DESIGN=F
  FORTRAN    DESIGN=G,SORCODE=EBCDIC,                            X
        STORMAP=MAP,LINECNT=57
  FORTRAN    DESIGN=H,SIZE=200K,SORCODE=EBCDIC,OPT=0,OBJID=ID,   X
        STORMAP=MAP,LINECNT=57
```

```
         FORTLIB     DESIGN=H,UNTABLE=99,BOUNDRY=ALIGN
         PL1         DESIGN=F,SIZE=200000,REFLIST=XREF,STMDIAG=STMT,     X
               ATRLIST=ATR
         PL1LIB      LIBFCNS=COMPLEX
         RPG
         EJECT
*
*              THE GENERATE MACRO BEGINS THE OUTPUT PROCESSING
         SPACE 2
         GENERATE    RESNAME=DISK,RESVOL=GENSTD,RESTYPE=2314,            X
               OBJPDS=SYS1.OBJMODS,UT1SDS=SYS1.SYSUT2,                   X
               UT2SDS=SYS1.SYSUT1,UT3SDS=SYS1.SYSUT3,                    X
               LBMAINT=E,ASMPRT=ON,LEPRT=(LIST,XREF),DIRDATA=PDS,        X
               GENTYPE=ALL
         END


SYSG     TITLE   STAGE 1 MACROS FOR MODEL 75 SYSTEM - ARGONNE NATIONAL X
               LABORATORY RELEASE 13
*
*              DESCRIPTION OF HARDWARE
*
CPU      CENPROCS    MODEL=75,STORAGE=J
CH1      CHANNEL     ADDRESS=1,TYPE=SELECTOR
DRMCTL   IOCONTRL    UNIT=2820,ADDRESS=1C
DRUM     IODEVICE    UNIT=2301,ADDRESS=1C0
TPCTL1   IOCONTRL    UNIT=2403,ADDRESS=18,MODEL=3,                       X
               FEATURE=(DATACONV,7-TRACK)
TAPE1    IODEVICE    UNIT=2403,ADDRESS=180,MODEL=3,                      X
               FEATURE=7-TRACK,OPTCHAN=2
TAPE2    IODEVICE    UNIT=2403,ADDRESS=181,MODEL=3,                      X
               FEATURE=7-TRACK,OPTCHAN=2
TAPE3    IODEVICE    UNIT=2402,ADDRESS=182,MODEL=3,                      X
               FEATURE=9-TRACK,OPTCHAN=2
TAPE4    IODEVICE    UNIT=2402,ADDRESS=183,MODEL=3,                      X
               FEATURE=9-TRACK,OPTCHAN=2
TAPE5    IODEVICE    UNIT=2402,ADDRESS=184,MODEL=3,                      X
               FEATURE=9-TRACK,OPTCHAN=2
TAPE6    IODEVICE    UNIT=2402,ADDRESS=185,MODEL=3,                      X
               FEATURE=9-TRACK,OPTCHAN=2
CH2      CHANNEL     ADDRESS=2,TYPE=SELECTOR
CN1      IODEVICE    UNIT=1052,ADDRESS=21F,MODEL=7
DISK1    IODEVICE    UNIT=2314,ADDRESS=230
TPCTL2   IOCONTRL    UNIT=2403,ADDRESS=28,MODEL=3,                       X
               FEATURE=(DATACONV,7-TRACK)
CH3      CHANNEL     ADDRESS=3,TYPE=SELECTOR
DISK2    IODEVICE    UNIT=2314,ADDRESS=330
DCCU     IOCONTRL    UNIT=2841,ADDRESS=39,FEATURE=RECOFLO
DCELLA   IODEVICE    UNIT=2321,ADDRESS=393
DCELLB   IODEVICE    UNIT=2321,ADDRESS=394
CTCCN    IODEVICE    UNIT=1052,ADDRESS=370,MODEL=7
CTCCU    IOCONTRL    UNIT=2403,ADDRESS=37,MODEL=3,                       X
               FEATURE=16-DRIVE
CTC1     IODEVICE    UNIT=2402,ADDRESS=371,MODEL=3,FEATURE=9-TRACK
CTC2     IODEVICE    UNIT=2402,ADDRESS=372,MODEL=3,FEATURE=9-TRACK
CTC3     IODEVICE    UNIT=2402,ADDRESS=373,MODEL=3,FEATURE=9-TRACK
CTC4     IODEVICE    UNIT=2402,ADDRESS=374,MODEL=3,FEATURE=9-TRACK
CTC5     IODEVICE    UNIT=2402,ADDRESS=375,MODEL=3,FEATURE=9-TRACK
CTC6     IODEVICE    UNIT=2402,ADDRESS=376,MODEL=3,FEATURE=9-TRACK
```

```
CTC7      IODEVICE   UNIT=2402,ADDRESS=377,MODEL=3,FEATURE=9-TRACK
CTC8      IODEVICE   UNIT=2402,ADDRESS=378,MODEL=3,FEATURE=9-TRACK
CTC9      IODEVICE   UNIT=2402,ADDRESS=379,MODEL=3,FEATURE=9-TRACK
CTC10     IODEVICE   UNIT=2402,ADDRESS=37A,MODEL=3,FEATURE=9-TRACK
CTC11     IODEVICE   UNIT=2402,ADDRESS=37B,MODEL=3,FEATURE=9-TRACK
CTC12     IODEVICE   UNIT=2402,ADDRESS=37C,MODEL=3,FEATURE=9-TRACK
CTC13     IODEVICE   UNIT=2402,ADDRESS=37D,MODEL=3,FEATURE=9-TRACK
CTC14     IODEVICE   UNIT=2402,ADDRESS=37E,MODEL=3,FEATURE=9-TRACK
CTC15     IODEVICE   UNIT=2402,ADDRESS=37F,MODEL=3,FEATURE=9-TRACK
*
*               END OF HARDWARE DESCRIPTION
*               GIVE ALTERNATE NAMES TO I/O DEVICES
*
SYSDA     UNITNAME   UNIT=(230,231,232,233,234,235,236,237,330,331,332, X
               333,334,335,336,337),NAME=SYSDA
SYSSQ     UNITNAME   UNIT=(230,231,232,233,234,235,236,237,330,331,332, X
               333,334,335,336,337),NAME=SYSSQ
DISK      UNITNAME   UNIT=(237,236,235,234,233,232,231,230,337,336,335, X
               334,333,332,331,330),NAME=DISK
TAPE      UNITNAME   UNIT=(182,183,184,185),NAME=TAPE
TAPE9TRK  UNITNAME   UNIT=(182,183,184,185),NAME=TAPE9TRK
TAPE7TRK  UNITNAME   UNIT=(180,181),NAME=TAPE7TRK
SYSCP     UNITNAME   UNIT=373,NAME=SYSCP
DRUM      UNITNAME   UNIT=1C0,NAME=DRUM
CTC       UNITNAME   UNIT=(374,375,376,377,378,379,37A,37B,37C,37D,37E, X
               37F),NAME=CTC
CELL      UNITNAME   UNIT=(393,394),NAME=CELL
          EJECT
*
*               DESCRIPTION OF OS/360 CONFIGURATION
*
          SPACE 3
          CTRLPROG   TYPE=PCP,MAXIO=120,OVERLAY=ADVANCED,FETCH=PCI
          SCHEDULR   TYPE=SEQUENTIAL,CONSOLE=370,ALTCONS=21F,          X
               OPTIONS=BYLABEL,STARTR=(A-371),STARTW=(A-372),          X
               ACCTRTN=SUPPLIED,DESIGN=100K,RESJOBQ=100,               X
               CANCEL=NOACCNUM,TSYSIN=556,TSYSOUT=556
          SUPRVSOR   RESIDNT=(ATTACH,EXTRACT,IDENTIFY,SPIE,BLDLTAB,    X
               ACSMETH,TRSVC),OPTIONS=(TRSVCTBL,PROTECT,ONLNTEST,COMM),X
               WAIT=MULTIPLE,TIMER=INTERVAL,TRACE=512,SER=SER1
      SVCTABLE SVC-255-T4-S6,SVC-254-T3-S6,SVC-253-T2-S6,SVC-252-T1-S0,X
               SVC-251-T4-S6,SVC-250-T3-S6,SVC-249-T2-S6,SVC-248-T1-S0,X
               SVC-247-T4-S6,SVC-246-T3-S6,SVC-245-T2-S6,SVC-244-T1-S0,X
               SVC-243-T4-S6,SVC-242-T3-S6,SVC-241-T2-S6,SVC-240-T1-S0,X
               SVC-239-T4-S6,SVC-238-T3-S6,SVC-237-T2-S6,SVC-236-T1-S0,X
               SVC-235-T4-S6,SVC-234-T3-S6,SVC-233-T2-S6,SVC-232-T1-S0,X
               SVC-231-T4-S6,SVC-230-T3-S6,SVC-229-T2-S6,SVC-228-T1-S0,X
               SVC-227-T4-S6,SVC-226-T3-S6,SVC-225-T2-S6,SVC-224-T1-S0,X
               SVC-223-T4-S6,SVC-222-T3-S6,SVC-221-T2-S6,SVC-220-T1-S0,X
               SVC-219-T4-S6,SVC-218-T3-S6,SVC-217-T2-S6,SVC-216-T1-S0,X
               SVC-215-T4-S6,SVC-214-T3-S6,SVC-213-T2-S6,SVC-212-T1-S0,X
               SVC-211-T4-S6,SVC-210-T3-S6,SVC-209-T2-S6,SVC-208-T1-S0,X
               SVC-207-T4-S6,SVC-206-T3-S6,SVC-205-T2-S6,SVC-204-T1-S0,X
               SVC-203-T4-S6,SVC-202-T3-S6,SVC-201-T2-S6,SVC-200-T1-S0
          RESMODS    PDS=SYS1.AMDSVCS,MEMBERS=(ANLTYPE1,ANLTYPE2)
          SVCLIB     PDS=SYS1.AMDSVCS,MEMBERS=(IGC0025D,IGC0025E)
          PROCLIB
          LINKLIB    PDS=SYS1.OBJPDS,MEMBERS=(SUPERSCR,COPYAGO)
          DATAMGT    ACSMETH=(BDAM,ISAM)
          SYSUTILS   SIZE=800000
          EDITOR     DESIGN=E18
          EDITOR     DESIGN=E44
```

```
ASSEMBLR    DESIGN=F
TESTRAN     PHASES=(INTER,EDITOR),MODE=TRACE,PAGES=100,EXEC=256
MACLIB      UNIT=2314,VOLNO=GENLIB
CHKPOINT
SORTMERG    SIZE=800000,SORTOPT=FULLIB
SORTLIB     UNIT=2314,VOLNO=GENLIB
ALGOL       SIZE=800000
ALGLIB      UNIT=2314,VOLNO=GENLIB
COBOL       DESIGN=E,DATAMAP=DMAP,BUFSIZE=3600
COBOL       DESIGN=F,SIZE=800000,STORMAP=MAP
COBLIB      DESIGN=E,UNIT=2314,VOLNO=GENLIB
COBLIB      DESIGN=F,UNIT=2314,VOLNO=GENLIB
FORTRAN     DESIGN=G,STORMAP=MAP,LINECNT=57
FORTRAN     DESIGN=H,STORMAP=MAP,LINECNT=57,OBJID=ID
FORTLIB     DESIGN=H,UNIT=2314,VOLNO=GENLIB,UNTABLE=99
PL1         DESIGN=F,SIZE=800000,REFLIST=XREF,STMDIAG=STMT,     X
      ATRLIST=ATR
PL1LIB      UNIT=2314,VOLNO=GENLIB,LIBFCNS=COMPLEX
RPG
GENERATE    GENTYPE=ALL,UT1SDS=SYS1.SYSUT1,UT2SDS=SYS1.SYSUT2, X
      UT3SDS=SYS1.SYSUT3,OBJPDS=SYS1.OBJPDS,RESNAME=DISK,      X
      RESVOL=GENSYS,RESTYPE=2314,LNKNAME=DISK,LNKVOL=GENSYS,   X
      ASMPRT=ON,LEPRT=(LIST,XREF),DIRDATA=PDS
END
```

b. OS/360 Model 75 Memory required = 100,384 bytes

2. Catalogued Procedures

a. Catalogue procedures for the software systems provided at this installation are prefixed as follows:

1)	FORTRAN H	FTH
2)	FORTRAN G	FTG
3)	Assembler language	ASM
4)	Programming Language/I	PLI
5)	ALGOL	ALG
6)	Report Program Generator	RPG
7)	COBOL	COB

b. Suffixes used with each of the above include:

1)	Compile	C
2)	Deck	D
3)	Private library	P
4)	Link edit	L
5)	Execute	G

c. A detailed listing follows:

```
MEMBER NAME  FTHCD
//FTH      EXEC PGM=FORTRANH,PARM= NOLOAD,DECK                          0000001
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                      0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000004

MEMBER NAME  FTHCP
//FTH      EXEC PGM=FORTRANH                                            0000001
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000002
//SYSLIN     DD DDNAME=SYSOBJ                                           0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000004

MEMBER NAME  FTHCLG
//FTH      EXEC PGM=FORTRANH                                            0000001
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000003
//              DISP=(OLD,PASS)                                         0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000005
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,FTH),PARM= LIST,MAP             0000006
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000007
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000008
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                            0000009
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000010
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                            0000011
//           DD DDNAME=SYSIN                                            0000012
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTH),(5,LT,EDT))          0000013
//FT05F001   DD DDNAME=SYSIN                                            0000014
//FT06F001   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)          0000015
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                      0000016

MEMBER NAME  FTHCLP
//FTH      EXEC PGM=FORTRANH                                            0000001
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000003
//              DISP=(OLD,PASS)                                         0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000005
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,FTH),PARM= LIST,MAP             0000006
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000007
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000008
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                            0000009
//SYSLMOD    DD DDNAME=SYSPVT                                           0000010
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                            0000011
//           DD DDNAME=SYSIN                                            0000012

MEMBER NAME  FTHLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                             0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                            0000004
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN     DD DDNAME=SYSIN                                            0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000007
//FT05F001   DD DDNAME=SYSIN                                            0000008
//FT06F001   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)          0000009
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                      0000010

MEMBER NAME  FTHLP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                             0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                            0000004
//SYSLMOD    DD DDNAME=SYSPVT                                           0000005
//SYSLIN     DD DDNAME=SYSIN                                            0000006
```

```
MEMBER NAME  FTGCD
//FTG      EXEC PGM=FORTRANG,PARM= NOLOAD,DECK
//SYSPRINT  DD SYSOUT=A
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)
```

```
MEMBER NAME  FTGCP
//FTG      EXEC PGM=FORTRANG
//SYSPRINT  DD SYSOUT=A
//SYSLIN    DD DDNAME=SYSOBJ
```

```
MEMBER NAME  FTGCLG
//FTG      EXEC PGM=FORTRANG
//SYSPRINT  DD SYSOUT=A
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),
//            DISP=(OLD,PASS)
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,FTG),PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD
//          DD DDNAME=SYSIN
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTG),(5,LT,EDT))
//FT05F001  DD DDNAME=SYSIN
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)
//FT07F001 DD  UNIT=(SYSCP,,DEFER)
```

```
MEMBER NAME  FTGCLP
//FTG      EXEC PGM=FORTRANG
//SYSPRINT  DD SYSOUT=A
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),
//            DISP=(OLD,PASS)
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,FTG),PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD   DD DDNAME=SYSPVT
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD
//          DD DDNAME=SYSIN
```

```
MEMBER NAME  FTGLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)
//SYSLIN    DD DDNAME=SYSIN
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)
//FT05F001  DD DDNAME=SYSIN
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)
//FT07F001 DD  UNIT=(SYSCP,,DEFER)
```

```
MEMBER NAME  FTGLP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD
//SYSLMOD   DD DDNAME=SYSPVT
//SYSLIN    DD DDNAME=SYSIN
```

```
MEMBER NAME  ASMCD
//ASM      EXEC PGM=ASMBLR,PARM= NOLOAD,DECK                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)              0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                         0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000006
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                                0000007

MEMBER NAME  ASMCP
//ASM      EXEC PGM=ASMBLR,PARM= NODECK,LOAD                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)              0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000005
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                                0000006
//SYSGO      DD DDNAME=SYSOBJ                                              0000007

MEMBER NAME  ASMCLG
//ASM      EXEC PGM=ASMBLR,PARM= NODECK,LOAD                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)              0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000005
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                                0000006
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),     C0000007
//             DISP=(OLD,PASS)                                             0000008
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,ASM),PARM= LIST,MAP                 0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000010
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000011
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)  0000012
//SYSLIN     DD DSNAME=*.ASM.SYSGO,DISP=OLD                                0000013
//           DD DDNAME=SYSIN                                               0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ASM),(5,LT,EDT))            0000015
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000016
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                         0000017

MEMBER NAME  ASMCLP
//ASM      EXEC PGM=ASMBLR,PARM= NODECK,LOAD                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)              0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000005
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                                0000006
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),     C0000007
//             DISP=(OLD,PASS)                                             0000008
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,ASM),PARM= LIST,MAP                 0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000010
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000011
//SYSLMOD    DD DDNAME=SYSPVT                                              0000012
//SYSLIN     DD DSNAME=*.ASM.SYSGO,DISP=OLD                                0000013
//           DD DDNAME=SYSIN                                               0000014

MEMBER NAME  ASMLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                                 0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000003
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)  0000004
//SYSLIN     DD DDNAME=SYSIN                                               0000005
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                          0000006
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000007
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                         0000008
```

```
MEMBER NAME  ASMLP
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLMOD   DD DDNAME=SYSPVT
//SYSLIN    DD DDNAME=SYSIN
```

```
MEMBER NAME  PL1CD
//PL1     EXEC PGM=PL1,PARM= NOLOAD,DECK
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY
```

```
MEMBER NAME  PL1CP
//PL1     EXEC PGM=PL1
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY
//SYSLIN    DD DDNAME=SYSOBJ
```

```
MEMBER NAME  PL1CLG
//PL1     EXEC PGM=PL1
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY
//SYSLIN   DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C
//             DISP=(OLD,PASS)
//EDT     EXEC PGM=LINKEDIT,COND=(5,LT,PL1),PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.PL1LIB,DISP=OLD
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)
//SYSLIN    DD DSNAME=*.PL1.SYSLIN,DISP=OLD
//           DD DDNAME=SYSIN
//GO      EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,PL1),(9,LT,EDT))
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)
//SYSPNCH   DD UNIT=(SYSCP,,DEFER)
```

```
MEMBER NAME  PL1CLP
//PL1     EXEC PGM=PL1
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY
//SYSLIN   DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C
//             DISP=(OLD,PASS)
//EDT     EXEC PGM=LINKEDIT,COND=(5,LT,PL1),PARM= LIST,MAP
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD
//SYSLIB    DD DSNAME=SYS1.PL1LIB,DISP=OLD
//SYSLMOD   DD DDNAME=SYSPVT
//SYSLIN    DD DSNAME=*.PL1.SYSLIN,DISP=OLD
//           DD DDNAME=SYSIN
```

```
MEMBER NAME  PL1LG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                                0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                               0000004
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN     DD DDNAME=SYSIN                                              0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(9,LT,EDT)                         0000007
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000008
//SYSPNCH    DD UNIT=(SYSCP,,DEFER)                                       0000009

MEMBER NAME  PL1LP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                                0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                               0000004
//SYSLMOD    DD DDNAME=SYSPVT                                             0000005
//SYSLIN     DD DDNAME=SYSIN                                              0000006

MEMBER NAME  ALGCD
//ALG      EXEC PGM=ALGOL,PARM= NOLOAD,DECK                                0000001
//SYSPRINT   DD SYSOUT=A                                                   0000002
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                        0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000006

MEMBER NAME  ALGCP
//ALG      EXEC PGM=ALGOL                                                  0000001
//SYSPRINT   DD SYSOUT=A                                                   0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000005
//SYSLIN     DD DDNAME=SYSOBJ                                             0000006

MEMBER NAME  ALGCLG
//ALG      EXEC PGM=ALGOL                                                  0000001
//SYSPRINT   DD SYSOUT=A                                                   0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000005
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000006
//              DISP=(OLD,PASS)                                            0000007
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,ALG),PARM= LIST,MAP                0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000010
//SYSLIB     DD DSNAME=SYS1.ALGLIB,DISP=OLD                               0000011
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000012
//SYSLIN     DD DSNAME=*.ALG.SYSLIN,DISP=OLD                              0000013
//            DD DDNAME=SYSIN                                             0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ALG),(5,LT,EDT))            0000015
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000016
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000017
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000018

MEMBER NAME  ALGCLP
//ALG      EXEC PGM=ALGOL                                                  0000001
//SYSPRINT   DD SYSOUT=A                                                   0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000005
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000006
```

```
//              DISP=(OLD,PASS)                                          0000007
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,ALG),PARM= LIST,MAP              0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000009
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000010
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                              0000011
//SYSLMOD   DD DDNAME=SYSPVT                                            0000012
//SYSLIN    DD DSNAME=*.ALG.SYSLIN,DISP=OLD                             0000013
//          DD DDNAME=SYSIN                                             0000014
```

```
MEMBER NAME  ALGLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                             0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                              0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN    DD DDNAME=SYSIN                                             0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000007
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000008
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000009
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000010
```

```
MEMBER NAME  ALGLP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                             0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                              0000004
//SYSLMOD   DD DDNAME=SYSPVT                                            0000005
//SYSLIN    DD DDNAME=SYSIN                                             0000006
```

```
MEMBER NAME  RPGCD
//RPG      EXEC PGM=RPG,PARM= NOLOAD,DECK                               0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
```

```
MEMBER NAME  RPGCP
//RPG      EXEC PGM=RPG                                                 0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSGO     DD DDNAME=SYSOBJ                                            0000006
```

```
MEMBER NAME  RPGCLG
//RPG      EXEC PGM=RPG                                                 0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000006
//              DISP=(OLD,PASS)                                         0000007
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,RPG),PARM= LIST,MAP             0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000009
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000010
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000011
//SYSLIN    DD DSNAME=*.RPG.SYSGO,DISP=OLD                              0000012
//          DD DDNAME=SYSIN                                             0000013
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,RPG),(5,LT,EDT))          0000014
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000015
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      0000016
```

```
MEMBER NAME  RPGCLP
//RPG      EXEC PGM=RPG                                                  0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSGO   DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000006
//              DISP=(OLD,PASS)                                          0000007
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,RPG),PARM= LIST,MAP             0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000010
//SYSLMOD    DD DDNAME=SYSPVT                                            0000011
//SYSLIN     DD DSNAME=*.RPG.SYSGO,DISP=OLD                              0000012
//           DD DDNAME=SYSIN                                             0000013

MEMBER NAME  RPGLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                              0000001
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000004
//SYSLIN     DD DDNAME=SYSIN                                             0000005
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                        0000006
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000007
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                      0000008

MEMBER NAME  RPGLP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                              0000001
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSLMOD    DD DDNAME=SYSPVT                                            0000004
//SYSLIN     DD DDNAME=SYSIN                                             0000005

MEMBER NAME  COBCD
//COB      EXEC PGM=COBOL,PARM= NOLOAD,DECK                              0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                       0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                            0000007
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000008

MEMBER NAME  COBCP
//COB      EXEC PGM=COBOL                                                0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                            0000006
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000007
//SYSLIN     DD DDNAME=SYSOBJ                                            0000008

MEMBER NAME  COBCLG
//COB      EXEC PGM=COBOL                                                0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                            0000006
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000007
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000008
//              DISP=(OLD,PASS)                                          0000009
```

```
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,COB),PARM= LIST,MAP                0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000011
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000012
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                                 0000013
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000014
//SYSLIN    DD DSNAME=*.COB.SYSLIN,DISP=OLD                                0000015
//          DD DDNAME=SYSIN                                                0000016
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,COB),(5,LT,EDT))             0000017
//SYSOUT    DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000018
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                         0000019
```

```
MEMBER NAME  COBCLP
//COB      EXEC PGM=COBOL                                                  0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000005
//SYSUT4    DD UNIT=DISK,SPACE=(TRK,(20,20))                               0000006
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY              0000007
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),      C0000008
//             DISP=(OLD,PASS)                                             0000009
//EDT      EXEC PGM=LINKEDIT,COND=(5,LT,COB),PARM= LIST,MAP                0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000011
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000012
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                                 0000013
//SYSLMOD   DD DDNAME=SYSPVT                                               0000014
//SYSLIN    DD DSNAME=*.COB.SYSLIN,DISP=OLD                                0000015
//          DD DDNAME=SYSIN                                                0000016
```

```
MEMBER NAME  COBLG
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                                 0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000005
//SYSLIN    DD DDNAME=SYSIN                                                0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                          0000007
//SYSOUT    DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000008
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                         0000009
```

```
MEMBER NAME  COBLP
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,MAP                                0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                                 0000004
//SYSLMOD   DD DDNAME=SYSPVT                                               0000005
//SYSLIN    DD DDNAME=SYSIN                                                0000006
```

```
MEMBER NAME  FTHCDE
//FTH      EXEC PGM=FORTRANH,PARM= LIST,NOLOAD,DECK                        0000001
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000002
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                        0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                *0000005
//             * PROCEDURE COMPILES FTH.SYSIN 80-BYTE SOURCE IMAGES       *0000006
//             * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.     *0000007
//             *                                                 PRO FORMA0000008
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000009
//             * //XXXXXXXX EXEC FTHCDE                                    *0000010
//             * //FTH.SYSIN DD *                                          *0000011
//             * ..................... SOURCE DECK ......................*0000012
//             * /*                                                       0000013
```

```
MEMBER NAME  FTHCPE
//FTH      EXEC PGM=FORTRANH,PARM= LIST,NODECK,LOAD                          0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000003
//SYSLIN     DD DDNAME=SYSOBJ                                            0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             *0000005
//              * PROCEDURE COMPILES FTH.SYSIN 80-BYTE SOURCE IMAGES    *0000006
//              * INTO FTH.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000007
//              *                                                 PRO FORMA00000008
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000009
//              * //XXXXXXXX EXEC FTHCPE                                 *0000010
//              * //FTH.SYSIN DD *                                       *0000011
//              * ..................... SOURCE DECK ....................*0000012
//              * /*                                                     0000013

MEMBER NAME  FTHCLGE
//FTH      EXEC PGM=FORTRANH,PARM= LIST,NODECK,LOAD                          0000001
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000003
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000004
//             DISP=(OLD,PASS)                                            0000005
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000006
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,FTH),PARM= LIST,XREF             0000007
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000010
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000011
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000012
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                             0000013
//           DD DDNAME=SYSIN                                             0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTH),(5,LT,EDT))           0000015
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000016
//FT05F001  DD DDNAME=SYSIN                                              0000017
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000018
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                      X0000019
//              * PROCEDURE COMPILES FTH.SYSIN 80-BYTE SOURCE IMAGES    *0000020
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.           *0000021
//              *                                                 PRO FORMA00000022
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000023
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE            *0000024
//              * //XXXXXXXX EXEC FTHCLGE                                *0000025
//              * //FTH.SYSIN DD *                                       *0000026
//              * ..................... SOURCE DECK ....................*0000027
//              * /*                                                     *0000028
//              * //EDT.SYSIN DD *                                       *0000029
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000030
//              * /*                                                     0000031

MEMBER NAME  FTHCLPE
//FTH      EXEC PGM=FORTRANH,PARM= LIST,NODECK,LOAD                          0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER)                                         0000002
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000003
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000004
//             DISP=(OLD,PASS)                                            0000005
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000006
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,FTH),PARM= LIST,XREF             0000007
//SYSUDUMP DD  UNIT=(CTC,,DEFER)                                         0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000010
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000011
//SYSLMOD    DD DDNAME=SYSPVT                                            0000012
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                             0000013
//           DD DDNAME=SYSIN                                            *0000014
//              * PROCEDURE COMPILES FTH.SYSIN 80-BYTE SOURCE IMAGES    *0000015
```

```
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.       *0000016
                                                          PRO FORMA00000017
//              *                                                        *0000018
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000019
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000020
//              * //XXXXXXXX EXEC FTHCLPE                                *0000021
//              * //FTH.SYSIN DD *                                       *0000022
//              * ..................... SOURCE DECK ....................*0000023
//              * /*                                                     *0000024
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000025
//              * //EDT.SYSIN DD *                                       *0000026
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........ 0000027
//              * /*

MEMBER NAME  FTHLGE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                              0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER)                                          0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                              0000005
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000006
//SYSLIN    DD DDNAME=SYSIN                                              0000007
//GO      EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                         0000008
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000009
//FT05F001  DD DDNAME=SYSIN                                              0000010
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000011
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                      X0000012
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000013
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.            *0000014
//              *                                                 PRO FORMA00000015
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000016
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE             *0000017
//              * //XXXXXXXX EXEC FTHLGE                                 *0000018
//              * //EDT.SYSIN DD *                                       *0000019
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........ *0000020
//              * /*                                                     0000021

MEMBER NAME  FTHLPE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                              0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                              0000005
//SYSLMOD   DD DDNAME=SYSPVT                                             *0000006
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000007
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.       *0000008
//              *                                                 PRO FORMA00000009
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000010
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000011
//              * //XXXXXXXX EXEC FTHLPE                                 *0000012
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000013
//              * //EDT.SYSIN DD *                                       *0000014
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........ *0000015
//              * /*                                                     0000016
//SYSLIN    DD DDNAME=SYSIN                                              0000017

MEMBER NAME  FTGCDE
//FTG     EXEC PGM=FORTRANG,PARM= LIST,NOLOAD,DECK                       0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A                                                  0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                      X0000004
//              * PROCEDURE COMPILES FTG.SYSIN 80-BYTE SOURCE IMAGES    *0000005
//              * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.  *0000006
//              *                                                 PRO FORMA00000007
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000008
```

```
//              * //XXXXXXXX EXEC FTGCDE                                    *0000009
//              * //FTG.SYSIN DD *                                          *0000010
//              * ..................... SOURCE DECK ......................*0000011
//              * /*                                                        0000012
```

```
MEMBER NAME  FTGCPE
//FTG      EXEC PGM=FORTRANG,PARM= LIST,NODECK,LOAD                         0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A                                                    0000003
//SYSLIN    DD DDNAME=SYSOBJ                                               *0000004
//              * PROCEDURE COMPILES FTG.SYSIN 80-BYTE SOURCE IMAGES       *0000005
//              * INTO FTG.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000006
//              *                                              PRO FORMA00000007
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000008
//              * //XXXXXXXX EXEC FTGCPE                                   *0000009
//              * //FTG.SYSIN DD *                                          *0000010
//              * ..................... SOURCE DECK ......................*0000011
//              * /*                                                        0000012
```

```
MEMBER NAME  FTGCLGE
//FTG      EXEC PGM=FORTRANG,PARM= LIST,NODECK,LOAD                         0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A                                                    0000003
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),       C0000004
//              DISP=(OLD,PASS)                                            0000005
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,FTG),PARM= LIST,XREF               0000006
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000007
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000008
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000009
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                                0000010
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000011
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD                                0000012
//          DD DDNAME=SYSIN                                                0000013
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTG),(5,LT,EDT))             0000014
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000015
//FT05F001  DD DDNAME=SYSIN                                                0000016
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000017
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                         X0000018
//              * PROCEDURE COMPILES FTG.SYSIN 80-BYTE SOURCE IMAGES       *0000019
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.              *0000020
//              *                                              PRO FORMA00000021
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000022
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE               *0000023
//              * //XXXXXXXX EXEC FTGCLGE                                   *0000024
//              * //FTG.SYSIN DD *                                          *0000025
//              * ..................... SOURCE DECK ......................*0000026
//              * /*                                                        *0000027
//              * //EDT.SYSIN DD *                                          *0000028
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000029
//              * /*                                                        0000030
```

```
MEMBER NAME  FTGCLPE
//FTG      EXEC PGM=FORTRANG,PARM= LIST,NODECK,LOAD                         0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A                                                    0000003
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),       C0000004
//              DISP=(OLD,PASS)                                            0000005
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,FTG),PARM= LIST,XREF               0000006
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000007
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000008
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000009
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                                0000010
//SYSLMOD   DD DDNAME=SYSPVT                                               0000011
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD                                0000012
```

```
//          DD DDNAME=SYSIN                                                 *0000013
//             * PROCEDURE COMPILES FTG.SYSIN 80-BYTE SOURCE IMAGES         *0000014
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.           *0000015
//             *                                                  PRO FORMA00000016
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                   *0000017
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT                *0000018
//             * //XXXXXXXX EXEC FTGCLPE                                     *0000019
//             * //FTG.SYSIN DD *                                            *0000020
//             * ..................... SOURCE DECK ......................*0000021
//             * /*                                                         *0000022
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000023
//             * //EDT.SYSIN DD *                                           *0000024
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000025
//             * /*                                                          0000026


MEMBER NAME  FTGLGE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                                 0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000004
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                                 0000005
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)     0000006
//SYSLIN    DD DDNAME=SYSIN                                                 0000007
//GO      EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                            0000008
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000009
//FT05F001  DD DDNAME=SYSIN                                                 0000010
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)               0000011
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                         X0000012
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000013
//             * FOR LINKING INTO LOAD MODULE FOR EXECUTION.               *0000014
//             *                                                  PRO FORMA00000015
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                   *0000016
//             * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE                 *0000017
//             * //XXXXXXXX EXEC FTGLGE                                      *0000018
//             * //EDT.SYSIN DD *                                            *0000019
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000020
//             * /*                                                          0000021


MEMBER NAME  FTGLPE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                                 0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000004
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                                 0000005
//SYSLMOD   DD DDNAME=SYSPVT                                               *0000006
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000007
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.          *0000008
//             *                                                  PRO FORMA00000009
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                   *0000010
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT                *0000011
//             * //XXXXXXXX EXEC FTGLPE                                      *0000012
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000013
//             * //EDT.SYSIN DD *                                           *0000014
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000015
//             * /*                                                          0000016
//SYSLIN    DD DDNAME=SYSIN                                                 0000017


MEMBER NAME  ASMCDE
//ASM     EXEC PGM=ASMBLR,PARM= LIST,XREF,NOLOAD,DECK                       0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)               0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                          0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000005
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                  0000006
```

```
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000007
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                               X0000008
//              * PROCEDURE COMPILES ASM.SYSIN 80-BYTE SOURCE IMAGES     *0000009
//              * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.   *0000010
//              *                                                 PRO FORMA00000011
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000012
//              * //XXXXXXXX EXEC ASMCDE                                  *0000013
//              * //ASM.SYSIN DD *                                        *0000014
//              * ..................... SOURCE DECK ....................*0000015
//              * /*                                                      0000016

MEMBER NAME  ASMCPE
//ASM      EXEC PGM=ASMBLR,PARM= LIST,XREF,NODECK,LOAD                    0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)   0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)            0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000007
//SYSGO      DD DDNAME=SYSOBJ                                           *0000008
//              * PROCEDURE COMPILES ASM.SYSIN 80-BYTE SOURCE IMAGES     *0000009
//              * INTO ASM.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000010
//              *                                                 PRO FORMA00000011
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000012
//              * //XXXXXXXX EXEC ASMCPE                                  *0000013
//              * //ASM.SYSIN DD *                                        *0000014
//              * ..................... SOURCE DECK ....................*0000015
//              * /*                                                      0000016

MEMBER NAME  ASMCLGE
//ASM      EXEC PGM=ASMBLR,PARM= LIST,XREF,NODECK,LOAD                    0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)   0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)            0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIB     DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000007
//SYSGO   DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000008
//              DISP=(OLD,PASS)                                          0000009
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,ASM),PARM= LIST,XREF             0000010
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)   0000011
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000012
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000013
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000014
//SYSLIN     DD DSNAME=*.ASM.SYSGO,DISP=OLD                              0000015
//              DD DDNAME=SYSIN                                          0000016
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ASM),(5,LT,EDT))           0000017
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)   0000018
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000019
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      *0000020
//              * PROCEDURE COMPILES ASM.SYSIN 80-BYTE SOURCE IMAGES     *0000021
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.            *0000022
//              *                                                 PRO FORMA00000023
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000024
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE             *0000025
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000026
//              * //XXXXXXXX EXEC ASMCLGE                                 *0000027
//              * //ASM.SYSIN DD *                                        *0000028
//              * ..................... SOURCE DECK ....................*0000029
//              * /*                                                      *0000030
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000031
//              * //EDT.SYSIN DD *                                       *0000032
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000033
//              * /*                                                      0000034
```

```
MEMBER NAME  ASMCLPE
//ASM      EXEC PGM=ASMBLR,PARM= LIST,XREF,NODECK,LOAD                    0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIB    DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000007
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000008
//             DISP=(OLD,PASS)                                          0000009
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,ASM),PARM= LIST,XREF            0000010
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000011
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000012
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000013
//SYSLMOD   DD DDNAME=SYSPVT                                            0000014
//SYSLIN    DD DSNAME=*.ASM.SYSGO,DISP=OLD                              0000015
//           DD DDNAME=SYSIN                                           *0000016
//             * PROCEDURE COMPILES ASM.SYSIN 80-BYTE SOURCE IMAGES    *0000017
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.      *0000018
//             *                                          PRO FORMA0000019
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000020
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT           *0000021
//             * //XXXXXXXX EXEC ASMCLPE                                *0000022
//             * //ASM.SYSIN DD *                                       *0000023
//             * ..................... SOURCE DECK ......................*0000024
//             * /*                                                    *0000025
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000026
//             * //EDT.SYSIN DD *                                       *0000027
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000028
//             * /*                                                     0000029
```

```
MEMBER NAME  ASMLGE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                            0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN    DD DDNAME=SYSIN                                             0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000007
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000009
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                     *0000010
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000011
//             * FOR LINKING INTO LOAD MODULE FOR EXECUTION.           *0000012
//             *                                          PRO FORMA0000013
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000014
//             * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE            *0000015
//             * //XXXXXXXX EXEC ASMLGE                                 *0000016
//             * //EDT.SYSIN DD *                                       *0000017
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000018
//             * /*                                                     0000019
```

```
MEMBER NAME  ASMLPE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                            0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLMOD   DD DDNAME=SYSPVT                                            0000005
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000006
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.      *0000007
//             *                                          PRO FORMA0000008
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000009
//             * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE            *0000010
//             * //XXXXXXXX EXEC ASMLPE                                 *0000011
```

```
//              * //EDT.SYSIN DD *                                          *0000012
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000013
//              * /*                                                        0000014
//SYSLIN     DD DDNAME=SYSIN                                                0000015
```

```
MEMBER NAME  PL1CDE
//PL1       EXEC PGM=PL1,PARM= E,L,D,NLD                                   0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)              0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                         0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000006
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000007
//              * PROCEDURE COMPILES PL1.SYSIN 80-BYTE SOURCE IMAGES     *0000008
//              * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.   *0000009
//              *                                                 PRO FORMA00000010
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000011
//              * //XXXXXXXX EXEC PL1CDE                                  *0000012
//              * //PL1.SYSIN DD *                                        *0000013
//              * ..................... SOURCE DECK ......................*0000014
//              * /*                                                        0000015
```

```
MEMBER NAME  PL1CPE
//PL1       EXEC PGM=PL1,PARM= E,L,ND,LD                                   0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)              0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000005
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000006
//SYSLIN     DD DDNAME=SYSOBJ                                             *0000007
//              * PROCEDURE COMPILES PL1.SYSIN 80-BYTE SOURCE IMAGES     *0000008
//              * INTO PL1.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000009
//              *                                                 PRO FORMA00000010
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000011
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT            *0000012
//              * //XXXXXXXX EXEC PL1CPE                                  *0000013
//              * //PL1.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000014
//              * //PL1.SYSIN DD *                                        *0000015
//              * ..................... SOURCE DECK ......................*0000016
//              * /*                                                        0000017
```

```
MEMBER NAME  PL1CLGE
//PL1       EXEC PGM=PL1,PARM= E,L,ND,LD                                   0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)              0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000005
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000006
//SYSLIN    DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C0000007
//              DISP=(OLD,PASS)                                            0000008
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,PL1),PARM= LIST,XREF              0000009
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000011
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000012
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                                0000013
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)   0000014
//SYSLIN     DD DSNAME=*.PL1.SYSLIN,DISP=OLD                               0000015
//           DD DDNAME=SYSIN                                               0000016
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,PL1),(9,LT,EDT))            0000017
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000018
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000019
//SYSPNCH    DD UNIT=(SYSCP,,DEFER)                                       *0000020
//              * PROCEDURE COMPILES PL1.SYSIN 80-BYTE SOURCE IMAGES     *0000021
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.            *0000022
//              *                                                 PRO FORMA00000023
```

```
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                    *0000024
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE                  *0000025
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT                 *0000026
//              * //XXXXXXXX EXEC PL1CLGE                                      *0000027
//              * //PL1.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000028
//              * //PL1.SYSIN DD *                                             *0000029
//              * ..................... SOURCE DECK ......................*0000030
//              * /*                                                          *0000031
//              * //EDT.SYSIN DD *                                             *0000032
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000033
//              * /*                                                           0000034


MEMBER NAME  PL1CLPE
//PL1       EXEC PGM=PL1,PARM= E,L,ND,LD                                      0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)           0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000005
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY             0000006
//SYSLIN    DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C0000007
//              DISP=(OLD,PASS)                                                0000008
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,PL1),PARM= LIST,XREF                  0000009
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000011
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000012
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                                 0000013
//SYSLMOD    DD DDNAME=SYSPVT                                               0000014
//SYSLIN     DD DSNAME=*.PL1.SYSLIN,DISP=OLD                                0000015
//           DD DDNAME=SYSIN                                               *0000016
//              * PROCEDURE COMPILES PL1.SYSIN 80-BYTE SOURCE IMAGES       *0000017
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.         *0000018
//              *                                          PRO FORMA00000019
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000020
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT              *0000021
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT              *0000022
//              * //XXXXXXXX EXEC PL1CLGE                                   *0000023
//              * //PL1.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000024
//              * //PL1.SYSIN DD *                                          *0000025
//              * ..................... SOURCE DECK ......................*0000026
//              * /*                                                        *0000027
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000028
//              * //EDT.SYSIN DD *                                          *0000029
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000030
//              * /*                                                         0000031


MEMBER NAME  PL1LGE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                                 0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                                 0000005
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000006
//SYSLIN     DD DDNAME=SYSIN                                                0000007
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(9,LT,EDT)                            0000008
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000010
//SYSPNCH   DD UNIT=(SYSCP,,DEFER)                                        *0000011
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000012
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.             *0000013
//              *                                          PRO FORMA00000014
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000015
//              * //XXXXXXXX EXEC PL1LGE                                    *0000016
//              * //EDT.SYSIN DD *                                          *0000017
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000018
//              * /*                                                         0000019
```

```
MEMBER NAME  PL1LPE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                                0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLIB    DD DSNAME=SYS1.PL1LIB,DISP=OLD                              0000005
//SYSLMOD   DD DDNAME=SYSPVT                                           *0000006
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000007
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.      *0000008
//             *                                                 PRO FORMA00000009
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000010
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT           *0000011
//             * //XXXXXXXX EXEC PL1LPE                                 *0000012
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000013
//             * //EDT.SYSIN DD *                                      *0000014
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000015
//             * /*                                                    0000016
//SYSLIN    DD DDNAME=SYSIN                                             0000017

MEMBER NAME  ALGCDE
//ALG      EXEC PGM=ALGOL,PARM= S,T,NOLOAD,DECK                         0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A                                                 0000003
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000005
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000006
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000007
//             * PROCEDURE COMPILES ALG.SYSIN 80-BYTE SOURCE IMAGES    *0000008
//             * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.  *0000009
//             *                                                 PRO FORMA00000010
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000011
//             * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD               *0000012
//             * //XXXXXXXX EXEC ALGCDE                                 *0000013
//             * //ALG.SYSIN DD *                                      *0000014
//             * ..................... SOURCE DECK ......................*0000015
//             * /*                                                    0000016

MEMBER NAME  ALGCPE
//ALG      EXEC PGM=ALGOL,PARM= S,T,NODECK,LOAD                         0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A                                                 0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIN    DD DDNAME=SYSOBJ                                           *0000007
//             * PROCEDURE COMPILES ALG.SYSIN 80-BYTE SOURCE IMAGES    *0000008
//             * INTO ALG.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000009
//             *                                                 PRO FORMA00000010
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000011
//             * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD               *0000012
//             * //XXXXXXXX EXEC ALGCPE                                 *0000013
//             * //ALG.SYSIN DD *                                      *0000014
//             * ..................... SOURCE DECK ......................*0000015
//             * /*                                                    0000016

MEMBER NAME  ALGCLGE
//ALG      EXEC PGM=ALGOL,PARM= S,T,NODECK,LOAD                         0000001
//SYSPRINT  DD SYSOUT=A                                                 0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000006
//             DISP=(OLD,PASS)                                          0000007
```

```
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,ALG),PARM= LIST,XREF                0000008
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)      0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000010
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000011
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000012
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000013
//SYSLIN    DD DSNAME=*.ALG.SYSLIN,DISP=OLD                                0000014
//          DD DDNAME=SYSIN                                                0000015
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ALG),(5,LT,EDT))             0000016
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000017
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000018
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000019
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                *0000020
//             * PROCEDURE COMPILES ALG.SYSIN 80-BYTE SOURCE IMAGES       *0000021
//             * FOR LINKING INTO LOAD MODULE FOR EXECUTION.              *0000022
//             *                                                 PRO FORMA00000023
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000024
//             * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                  *0000025
//             * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE               *0000026
//             * //XXXXXXXX EXEC ALGCLGE                                   *0000027
//             * //ALG.SYSIN DD *                                          *0000028
//             * ..................... SOURCE DECK ......................*0000029
//             * /*                                                       *0000030
//             * //EDT.SYSIN DD *                                          *0000031
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000032
//             * /*                                                        0000033
```

```
MEMBER NAME  ALGCLPE
//ALG      EXEC PGM=ALGOL,PARM= S,T,NODECK,LOAD                            0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT  DD SYSOUT=A                                                    0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000006
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),      C0000007
//          DISP=(OLD,PASS)                                                0000008
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,ALG),PARM= LIST,XREF               0000009
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000011
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000012
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000013
//SYSLMOD   DD DDNAME=SYSPVT                                               0000014
//SYSLIN    DD DSNAME=*.ALG.SYSLIN,DISP=OLD                                0000015
//          DD DDNAME=SYSIN                                               *0000016
//             * PROCEDURE COMPILES ALG.SYSIN 80-BYTE SOURCE IMAGES       *0000017
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.         *0000018
//             *                                                 PRO FORMA00000019
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                 *0000020
//             * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                  *0000021
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT             *0000022
//             * //XXXXXXXX EXEC ALGCLPE                                   *0000023
//             * //ALG.SYSIN DD *                                          *0000024
//             * ..................... SOURCE DECK ......................*0000025
//             * /*                                                       *0000026
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000027
//             * //EDT.SYSIN DD *                                         *0000028
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000029
//             * /*                                                        0000030
```

```
MEMBER NAME  ALGLGE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                               0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000005
```

```
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000006
//SYSLIN     DD DDNAME=SYSIN                                            0000007
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000008
//SYSUDUMP   DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726) 0000009
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)          0000010
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000011
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                            *0000012
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000013
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.           *0000014
//              *                                          PRO FORMA00000015
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000016
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE            *0000017
//              * //XXXXXXXX EXEC ALGLGE                                 *0000018
//              * //EDT.SYSIN DD *                                       *0000019
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000020
//              * /*                                                     0000021
```

```
MEMBER NAME  ALGLPE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                            0000001
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726) 0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000004
//SYSLIB     DD DSNAME=SYS1.ALGLIB,DISP=OLD                             0000005
//SYSLMOD    DD DDNAME=SYSPVT                                          *0000006
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000007
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.      *0000008
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000009
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT           *0000010
//              * //XXXXXXXX EXEC ALGLPE                                 *0000011
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000012
//              * //EDT.SYSIN DD *                                       *0000013
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000014
//              * /*                                                     0000015
//SYSLIN     DD DDNAME=SYSIN                                            0000016
```

```
MEMBER NAME  RPGCDE
//RPG      EXEC PGM=RPG,PARM= LIST,NOLOAD,DECK                          0000001
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726) 0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000003
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                     0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000005
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                             0000006
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                            *0000007
//              * PROCEDURE COMPILES RPG.SYSIN 80-BYTE SOURCE IMAGES    *0000008
//              * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.  *0000009
//              *                                          PRO FORMA00000010
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000011
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD               *0000012
//              * //XXXXXXXX EXEC RPGCDE                                 *0000013
//              * //RPG.SYSIN DD *                                       *0000014
//              * ..................... SOURCE DECK ....................*0000015
//              * /*                                                     0000016
```

```
MEMBER NAME  RPGCPE
//RPG      EXEC PGM=RPG,PARM= LIST,NODECK,LOAD                          0000001
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726) 0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)          0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                             0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                             0000006
//SYSGO      DD DDNAME=SYSOBJ                                          *0000007
//              * PROCEDURE COMPILES RPG.SYSIN 80-BYTE SOURCE IMAGES    *0000008
//              * INTO RPG.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.*0000009
//              *                                          PRO FORMA00000010
```

```
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000011
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                 *0000012
//              * //XXXXXXXX EXEC RPGCPE                                  *0000013
//              * //RPG.SYSIN DD *                                        *0000014
//              * ..................... SOURCE DECK ......................*0000015
//              * /*                                                      0000016
//
MEMBER NAME  RPGCLGE                                                      0000001
//RPG     EXEC PGM=RPG,PARM= LIST,NODECK,LOAD                             0000002
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000003
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)             0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000005
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000006
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000007
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),     C0000008
//             DISP=(OLD,PASS)                                            0000009
//EDT     EXEC PGM=LINKEDIT,COND=(9,LT,RPG),PARM= LIST,XREF               0000010
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000011
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)             0000012
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000013
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)   0000014
//SYSLIN    DD DSNAME=*.RPG.SYSGO,DISP=OLD                                0000015
//           DD DDNAME=SYSIN                                              0000016
//GO      EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,RPG),(5,LT,EDT))             0000017
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000018
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)             *0000019
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                        *0000020
//              * PROCEDURE COMPILES RPG.SYSIN 80-BYTE SOURCE IMAGES      *0000021
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.             *0000022
//              *                                                 PRO FORMA
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000023
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                 *0000024
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE             *0000025
//              * //XXXXXXXX EXEC RPGCLGE                                  *0000026
//              * //RPG.SYSIN DD *                                        *0000027
//              * ..................... SOURCE DECK ......................*0000028
//              * /*                                                      *0000029
//              * //EDT.SYSIN DD *                                        *0000030
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000031
//              * /*                                                      0000032

MEMBER NAME  RPGCLPE
//RPG     EXEC PGM=RPG,PARM= LIST,NODECK,LOAD                             0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)             0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                0000006
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),     C0000007
//             DISP=(OLD,PASS)                                            0000008
//EDT     EXEC PGM=LINKEDIT,COND=(9,LT,RPG),PARM= LIST,XREF               0000009
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)    0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)             0000011
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000012
//SYSLMOD   DD DDNAME=SYSPVT                                              0000013
//SYSLIN    DD DSNAME=*.RPG.SYSGO,DISP=OLD                                0000014
//           DD DDNAME=SYSIN                                              *0000015
//              * PROCEDURE COMPILES RPG.SYSIN 80-BYTE SOURCE IMAGES      *0000016
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.        *0000017
//              *                                                 PRO FORMA
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000019
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                 *0000020
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000021
//              * //XXXXXXXX EXEC RPGCLPE                                  *0000022
```

```
//              * //RPG.SYSIN DD *                                          *0000023
//              * ..................... SOURCE DECK ......................*0000024
//              * /*                                                        *0000025
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000026
//              * //EDT.SYSIN DD *                                          *0000027
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000028
//              * /*                                                         0000029


MEMBER NAME  RPGLGE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                                 0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN    DD DDNAME=SYSIN                                              0000006
//GO      EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                         0000007
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000009
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                       *0000010
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000011
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.            *0000012
//              *                                                 PRO FORMA00000013
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000014
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE             *0000015
//              * //XXXXXXXX EXEC RPGLGE                                  *0000016
//              * //EDT.SYSIN DD *                                        *0000017
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000018
//              * /*                                                         0000019


MEMBER NAME  RPGLPE
//EDT     EXEC PGM=LINKEDIT,PARM= LIST,XREF                                 0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000004
//SYSLMOD   DD DDNAME=SYSPVT                                             *0000005
//              * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000006
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.       *0000007
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000008
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000009
//              * //XXXXXXXX EXEC RPGLPE                                  *0000010
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000011
//              * //EDT.SYSIN DD *                                        *0000012
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000013
//              * /*                                                         0000014
//SYSLIN    DD DDNAME=SYSIN                                              0000015


MEMBER NAME  COBCDE
//COB     EXEC PGM=COBOL,PARM= SOURCE,MAP,SEQ,FLAGW,NOLOAD,DECK          0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                       0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000005
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000006
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000007
//SYSUT4    DD UNIT=DISK,SPACE=(TRK,(20,20))                             0000008
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY            0000009
//              * PROCEDURE COMPILES COB.SYSIN 80-BYTE SOURCE IMAGES    *0000010
//              * INTO OBJECT MODULE CARD DECK FOR LATER LINK EDITING.  *0000011
//              *                                                 PRO FORMA00000012
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1               *0000013
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                *0000014
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT            *0000015
//              * //XXXXXXXX EXEC COBCDE                                  *0000016
//              * //COB.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000017
```

```
//              * //COB.SYSIN DD *                                          *0000018
//              * ..................... SOURCE DECK ......................*0000019
//              * /*                                                        0000020


MEMBER NAME  COBCPE
//COB      EXEC PGM=COBOL,PARM= SOURCE,MAP,SEQ,FLAGW,NODECK,LOAD            0000001
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT DD   SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000006
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                               0000007
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY             0000008
//SYSLIN     DD DDNAME=SYSOBJ                                              *0000009
//              * PROCEDURE COMPILES COB.SYSIN 80-BYTE SOURCE IMAGES      *0000010
//              * INTO COB.SYSOBJ OBJECT LIBRARY FOR LATER LINK EDITING.  *0000011
//              *                                                 PRO FORMA00000012
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000013
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                 *0000014
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT             *0000015
//              * //XXXXXXXX EXEC COBCPE                                   *0000016
//              * //COB.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000017
//              * //COB.SYSIN DD *                                         *0000018
//              * ..................... SOURCE DECK ......................*0000019
//              * /*                                                        0000020


MEMBER NAME  COBCLGE
//COB      EXEC PGM=COBOL,PARM= SOURCE,MAP,SEQ,FLAGW,NODECK,LOAD            0000001
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000002
//SYSPRINT DD   SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000006
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                               0000007
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY             0000008
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),       C0000009
//              DISP=(OLD,PASS)                                             0000010
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,COB),PARM= LIST,XREF                0000011
//SYSUDUMP DD   UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000012
//SYSPRINT  DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000013
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000014
//SYSLIB     DD DSNAME=SYS1.COBLIB,DISP=OLD                                 0000015
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000016
//SYSLIN     DD DSNAME=*.COB.SYSLIN,DISP=OLD                                0000017
//              DD DDNAME=SYSIN                                             0000018
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,COB),(5,LT,EDT))              0000019
//SYSUDUMP  DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)     0000020
//SYSOUT     DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000021
//SYSPUNCH  DD  UNIT=(SYSCP,,DEFER)                                         *0000022
//              * PROCEDURE COMPILES COB.SYSIN 80-BYTE SOURCE IMAGES      *0000023
//              * FOR LINKING INTO LOAD MODULE FOR EXECUTION.             *0000024
//              *                                                 PRO FORMA00000025
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000026
//              * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD                 *0000027
//              * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE              *0000028
//              * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT             *0000029
//              * //XXXXXXXX EXEC COBCLGE                                  *0000030
//              * //COB.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000031
//              * //COB.SYSIN DD *                                         *0000032
//              * ..................... SOURCE DECK ......................*0000033
//              * /*                                                        *0000034
//              * //EDT.SYSCALL DD DSNAME=XXX,UNIT=DISK,VOLUME=SER=SETUP*0000035
//              * //EDT.SYSIN DD *                                         *0000036
//              * ........CONTROL STATEMENTS AND/OR OBJECT DECKS.........*0000037
//              * /*                                                        0000038
```

```
MEMBER NAME  COBCLPE
//COB      EXEC PGM=COBOL,PARM= SOURCE,MAP,SEQ,FLAGW,NODECK,LOAD          0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                            0000007
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY          0000008
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000009
//              DISP=(OLD,PASS)                                          0000010
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,COB),PARM= LIST,XREF             0000011
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000012
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000013
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000014
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                              0000015
//SYSLMOD   DD DDNAME=SYSPVT                                            0000016
//SYSLIN    DD DSNAME=*.CCB.SYSLIN,DISP=OLD                             0000017
//          DD DDNAME=SYSIN                                            *0000018
//             * PROCEDURE COMPILES COB.SYSIN 80-BYTE SOURCE IMAGES    *0000019
//             * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.      *0000020
//             *                                          PRO FORMA0000021
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000022
//             * //JOBLIB DD DSNAME=SYS1.LNKLIB,DISP=OLD               *0000023
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSTXT           *0000024
//             * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000025
//             * //XXXXXXXX EXEC COBCLPE                               *0000026
//             * //COB.SYSTXT DD DSNAME=XXXX,UNIT=DISK,VOLUME=SER=SETUP*0000027
//             * //COB.SYSIN DD *                                      *0000028
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000029
//             * /*                                                    *0000030
//             * /*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT           *0000031
//             * //EDT.SYSIN DD *                                      *0000032
//             * /*                                                     0000033
```

```
MEMBER NAME  COBLGE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                             0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                              0000005
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000006
//SYSLIN    DD DDNAME=SYSIN                                             0000007
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000008
//SYSUDUMP  DD UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000009
//SYSOUT    DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000010
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      *0000011
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000012
//             * FOR LINKING INTO LOAD MODULE FOR EXECUTION.           *0000013
//             *                                          PRO FORMA0000014
//             * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1              *0000015
//             * /*FORMAT PR,DDNAME=SYSUDUMP,CONTROL=SINGLE            *0000016
//             * //XXXXXXXX EXEC COBLGE                                *0000017
//             * //EDT.SYSIN DD *                                      *0000018
//             * ........CONTROL STATEMENTS AND/OR OBJECT DECKS........*0000019
//             * /*                                                     0000020
```

```
MEMBER NAME  COBLPE
//EDT      EXEC PGM=LINKEDIT,PARM= LIST,XREF                             0000001
//SYSUDUMP DD  UNIT=(CTC,,DEFER),DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)  0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLIB    DD DSNAME=SYS1.COBLIB,DISP=OLD                              0000005
//SYSLMOD   DD DDNAME=SYSPVT                                            *0000006
//             * PROCEDURE READS EDT.SYSIN STATEMENTS AND OBJECT DECKS *0000007
```

```
//              * FOR LINKING INTO EDT.SYSPVT LOAD MODULE LIBRARY.        *0000008
//              * //XXXXXXXX JOB (XXXXXX,X,X,X),MSGLEVEL=1                *0000009
//              * //*SETUP DEVICE=2314,ID=XXXXXX,DDNAME=SYSPVT            *0000010
//              * //XXXXXXXX EXEC COBLPE                                  *0000011
//              * //EDT.SYSPVT DD DSNAME=X(X),UNIT=DISK,VOLUME=SER=SETUP*0000012
//              * //EDT.SYSIN DD *                                        *0000013
//              * /*                                                      0000014
//SYSLIN     DD DDNAME=SYSIN                                             0000015

MEMBER NAME  FTHCDX
//FTH      EXEC PGM=FORTRANH,PARM= NOSOURCE,NOMAP,NOLOAD,DECK            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                       0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004

MEMBER NAME  FTHCPX
//FTH      EXEC PGM=FORTRANH,PARM= NOSOURCE,NOMAP,NODECK,LOAD            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSLIN     DD DDNAME=SYSOBJ                                            0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004

MEMBER NAME  FTHCLGX
//FTH      EXEC PGM=FORTRANH,PARM= NOSOURCE,NOMAP,NODECK,LOAD            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000003
//             DISP=OLD                                                  0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000005
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,FTH)                             0000006
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000007
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000008
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000009
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000010
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                             0000011
//           DD DDNAME=SYSIN                                             0000012
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTH),(5,LT,EDT))           0000013
//FT05F001  DD DDNAME=SYSIN                                              0000014
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000015
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                       0000016

MEMBER NAME  FTHCLPX
//FTH      EXEC PGM=FORTRANH,PARM= NOSOURCE,NOMAP,NODECK,LOAD            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),    C0000003
//             DISP=(OLD,PASS)                                           0000004
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000005
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,FTH)                             0000006
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000007
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000008
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000009
//SYSLMOD    DD DDNAME=SYSPVT                                            0000010
//SYSLIN     DD DSNAME=*.FTH.SYSLIN,DISP=OLD                             0000011
//           DD DDNAME=SYSIN                                             0000012

MEMBER NAME  FTHLGX
//EDT      EXEC PGM=LINKEDIT                                             0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSLIB     DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000004
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN     DD DDNAME=SYSIN                                             0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                        0000007
//FT05F001  DD DDNAME=SYSIN                                              0000008
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000009
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                       0000010
```

```
MEMBER NAME  FTHLPX
//EDT      EXEC PGM=LINKEDIT                                          0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)         0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                            0000003
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                           0000004
//SYSLMOD   DD DDNAME=SYSPVT                                          0000005
//SYSLIN    DD DDNAME=SYSIN                                           0000006


MEMBER NAME  FTGCDX
//FTG      EXEC PGM=FORTRANG,PARM= NOSOURCE,NOMAP,NOLOAD,DECK         0000001
//SYSPRINT  DD SYSOUT=A                                               0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                    0000003


MEMBER NAME  FTGCPX
//FTG      EXEC PGM=FORTRANG,PARM= NOSOURCE,NOMAP,NODECK,LOAD         0000001
//SYSPRINT  DD SYSOUT=A                                               0000002
//SYSLIN    DD DDNAME=SYSCBJ                                          0000003


MEMBER NAME  FTGCLGX
//FTG      EXEC PGM=FORTRANG,PARM= NOSOURCE,NOMAP,NODECK,LOAD         0000001
//SYSPRINT  DD SYSOUT=A                                               0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),  C0000003
//             DISP=(OLD,PASS)                                        0000004
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,FTG)                          0000005
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)         0000006
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                            0000007
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                           0000008
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000009
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD                           0000010
//           DD DDNAME=SYSIN                                          0000011
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTG),(5,LT,EDT))        0000012
//FT05F001  DD DDNAME=SYSIN                                           0000013
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)         0000014
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                    0000015


MEMBER NAME  FTGCLPX
//FTG      EXEC PGM=FORTRANG,PARM= NOSOURCE,NOMAP,NODECK,LOAD         0000001
//SYSPRINT  DD SYSOUT=A                                               0000002
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),  C0000003
//             DISP=(OLD,PASS)                                        0000004
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,FTG)                          0000005
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)         0000006
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                            0000007
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                           0000008
//SYSLMOD   DD DDNAME=SYSPVT                                          0000009
//SYSLIN    DD DSNAME=*.FTG.SYSLIN,DISP=OLD                           0000010
//           DD DDNAME=SYSIN                                          0000011


MEMBER NAME  FTGLGX
//EDT      EXEC PGM=LINKEDIT                                          0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)         0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                            0000003
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                           0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN    DD DDNAME=SYSIN                                           0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                     0000007
//FT05F001  DD DDNAME=SYSIN                                           0000008
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)         0000009
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                    0000010
```

```
MEMBER NAME  FTGLPX                                                     0000001
//EDT      EXEC PGM=LINKEDIT                                            0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSLIB    DD DSNAME=SYS1.FORTLIB,DISP=OLD                             0000005
//SYSLMOD   DD DDNAME=SYSPVT                                            0000006
//SYSLIN    DD DDNAME=SYSIN
```

```
MEMBER NAME  ASMCDX                                                     0000001
//ASM      EXEC PGM=ASMBLR,PARM= NOLIST,NOXREF,NOLOAD,DECK              0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)           0000003
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                      0000004
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000005
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000006
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000007
//SYSLIB    DD DSNAME=SYS1.MACLIB,DISP=OLD
```

```
MEMBER NAME  ASMCPX                                                     0000001
//ASM      EXEC PGM=ASMBLR,PARM= NOLIST,NOXREF,NODECK,LOAD              0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIB    DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000007
//SYSGO     DD DDNAME=SYSOBJ
```

```
MEMBER NAME  ASMCLGX                                                    0000001
//ASM      EXEC PGM=ASMBLR,PARM= NOLIST,NOXREF,NODECK,LOAD              0000002
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)           0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000006
//SYSLIB    DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000007
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000007
//             DISP=(OLD,PASS)                                          0000008
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,ASM)                            0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000010
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000011
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000012
//SYSLIN    DD DSNAME=*.ASM.SYSGO,DISP=OLD                              0000013
//           DD DDNAME=SYSIN                                            0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ASM),(5,LT,EDT))          0000015
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000016
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                      0000017
```

```
MEMBER NAME  ASMCLPX
//ASM      EXEC PGM=ASMBLR,PARM= NOLIST,NOXREF,NODECK,LOAD              0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)           0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                              0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                              0000005
//SYSLIB    DD DSNAME=SYS1.MACLIB,DISP=OLD                              0000006
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000007
//             DISP=OLD                                                 0000008
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,ASM)                            0000009
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000010
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                              0000011
//SYSLMOD   DD DDNAME=SYSPVT                                            0000012
//SYSLIN    DD DSNAME=*.ASM.SYSGO,DISP=OLD                              0000013
//           DD DDNAME=SYSIN                                            0000014
```

```
MEMBER NAME  ASMLGX
//EDT      EXEC PGM=LINKEDIT                                          0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)        0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000003
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000004
//SYSLIN     DD DDNAME=SYSIN                                          0000005
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                     0000006
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)        0000007
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                   0000008

MEMBER NAME  ASMLPX
//EDT      EXEC PGM=LINKEDIT                                          0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)        0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000003
//SYSLMOD    DD DDNAME=SYSPVT                                         0000004
//SYSLIN     DD DDNAME=SYSIN                                          0000005

MEMBER NAME  PL1CDX
//PL1      EXEC PGM=PL1,PARM= NS,NE,NA,NX,FS,NLD,D                    0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)         0000002
//SYSPUNCH DD  UNIT=(SYSCP,,DEFER)                                    0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                           0000005
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY         0000006

MEMBER NAME  PL1CPX
//PL1      EXEC PGM=PL1,PARM= NS,NE,NA,NX,FS,ND,LD                    0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)         0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000003
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                           0000004
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY         0000005
//SYSLIN     DD DDNAME=SYSCBJ                                         0000006

MEMBER NAME  PL1CLGX
//PL1      EXEC PGM=PL1,PARM= NS,NE,NA,NX,FS,ND,LD                    0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)         0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000003
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                           0000004
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY         0000005
//SYSLIN   DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C0000006
//             DISP=(OLD,PASS)                                        0000007
//EDT      EXEC PGM=LINKEDIT,COND=(9,LT,PL1)                          0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)        0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000010
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                           0000011
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000012
//SYSLIN     DD DSNAME=*.PL1.SYSLIN,DISP=OLD                          0000013
//            DD DDNAME=SYSIN                                         0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,PL1),(9,LT,EDT))        0000015
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)        0000016
//SYSPNCH    DD UNIT=(SYSCP,,DEFER)                                   0000017

MEMBER NAME  PL1CLPX
//PL1      EXEC PGM=PL1,PARM= NS,NE,NA,NX,FS,ND,LD                    0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)         0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                           0000003
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                           0000004
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY         0000005
//SYSLIN   DD  DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=400), C0000006
//             DISP=(OLD,PASS)                                        0000007
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,PL1)                          0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)        0000009
```

```
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                0000010
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                                0000011
//SYSLMOD    DD DDNAME=SYSPVT                                              0000012
//SYSLIN     DD DSNAME=*.PL1.SYSLIN,DISP=OLD                               0000013
//           DD DDNAME=SYSIN                                               0000014
```

```
MEMBER NAME  PL1LGX                                                       0000001
//EDT      EXEC PGM=LINKEDIT                                              0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                               0000005
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)  0000006
//SYSLIN     DD DDNAME=SYSIN                                              0000007
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(9,LT,EDT)                         0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000009
//SYSPNCH    DD UNIT=(SYSCP,,DEFER)
```

```
MEMBER NAME  PL1LPX                                                       0000001
//EDT      EXEC PGM=LINKEDIT                                              0000002
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)            0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSLIB     DD DSNAME=SYS1.PL1LIB,DISP=OLD                               0000005
//SYSLMOD    DD DDNAME=SYSPVT                                             0000006
//SYSLIN     DD DDNAME=SYSIN
```

```
MEMBER NAME  ALGCDX
//ALG      EXEC PGM=ALGOL,PARM= NS,NT,NOLOAD,DECK                         0000001
//SYSPRINT   DD SYSOUT=A                                                  0000002
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                       0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000006
```

```
MEMBER NAME  ALGCPX
//ALG      EXEC PGM=ALGOL,PARM= NS,NT,NODECK,LOAD                         0000001
//SYSPRINT   DD SYSOUT=A                                                  0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000005
//SYSLIN     DD DDNAME=SYSOBJ                                             0000006
```

```
MEMBER NAME  ALGCLGX
//ALG      EXEC PGM=ALGOL,PARM= NS,NT,NODECK,LOAD                         0000001
//SYSPRINT   DD SYSOUT=A                                                  0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                               0000005
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),     C0000006
//             DISP=(OLD,PASS)                                            0000007
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,ALG)                              0000008
//SYSPRINT   DD DUMMY                                                     0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000010
//SYSLIB     DD DSNAME=SYS1.ALGLIB,DISP=OLD                               0000011
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)  0000012
//SYSLIN     DD DSNAME=*.ALG.SYSLIN,DISP=OLD                              0000013
//           DD DDNAME=SYSIN                                              0000014
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,ALG),(5,LT,EDT))            0000015
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)            0000016
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                               0000017
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                               0000018
```

```
MEMBER NAME  ALGCLPX
//ALG      EXEC PGM=ALGOL,PARM= NS,NT,NODECK,LOAD                          0000001
//SYSPRINT  DD SYSOUT=A                                                    0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000005
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),      C0000006
//             DISP=(OLD,PASS)                                             0000007
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,ALG)                               0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000009
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000010
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000011
//SYSLMOD   DD DDNAME=SYSPVT                                               0000012
//SYSLIN    DD DSNAME=*.ALG.SYSLIN,DISP=OLD                                0000013
//          DD DDNAME=SYSIN                                                0000014


MEMBER NAME  ALGLGX
//EDT      EXEC PGM=LINKEDIT                                               0000001
//SYSPRINT  DD DUMMY                                                       0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000004
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)    0000005
//SYSLIN    DD DDNAME=SYSIN                                                0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                          0000007
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)              0000008
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000009
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000010


MEMBER NAME  ALGLPX
//EDT      EXEC PGM=LINKEDIT                                               0000001
//SYSPRINT  DD DUMMY                                                       0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSLIB    DD DSNAME=SYS1.ALGLIB,DISP=OLD                                 0000004
//SYSLMOD   DD DDNAME=SYSPVT                                               0000005
//SYSLIN    DD DDNAME=SYSIN                                                0000006


MEMBER NAME  RPGCDX
//RPG      EXEC PGM=RPG,PARM= NOLIST,NOLOAD,DECK                           0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                         0000003
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000004
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000005
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000006


MEMBER NAME  RPGCPX
//RPG      EXEC PGM=RPG,PARM= NOLIST,NODECK,LOAD                           0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000005
//SYSGO     DD DDNAME=SYSOBJ                                               0000006


MEMBER NAME  RPGCLGX
//RPG      EXEC PGM=RPG,PARM= NOLIST,NODECK,LOAD                           0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000002
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                 0000003
//SYSUT2    DD DSNAME=SYS1.SYSUT2,DISP=OLD                                 0000004
//SYSUT3    DD DSNAME=SYS1.SYSUT3,DISP=OLD                                 0000005
//SYSGO  DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),      C0000006
//             DISP=OLD                                                    0000007
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,RPG)                               0000008
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)              0000009
```

```
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000010
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)     0000011
//SYSLIN     DD DSNAME=*.RPG.SYSGO,DISP=OLD                                  0000012
//           DD DDNAME=SYSIN                                                 0000013
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((9,LT,RPG),(5,LT,EDT))               0000014
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)               0000015
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                          0000016
```

```
MEMBER NAME  RPGCLPX
//RPG      EXEC PGM=RPG,PARM= NOLIST,NODECK,LOAD                             0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                  0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                  0000005
//SYSGO   DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),       C0000006
//              DISP=OLD                                                     0000007
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,RPG)                                 0000008
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000009
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000010
//SYSLMOD    DD DDNAME=SYSPVT                                                0000011
//SYSLIN     DD DSNAME=*.RPG.SYSGO,DISP=OLD                                  0000012
//           DD DDNAME=SYSIN                                                 0000013
```

```
MEMBER NAME  RPGLGX
//EDT      EXEC PGM=LINKEDIT                                                 0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000003
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G)     0000004
//SYSLIN     DD DDNAME=SYSIN                                                 0000005
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                            0000006
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)               0000007
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                          0000008
```

```
MEMBER NAME  RPGLPX
//EDT      EXEC PGM=LINKEDIT                                                 0000001
//SYSPRINT   DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000003
//SYSLMOD    DD DDNAME=SYSPVT                                                0000004
//SYSLIN     DD DDNAME=SYSIN                                                 0000005
```

```
MEMBER NAME  COBCDX
//COB      EXEC PGM=COBOL,PARM= NOSOURCE,NOMAP,NOSEQ,FLAGE,NOLOAD,DECK      0000001
//SYSPRINT DD   SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000002
//SYSPUNCH DD   UNIT=(SYSCP,,DEFER)                                          0000003
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000004
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                  0000005
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                  0000006
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                                0000007
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY              0000008
```

```
MEMBER NAME  COBCPX
//COB      EXEC PGM=COBOL,PARM= NOSOURCE,NOMAP,NOSEQ,FLAGE,NODECK,LOAD      0000001
//SYSPRINT DD   SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                                  0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                                  0000005
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                                0000006
//SYSLIB    DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY              0000007
//SYSLIN     DD DDNAME=SYSOBJ                                                0000008
```

```
MEMBER NAME  COBCLGX
//COB      EXEC PGM=COBOL,PARM= NOSOURCE,NOMAP,NOSEQ,FLAGE,NODECK,LOAD   0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                             0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                             0000005
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                           0000006
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY           0000007
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000008
//             DISP=(OLD,PASS)                                          0000009
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,COB)                            0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000011
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000012
//SYSLIB     DD DSNAME=SYS1.COBLIB,DISP=OLD                             0000013
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000014
//SYSLIN     DD DSNAME=*.CCB.SYSLIN,DISP=OLD                            0000015
//           DD DDNAME=SYSIN                                            0000016
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,COB),(5,LT,EDT))          0000017
//SYSOUT     DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)          0000018
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                     0000019

MEMBER NAME  COBCLPX
//COB      EXEC PGM=COBOL,PARM= NOSOURCE,NOMAP,NOSEQ,FLAGE,NODECK,LOAD   0000001
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSUT2     DD DSNAME=SYS1.SYSUT2,DISP=OLD                             0000004
//SYSUT3     DD DSNAME=SYS1.SYSUT3,DISP=OLD                             0000005
//SYSUT4     DD UNIT=DISK,SPACE=(TRK,(20,20))                           0000006
//SYSLIB   DD  DDNAME=SYSTXT            COMPILE-TIME FACILITY           0000007
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),   C0000008
//             DISP=(OLD,PASS)                                          0000009
//EDT      EXEC PGM=LINKEDIT,COND=(1,LT,COB)                            0000010
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000011
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000012
//SYSLIB     DD DSNAME=SYS1.COBLIB,DISP=OLD                             0000013
//SYSLMOD    DD DDNAME=SYSPVT                                           0000014
//SYSLIN     DD DSNAME=*.CCB.SYSLIN,DISP=OLD                            0000015
//           DD DDNAME=SYSIN                                            0000016

MEMBER NAME  COBLGX
//EDT      EXEC PGM=LINKEDIT                                            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSLIB     DD DSNAME=SYS1.COBLIB,DISP=OLD                             0000004
//SYSLMOD    DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),DSNAME=+G(G) 0000005
//SYSLIN     DD DDNAME=SYSIN                                            0000006
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                       0000007
//SYSOUT     DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133.BLKSIZE=798)          0000008
//SYSPUNCH   DD UNIT=(SYSCP,,DEFER)                                     0000009

MEMBER NAME  COBLPX
//EDT      EXEC PGM=LINKEDIT                                            0000001
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)           0000002
//SYSUT1     DD DSNAME=SYS1.SYSUT1,DISP=OLD                             0000003
//SYSLIB     DD DSNAME=SYS1.COBLIB,DISP=OLD                             0000004
//SYSLMOD    DD DDNAME=SYSPVT                                           0000005
//SYSLIN     DD DDNAME=SYSIN                                            0000006

MEMBER NAME  COPYAGO
//GO EXEC PGM=COPYAGO                                                   0000001
//SYSPRINT DD SYSOUT=A                                                  0000002
//SYSUT1   DD  UNIT=(DISK,,DEFER),DISP=OLD,VOLUME=SER=SCRTWO            0000003
//TAPE     DD  UNIT=(TAPE,,DEFER),DISP=OLD,VOLUME=SER=DUMMY             0000004
//POINTER  DD  UNIT=(DISK,,DEFER),VOLUME=SER=SYSSCR,DSNAME=REPLACE,    X0000005
//             DISP=(OLD,DELETE)                                        0000006
//FT06F001 DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)           0000007
//FT07F001 DD  UNIT=(SYSCP,,DEFER)                                      0000008
```

B. Hardware Description

1. Equipment List

ARGONNE IBM 75/50 CONFIGURATION
SEPTEMBER 15, 1967

NO.	UNIT	DESCRIPTION	
CENTRAL PROCESSOR AND CORE STORAGE		**********	
1	2075-J	CENTRAL PROCESSING UNIT	
4	2365/3	CORE STORAGE	1024BK BYTES
1	8080	2361 ADAPTER FOR 2075 CPU	
1	2150-1	OPERATOR CONSOLE	
1	5475	OPERATOR CONTROL PANEL	
1	1052-7	PRINTER-KEYBOARD	
SUPPORT PROCESSOR AND CORE STORAGE		**********	
1	2050-I	CENTRAL PROCESSING UNIT	512BK BYTES
1	8080	2361 ADAPTER FOR 2050 CPU	
1	7920	1052 ADAPTER	
1	3274	DIRECT CONTROL	
1	1850	CHANNEL TO CHANNEL ADAPTER	
1	6980	SELECTOR CHANNEL	
1	6981	SELECTOR CHANNEL	
1	6982	SELECTOR CHANNEL	
1	1052-7	PRINTER-KEYBOARD	
LARGE CAPACITY CORE STORAGE		**********	
1	2361-2	LARGE CAPACITY CORE STORAGE	2048BK BYTES
1	7131	SHARED STORAGE	
MAGNETIC TAPE EQUIPMENT		**********	
2	2403-3	TAPE CONTROL AND TAPE	
1	RPQ	2-CHANNEL SWITCH (CONTROLLER)	
2	3228	DATA CONVERSION	
2	7125	7-TRACK COMPATIBILITY	
2	2402-3	DOUBLE TAPE	
1	2816	SWITCHING UNIT	
1	1050	ADDITIONAL DRIVES	
2	9557	7-TRACK READ/WRITE HEAD	
4	9558	9-TRACK READ/WRITE HEAD	
UNIT RECORD EQUIPMENT		**********	
1	2821-1	CONTROL UNIT	
1	1990	COLUMN BINARY	
1	8637	UNIVERSAL CHARACTER SET ADAPTER	
1	3615	1100 LPM ADAPTER	
1	1403-N1	PRINTER 1100 LPM	1100 LPM
1	8640	UNIVERSAL CHARACTER SET FEATURE	
1	1416-1	INTERCHANGEABLE TRAIN CARTRIDGE	
1	9632	PRINT TRAIN ARRANGEMENT ON	
1	2540-1	CARD READER-PUNCH	1000/300 CPM
1	2821-5	CONTROL UNIT	
1	1990	COLUMN BINARY	
1	8637	UNIVERSAL CHARACTER SET ADAPTER	
1	8638	UNIVERSAL CHARACTER SET ADAPTER	
1	8639	UNIVERSAL CHARACTER SET ADAPTER	
1	7945	3RD PRINTER CONTROL	
3	3615	1100 LPM ADAPTER	
3	1403-N1	PRINTER 1100 LPM	1100 LPM
3	8640	UNIVERSAL CHARACTER SET FEATURE	
3	1416-1	INTERCHANGEABLE TRAIN CARTRIDGE	
3	9632	PRINT TRAIN ARRANGEMENT ON	
1	2540-1	CARD READER-PUNCH	

```
SELECTOR CHANNELS                          **********
1     2860-3       THREE SELECTOR CHANNELS

DATA CELL                                  **********
1     2841-1       STORAGE CONTROL
1       4385       FILE SCAN
1       6118       RECORD OVERFLOW
1       8079       2321 ATTACHMENT
1       8100       2-CHANNEL SWITCH
2     2321-1       DATA CELL DRIVE                        800,000,000 BYTES

DISPLAY STATIONS                           **********
1     2848-3       DISPLAY CONTROL
4       3357       DISPLAY ADAPTER
1       3857       EXPANSION UNIT
1       4787       LINE-ADDRESSING
1       5340       NON-DESTRUCTIVE CURSOR
4       5341       NON-DESTRUCTIVE CURSOR ADAPTER
1       7928 PRINTER ADAPTER
8     2260-1       DISPLAY STATION
8       4766       ALPHAMERIC KEYBOARD
1     1053-4       PRINTER

GRAPHIC DISPLAYS AND RECORDER              **********
1     2840-2       DISPLAY CONTROL
1       4395       FILM ATTACHMENT
1     2280-1       FILM RECORDER
1        RPQ       EXPANDED STROKE MODE
1        RPQ       FILM INCREMENTAL POSITIONING
1     2250-3       DISPLAY STATION
1       1245       ALPHAMERIC KEYBOARD
1       5855       PROGRAMMED FUNCTION KEYBOARD

DISK STORAGE                               **********
2     2314-1       DIRECT ACCESS STORAGE FACILITY         466,816,000 BYTES
1       8170       2-CHANNEL SWITCH

DRUM STORAGE                               **********
1     2820-1       DRUM STORAGE CONTROL
1     2301-1       DRUM STORAGE UNIT                      4,096,600 BYTES

TELEPROCESSING AND COMMUNICATIONS          **********
1     2701-1       DATA ADAPTER UNIT
1       3855       EXPANSION FEATURE
2       5500       PARALLEL DATA ADAPTER
1     2702-1       TRANSMISSION CONTROL
3       3233       DATA SET LINE ADAPTER
1       3853       EXPANSION BASE
1       4615       IBM TERMINAL CONTROL TYPE I
2       4635       IBM LINE ADAPTER-4 WIRE
1       7912       TEL TERMINAL CONTROL TYPE II
2     2741-1       COMMUNICATION TERMINAL
```

2. Configuration Chart

A configuration chart for the IBM System/360 System is shown in Fig. 1.

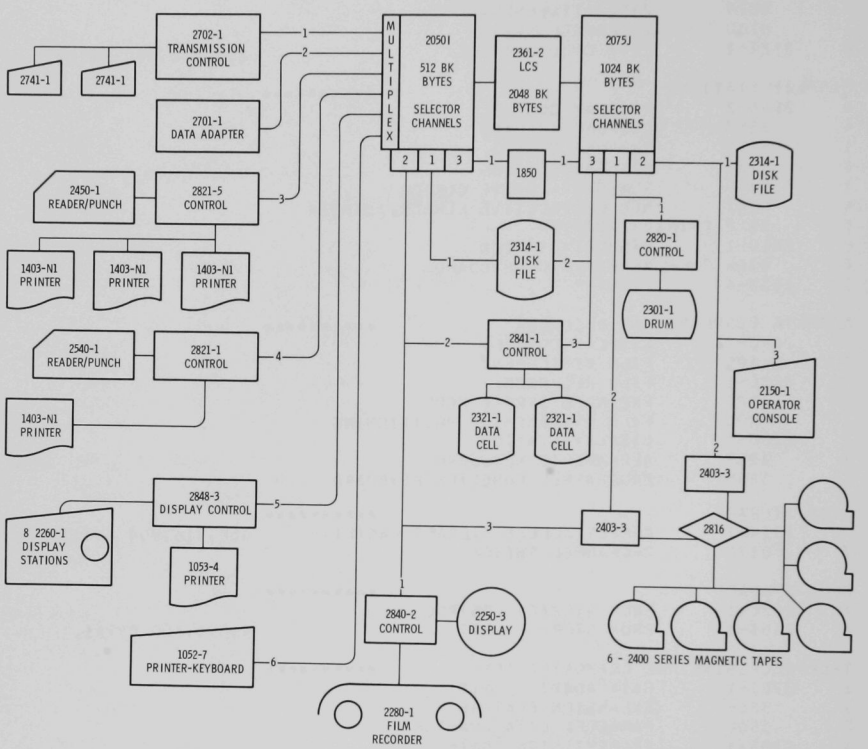


Fig. 1. Configuration Chart of IBM System/360 System

C. Bibliography

IBM Form No.	Title
A22-6898	Model 50 Functional Characteristics
A22-6908	Model 50 Operating Procedures
A22-6889	Model 75 Functional Characteristics
A22-6909	Model 75 Operating Procedures
A22-6877	1052 Printer Keyboard Model 7 with 2150 Console
A24-3073	1403 Printer
A27-2700	2260 Display Station, 2848 Display Control Component Description
A22-6853	2280 Film Recorder, 2281 Film Scanner and 2282 Film Recorder/Scanner
A21-9033	2540 Component Description and Operating Procedures
A22-6866	2400 and 2816 Model 1 Component Description
A26-5988	2841 Storage Control Unit, 2302 Disk Storage Models 3 and 4, 2311 Disk Storage Drive, 2321 Data Cell Drive Model 1, and 7320 Drum Storage
A22-6895	2301 Drum Storage-2820 Storage Control, Component Description
A26-3599	2314 Direct Access Storage Facility, Component Description
A26-3633	Data Cell Handling Guide (2321)
A22-6864	2701 Data Adapter Unit, Principles of Operation
A22-6846	2702 Transmission Control, Component Description
A24-3415	2741 Communications Terminal
	Operating System/360(OS)
C28-6534	Operating System Introduction
C28-6550	System Programmer's Guide
C28-6551	Storage Estimates
C28-6630	Starter Operating System Guide
C28-6631	Messages, Completion Codes and Storage Dumps
C28-6514	Assembler Language

IBM Form No.	Title
C26-3756	Assembler (E and F) Programmer's Guide
C28-6516	COBOL Language
C28-6380	COBOL (F) Programmer's Guide
C28-6629	Basic FORTRAN IV Language
C28-6639	FORTRAN IV (G) Programmer's Guide
C28-6602	FORTRAN IV (H) Programmer's Guide
C28-6515	FORTRAN IV Language
C28-6615	ALGOL Language
C24-3337	Report Program Generator Language
C28-6571	PL/I: Language Specifications
C28-6590	PL/I: Subroutine Library Computational Subroutines
C28-6594	Program Language/One (F) Programmer's Guide
C27-6909	Graphic Programming Services for 2250 Display Unit
C27-6925	Graphic Programming Services for 2260 Display Station
C27-6912	Graphic Programming Services for 2262 Display Station
C27-6927	Graphic Programming Services for IBM 2280 and 2282 Film Units
C30-2004	Basic Telecommunications Access Method
C30-2005	Queued Telecommunications Access Method Message Control Program
C30-2003	Queued Telecommunications Access Method Message Processing Program Services
C28-6538	Linkage Editor
C27-6918	Maintenance Program
C28-6554	System Generation
C28-6586	Utilities
C28-6543	Sort/Merge Program
C27-6926	Multiprogramming with a Fixed Number of Tasks: Concepts and Considerations
C28-6535	Concepts and Facilities
C28-6539	Job Control Language

IBM Form No.	Title
C28-6540	Operator's Guide
C28-6646	Supervisor and Data Management Services
C28-6647	Supervisor and Data Management Macro Instructions
C28-6628	System Control Blocks
C28-6632	Job Control Language Charts
C20-1652	TESTRAN User's Guide
	Attached Support Processor System/ASP
H20-0223	ASP System Description Version 1
H20-0466	ASP System Description Version 2
H20-0321	Operator's Manual
H20-0322	Application Programmer's Manual
H20-0323	System Programmer's Manual
Y20-0069	System Manual

III. CDC 3600 COMPUTER ENVIRONMENT

Since the fall of 1963 Argonne has had in operation a two-bank Control Data Corporation 3600 computer operating in conjunction with satellite and offline 160A computers. This facility is described in a similar manner to the 360 with the three breakdowns: software systems, hardware description, and bibliography.

A. Software Systems

The main processor, the 3600, is operated under the Control Data SCOPE Operating System, with the satellite 160A under the ANL SATCOPS program providing the input/output processing required. The two computers communicate via the satellite coupler.

1. Resident System

15,536 48-bit words of memory are required for the resident system. Since the Table of Contents resides in Bank 1 in locations 77001_8 through 77776_8, locations 1 through 77000_8 are available for use in that bank; the rest of the 50,000 available locations are in high-address Bank 0.

2. Software Systems Available

   a. 3600 FORTRAN

   b. ALGOL

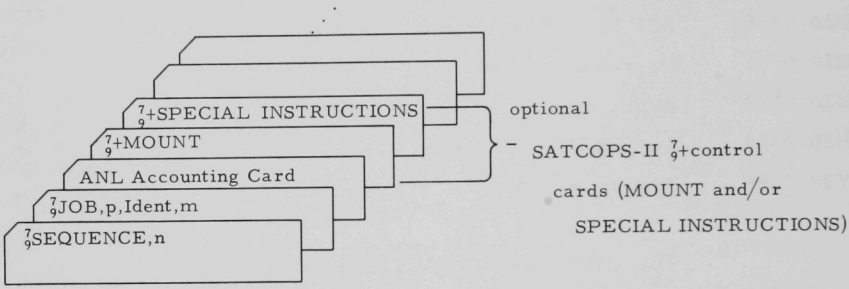
   c. COMPASS

   d. SORT

3. ANL SCOPE/SATCOPS-II Control Cards

   a. SCOPE Input Deck Format

The format of a job input deck under the ANL SCOPE operating system is as follows:



   b. 7/9JOB Card

The format of an ANL JOB card is as follows:

Col 1	Cols 2-80
7,9	JOB,p,Ident,m

p = estimated number of pages of standard output to be printed. If the first character is a T, all standard output will be written on tape, rather than disk;

Ident = output station and requester identification;

m = job time limit (minutes).

   c. 7/9ID Card

The format of an ANL ID card is as follows:

Col 1	Cols 2-80
7,9	ID, 6-character alphanumeric field

This identification card has the following effect:

1) The card image with the column 1 7,9 replaced by a blank is written on the standard output tape.

2) The six-character field preceded and followed by a blank is written on the standard punch tape as a binary record from which the peripheral processor will produce a single card with the identification in legible form for use in easily separating and identifying binary output decks.

d. $^{7}_{9}$GOANYWAY Card

The format of an ANL GOANYWAY card is as follows:

Col 1	Cols 2-80
7,9	GOANYWAY

This card allows a user to specify that a run be loaded and executed in spite of errors returned from a named library program. This control card must immediately precede the control card for the first named library program called in a run. The GOANYWAY statement will affect that particular run only. The card image with the column 1 7,9 replaced by a blank is written on the standard output tape.

e. $^{7}_{9}$+MOUNT and $^{7}_{9}$+SPECIAL INSTRUCTIONS Cards

The formats of the ANL MOUNT and SPECIAL INSTRUCTIONS cards are as follows:

Col 1	Col 2	Col 3
7,9	+	MOUNT,xxxxxxx
7,9	+	SPECIAL INSTRUCTIONS

A $^{7}_{9}$+MOUNT card must be supplied for each tape to be mounted for a job; the seven-character quantity is a fixed-field tape identifier. No more than nine $^{7}_{9}$+MOUNT cards may be included in an input deck.

A $^{7}_{9}$+SPECIAL INSTRUCTIONS card is used if any special action of the 3600 operator is required in running the job. The instructions themselves must be written on a special form. If more than one $^{7}_{9}$+SPECIAL INSTRUCTIONS card appears in a deck, all but the first are ignored.

To reduce the need for these forms, special mnemonics have been assigned the more common requests, and these may be used in the seven-character field of the $^{7}_{9}$+MOUNT,xxxxxxx cards to convey the assigned special instructions, as follows:

Seven-character Field	Interpretation
Lxxxxx	Mount library tape xxxxx without ring (556 bpi).
LxxxxxL	Mount library tape xxxxx without ring at low density (200 bpi).
LxxxxxR	Mount library tape xxxxx with write enable ring in (556 bpi).
LxxxxxI	Mount library tape xxxxx with write enable ring in (200 bpi).
Rxxxxx	Mount tape reel xxxxx without ring (556 bpi).
RxxxxxL	Mount tape reel xxxxx without ring at low density (200 bpi).
RxxxxxR	Mount tape reel xxxxx with write enable ring in (556 bpi).
RxxxxxI	Mount tape reel xxxxx with write enable ring in (200 bpi).
xxxFILM	xxx number of feet of film needed for this job; unnecessary if fewer than 50 ft (15 frames ≡ 1 ft).
SCRATCH	Mount a SCRATCH tape.
CALCOMP	Mount a CALCOMP tape.
DISK	The disk will be used.
fulrlxx	A full reel (2400 ft) of magnetic tape is required for LUNxx.
TOTUnn	Total number of user tapes for this job is nn.
TOTSnn	Total number of scratch tapes used on this job is nn.
LUN$\ell\ell$	The previously indicated tape is LUN$\ell\ell$.
LUN$\ell\ell$**	The previously indicated tape is LUN$\ell\ell$ unlabeled.
$^{7}_{9}$+MOUNT,FORCE $^{7}_{9}$+MOUNT,LUNxx $^{7}_{9}$+MOUNT,yyyyy	These MOUNT cards indicate that a $^{7}_{9}$EQUIP,xx=DA card is in the user's input deck. When SCOPE types out CANNOT ASSIGN xx = MT, tape yyyyy is to be assigned.

B. Hardware Description

1. Equipment List

ARGONNE CDC 3600/160A CONFIGURATION
SEPTEMBER 15, 1967

NO.	UNIT	DESCRIPTION	
CENTRAL PROCESSOR AND CORE STORAGE		**********	
1	3601	CONSOLE	
1	3602	COMMUNICATION MODULE	
2	3603	STORAGE MODULE	65,536 48-BIT WORDS
1	3604	COMPUTATION MODULE	
4	3606	STANDARD BI-DIRECTIONAL DATA CHANNEL	
1	3607	SPECIAL 24-BIT DATA CHANNEL	
1	3667	POWER CONVERTER AND CONTROL	
2	3681	DATA CHANNEL CONVERTER	
1	3682	SATELLITE COUPLER	
MAGNETIC TAPE EQUIPMENT		**********	
1	3624	MAGNETIC TAPE CONTROLLER	
16	606	MAGNETIC TAPE TRANSPORT	
DATA DISPLAY AND RECORDER		**********	
1	DD80A	DATA DISPLAY UNIT	
SATELLITE PROCESSOR AND DISK STORAGE		**********	
1	160A	COMPUTER	8192 8-BIT WORDS
1	161	ON-LINE I/O TYPEWRITER	
1	3692	PROGRAM CONTROLLED I/O TYPEWRITER	
1	3632	DISK FILE CONTROLLER	
1	828	DISK FILE	4,194,304 48-BIT WORDS
OFF LINE PROCESSOR AND EQUIPMENT		**********	
1	160A	COMPUTER	8192 8-BIT WORDS
1	161	ON-LINE I/O TYPEWRITER	
1	162-2	MAGNETIC TAPE SYNCHRONIZER	
4	606	MAGNETIC TAPE TRANSPORT	
1	168-2	AUXILIARY ARITHMETIC UNIT	
1	3447	CARD READER CONTROLLER 1 CHANNEL	
1	3649	CARD READER CONTROLLER 2 CHANNEL	
2	405	CARD READER	1200 CPM
1	3644	CARD PUNCH CONTROLLER 2 CHANNEL	
1	415	CARD PUNCH	250 CPM
4	3659	LINE PRINTER CONTROLLER	
4	501	LINE PRINTER	1000 LPM
1	3681	DATA CHANNEL CONVERTER	
1	8528	DATA COMMUNICATIONS TERMINAL	
1	MR20	MICROWAVE	

2. Configuration Chart

A configuration chart for the CDC-3600 computer system is shown in Fig. 2.

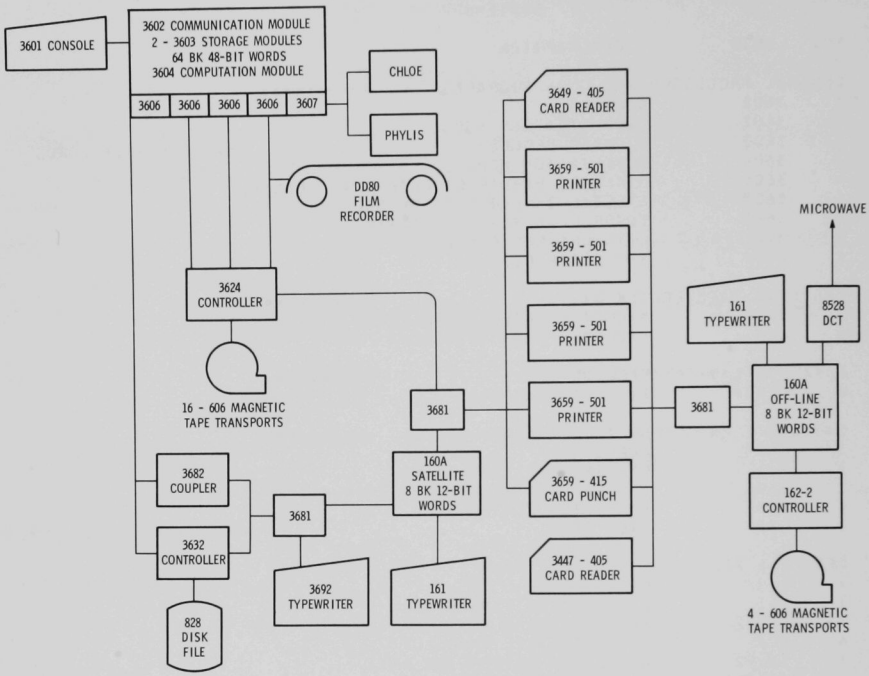


Fig. 2. Configuration Chart of CDC 3600 Computer System

C. Bibliography

1. 3600

G. A. Robinson and Ronald F. Krupp — A Cathode-ray-tube Plotting System for the Control Data 3600 Computer, ANL-7121 (Dec 1965).

CDC Form No.	Title
60108800C	Peripheral Equipment Reference Manual
60113400D	Peripheral Equipment Codes
60148800G	Input/Output Specifications

60021300J	3600 Reference Manual
60044400C	3600/3800 Instruction Codes
60052500D	3600 COMPASS Reference Manual
60053300B	3600 SCOPE Reference Manual
60056400B	Library Functions for the 3600
60058500A	3600 SORT II Reference Manual
60059500	3400/3600 PERT Reference Manual
60059800A	3400/3600 SORT III Reference Manual
60084800A	ALGOL for the 3400/3600
60132300	3400/3600 CDM4 Reference Manual
60132800	3400/3600 APT Reference Manual
60132900	3400/3600 FORTRAN Reference Manual
60134600	3600/3800 SIMSCRIPT Reference Manual
60170300	3600/3800 INFOL Reference Manual
60170500	3400/3600 COBOL Reference Manual
60170800	3600/3800 PERT-TIME Reference Manual
60172200	3600 OPHELIE Reference Manual
60174800	3600 MSID Reference Manual

2. 160A

60007300M	161 A/B On-line I/Ø Typewriter Reference/Instruction Manual
60023000C	162 Magnetic Tape Synchronizer Operation, Programming, Installation
60025800D	162-1/162-2 Reference Manual
60023800C	168-2 Auxiliary Arithmetic Unit Reference/Instruction Manual
40809300	405 High Speed Card Reader Reference Manual
40806900	415 Card Punch Reference Manual
40807200	501 High Speed Line Printer Reference Manual
PED2201A	606 Magnetic Tape Transport Reference Manual
60043500B	362X Magnetic Tape Controller Reference/Instruction Manual
60099600A	363X A Magnetic Disk File Controller Reference Manual
60107500A	3644 A Card Punch Controller Reference Manual
60097900B	3649 A Card Reader Controller Reference/Instruction Manual
60169100	3681 System Maintenance Monitor Reference Manual

IV. PERIPHERAL AND SPECIAL-PURPOSE EQUIPMENT

A. Peripheral Equipment Listing

ARGONNE PERIPHERAL EQUIPMENT
SEPTEMBER 15, 1967

NO.	UNIT	DESCRIPTION	
IBM 1401 PROCESSOR		**********	
1	1401-C6	PROCESSING UNIT	16000 6-BIT CHARACTERS
1	1406-3	ADDITIONAL STORAGE	
1	5540	PRINT CONTROL, ADDITION	
1	5275	MULTIPLY-DIVIDE	
1	5585	PRINT STORAGE	
1	1990	COLUMN BINARY	
1	4575	HIGH-LOW-EQUAL COMPARE	
1	1060	ADVANCED PROGRAMMING	
1	7246	SPACE SUPPRESSION	
1	7600	SENSE SWITCHES	
1	5895	PUNCH-FEED-READ	
1	1402-1	CARD READ/PUNCH	800/250 CPM
1	5890	PUNCH-FEED-READ	
1	1403-2	PRINTER	600 LPM
4	729-5	MAGNETIC TAPE UNIT	
CDC 160A COMPUTER BUILDING 362		**********	
1	160A	COMPUTER	8192 8-BIT WORDS
1	161	ON-LINE I/O TYPEWRITER	
1	162-2	MAGNETIC TAPE SYNCHRONIZER	
2	606	MAGNETIC TAPE TRANSPORT	
1	168-2	AUXILIARY ARITHMETIC UNIT	
1	170	CARD PUNCH CONTROLLER	
1	IBM523	GANG SUMMARY PUNCH	150 CPM
1	177	CARD READER CONTROLLER	
1	405	CARD READER	1200 CPM
1	1612	HIGH SPEED LINE PRINTER	1000 LPM
1	8528	DATA COMMUNICATIONS TERMINAL	
1	MR20	MICROWAVE	
CDC 160A COMPUTER BUILDING 208		**********	
1	160A	COMPUTER	8192 8-BIT WORDS
1	161	ON-LINE I/O TYPEWRITER	
1	162-2	MAGNETIC TAPE SYNCHRONIZER	
2	606	MAGNETIC TAPE TRANSPORT	
1	168-2	AUXILIARY ARITHMETIC UNIT	
1	170	CARD PUNCH CONTROLLER	
1	IBM523	GANG SUMMARY PUNCH	150 CPM
1	177	CARD READER CONTROLLER	
1	405	CARD READER	1200 CPM
1	1612	HIGH SPEED LINE PRINTER	1000 LPM
1	165-2	INCREMENTAL PLOTTER	300 STEPS/SEC
MAGNETIC TAPE PLOTTING		**********	
1	580	CALCOMP MAGNETIC TAPE PLOTTER	200/300 STEPS/SEC
1	765	CALCOMP DIGITAL PLOTTER (DRUM)	450/1687 STEPS/SEC
1	780	CALCOMP OFF-LINE MAGNETIC TAPE	

B. Peripheral Equipment Bibliography†

1. Control Data Corporation

60013900D	165/165-2 Incremental Plotter Instruction Manual
60021800A	170A Card Punch Controller Reference/Instruction Manual
60049100D	177A Card Reader Controller Reference/Instruction Manual
60098200D	1612G High Speed Line Printer with Off-line Buffer

2. California Computer Products

Bulletin 151B	CalComp Plotter Subroutine Package for CDC 1604/3600, September 14, 1963
Bulletin 170B	Plotter Programming for CalComp Digital Incremental Plotters, August 1965
Bulletin 175	Model 770 Digital Plotting System with Model 765 Plotter, 1965

C. Special-purpose Equipment

1. CHLOE is an automatic film-scanning system consisting of an ASI-210 computer and an optical scanner designed for the digital processing of photographic information from 35-mm film. CHLOE is equipped with twin scanning stations, each consisting of a cathode-ray-tube light source, which projects a spot of light onto the film, and a photomultiplier, which views the light transmitted through the film. The light spot is driven by a pair of counting registers, x and y, so as to appear sequentially at regular points over a designated rectangular area of the frame. The extent of this area, as well as the frequency of the spot's appearance, is determined by the ASI program. As many as 4096 x 4096 spots may be selected for scanning a 1.25- by 1.25-in. area. Whenever the photomultiplier unit detects a significant change in the light transmitted from one point to the next, the coordinates of that point (the contents of the x and y registers) are sent to the computer memory together with the density level registered. The ASI program designates the film condition to be scanned as either white on black, or black on white. The scanner design permits density readings at eight different levels, and the CRT light output level is adjustable to 1024 values under program control. The ASI-210 has paper tape input-output, a console typewriter, and a 200-bpi magnetic-tape unit.

Reference: D. Hodges, CHLOE, An Automatic Film Scanning Equipment, Hardware Reference Manual, ANL-AMD Technical Memorandum 61 (Nov 1963).

†Any references given in Section III.C are not repeated here.

2. PHYLIS is a system designed to process data acquired at the Physics Division's 12-MeV Tandem generator and 4.5-MeV Van de Graaff generator. The PHYsics On Line Information Station consists of an ASI-2100 computer with direct communication links to a Nuclear Data Multi-Channel Analyzer and to an ASI-210 remote station. Peripheral equipment available to the system includes magnetic-tape units, line printers, card-processing equipment, console typewriters, a digital plotter, and a graphic-display device.

PHYLIS is controlled by an executive program resident in the ASI 2100, which permits the execution of data-processing programs in the ASI 2100 or in the CDC 3600 on an interrupt basis. These programs fall into the two categories of routine experimental data manipulation or collection and analysis (e.g., γ-ray spectrum stripping reaction kinetics, least-squares fitting).

References: R. H. Vonderohe and D. S. Gemmell, "A Description of the PHYLIS On-Line Computer System"; W. F. Miller, M. A. Fisherkeller, and K. Hillstrom, "The PHYLIS Executive System"; D. S. Gemmell, "Data-Handling Programs in Use with the PHYLIS Computing System"; Automatic Acquisition and Reduction of Nuclear Data, Proc. EANDC Conference, Karlsruhe, 1964.

3. PAULETTE is a system designed to count automatically the tracks made by atomic particles in a fine-grain emulsion coated on a plate of glass. An optical system enlarges a segment of the plate onto the face of an image-dissector tube. Electrons from the tube photocathode corresponding to the incident image are accelerated toward the rear of the tube and focused onto a resolving plane with a small rectangular aperture 7 by 40 μ. A scan is performed by moving the entire image across the resolving aperture. Signals from a 12-stage photomultiplier, located immediately behind the aperture, correspond to electrons from the segment of the image currently incident upon the aperture. Coordinates are sent to the PDP-9 computer, which contains the analysis program for determining tracks as opposed to noise and for counting these as a function of linear plate distance, thereby determining the particle-energy distribution.

References: D. Hodges, Photographic Scanning Systems in the Applied Mathematics Division, ANL, AEC Symposium Series No. 10 Use of Computers in Analysis of Experimental Data and the Control of Nuclear Facilities, CONF660527, Argonne, May 1967, p. 182.

J. R. Erskine, R. H. Vonderohe, M. D. Machalek, and N. N. Sobol, Machine Counting of Tracks in Nuclear Emulsions, Physical Research Monthly Report for March 1967.

4. ARCADE (ARgonne Computer Aided Diffraction Equipment) is a computer-controlled system for acquisition and analysis of data from either X-ray or neutron diffractometers. The system uses an IBM 1130, with paper tape input-output, chosen primarily for its disk-file capability, and is designed to operate either as an isolated control system or as a member of an on-line data network. The control system is intended for use with four-circle, or single-crystal diffractometers; two-circle, or heavy-duty diffractometers; or the four-circle goniometers used in X-ray work. Control of the rotational axes is under program control, and up to six motors can be accommodated by the drive program. In addition, the shutters, environmental equipment, a rate meter, and a chart recorder can be either program-controlled or initiated by typewriter command, or console switch operation.

Basic to the system is the control program written for the 1130, which permits selection of main programs from the 1130 disk library by console typewriter command.

References: R. Aschenbrenner, Computer-Controlled Diffractometer Equipment, AEC Symposium Series No. 10 Use of Computers in Analysis of Experimental Data and the Control of Nuclear Facilities, CONF660527, Argonne, May 1967, p. 67.

L. W. Amiot and J. Becker, Stepping Motor Positioning System for the ARCADE Control System, ANL-AMD Technical Memorandum 143 (Aug 1967).

R. A. Aschenbrenner, Real Time Monitor for the ARCADE Control System, ANL-AMD Technical Memorandum 132 (March 1967).

5. POLLY is a high-precision film-scanning and -measuring device for 70-mm bubble-chamber film. The system uses two CRT scanners, one for the conventional television flying-spot scan and the other for precision measuring. In normal operation, the scanning CRT executes the flying-spot scan and displays the results on a monitor screen for the operator. Also on this screen is a marker under operator control. The operator can then request the computer to measure. The system measures by means of a high-speed minor-line scan with a fixed number of possible orientations. A 4096 x 4096 positioning system is used for this with a variable line-segment length. A DEC PDP-7 is used as the control computer, and the system design includes extensive human intervention and direction facilities. POLLY-2, currently under development, will use the SDS Sigma-7 computer.

Reference: D. Hodges, R. H. Wehman, and G. Wittmus, POLLY, A High Precision Film Scanning and Measuring System, ANL-AMD Technical Memorandum 91 (Dec 1964).

APPENDIX A

POINTR, A Dynamic Storage Allocation Program†

Allen S. Kennedy

1. Introduction

POINTR is a FORTRAN subroutine package developed to alleviate bookkeeping chores associated with the use of dynamic storage allocation techniques. The important features of POINTR are:

a. The program is easy to use. POINTR keeps track of pointer definitions, available storage, etc. The user simply defines the container array, then makes requests for storage and pointers as needed.

b. All references are by array name, making reading and understanding of listings easier. Also, the definition of integer variables for use as pointers is no longer necessary.

c. Automatic purging of storage areas no longer needed by the calling program allows larger problems to fit in fast-core.

d. Dynamic dump facility allows array-status checking at any point in a program during the checkout phase.

The potential user should be aware that the use of dynamic storage in FORTRAN requires structuring programs in subroutine form. A small control routine is used to define a large block of storage (called the container array) and make the appropriate calls to POINTR to control the allocation of storage within this block. Calls to calculational subroutines transmit pointers corresponding to appropriate array locations through the calling sequences. See the Usage section (Section 3) and the sample program in Section 5 for the details of this procedure.

This package has been used successfully in several reactor calculation codes where data requirements demand the use of dynamic storage allocation.

2. Machine

The POINTR package is written for the IBM 360 computer to be used with the FORTRAN IV(H) language.

---

†Originally published for internal distribution only as ANL-AMD Technical Memorandum No. 98 (Feb. 28, 1967).

3. Usage

The POINTR package has been written from a pseudo-language point of view. That is, all POINTR capabilities are accessed through an appropriate CALL to an entry point, subroutine, or function subroutine in POINTR.

a. Initialization

CALL POINTR(ARRAY,LSTLOC,MAXSIZ,IPRINT)

This entry defines the local container array (ARRAY) to POINTR. (ARRAY must be a double-precision variable.) All variable-dimensioned arrays to be referenced through POINTR are packed into the container array. Also defined are the first available location (LSTLOC) in ARRAY, the array length (MAXSIZ), and an optional trace printout flag (IPRINT--set to one for debugging purposes and zero after checkout). POINTR tables and the container array (between the limits LSTLOC and LSTLOC + MAXSIZ) are cleared, and a return is made to the calling program. This entry must be called before the first request for storage.

b. Requesting Storage

CALL PUTPNT (NAME,LENDUM,MULT)

This entry places the array name (NAME)† in the name table, along with its associated length (LENGTH) and a pointer (IPT).

LENDUM is the length of array NAME in the calling program, and LENGTH is the number of double-length words to be reserved in the container storage area. LENGTH is computed as

$$\text{LENGTH} = [(\text{LENDUM}-1)*\text{MULT}+8]/8,^{\dagger\dagger}$$

where

MULT = 2 for nonstandard integer variables,

MULT = 4 for standard real or integer variables,

and

MULT = 8 for nonstandard real variables.

---

†NAME = $\left\{\begin{array}{l}\text{'NAMEbbbb'}\\ \text{8HNAMEbbbb}\\ \text{VNAME}\end{array}\right\}$.

One of three methods is used to transmit the array name (NAME) to POINTR. The first two transmit the Hollerith literal NAMEbbbb directly through the calling sequence. (Note that eight characters must be present.) The third method transmits the FORTRAN double-precision variable (VNAME), which must contain the Hollerith representation of NAMEbbbb. The notation NAME in the calling sequences denotes the selection of one of these methods.

††Formula due to J. V. Zapatka, Applied Mathematics Division.

Then

IPT = LSTLOC,

and

LSTLOC = LSTLOC + LENGTH.

Before array NAME is entered in the name table, a test is made to see if the addition will cause storage overflow. If so, the container array is sifted. That is, arrays with blank names (see Section d) are purged. The storage area is repacked, and the new pointers are printed on the standard output medium. Another attempt is made to insert array NAME in the name table. If storage overflow again occurs, an error condition results. (See Section h for error procedures.)

The following calling sequence can be used to change the length of an array previously defined:

CALL REDEF (NAME,LENDUM,MULT)

This entry searches the name table for the entry corresponding to array NAME. If the new LENGTH is less than the old, the new length is inserted in the table; also, a new entry (with a blank name) corresponding to the storage gap is inserted in the name table. Thus, when more storage is needed, the gap will be eliminated. If the new LENGTH is greater than the old, a new array is defined and data are transferred from the old array to the new. The old array is then given a blank name and will be eliminated when storage is needed.

c. Requesting Pointers

There are two methods of requesting pointers. Each method searches the name table for the entry corresponding to array NAME. The associated pointer is then returned to the calling program. If the name cannot be found in the table, a return is made with a zero pointer.

Method 1

```
CALL SUB (ARRAY(IGET(NAME)))
```

Method 2

```
CALL GETPNT(NAME,IPOINT)
CALL SUB (ARRAY(IPOINT))
```

where SUB is a subroutine where NAME will be used. For example,

```
SUBROUTINE SUB (NAME)
DIMENSION NAME (1)
.
.
.
.
```

d. Array Purging

CALL WIPOUT(NAME)

This entry replaces NAME in the name table with a blank name. A blank name signals that this array is no longer needed and may be purged. No sifting of the container array takes place at this time, however. Only if some future request for storage (via entry PUTPNT) causes overflow will the container array be repacked. The deferred sifting of storage greatly reduces the overhead and frequency of this operation.

CALL PURGE (LAST)

This entry forces the container array to be sifted and eliminates arrays with blank names. Since this operation is time-consuming, this option should be used sparingly (i.e., not in tight loops). LAST contains the pointer in the container array of the first unused storage location.

e. Obtaining Available Storage

LAST=ILAST (DUMMY)

LAST contains the pointer in the container array of the first unused storage location; i.e., MAXSIZ-LAST is the amount of available storage left in the container array.

f. Array Clearing

CALL CLEAR (NAME)

This entry zeros the locations in the container array associated with array NAME.

g. Dynamic Dumps

CALL DUMP(NAME,ITYPE)

This entry causes the contents of the locations in the container array associated with array NAME to be printed on the standard output medium in a format designated by ITYPE. If ITYPE = 1, an integer array

is printed using the FORMAT (12I10). If ITYPE=2, a real array is printed using the FORMAT(8E15.7). Each array dump is preceded by the heading:

DUMP OF ARRAY NAMEbbbb POINTER= LENGTH=

h. Error Procedure

If an error occurs in requesting storage or the pointer for array NAME, one of the following messages is written on the standard output medium:

STORAGE EXCEEDED DURING REQUEST FOR STORAGE OR ARRAY NAME LENGTH=_____LAST AVAILABLE LOCATION=_____

MAXIMUM NUMBER OF ENTRIES IN THE NAME TABLE EXCEEDED

REQUEST FOR POINTER FOR ARRAY NAME HAS BEEN MADE BEFORE REQUEST FOR STORAGE

ARRAY NAME CANNOT BE FOUND IN THE NAME TABLE

An error counter(NPTERR, initially zero) is stepped up by one, and a normal return is made to the calling program. It is the responsibility of the calling program to check for the occurrence of an error in POINTR before the use of pointers. This is done as follows:

```
IF(IPTERR(DUMMY).GT.0)──→ ERROR EXIT
        ↓
NO ERRORS ENCOUNTERED--CONTINUE
```

IPTERR is a function subroutine that returns the value of NPTERR to the calling program. If NPTERR=0, no errors have been encountered in POINTR. If NPTERR>0, NPTERR errors have been encountered. If an error occurs in POINTR, it must be assumed that the program will continue to the next case or terminate. Results will be garbage if normal execution is continued.

Error checking using Method 1 for requesting pointers:

```
        .
        .
        .
      CALL SUB(ARRAY(IGET(NAME)),ϵ10)
      NORMAL RETURN--(Continue problem)
        .
        .
        .
   10 CONTINUE
      ERROR RETURN--(Go to next problem or STOP)
      END
```

```
      SUBROUTINE SUB (NAME,*)
      DIMENSION NAME(1)
            .
            .
            .
      IF(IPTERR(DUMMY).GT.0) RETURN 1 {First executable state-
        ment in SUB}
            .
            .
            .
      RETURN
      END
```

Error checking using Method 2 for requesting pointers:

```
            .
            .
            .
      CALL GETPNT(NAME,IPT)
      IF(IPTERR(DUMMY).GT.0) GO TO 10
      CALL SUB(ARRAY(IPT))
            .
            .
            .
   10 CONTINUE
      ERROR RETURN (GO to next problem or STOP)
      END
```

4. Restrictions

The tables in POINTR currently allow up to 100 entries. This can be easily modified by changing the appropriate dimension statements.

5. Sample Program

This section contains a sample program using all entry points to POINTR, along with sample output and diagrams of the container array at various steps during the operation (Table A.I).

a. Description of Sample Program by Steps

Step No.	Description
1	Initializes POINTR; clears tables and container array A between limits of LSTLOC and MAXSIZ.
2	Size of arrays input as NSIZE.
3-6	Defines storage for arrays X, Y, M, N.
7	Places pointer for array M in IPTM.

Step No.	Description
8	Places pointer for array N in IPTN.
9	Tests for errors in POINTR which may have occurred in steps 3-8.
10	Calls CALC subroutine, which places the integers 1-NSIZE in arrays X, Y, M, N. The variable return is provided for errors that may have occurred in transmitting the pointers for the arrays X and Y. (See subroutine CALC listing in Section b below.)
11-14	Dumps arrays X, Y, M, N.
15	Clears array X.
16	Dumps array X.
17	Places a BLANK name in the POINTR table in place of the array name X. Thus, space occupied by X is available for reuse if storage is needed.
18	Redefines SIZE of array Y to 1/2 NSIZE.
19	Dumps array Y.
20	Redefines SIZE of array Y to 2*NSIZE. Note by the sample output that storage was sifted to allow space for the redefinition of array Y.
21	Dumps array Y.
22	Same as step 17 for array Y.
23	Redefines size of array Z to 2*NSIZE. Since the array name Z has not been defined previously, the effect is the same as CALL PUTPNT('Zbbbbbbb', 2*NSIZE,8) Note again that storage has been sifted to make space for array Z.
24-25	Gets and prints first available location in the container array.
26-27	Same as step 17 for arrays M and N.
28	Forces sifting of container array, and returns first available storage location.

TABLE A.I. Contents of Container Array A at Various Steps in Sample Program

Step 1		Step 6		Step 10		Step 15		Step 18		Step 20		Step 23		Step 28	
Loc	Contents	Loc	Contents	Loc	Contents	Loc	Contents	Loc	Contents	Loc	Contents	Loc	Contents	Loc	Contents
1	0	1	0 0	1	1.0 2.0	1	0 0	1	0 0	1	1.0	1	1 2 3 4	1	0
			X		X		X		Blank		Blank		M		
		25	0 0	25	49.0 50.0	25	0 0	25	0 0	25	25.0	13	47 48 49 50		
		26	0	26	1.0			26	1.0	26	1 2 3 4	14	1 2		
											M		N		
									Y	38	47 48 49 50	38	49 50		
										39	1 2	39	0		Z
			Y		Y						N				
								50	25.0						
								51	26.0						
									Blank	63	49 50				
										64	1.0				
		75	0	75	50.0			75	50.0		Y				
		76	0 0 0 0	76	1 2 3 4	Same as Step 10	Same as Step 10						Z		
			M		M										
		88	0 0 0 0	88	47 48 49 50					88	25.0				
		89	0 0	89	1 2					89	0				
			N		N									100	0
		113	0 0	113	49 50			Same as Step 10	Same as Step 10		Y				
		114	0	114	0										
												138	0		
										163	0				
200	0	200	0	200	0					200	0	209	0	200	0

b. Output from Sample Program

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.103/13.26.27

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

                C     PROGRAM TO TEST POINTR
                C
  ISN 0002            REAL*8 A(200),M
                C
  ISN 0003            DATA M/'M'/
                C
  ISN 0004       1    CALL POINTR(A,1,200,1)
  ISN 0005       2    READ 100,NSIZE
  ISN 0006     100 FORMAT(II10)
  ISN 0007       3    CALL PUTPNT(8HX        ,NSIZE,4)
  ISN 0008       4    CALL PUTPNT(8HY        ,NSIZE,8)
  ISN 0009       5    CALL PUTPNT(8HM        ,NSIZE,2)
  ISN 0010       6    CALL PUTPNT(8HN        ,NSIZE,4)
  ISN 0011       7    CALL GETPNT(M,IPTM)
  ISN 0012       8    CALL GETPNT(8HN        ,IPTN)
  ISN 0013       9    IF(IPTERR(DUMMY).GT.0)GO TO 300
  ISN 0015      10    CALL CALC(A(IGET(8HX        )),A(IGET(8HY        )),
                     1          A(IPTM),A(IPTN),NSIZE,6300)
  ISN 0016      11    CALL DUMP('X        ',2)
  ISN 0017      12    CALL DUMP('Y        ',2)
  ISN 0018      13    CALL DUMP('M        ',1)
  ISN 0019      14    CALL DUMP('N        ',1)
  ISN 0020      15    CALL CLEAR(8HX        )
  ISN 0021      16    CALL DUMP(8HX        ,2)
  ISN 0022      17    CALL WIPOUT(8HX        )
  ISN 0023      18    CALL REDEF(8HY        ,NSIZE/2,8)
  ISN 0024      19    CALL DUMP(8HY        ,2)
  ISN 0025      20    CALL REDEF(8HY        ,2*NSIZE,8)
  ISN 0026      21    CALL DUMP(8HY        ,2)
  ISN 0027      22    CALL WIPOUT(8HY        )
  ISN 0028      23    CALL REDEF(8HZ        ,2*NSIZE,8)
  ISN 0029      24    LAST=ILAST(DUMMY)
  ISN 0030      25    PRINT 200,LAST
  ISN 0031     200 FORMAT('0FIRST AVAILABLE LOCATION IS'I10)
  ISN 0032      26    CALL WIPOUT(8HM        )
  ISN 0033      27    CALL WIPOUT(8HN        )
  ISN 0034      28    CALL PURGE(LAST)
  ISN 0035      29    PRINT 200,LAST
  ISN 0036     300 STOP
  ISN 0037           END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.31.33

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

  ISN 0002          SUBROUTINE  CALC(X,Y,M,N,IMAX,*)
            C
  ISN 0003          REAL*4 X(1)
  ISN 0004          REAL*8 Y(1)
  ISN 0005          INTEGER*2 M(1)
  ISN 0006          INTEGER*4 N(1)
            C
  ISN 0007          IF(IPTERR(DUMMY).GT.0)RETURN 1
  ISN 0009          DO 10 I=1,IMAX
  ISN 0010          X(I)=I
  ISN 0011          Y(I)=I
  ISN 0012          M(I)=I
  ISN 0013          N(I)=I
  ISN 0014       10 CONTINUE
  ISN 0015          RETURN
  ISN 0016          END
```

```
POINTR HAS BEEN INITIALIZED

THE CONTAINER ARRAY HAS BEEN CLEARED FROM LOCATION    1 TO LOCATION  200

ENTRY FOR ARRAY X         HAS BEEN MADE IN THE POINTER TABLE
POINTER=    1 DBL LEN=   25 ORIG LEN=   50 MULT=    4

ENTRY FOR ARRAY Y         HAS BEEN MADE IN THE POINTER TABLE
POINTER=   26 DBL LEN=   50 ORIG LEN=   50 MULT=    8

ENTRY FOR ARRAY M         HAS BEEN MADE IN THE POINTER TABLE
POINTER=   76 DBL LEN=   13 ORIG LEN=   50 MULT=    2

ENTRY FOR ARRAY N         HAS BEEN MADE IN THE POINTER TABLE
POINTER=   89 DBL LEN=   25 ORIG LEN=   50 MULT=    4

POINTER FOR ARRAY M         HAS BEEN EXTRACTED FROM POINTER TABLE
POINTER=         76

POINTER FOR ARRAY N         HAS BEEN EXTRACTED FROM POINTER TABLE
POINTER=         89

POINTER FOR ARRAY X         HAS BEEN EXTRACTED FROM POINTER TABLE
POINTER=          1

POINTER FOR ARRAY Y         HAS BEEN EXTRACTED FROM POINTER TABLE
POINTER=         26

DUMP OF ARRAY X         POINTER=          1 LENGTH=         50
 0.1000000E 01  0.2000000E 01  0.3000000E 01  0.4000000E 01  0.5000000E 01  0.6000000E 01  0.7000000E 01  0.8000000E 01
 0.9000000E 01  0.1000000E 02  0.1100000E 02  0.1200000E 02  0.1300000E 02  0.1400000E 02  0.1500000E 02  0.1600000E 02
 0.1700000E 02  0.1800000E 02  0.1900000E 02  0.2000000E 02  0.2100000E 02  0.2200000E 02  0.2300000E 02  0.2400000E 02
 0.2500000E 02  0.2600000E 02  0.2700000E 02  0.2800000E 02  0.2900000E 02  0.3000000E 02  0.3100000E 02  0.3200000E 02
 0.3300000E 02  0.3400000E 02  0.3500000E 02  0.3600000E 02  0.3700000E 02  0.3800000E 02  0.3900000E 02  0.4000000E 02
 0.4100000E 02  0.4200000E 02  0.4300000E 02  0.4400000E 02  0.4500000E 02  0.4600000E 02  0.4700000E 02  0.4800000E 02
 0.4900000E 02  0.5000000E 02

DUMP OF ARRAY Y         POINTER=         26 LENGTH=         50
 0.1000000D 01  0.2000000D 01  0.3000000D 01  0.4000000D 01  0.5000000D 01  0.6000000D 01  0.7000000D 01  0.8000000D 01
 0.9000000D 01  0.1000000D 02  0.1100000D 02  0.1200000D 02  0.1300000D 02  0.1400000D 02  0.1500000D 02  0.1600000D 02
 0.1700000D 02  0.1800000D 02  0.1900000D 02  0.2000000D 02  0.2100000D 02  0.2200000D 02  0.2300000D 02  0.2400000D 02
 0.2500000D 02  0.2600000D 02  0.2700000D 02  0.2800000D 02  0.2900000D 02  0.3000000D 02  0.3100000D 02  0.3200000D 02
 0.3300000D 02  0.3400000D 02  0.3500000D 02  0.3600000D 02  0.3700000D 02  0.3800000D 02  0.3900000D 02  0.4000000D 02
 0.4100000D 02  0.4200000D 02  0.4300000D 02  0.4400000D 02  0.4500000D 02  0.4600000D 02  0.4700000D 02  0.4800000D 02
 0.4900000D 02  0.5000000D 02

DUMP OF ARRAY M         POINTER=         76 LENGTH=         50
   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24
  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48
  49  50

DUMP OF ARRAY N         POINTER=         89 LENGTH=         50
        1         2         3         4         5         6         7         8         9        10        11        12
       13        14        15        16        17        18        19        20        21        22        23        24
       25        26        27        28        29        30        31        32        33        34        35        36
       37        38        39        40        41        42        43        44        45        46        47        48
       49        50

ARRAY X         HAS BEEN CLEARED
```

```
DUMP OF ARRAY X          POINTER=          1 LENGTH=        50
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0

ARRAY X          HAS BEEN PLACED IN THE PURGE TABLE

ENTRY FOR ARRAY Y          HAS BEEN MADE IN THE POINTER TABLE
POINTER=    26 DBL LEN=    25 ORIG LEN=    25 MULT=     8

DUMP OF ARRAY Y          POINTER=         26 LENGTH=        25
 0.1000000D 01  0.2000000D 01  0.3000000D 01  0.4000000D 01  0.5000000D 01  0.6000000D 01  0.7000000D 01  0.8000000D 01
 0.9000000D 01  0.1000000D 02  0.1100000D 02  0.1200000D 02  0.1300000D 02  0.1400000D 02  0.1500000D 02  0.1600000D 02
 0.1700000D 02  0.1800000D 02  0.1900000D 02  0.2000000D 02  0.2100000D 02  0.2200000D 02  0.2300000D 02  0.2400000D 02
 0.2500000D 02

STORAGE HAS BEEN SIFTED - NEW POINTER TABLE FOLLOWS

      NO      NAME    POINTER    DBL LEN  ORIG LEN      MULT
       1   DUMNAM          1         25        25         8
       2   M              26         13        50         2
       3   N              39         25        50         4

ENTRY FOR ARRAY Y          HAS BEEN MADE IN THE POINTER TABLE
POINTER=    64 DBL LEN=   100 ORIG LEN=   100 MULT=     8

DUMP OF ARRAY Y          POINTER=         64 LENGTH=       100
 0.1000000D 01  0.2000000D 01  0.3000000D 01  0.4000000D 01  0.5000000D 01  0.6000000D 01  0.7000000D 01  0.8000000D 01
 0.9000000D 01  0.1000000D 02  0.1100000D 02  0.1200000D 02  0.1300000D 02  0.1400000D 02  0.1500000D 02  0.1600000D 02
 0.1700000D 02  0.1800000D 02  0.1900000D 02  0.2000000D 02  0.2100000D 02  0.2200000D 02  0.2300000D 02  0.2400000D 02
 0.2500000D 02  0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0            0.0            0.0            0.0            0.0
 0.0            0.0            0.0            0.0

ARRAY Y          HAS BEEN PLACED IN THE PURGE TABLE

ARRAY Z          CANNOT BE FOUND IN THE NAME TABLE
```

```
STORAGE HAS BEEN SIFTED - NEW POINTER TABLE FOLLOWS

      NO      NAME   POINTER   DBL LEN  ORIG LEN      MULT
       1  M                1        13        50         2
       2  N               14        25        50         4

ENTRY FOR ARRAY Z        HAS BEEN MADE IN THE POINTER TABLE
POINTER=   39 DBL LEN=  100 ORIG LEN=  100 MULT=    8

FIRST AVAILABLE LOCATION IS       139

ARRAY M        HAS BEEN PLACED IN THE PURGE TABLE

ARRAY N        HAS BEEN PLACED IN THE PURGE TABLE


STORAGE HAS BEEN SIFTED - NEW POINTER TABLE FOLLOWS

      NO      NAME   POINTER   DBL LEN  ORIG LEN      MULT
       1  Z                1       100       100         8

FIRST AVAILABLE LOCATION IS       101
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.31.45

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

  ISN 0002              SUBROUTINE POINTR(BLK,LSTLOC,MAXSIZ,JPRINT)
                  C
                  C     MANIPULATES POINTERS FOR VARIABLE STORAGE
                  C     ADJUSTS STORAGE IF STORAGE EXCEEDED
                  C     MULT=2  NON STANDARD INTEGER
                  C     MULT=4  STANDARD INTEGER OR REAL
                  C     MULT=8  NON STANDARD REAL
                  C
  ISN 0003              REAL*8 BLK
  ISN 0004              REAL*8 NAME,NAMLST,BLANK
  ISN 0005              REAL NAM1,NAM2
                  C
  ISN 0006              DIMENSION BLK(1)
  ISN 0007              DIMENSION NAM1(2),NAM2(2)
                  C
  ISN 0008              COMMON/LOCATE/LSTBLK,MAXBLK,IPRINT
  ISN 0009              COMMON/TABLES/NAMLST(100),LENLST(100),IPTLST(100),
                       1LEN(100),MLT(100),NNAMS
  ISN 0010              COMMON/NAMBLK/NAME
  ISN 0011              COMMON/PTERR/NPTERR
                  C
  ISN 0012              EQUIVALENCE (NAME,NAM2)
                  C
  ISN 0013              DATA BLANK/'        '/
                  C
  ISN 0014              NNAMS=0
  ISN 0015              NPTERR=0
  ISN 0016              LSTBLK=LSTLOC
  ISN 0017              MAXBLK=MAXSIZ
  ISN 0018              IPRINT=JPRINT
  ISN 0019              CALL PRGSET(BLK)
  ISN 0020              CALL REDSET(BLK)
  ISN 0021              DO 10 I=1,100
  ISN 0022              NAMLST(I)=0
  ISN 0023              IPTLST(I)=0
  ISN 0024              LENLST(I)=0
  ISN 0025              LEN(I)=0
  ISN 0026              MLT(I)=0
  ISN 0027           10 CONTINUE
  ISN 0028              DO 20 I=LSTBLK,MAXBLK
  ISN 0029              BLK(I)=0
  ISN 0030           20 CONTINUE
  ISN 0031              IF(IPRINT.EQ.0)RETURN
  ISN 0033              PRINT 25,LSTBLK,MAXBLK
  ISN 0034           25 FORMAT('1POINTR HAS BEEN INITIALIZED'/
                       1'0THE CONTAINER ARRAY HAS BEEN CLEARED FROM LOCATION'I5,
                       1' TO LOCATION'I5)
  ISN 0035              RETURN
                  C
```

```
ISN 0036          ENTRY PUTPNT(NAM1,LENDUM,MULT)
            C
            C     PUT ARRAY NAME AND POINTER IN TABLE
            C
ISN 0037          NAM2(1)=NAM1(1)
ISN 0038          NAM2(2)=NAM1(2)
ISN 0039          LENGTH=(((LENDUM-1)*MULT+8)/8
ISN 0040          IF(LSTBLK+LENGTH-1.LE.MAXBLK)GO TO 30
            C     STORAGE WILL BE EXCEEDED UNLESS GARBAGE IS CLEARED OUT
ISN 0042          CALL PURGE(DUMMY)
ISN 0043          IF(LSTBLK+LENGTH.GT.MAXBLK)GO TO 1000
ISN 0045       30 CONTINUE
            C     ENOUGH STORAGE - RECORD NAME AND POINTER IN TABLE
ISN 0046          NNAMS=NNAMS+1
ISN 0047          IF(NNAMS.GT.100)GO TO 1020
ISN 0049          NAMLST(NNAMS)=NAME
ISN 0050          IPTLST(NNAMS)=LSTBLK
ISN 0051          LENLST(NNAMS)=LENGTH
ISN 0052          LEN(NNAMS)=LENDUM
ISN 0053          MLT(NNAMS)=MULT
ISN 0054          LSTBLK=LSTBLK+LENGTH
ISN 0055          IF(IPRINT.EQ.0)RETURN
ISN 0057          PRINT 40,NAME,IPTLST(NNAMS),LENLST(NNAMS),LEN(NNAMS),MLT(NNAMS)
ISN 0058       40 FORMAT(17HOENTRY FOR ARRAY A8,35H HAS BEEN MADE IN THE POINTER TAB
                 1LE/9H POINTER=I5,9H DBL LEN=I5,10H ORIG LEN=I5,6H MULT=I5)
ISN 0059          RETURN
            C
ISN 0060          ENTRY WIPOUT(NAM1)
            C
            C     RECORD NAME IN PURGE TABLE
            C
ISN 0061          CALL GETN(NAM1,N)
ISN 0062          IF(N.EQ.0)RETURN
ISN 0064          NAMLST(N)=BLANK
ISN 0065          IF(IPRINT.EQ.0)RETURN
ISN 0067          PRINT 70,NAME
ISN 0068       70 FORMAT(7HOARRAY A8,35H HAS BEEN PLACED IN THE PURGE TABLE)
ISN 0069          RETURN
            C
ISN 0070          ENTRY DUMP(NAM1,JUMP)
            C
            C     DUMP VARIABLE DIMENSION ARRAY ON STANDARD OUTPUT
            C
ISN 0071          CALL GETN(NAM1,N)
ISN 0072          IF(N.EQ.0)RETURN
ISN 0074          PRINT 130,NAME,IPTLST(N),LEN(N)
ISN 0075      130 FORMAT(15HODUMP OF ARRAY A8,9H POINTER=I10,8H LENGTH=I10)
ISN 0076          IPOINT=IPTLST(N)
ISN 0077          GO TO (140,160),JUMP
ISN 0078      140 CONTINUE
```

```
ISN 0079          IF(MLT(N).EQ.4)GO TO 150
ISN 0081          CALL PRTI1(BLK(IPOINT),LEN(N))
ISN 0082          RETURN
ISN 0083      150 CONTINUE
ISN 0084          CALL PRTI2(BLK(IPOINT),LEN(N))
ISN 0085          RETURN
ISN 0086      160 CONTINUE
ISN 0087          IF(MLT(N).EQ.8)GO TO 170
ISN 0089          CALL PRTR1(BLK(IPOINT),LEN(N))
ISN 0090          RETURN
ISN 0091      170 CONTINUE
ISN 0092          CALL PRTR2(BLK(IPOINT),LEN(N))
ISN 0093          RETURN
            C
ISN 0094          ENTRY GETPNT(NAM1,IPT)
            C
            C     GET POINTER CORRESPONDING TO ARRAY NAME
            C
ISN 0095          IPT=0
ISN 0096          CALL GETN(NAM1,N)
ISN 0097          IF(N.GT.0)GO TO 190
ISN 0099          NPTERR=NPTERR+1
ISN 0100          RETURN
ISN 0101      190 CONTINUE
ISN 0102          IPT=IPTLST(N)
ISN 0103          IF(IPRINT.EQ.0)RETURN
ISN 0105          PRINT 200,NAME,IPT
ISN 0106      200 FORMAT(19H0POINTER FOR ARRAY A8,38H HAS BEEN EXTRACTED FROM POINTE
                 1R TABLE/9H POINTER=I10)
ISN 0107          RETURN
            C
ISN 0108          ENTRY CLEAR(NAM1)
            C
            C     CLEARS ARRAY CORRESPONDING TO NAME
            C
ISN 0109          CALL GETN(NAM1,N)
ISN 0110          IF(N.EQ.0)RETURN
ISN 0112          LENGTH=LENLST(N)
ISN 0113          IPT=IPTLST(N)
ISN 0114          DO 230 I=1,LENGTH
ISN 0115          BLK(IPT+I-1)=0
ISN 0116      230 CONTINUE
ISN 0117          IF(IPRINT.EQ.0)RETURN
ISN 0119          PRINT 240,NAME
ISN 0120      240 FORMAT(7H0ARRAY A8,17H HAS BEEN CLEARED)
ISN 0121          RETURN
            C
            C     PRINT ERROR MESSAGES
            C
ISN 0122     1000 CONTINUE
```

```
ISN 0123          NPTERR=NPTERR+1
ISN 0124          PRINT 1010,NAME,LENGTH,LSTBLK
ISN 0125     1010 FORMAT(54H0STORAGE EXCEEDED DURING REQUEST FOR STORAGE OF ARRAY A8
                 1,/8H LENGTH=I10,25H LAST AVAILABLE LOCATION=I10)
ISN 0126          RETURN
ISN 0127     1020 CONTINUE
ISN 0128          NPTERR=NPTERR+1
ISN 0129          PRINT 1030
ISN 0130     1030 FORMAT(49H0MAXIMUM NUMBER OF ENTRIES IN NAME TABLE EXCEEDED)
ISN 0131          RETURN
ISN 0132          END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.32.13

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

   ISN 0002          SUBROUTINE GETN(NAM1,N)
              C
   ISN 0003          REAL*8 NAME,NAMLST
   ISN 0004          REAL NAM1,NAM2
              C
   ISN 0005          DIMENSION NAM1(2),NAM2(2)
              C
   ISN 0006          COMMON/TABLES/NAMLST(100),LENLST(100),IPTLST(100),
                    1LEN(100),MLT(100),NNAMS
   ISN 0007          COMMON/NAMBLK/NAME
              C
   ISN 0008          EQUIVALENCE (NAME,NAM2)
              C
   ISN 0009          N=0
   ISN 0010          NAM2(1)=NAM1(1)
   ISN 0011          NAM2(2)=NAM1(2)
   ISN 0012          IF(NNAMS.EQ.0)GO TO 20
   ISN 0014          DO 10 N=1,NNAMS
   ISN 0015          IF(NAMLST(N).EQ.NAME)RETURN
   ISN 0017       10 CONTINUE
   ISN 0018       20 CONTINUE
   ISN 0019          PRINT   30,NAME
   ISN 0020       30 FORMAT(7H0ARRAY A8,34H CANNOT BE FOUND IN THE NAME TABLE)
   ISN 0021          RETURN
   ISN 0022          END
```

COMPILER OPTIONS - NAME= MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

```
ISN 0002          SUBROUTINE REDSET(BLK)
            C
            C     REDEFINES ARRAY NAME
            C
ISN 0003          REAL*8 BLK
ISN 0004          REAL*8 NAME,NAMLST,BLANK,DUMNAM
ISN 0005          REAL NAM1,NAM2
            C
ISN 0006          DIMENSION BLK(1)
ISN 0007          DIMENSION NAM1(2),NAM2(2),DUM(2)
            C
            C
ISN 0008          COMMON/LOCATE/LSTBLK,MAXBLK,IPRINT
ISN 0009          COMMON/TABLES/NAMLST(100),LENLST(100),IPTLST(100),
                 1LEN(100),MLT(100),NNAMS
ISN 0010          COMMON/NAMBLK/NAME
ISN 0011          COMMON/PTERR/NPTERR
            C
            C
ISN 0012          EQUIVALENCE (NAME,NAM2),(DUMMY,DUM(1))
            C
ISN 0013          DATA BLANK/'        '/,DUMNAM/'DUMNAM'/
            C
ISN 0014          RETURN
            C
ISN 0015          ENTRY REDEF(NAM1,LENDUM,MULT)
            C
ISN 0016          CALL GETN(NAM1,N)
ISN 0017          IF(N.GT.0)GO TO 30
            C
            C     NAME NOT IN TABLE SO PUT IT THERE
            C
ISN 0019          CALL PUTPNT(NAME,LENDUM,MULT)
ISN 0020          RETURN
ISN 0021       30 CONTINUE
ISN 0022          LENNEW=((LENDUM-1)*MULT+8)/8
ISN 0023          LENOLD=LENLST(N)
ISN 0024          IF(LENNEW.EQ.LENOLD)RETURN
ISN 0026          IF(LENNEW.GT.LENOLD)GO TO 60
            C
            C     NEW LENGTH LESS THAN OLD LENGTH
            C
ISN 0028          NNAMS=NNAMS+1
ISN 0029          IF(NNAMS.LE.100)GO TO 50
ISN 0031          NPTERR=NPTERR+1
ISN 0032          PRINT 40
ISN 0033       40 FORMAT(49H0MAXIMUM NUMBER OF ENTRIES IN NAME TABLE EXCEEDED)
ISN 0034          RETURN
ISN 0035       50 CONTINUE
```

```
ISN 0036          L1=NNAMS-N-1
ISN 0037          DO 55 L=1,L1
ISN 0038          K=NNAMS-L
ISN 0039          NAMLST(K+1)=NAMLST(K)
ISN 0040          IPTLST(K+1)=IPTLST(K)
ISN 0041          LENLST(K+1)=LENLST(K)
ISN 0042          LEN   (K+1)=LEN   (K)
ISN 0043          MLT   (K+1)=MLT   (K)
ISN 0044       55 CONTINUE
ISN 0045          LENLST(N)=LENNEW
ISN 0046          LEN(N)=LENDUM
ISN 0047          MLT(N)=MULT
ISN 0048          NAMLST(N+1  )=BLANK
ISN 0049          IPTLST(N+1  )=IPTLST(N)+LENNEW
ISN 0050          LENLST(N+1  )=LENOLD-LENNEW
ISN 0051          IF(IPRINT.EQ.0)RETURN
ISN 0053          PRINT 56,NAME,IPTLST(N    ),LENLST(N    ),LEN(N    ),MLT(N    )
ISN 0054       56 FORMAT(17HOENTRY FOR ARRAY A8,35H HAS BEEN MADE IN THE POINTER TAB
                 1LE/9H POINTER=I5,9H DBL LEN=I5,10H ORIG LEN=I5,6H MULT=I5)
ISN 0055          RETURN
ISN 0056       60 CONTINUE
            C
            C     NEW LENGTH GREATER THAN OLD LENGTH
            C
ISN 0057          NAMLST(N)=DUMNAM
ISN 0058          CALL PUTPNT(NAME,LENDUM,MULT)
ISN 0059          IF(NAMLST(N).EQ.DUMNAM)GO TO 90
ISN 0061          CALL GETN(DUMNAM,N)
ISN 0062          IF(N.GT.0)GO TO 90
ISN 0064          NPTERR=NPTERR+1
ISN 0065          RETURN
ISN 0066       90 CONTINUE
ISN 0067          NAMLST(N)=BLANK
ISN 0068          DO 100 L=1,LENOLD
ISN 0069          LSUB1=IPTLST(N)+L-1
ISN 0070          LSUB2=IPTLST(NNAMS)+L-1
ISN 0071          BLK(LSUB2)=BLK(LSUB1)
ISN 0072      100 CONTINUE
ISN 0073          RETURN
ISN 0074          END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.32.44

        COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

 ISN 0002            SUBROUTINE PRGSET(BLK)
                C
                C    SIFTS STORAGE AND ELIMINATES ARRAYS IN THE PURGE TABLE
                C
 ISN 0003            REAL*8 BLK
 ISN 0004            REAL*8      NAMLST,BLANK
                C
 ISN 0005            DIMENSION BLK(1)
                C
 ISN 0006            COMMON/LOCATE/LSTBLK,MAXBLK,IPRINT
 ISN 0007            COMMON/TABLES/NAMLST(100),LENLST(100),IPTLST(100),
                    1LEN(100),MLT(100),NNAMS
                C
 ISN 0008            DATA BLANK/'        '/
                C
 ISN 0009            RETURN
                C
 ISN 0010            ENTRY PURGE(LSTLOC)
                C
 ISN 0011            IF(NNAMS.GT.0)GO TO 5
 ISN 0013            LSTLOC=LSTBLK
 ISN 0014            RETURN
 ISN 0015          5 CONTINUE
 ISN 0016            IMOV=1
 ISN 0017            INAM=0
 ISN 0018            DO 40 N=1,NNAMS
 ISN 0019            IF(NAMLST(N).EQ.BLANK    )GO TO 40
 ISN 0021         10 CONTINUE
                C    NAME NOT BLANK          - SO SAVE IT
 ISN 0022            IMAX=LENLST(N)
 ISN 0023            IF(IMOV.EQ.IPTLST(N))GO TO 30
 ISN 0025            DO 20 I=1,IMAX
 ISN 0026            ISUB1=IMOV+I-1
 ISN 0027            ISUB2=IPTLST(N)+I-1
 ISN 0028            BLK(ISUB1)=BLK(ISUB2)
 ISN 0029         20 CONTINUE
 ISN 0030         30 CONTINUE
 ISN 0031            INAM=INAM+1
 ISN 0032            NAMLST(INAM)=NAMLST(N)
 ISN 0033            IPTLST(INAM)=IMOV
 ISN 0034            LENLST(INAM)=IMAX
 ISN 0035            LEN(INAM)=LEN(N)
 ISN 0036            MLT(INAM)=MLT(N)
 ISN 0037            IMOV=IMOV+IMAX
 ISN 0038         40 CONTINUE
 ISN 0039            NNAMS=INAM
 ISN 0040            LSTBLK=IMOV
 ISN 0041            LSTLOC=LSTBLK
 ISN 0042            DO 45 I=LSTBLK,MAXBLK
```

```
ISN 0043          BLK(I)=0.
ISN 0044       45 CONTINUE
ISN 0045          IF(IPRINT.EQ.0)RETURN
ISN 0047          PRINT 50
ISN 0048       50 FORMAT(52H1STORAGE HAS BEEN SIFTED - NEW POINTER TABLE FOLLOWS/
                 11H07X2HNO6X4HNAME3X7HPOINTER3X7HDBL LEN2X8HORIG LEN6X4HMULT)
ISN 0049          DO 70 N=1,NNAMS
ISN 0050          PRINT 60,N,NAMLST(N),IPTLST(N),LENLST(N),LEN(N),MLT(N)
ISN 0051       60 FORMAT(1I10,2XA8,4I10)
ISN 0052       70 CONTINUE
ISN 0053          RETURN
ISN 0054          END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.33.01

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

ISN 0002          FUNCTION IGET(NAM1)
            C
            C     GET POINTER CORRESPONDING TO ARRAY NAME
            C
ISN 0003          REAL*8            NAME,NAMLST,BLANK
ISN 0004          REAL NAM1,NAM2
            C
ISN 0005          DIMENSION NAM1(2),NAM2(2)
            C
ISN 0006          COMMON/LOCATE/LSTBLK,MAXBLK,IPRINT
ISN 0007          COMMON/TABLES/NAMLST(100),LENLST(100),IPTLST(100),
                 1LEN(100),MLT(100),NNAMS
ISN 0008          COMMON/NAMBLK/NAME
ISN 0009          COMMON/PTERR/NPTERR
            C
ISN 0010          EQUIVALENCE (NAME,NAM2)
            C
ISN 0011          IPT=0
ISN 0012          CALL GETN(NAM1,N)
ISN 0013          IF(N.GT.0)GO TO 10
ISN 0015          NPTERR=NPTERR+1
ISN 0016          RETURN
ISN 0017       10 CONTINUE
ISN 0018          IGET=IPTLST(N)
ISN 0019          IF(IPRINT.EQ.0)RETURN
ISN 0021          PRINT 60,NAME,IPTLST(N)
ISN 0022       60 FORMAT(19H0POINTER FOR ARRAY A8,38H HAS BEEN EXTRACTED FROM POINTE
                 1R TABLE/9H POINTER=I10)
ISN 0023          RETURN
ISN 0024          END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.33.15

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

  ISN 0002              SUBROUTINE PRTI1 (J,LENGTH)
                  C
                  C     PRINTS NON STANDARD INTEGER ARRAY
                  C
  ISN 0003              INTEGER J*2
                  C
  ISN 0004              DIMENSION J(1)
                  C
  ISN 0005              PRINT 10,(J(L),L=1,LENGTH)
  ISN 0006           10 FORMAT(24I5)
  ISN 0007              RETURN
  ISN 0008              END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.33.26

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

  ISN 0002              SUBROUTINE PRTI2 (J,LENGTH)
                  C
                  C     PRINTS STANDARD INTEGER ARRAY
                  C
  ISN 0003              DIMENSION J(1)
                  C
  ISN 0004              PRINT 10,(J(L),L=1,LENGTH)
  ISN 0005           10 FORMAT(12I10)
  ISN 0006              RETURN
  ISN 0007              END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.33.38

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

  ISN 0002              SUBROUTINE PRTR1 (A,LENGTH)
                  C
                  C     PRINTS STANDARD REAL ARRAY
                  C
  ISN 0003              DIMENSION A(1)
                  C
  ISN 0004              PRINT 10,(A(L),L=1,LENGTH)
  ISN 0005           10 FORMAT(8E15.7)
  ISN 0006              RETURN
  ISN 0007              END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.33.49

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

   ISN 0002            SUBROUTINE PRTR2 (A,LENGTH)
               C
               C       PRINTS NON STANDARD REAL ARRAY
               C
   ISN 0003            REAL A*8
               C
   ISN 0004            DIMENSION A(1)
               C
   ISN 0005            PRINT 10,(A(L),L=1,LENGTH)
   ISN 0006         10 FORMAT(8E15.7)
   ISN 0007            RETURN
   ISN 0008            END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.34.01

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

   ISN 0002            FUNCTION IPTERR(DUMMY)
               C
               C       RETURNS NUMBER OF ERRORS THAT HAVE OCCURED IN POINTER ROUTINES
               C
   ISN 0003            COMMON/PTERR/NPTERR
               C
   ISN 0004            IPTERR=NPTERR
   ISN 0005            NPTERR=0
   ISN 0006            RETURN
   ISN 0007            END
```

```
LEVEL 18 NOV 66  ARGONNE NATIONAL LABORATORY    OS/360  FORTRAN H  CONTROLLED RELEASE VERSION 4   DATE  67.102/16.34.12

          COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=50,SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID

   ISN 0002            FUNCTION ILAST(DUMMY)
               C
               C       RETURNS FIRST AVAILABLE LOCATION IN CONTAINER ARRAY
               C
   ISN 0003            COMMON/LOCATE/LSTBLK,MAXBLK,IPRINT
               C
   ISN 0004            ILAST=LSTBLK
   ISN 0005            RETURN
   ISN 0006            END
```

APPENDIX B

A Disk-file Control System for the Control Data 3600†

George A. Robinson, Ronald F. Krupp, and Donald Jordan

1. Introduction

This appendix describes the disk-file control system being implemented on Argonne National Laboratory's Control Data 3600 (SCOPE Monitor) to allow use of the CDC 3632-828 disk-file system.

The Model 828 disk file consists of 16 physical disks, of which eight are directly available to the user and eight are used by the operating system for standard input-output purposes. Each disk has a positioner with eight read/write heads. There are 64 positions for each positioner, giving a total of 512 tracks per disk. For each position, four of the tracks are inner tracks (twelve 32-work blocks each) and four are outer tracks (twenty 32-word blocks each) yielding 128 blocks (4096 48-bit words), or tallying 8192 blocks (256 bK†† words) per disk. The eight-disk user portion thus has a capacity of 62 bK†† blocks (2^{21} or over 2 million words).

The computer-to-disk access time is equal to the sum of the positioner switching, positioning, confirmation, and latency times. The maximum is 373 msec, and the average is 225 msec. Interference arises from system use of the single control available to the 3600.

a. Positioner switching time is the interval between a power-off pulse to one positioner and a power-on pulse to another. This time does not exceed 23 msec, but must be added to the access time if sequential switching of positioners takes place. The user has control over this if and only if he is not using standard input-output; otherwise this interference will occur at unpredicted moments.

b. Positioning time is the interval between the time power is turned on for a positioner and the time its first block address is correctly read. The most important factor in determining positioning time is the number of positions the arm must move. The maximum is 250 msec, and the average is 145 msec.

c. Confirmation time is the interval between positioning time and the time when enough address headers have been read to ensure that the positioner is on tracks. This time is not less than 80% of one disk revolution (approximately 40 msec).

---

†Originally published for internal distribution only as ANL-AMD Technical Memorandum No. 122 (Oct 1966).
††bK denotes 1024.

d. Latency time is the interval between confirmation time and the time when the addressed block arrives under the head. This is a maximum of one revolution of the disk or 52 msec; the average time is 28 msec.

Data-bit transfer rates are about 418 kc (114 μsec per 48-bit word) for inner tracks and 698 kc (68 μsec per 48-bit word) for outer tracks.

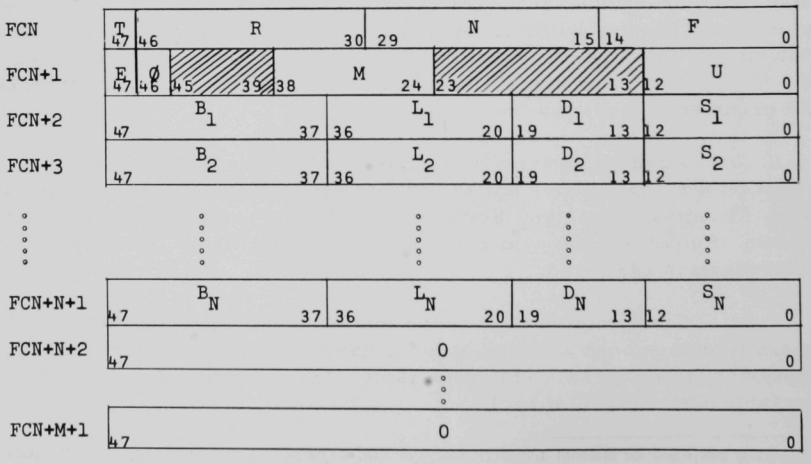
2. Organization of User Data on the Disk

The user is expected to think of the disk storage area as eight non-consecutive intervals of 8192 blocks (256 bK words). Each of these intervals is called a logical disk. Two methods of disk use are available--tape-like and random access. The tape-like mode is for compatible use of either disk or tape. Operations that are physically impossible on tape are not allowed on the disk, e.g., writing in the middle of a file. Similarly, operations that are undefined for a random-access device are illegal in that mode, e.g., write end-of-file.

For each SCOPE logical unit defined on the disk, the user must provide the system with a description of his organization of the data for that unit. This description is called a file complex control array (FCCA).

A logical record consists of one or more consecutive 32-word blocks on the disk. A logical file consists of one or more logical records of equal size. A file complex is an arbitrary collection of one or more logical files assigned to the SCOPE logical unit.

The parameters in the FCCA are defined below. Their positions in core storage are shown in the diagram below.



In the above diagram, the address of FCN is called the file complex name.

T = tape-like/random-access flag. This flag indicates in which device mode the disk is used.

$$T = \begin{cases} 0, \text{ tape-like device} \\ 1, \text{ random-access device.} \end{cases}$$

R = logical record pointer, an integer one less than the ordinal of the next record to be processed.†

N = number of logical files currently defined in the file complex.

F = logical file pointer (initially F=1, pointing to the first logical file).

E = end-of-tape flag;

$$E = \begin{cases} 0, \text{ not end of tape} \\ 1, \text{ end of tape.} \end{cases}$$

ϕ = last-operation flag;

$$\phi = \begin{cases} 0, \text{ last operation was a read type} \\ 1, \text{ last operation was a write type.} \end{cases}$$

M = maximum number of logical files that may be defined in the complex.

U = number of blocks reserved for the file complex, but not assigned to a logical file.

B_i = number of blocks per logical record in the ith file.

L_i = number of logical records in the ith file.

D_i = physical disk number; $8 \leq D_i \leq 15$, out-of-range values cause job abortion.

S_i = block address of the first block in the ith file.

If the disk is used as a random-access device (T=1), then M, the maximum number of logical files that may be defined, is not used.

---

†R ordinarily is zero initially.

The file pointer and record pointer indicate the current logical file within the file complex and the logical record within that file. This position changes in an abstract sense only, since mechanical movement is not necessarily associated with a pointer change. The disk itself is not accessed by the system until a data-transmission request is issued by the user (e.g., READ, WRITE). Then the positioner arm will be moved to block address $R*B_F+S_F$ on disk D_F.

The end-of-tape and last-operation flags are used to ensure tape compatibility; their initial values are usually 0.

3. Using the Disk File

a. Defining a Logical Unit on the Disk File

To define a logical unit on the disk file, the user must insert a SCOPE EQUIP control card

$^{7}_{9}$EQUIP, ℓ=DF

where ℓ = logical unit number ($1 \leq \ell < 49$). This card should be placed with other EQUIP cards near the beginning of the job deck. More than one logical unit may be defined on the disk. Example:

$^{7}_{9}$JØB

Accounting card

$^{7}_{9}$EQUIP, 3=DF

$^{7}_{9}$EQUIP, 41=DF

b. Defining and Creating a File Complex Control Array

Before using a logical unit defined on the disk, the user must define the file complex control array. Two standard methods are available. One is FORTRAN-oriented; the definition is made by a CALL DFDEFINE statement. The array is created in "free" storage--storage not occupied by data or program (see 1 below). The other method is for the user who wants complete control over the pointers and the array. Here the user must allocate storage and supply parameters. He is free to make dynamic changes to the array parameters (see 4 below).

1) The following FORTRAN statement defines the file complex control array:

```
CALL DFDEFINE(LU,LD,KT,...)
```

where

LU = logical unit number;

LD = logical disk number ($1 \leq LD \leq 8$),†
(If LD is out of range, the job will be aborted through a Q8QERROR call with the message "ILLEGAL LOGICAL DISK.")

KT = tape-like/random-access device flag and if

$$KT = \begin{cases} 0, & \text{tape-like device} \\ 1, & \text{random-access device.} \end{cases}$$

The remaining parameters in the argument list depend on the value of KT.

a) If KT=0 (tape-like device), the remaining parameters are

,...,MAXNF,MAXBL,KB_1).

b) If KT=1 (random-access device), the remaining parameters are

,...,NF,MAXBL,KB_1,LR_1,...,KB_J,LR_J).

where

MAXNF is M, the maximum number of logical files that may be defined in this complex.

MAXBL is the maximum number of blocks in the file complex.

KB_i is B_i, the number of blocks per logical record in the ith file.

NF is N, the number of logical files to be defined in the file complex.

LR_i is L_i, the number of logical records in the ith file.

If too many parameters are given for a random-access device (KT=1 and $J > NF$) or not enough disk space is allocated,

†The subroutine maps LD onto a physical disk automatically.

$$\sum_{i=1}^{J} KB_i * LR_i > MAXBL,$$

the job will be aborted by a Q8QERROR call with the diagnostic message "ILLEGAL DFDEFINE."

If not enough "free" storage is available to define the file complex control array, the job will be aborted through a Q8QERROR call with the message "NO MEM AVAILABLE FOR FCCA."

If not enough disk space is available the job will be aborted through a Q8QERROR call with the message "DISK SPACE EXCEEDED." DFDEFINE may be used to redefine the file complex on a logical unit. But all the information contained in the preceding file complex control array will be lost.

Examples

a) To define a file complex control array for a logical unit number 3 on logical disk number 2, so that

i. The disk is to be treated as a tape-like device,

ii. The maximum number of files that may be defined is 4,

iii. The maximum number of blocks is 8192 (note that this is an entire disk), and

iv. The number of blocks per logical record is 8 (256 words),

one would execute the FORTRAN statement

CALL DFDEFINE(3,2,0,4,8192,8).

Below is the file complex control array that would be generated for this call (note that the number of logical records in the first file is 8192/8 = 1024, and the logical disk number 2 is mapped onto physical disk number 9):

FCN	0	0							1					1		
	47	46						30	29				15	14		0
FCN+1	0	0	0			4				0				0		
	47	46	45		39	38			24	23			13	12		0
FCN+2	8							1024				9			0	
	47				37	36			20	19		13		12		0
FCN+3	0															
	47															0
FCN+4	0															
	47															0
FCN+5	0															
	47															

b) To define a file complex control array for logical unit number 4 on logical disk number 6, so that

i. The disk is to be treated as a random-access device,

ii. The number of files to be defined is 3,

iii. The maximum number of blocks is 7000,

iv. The number of blocks per logical record and the number of logical records for the first file are 10 and 200, respectively,

v. The number of blocks per logical record and the number of logical records for the second file, are 20 and 50, respectively, and

vi. The number of blocks per logical record and the number of logical records for the third file are 30 and 100, respectively,

one would execute the FORTRAN statement

CALL DFDEFINE(4,6,1,3,7000,10,200,20,50,30,100).

Below is the file complex control array that would be generated for this call (note that logical disk number 6 is mapped onto physical disk number 13):

FCN	1 (47)	0 (46–30)	3 (29–15)	1 (14–0)			
FCN+L	0 (47)	0 (46)	0 (45–39)	3 (38–24)	0 (23–13)	1000 (12–0)	
FCN+2	10 (47–37)	200 (36–20)	13 (19–13)	0 (12–0)			
FCN+3	20 (47–37)	50 (36–20)	13 (19–13)	2000 (12–0)			
FCN+4	30 (47–37)	100 (36–20)	13 (19–13)	3000 (12–0)			

2) In the tape-like mode, the user may want to change the blocking factor (number of blocks per logical record) in a new file. A CALL RDFBLSIZ FORTRAN statement is used and is legal only immediately after a request to mark end-of-file. If the current blocking factor (that of the file just ended) is satisfactory, no use of RDFBLSIZ is necessary. The FORTRAN statement

CALL RDFBLSIZ(LU,KB)

redefines the blocking factor B in the current file. Here

LU = logical unit number,

and

KB = number of blocks per logical record.

The number of logical records in the file will be

$$\left[\frac{B_i * L_i}{KB}\right],$$

where [x] is the largest integer not exceeding x. KB replaces B_i; any extra blocks are added to U, the number of blocks unassigned.

If CALL RDFBLSIZ(LU,KB) is other than the first statement after an END FILE LU statement, the job will be aborted through a Q8QERROR call with the diagnostic message "ILLEGAL REDEFINE FILE."

Example

Assume that the user has defined a file complex control array as described in example 1a above, and has written 100 logical records and then marked an end-of-file. Further, assume that the user executes the FORTRAN statement

CALL RDFBLSIZ (3,16)

to change the blocking factor from 8 to 16. The following two diagrams show the file complex control array before and after the execution of this statement.

a) Before

FCN	0 (47)	0 (46–30)			2 (29–15)		2 (14–0)
FCN+1	0 (47)	1 (46)	0 (45–39)	4 (38–24)		0 (23–13)	4 (12–0)
FCN+2	8 (47–37)			100 (36–20)		9 (19–13)	0 (12–0)
FCN+3	8 (47–37)			115 (36–20)		9 (19–13)	800 (12–0)
FCN+4	0 (47–0)						
FCN+5	0 (47–0)						

b) After

FCN	0 (47)	0 (46–30)			2 (29–15)		2 (14–0)
FCN+1	0 (47)	1 (46)	0 (45–39)	4 (38–24)		0 (23–13)	12 (12–0)
FCN+2	8 (47–37)			100 (36–20)		9 (19–13)	0 (12–0)
FCN+3	16 (47–37)			57 (36–29)		9 (19–13)	800 (12–0)
FCN+4	0 (47–0)						
FCN+5	0 (47–0)						

3) The user may wish to write a file in the tape-like mode, then read or edit it in a random-access manner. To change the tape-like/random-access device flag, one uses the FORTRAN statement

CALL REDEFKT(LU,KT),

where

LU = logical unit number,

and

$$KT = \begin{cases} 0, \text{ tape-like device} \\ 1, \text{ random-access device.} \end{cases}$$

Example

The FORTRAN statement

CALL REDEFKT(3,1)

will set the tape-like/random-access device flag in the file complex control array defined for logical unit number 3 to indicate that the disk is to be treated as a random-access device.

4) The FORTRAN statement

CALL CREATE(LU,LD,FCN)

is used to tell the subroutine that the user has constructed a file complex control array for a logical unit on a logical disk. Again,

LU = logical unit,

LD = logical disk number,

and

FCN = file complex name.

If LD is out of range, $1 \leq LD \leq 8$, then the job will be aborted through a Q8QERROR with the message "ILLEGAL LOGICAL DISK."

A COMPASS-coded routine may execute a SCOPE REØT request (see SCOPE Calls and Driver Specifications, Section 4a below) to create a file complex control array.

Example

The FORTRAN statement

```
CALL CREATE(5,4,FILECCA)
```

informs the subroutine that the user has constructed the file complex control array whose file complex name FILECCA, for logical unit number 5 on logical disk number 4.

Note: Calls to DFDEFINE, RDFBLSIZ, REDEFKT, or CREATE when the logical unit is not a disk are ignored. Programmers using only tape-like files can thus use disk or tape by inserting or omitting EQUIP cards.

c. Changing Pointer Positions

The following FORTRAN statements will change the pointers without accessing the disk file. File marks will be treated in a manner analogous to those of tape; i.e., an end-of-file is a record.

1) REWIND i,

where i = logical unit number, will set the file pointer to 1 and the record pointer to 0.

2) BACKSPACE i,

where i = logical unit number, will decrease the record pointer by one unless the unit was written through a FORTRAN WRITE TAPE statement. In this case, the pointer is changed by the appropriate number.†

†See WRITE (binary) in the 3600 FORTRAN Reference Manual for details.

3) CALL BACKFILE (i),

where i = logical unit number, will decrease the file pointer by one and set the record pointer to point to the end of the file. If the file pointer is 1, then the record pointer is set to 0 to point to the first logical record in the first logical file.

4) CALL SKIPFILE (i),

where i = logical unit number, will increase the file pointer by one and set the record pointer to 0, to point to the first logical record in the next logical file.

5) CALL SKIPLREC (i),

where i is the logical unit number, will increase the record pointer by one unless the unit was written through a FORTRAN WRITE TAPE statement; the appropriate change is made in this case.†

d. Data Transmission

Normal FORTRAN input-output procedures can be carried out after the file complex control array is defined. The following restrictions must be observed to match the disk's structure with FORTRAN conventions.

In the following FORTRAN statements,

i = logical unit number,

n = FORMAT statement number,

L = argument list,

and

p = parity mode,

where

$$p = \begin{cases} 0, \text{ even parity (BCD mode)} \\ 1, \text{ odd parity (binary mode).} \end{cases}$$

†See WRITE (binary) in the 3600 FORTRAN Reference Manual for details.

1) Read/Write in BCD Mode

The FORTRAN statements

READ (i,n) L

READ INPUT TAPE i,n,L

WRITE (i,n) L

WRITE OUTPUT TAPE i,n,L

are requests to transmit data in BCD format. The maximum record length in BCD mode is 136 characters (17 words).

To use BCD I/O operations on the disk file, each logical record in the file complex must be one block (32 words) long. Try to avoid BCD I/O on the disk, as this is inefficient.

2) Read/Write in Binary Mode

The FORTRAN statements

READ (i)L

READ TAPE i,L

WRITE (i)L

WRITE TAPE i,L

are requests to transmit one FORTRAN logical record in binary format. Such a record consists of one or more physical records,† depending on the number of words to be transmitted. The maximum size of a physical record† is 256 words--255 data words and one control word used by the input-output processor. The last physical record† may be 2 to 256 words long. For example, assume that 260 words are to be transmitted, the processor will transmit two physical records,† the first 256 words long, and the second 6 words long.

To use binary I/O operations on the disk, each logical record must be defined in the file complex to be eight blocks (256 words) long.

---

† A "physical record" referred to here is a 256-word block on tape; this entity becomes eight 32-word blocks on disk.

3) Buffering

The FORTRAN statement

BUFFER IN (i,p) (A,B)

will transmit one physical record to storage locations A through B when using a tape. When the unit is a disk, the number of words transmitted is precisely that amount required to fill the buffer A through B, irrespective of the logical record size defined in the file complex control array. No matter how many words are transmitted, the record pointer is increased by one.

The FORTRAN statement

BUFFER OUT (i,p) (A,B)

will transmit one record from storage locations A through B. When the unit is a disk, the number of words written is precisely the amount requested by the call, irrespective of the logical record size defined in the file complex control array. No matter how many words are written, the record pointer is increased by one. The size of the buffer should not exceed the size of the logical record defined in the file complex control array; otherwise, data may be lost.

To use buffering operations on the disk, each logical record in the file complex must be a multiple of a block (32 words) long.

Note: The FORTRAN LENGTHF is useless in BUFFER IN, BUFFER OUT operations since the number of words transmitted is purely a function of the space allocated by the user in his BUFFER statement. This is a basic incompatibility between disk and tape.

e. Marking an End-of-File

The FORTRAN statement

ENDFILE i,

where i = logical unit number, will mark an end-of-file on logical unit i.

When one marks an end-of-file in a file complex, the system checks to see if the logical unit is a tape-like device (T=0, in the file complex control array). If it is, the value of the record pointer replaces the number of logical records in the file to indicate the number of records in the file. If another file can be defined, the system constructs a new file

with the remaining blocks; the blocking factor in the new file is that of the previous file. If a new file cannot be defined (e.g., no array space), the remaining blocks are added to the number of blocks reserved but unassigned, U.

If the logical unit is a random-access device, the job is aborted with the recovery dump diagnostic "DF MARKEF ERR."

f. Checking Status

1) To check the status of units used in FORTRAN "BUFFER IN" or "BUFFER OUT" operations, one uses the FORTRAN statements

IF (UNIT,i)n_1,n_2,n_3,n_4,

where i = logical unit number and the transfer points are interpreted as follows:

n_1 not ready,

n_2 ready and no previous error,

n_3 EOF sensed on last operation,

n_4 lost data sensed on last operation;

IF(UNIT,i)n_1,n_2,n_3

n_1 not ready,

n_2 ready and no previous error,

n_3 EOF or lost data;

IF(UNIT,i)n_1,n_2

n_1 not ready,

n_2 ready, EOF, or lost data.

Only n_1 and n_2 are meaningful for BUFFER OUT operations.

2) To check end-of-file after using a FORTRAN READ statement, the FORTRAN statement

IF(EOF,i)n_1,n_2

is used, where i = logical unit number and the transfer points are interpreted as follows:

n_1 EOF sensed on last operation;

n_2 EOF not sensed on last operation.

3) When the records in a file have been exhausted while being read from that file, the system returns an EOF signal and points the file pointer and record pointer at the first record of the next file if it is defined. If another file is not defined, the end-of-tape flag is set.

4. Disk-file Driver Specifications

The disk-file driver keeps track of assignment of file complexes to logical units by storing the file complex name in the appropriate Disk Assignment Table (DAT) entry for the logical unit involved.

a. SCOPE Calls and Driver Specifications

In the following SCOPE calls,

ℓ = logical unit number,

c = control word address,

r = reject address,

and

i = interrupt address.

1) Create

The SCOPE call

REØT(ℓ,c,r,i)

will associate the file complex control array, whose file complex name, FCN, is given in the low order 18 bits of the word at location c, with the logical unit number ℓ.

Driver specification

Store the file complex name into the Disk Assignment Table as the entry for logical unit number ℓ, return.

2) Read

The SCOPE call

READ(ℓ,c,r,i)

will read one logical record from logical unit number ℓ.

Driver specification

If tape-like device, and last operation was a "write," then abort job with diagnostic "DF W-R SEQ ERR."
Else set "read" into last operation flag.
If file pointer greater than number of files defined, then if end-of-tape flag is on, then abort job with diagnostic "DF EOT ERROR."
Else set end-of-tape flag on, set Status Reply Bits = $(401)_8$ to indicate end-of-tape, return.
Else if record pointer is pointing at the end of the file, then increase the file pointer by one, set the record pointer to 0, set Status Reply Bits = $(201)_8$ to indicate end-of-file, return.
Else if control word is IOSR with first word address = 0, then increase record pointer by one, set Status Reply Bits = 1, to indicate unit not busy, return.
Else begin transmission, increase record pointer by one, set unit busy bits, return.

3) Write

The SCOPE call

WRITE(ℓ,c,r,i)

will write one logical record on logical unit number ℓ.

Driver specification

If random-access device, then go to x.
Else if last operation was a "mark end-of-file," then if the file pointer is greater than the maximum number of files that can be defined, then abort the job with diagnostic "DF EOFCCA ERROR."
Else replace the number of files defined with the file pointer, go to x.
Else if last operation was a "write," then go to x.

Else add to the number of blocks unassigned all the blocks remaining in the current file and all the files defined from this point, replace the number of logical records by the sum of the record pointer and the integer part obtained from dividing the number of blocks per record into the number of blocks unassigned; the remainder of this division replaces the number of blocks unassigned, the file pointer replaces the number of files defined; set "write" into last operation flag.

x: If file not exhausted, then go to y.

Else if no more files defined in the complex, then abort the job with diagnostic "DF EOFCCA ERROR."

Else increase file pointer by one, initialize record pointer.

y: begin transmission, increase record pointer by one, set unit busy bits, return.

4) Backspace Record

The SCOPE call

BSPR(ℓ,r,i)

will decrease the record pointer by one to backspace over one logical record.

Driver specification

If random-access device, then go to u.

Else if last operation was a "read," then go to u.

Else if record pointer is 0 (i.e., last operation was a MARKEF), then go to v.

Else add to the blocks assigned the blocks remaining in the current file, replace the number of logical records in the file by the record pointer.

v: set "read" into last operation flag.

u: If end-of-tape, then clear end-of-tape, return.

Else if record pointer not pointing to first record in file, then decrease record pointer by one, return.

Else if in first file, then set Status Reply Bits = $(101)_8$ to indicate "load point," return.

Else decrease file pointer by one, and set record pointer to point to the end of the file, return.

5) Backspace File

The SCOPE call

BSPF(ℓ,r,i)

will decrease the file pointer by one to backspace over one file.

Driver specification

If random-access device, then go to s.

Else if last operation was a "read," then go to s.

Else if record pointer is 0 (i.e., last operation was a MARKEF), then go to t.

Else add to the blocks unassigned the blocks remaining in the current file, replace the number of logical records in the file by the record pointer.

t: set "read" into last operation flag.

s: clear end-of-tape flag.

If in first file, then set record pointer to first record in the file, set Status Reply Bits = $(101)_8$ to indicate "load point," return.

Else decrease file pointer by one, set record pointer to point to the end of the file, return.

6) Skip File

The SCOPE call

SKIP (ℓ,r,i)

will increase the file pointer by one to skip over one file.

Driver specification

If tape-like device, and last operation was a "write," then abort job with diagnostic "DF W-R SEQ. ERR."
Else if file pointer is greater than the number of defined files, then if end-of-tape flag is not set, then set the end-of-tape flag, return.
Else abort job with diagnostic "DF EOT ERROR."
Else increase file pointer by one, set record pointer to point to the first record in the file, return.

7) Rewind

The SCOPE call

REWIND(ℓ,r,i)

will reset the file pointer and record pointer to reflect the first record in the first file.

Driver specification

If random-access device, then go to q.
Else if last operation was a "read," then go to q.
Else if record pointer is 0 (i.e., last operation was a MARKEF), then go to r.
Else add to the blocks unassigned the blocks remaining in the current file, replace the number of logical records in the file by the record pointer.
r: set "read" into last operation flag.
q: set file pointer and record pointer to point to the first record in the file complex, set Status Reply Bits = $(101)_8$ to indicate "load point," return.

8) Mark End-of-File

The SCOPE call

MARKEF(ℓ,r,i)

will mark an end-of-file at the current position in the file complex.

Driver specification

If random-access device, then abort job with diagnostic "DF MARKEF ERR."

Else add to the number of blocks unassigned all the blocks remaining in the current file and all of the files defined from this point, replace the number of logical records by the record pointer, replace the number of files defined by the file pointer, increase the file pointer by 1, if the number of files defined equals the maximum number of files that may be defined, then go to w.

Else increase the file pointer by one, define a new file with the blocks unassigned so that the logical record length is the same as the previous file.

w: clear the record pointer to zero, set "write" into last operation flag, return.

9) Request for First Word of File Complex Control Array

The SCOPE call

WEOT (ℓ,c,r,i)

will store the first word of the file complex control array associated with logical unit ℓ into location c.

Driver specification

Store first word of file complex control array into location c.

10) Illegal SCOPE Calls

The SCOPE calls ERASE, and UNLOAD are illegal.

Driver specification

If ERASE or UNLOAD, then abort job with diagnostic "DF ILLEG. SCC."

b. Control Words

Since the physical record size in the disk-file hardware is fixed at 32 words, complications may arise when one uses an IOTR or IOSR control word, and these should be avoided.

If an unchained IOTR control word is encountered during a read request, the driver executes instead an IOTW control word with the same word count and first word address as specified.

If an unchained, IOSR control word with a 0 first word address is encountered during a read request, the driver will increase the record pointer by one and return to the calling program. The effect is to skip over one logical record without accessing the disk file.

c. Referencing a Systems Disk

When the user executes an operation that references a disk that is reserved for the System, the driver terminates the job abnormally, writing out the diagnostic "DF REF. SYS DISK." Disks 0 through 7 are reserved for Systems use.

d. Write Lockout

For each physical disk there is a hardware switch called a Write Lockout Switch. In the ON position, the disk-file controller prevents any writing on the associated disk through the corresponding access. A READ operation to any disk in the Write Lockout state can be performed in the normal manner.

If one attempts a WRITE operation on a disk in the Write Lockout state, the driver will abort the job abnormally, writing out the diagnostic "NO WRITE ENABLE."

5. Diagnostics

a. SCOPE Recovery Dump

Message	Meaning
NO WRITE ENABLE	Attempted a WRITE on a disk that has the Write Lockout switch on.
DF REF. SYS DISK	Attempted a READ/WRITE operation on a disk reserved for Systems usage. (Disks 0 through 7 are reserved.)
DF ILLEG. SCC	Attempted an ERASE or UNLOAD SCOPE function on a disk.
DF W-R SEQ. ERR	Attempted a READ or SKIP SCOPE function after a WRITE on a disk logical unit in tape mode.

Message	Meaning
DF MARKEF ERR	Attempted a MARKEF SCOPE function on a disk logical unit in random-access mode.
DF EOT ERROR	Attempted a READ or SKIP SCOPE function past a logical end of data.
DF EOFCCA ERROR	Attempted a WRITE SCOPE function past the end of the file complex.

b. Q8QERROR

Message	Meaning
ILLEGAL LOGICAL DISK.	DFDEFINE or CREATE call with an illegal logical disk number, LD. The LD must be in the range $1 \leq LD \leq 8$.
NO MEM AVAILABLE FOR FCCA.	No free storage available to define the file complex control array.
DISK SPACE EXCEEDED.	No space available on the disk to assign to the logical unit.
ILLEGAL DFDEFINE.	Too many blocking factor and logical record parameters, or not enough disk space requested for a logical unit in random-access mode.
ILLEGAL REDEFINE FILE.	Attempt to redefine a file without having an END FILE statement precede the redefine call.

APPENDIX C

S/360 Programming Techniques for the CalComp 780†

Ronald F. Krupp

1. Introduction

This appendix describes the programming techniques and methods available for generating a seven-track tape on the IBM System 360, to be plotted on the California Computer Products, Incorporated, Model 780, Magnetic Tape Plotting System. It is intended to be a guide for those who are writing FORTRAN programs.

Anyone acquainted with the CalComp Plotting Packages in use on Argonne's Control Data Corporation 3600 Computer should note that there are some major changes in this package:

a. To close a tape at the end of plotting, one would execute a CALL ENDPLOT statement in the CDC 3600 system. For this package, one must execute a CALL PLOT(X,Y,999).

b. This package assumes that the storage areas assigned to the adjusted minimum and delta values used in the SCALE, LINE, and AXIS routines are located at the end of the array (see the description of SCALE, LINE, and AXIS in Section 3.c below).

c. This package requires an additional parameter for the SCALE and AXIS routines (see the description of SCALE and AXIS in Section 3.c below) that describes the type of paper being used.

d. One cannot describe the format of a floating-point number by an "n H format" parameter in the argument list of a NUMBER call (see the description of NUMBER in Section 3.c below) in this package.

e. The values of the variables in the array are not scaled to page dimensions when SCALE is used. To do point symbol plotting for an array of values, use SCALE and LINE (see the description of LINE in Section 3.c below).

Binary decks of the plot package are available at the scheduler's desk.

---

†Originally published for internal distribution only as ANL-AMD Technical Memorandum No. 130 (Jan. 6, 1967).

Before execution, one must insert the following Job Control Cards:

```
                                                          col. 72
//GØ.PLØTTAPE DD DSNAME=PLØT780,DISP=(NEW,KEEP),           C

                    col. 16
//                    UNIT=(2400-2,,DEFER),LABEL=(,NL)
```

2. General Discussion of Plotter Programming

Six functions must be considered in order to obtain useful graphic output from the computer, although any specific application may not require the use of all six. The form of the graphic output obtained with these functions is restricted only by the size of the plotter and the ingenuity of the programmer. The six functions are:

a. Planning

In some cases, the program accomplishes the planning function in advance. For example, in a simple graph of numeric data, the programmer can easily decide the dimensions or positions of margins, heights, widths, labels, and data identifications. In other applications, the computer may need to make these decisions on the basis of input data. A typical example is a PERT chart. The events can easily be located on the time axis, but the arrangement in the other axis must be based on the interrelationship of events, the number of events occurring at any point in time, and the prevention of task lines passing through events.

b. Scaling

For some problems, the range of data is predictable and the programmer can easily set the conversion factors. However, in the general case, the computer must scan the data to find maximum and minimum values. These values must then be scaled for optimum plot size and ease of interpolation between divisions.

c. Line or Curve Generation

The magnitude and units of data must be converted to page dimensions. The data may then be presented as points, as straight lines between adjacent points, or by curve fitting between points.

d. Data Identification

The most common method of data identification is by means of axes. Each axis normally includes a title for the nature of the variable and its units of measurement, plus the magnitude of quantities evenly spaced along the axis.

e. Labeling

If the plot is affected by a third variable, data identification by means of axes may not be sufficient. Also, a plot may be valid only when identified by date or time. Thus, the ability to provide alphabetic or numeric labels is an essential function in the program.

f. Special Functions

This category includes any special instructions not covered by the five functions described above. One of the most important special functions is the ability to change the reference point at the programmer's discretion. The three basic CalComp routines PLOT, SYMBOL, and NUMBER are highly flexible and most special functions can be provided by direct use of these routines by the programmer.

The CalComp package of subroutines provides all the functions described above. All the subroutines are written to permit ease of handling by the programmer.

3. Detailed Descriptions

a. Sign Conventions

The sign conventions for CalComp plotter subroutines are designed for standard report-type graphs, with the vertical dimension designated as the Y-axis and the horizontal dimension as the X-axis. Standard graphs are usually presented on $8\frac{1}{2}$- by 11-in. or 11- by 17-in. sheets. As shown in Fig. C.1, the Y-axis is plotted horizontally across the narrow dimension of the paper and the X-axis vertically along the long dimension.

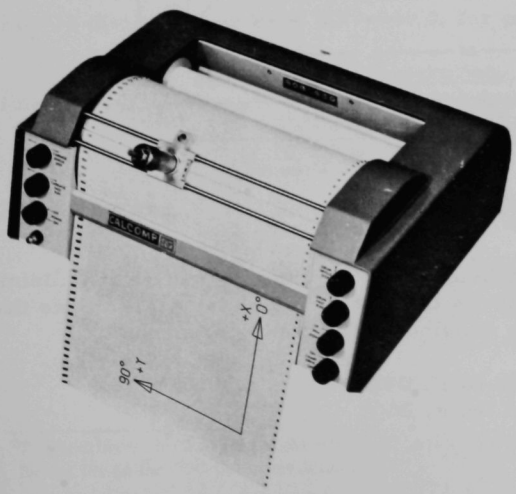


Fig. C.1

Sign Conventions, Shown on CalComp Model 565 Digital Incremental Plotter

b. Plotting Area

The CalComp 780 Digital Incremental Plotter uses chart paper rolls 120 ft long with a plotting width (Y-axis) of 11 in.

c. CalComp Plotter Subroutines

The following paragraphs describe the standard CalComp subroutines for the six programming functions discussed in Section 2. Note that FORTRAN conventions are used in the argument list for each subroutine. Arguments starting with I, J, K, L, M, or N are fixed-point; all others are floating-point, with the exception of ALPHA arrays, which are noted in the text.

1) Planning. Figure C.2 illustrates a planning layout to program a series of graphs. This layout is taken from the planning for the graphs in the sample problem described in Section 4 of this appendix.

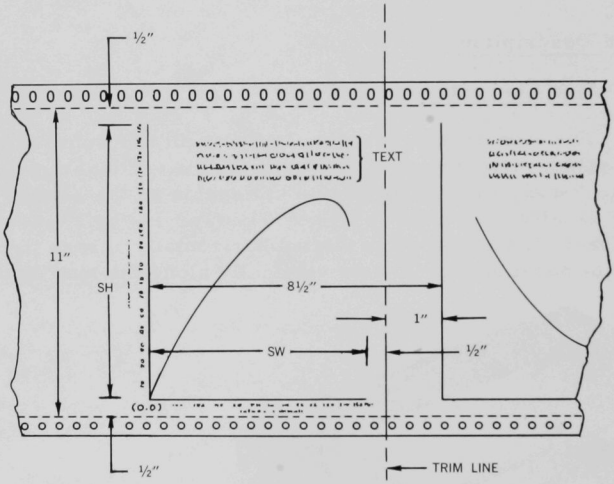


Fig. C.2. Planning Layout

2) Scaling. The SCALING routine finds maximum and minimum values and adjusts these to optimize the plot and establish scale values that are easy to interpolate between divisions. The calling sequence is

CALL SCALE (ORD, SH, N, K, DV)
CALL SCALE (ABS, SW, N, K, DV)

where

ORD is the name of array containing the variable to be plotted as the ordinate,

ABS is the name of array containing the variable to be plotted as the abcissa,

SW is the maximum width of the plotted line,

SH is the maximum height of the plotted line,

N is the number of data points in each array,

K is the repeat cycle, in case these values are in a mixed array,

and

DV describes the type of plotting paper being used by indicating the number of divisions per inch.†

NOTE A

a) The adjusted ORD (minimum) and adjusted delta ORD ((maximum-minimum)/SH) are stored following the array of data. The adjusted minimum is stored in ORD (N*K+1) and delta in ORD (N*K+K+1). Similarly, these two values for ABS will be found in ABS (N*K+1) and ABS(N*K+K+1).†

b) The values of the variables in the ORD and ABS arrays will not be scaled into page dimensions.†

c) The argument DV describes the type of paper to be used. For 10 divisions per inch, DV = 10.0; for centimeter paper, DV = 25.4.

3) Line or Curve Generation. The LINE subroutine draws a line or symbol through each successive data point. The calling sequence is

CALL LINE (ABS, ORD, N, K, J, L)

For ORD, ABS, N, and K, see the SCALE discussion above.

J is used to control symbols placed on data points. J=0 will produce only a line plot, J=1 will produce a symbol at every data point, J=2 will produce a symbol at every second data point, etc. A negative J suppresses the line between data points.†

L is used to specify the symbol to be drawn.†

---

†This is different from the use of the ANL PLOT package for the CalComp 580 and the ANL PLOT package for the 780 on the CDC Computer System.

NOTE B

a) LINE requires the adjusted minimum and delta values as described in SCALE.†

b) Symbols used for annotation by the LINE routine are the special centered symbols. See SYMBOL routine listing for symbol numbers (Note D below).

4) Data Identification. The AXIS subroutine draws a line with tick marks at 1-in. intervals. It annotates the value of the variable at each of these tick marks and provides an axis identification label. The calling sequence is

```
CALL AXIS (XO,YO,VORD,NCO,SH,THETA(O),
           ORD(N*K+1),ORD(N*K+K+1),DV(O))

CALL AXIS (XA,YA,VABS,NCA,SW,THETA(A),
           ABS(N*K+1),ABS(N*K+K+1),DV(A))
```

(XO,YO)	is the coordinate of the starting point of the ordinate axis.
(XA,YA)	is the coordinate of the starting point of the abscissa axis.
VORD	is the name of the alphabetic array of data for the ordinate axis title.
VABS	is the name of the alphabetic array of data for the abscissa axis title.
NCO,NCA	are the number of characters in each of the two axis titles. A negative sign places the annotation on the clockwise side of the axis instead of the counterclockwise side.
SH	is the length of the ordinate axis.
SW	is the length of the abscissa axis.
THETA(O), THETA(A)	are the angles (in degrees of the ordinate and abscissa axes, respectively).
ORD(N*K+1)	is the value of annotation at the first tick mark of the ordinate axis.
ORD(N*K+K+1)	is the difference in an ordinate variable for a 1-in. distance along the ordinate axis.

†This is different from the use of the ANL PLOT Package for the CalComp 580 and the ANL PLOT Package for the CalComp 780 on the CDC 3600 Computer System.

ABS(N*K+1)	is similar to the ORD(N*K+1) description.
ABS(N*K+K+1)	is similar to the ORD(N*K+K+1) description.
DV(O), DV(A)	describe the type of plotting paper being used, by indicating the divisions per inch of the ordinate and abscissa axes, respectively (see Note A on p. 115).†

NOTE C

a) The ORD(N*K+1) and ORD(N*K+K+1) arguments in CALL AXIS are the same values computed by SCALE. (See SCALE description on p. 114 for more detail.)†

b) The arguments SW and SH are usually the same values as the ones used in the SCALE routine.

5) Labeling. Labeling of the plots may be accomplished by the use of the symbol and number routines, whose calling sequences are

```
CALL SYMBOL (X,Y,HEIGHT,BCD,THETA,NS)
CALL NUMBER (X,Y,HEIGHT,FPN,THETA,NN)
```

(X,Y)	are the coordinates of the lower left corner of the first character to be drawn.
HEIGHT	is the height of the characters to be drawn.
BCD	is the location of the first byte containing alphabetic (or special character) information if NS is positive. If NS is negative, BCD is the integer value of the character to be plotted (a four-byte word with the character right-justified).
FPN	is the floating-point number.
THETA	is the angle (in degrees) at which the label is to be written.
NS	is either a positive or negative integer. If positive, NS is the number of characters to be plotted. If negative, only one character will be plotted and $NS = \begin{cases} -1, & \text{the pen will be raised before moving to (X,Y).} \\ -2, & \text{the pen will be lowered before moving.} \end{cases}$

†This is different from the use of the ANL PLOT Package for the CalComp 580 and the ANL PLOT Package for the CalComp 780 on the CDC 3600 Computer System.

NN† > 0, the NN places to right of the decimal point will be drawn.

$= 0$, the integral portion will be drawn with a decimal point.

≤ -1, no decimal point will be drawn and NN -1 digits will be dropped from the number.

NOTE D

The symbols available are shown in Fig. C.3 along with their hexadecimal values. Either the value shown, or that value plus 80_{16} may be used to generate a symbol. Thus, either one of two allowable punched card codes may be used to generate any character.

Four special characters that are not plotted are shown in Fig. C.3. These are Backspace (BS, Hex-11), Carriage Return (CR, Hex-15), Superscript (UC, Hex-2E), and Subscript (LC, Hex-2F). Backspace and Carriage Return act as in a typewriter and cause the following character to be plotted as superscripts or subscripts at 70% of the given height until the opposite character (subscript or superscript) is given.

If one of the first 13 characters shown in Fig. C.3 is used, then its height will be multiplied by 4/7 and it will be centered. To do point symbol plotting, use one of the first 13 characters with a negative NS parameter.

If X, Y, or HEIGHT is -0.0, then use the final values from the last call to SYMBOL (or NUMBER). For X, or Y, this means the location where the next character would have started.

$FPN*10^N$ should be less than 8,400,000.

6) Special Functions. The basic plotting routine provides several entries. The first entry is to allocate a work region to the PLOT routine. The calling sequence is

CALL PLOTS (BUFFER, NBUF, NTAPE)

BUFFER is the low-order core location of the work area.

NBUF is the number of bytes in the work area.

NTAPE is the logical tape unit for output. (NTAPE is not used in this version of the package.)

The second entry to PLOT is an instruction to move to the specified page coordinate with pen lifted or lowered. The third argument may also specify a new reference. The routine is called by

† This is different from the use of the ANL PLOT Package for the CalComp 580 and the ANL PLOT Package for the CalComp 780 on the CDC 3600 Computer System.

00	□	10	|	20	}	30	∑	40		50	&	60	–	70	0
01	⊖	11	BS	21	{	31	÷	41	A	51	J	61	/	71	1
02	△	12	∧	22	μ	32	≤	42	B	52	K	62	S	72	2
03	+	13	≡	23	π	33	≥	43	C	53	L	63	T	73	3
04	×	14	→	24	Φ	34	△	44	D	54	M	64	U	74	4
05	◇	15	CR	25	⊖	35	[	45	E	55	N	65	V	75	5
06	⊕	16	≠	26	ψ	36	]	46	F	56	Ø	66	W	76	6
07	⊠	17	±	27	χ	37	\	47	G	57	P	67	X	77	7
08	Z	18	_	28	ω	38	↑	48	H	58	Q	68	Y	78	8
09	Y	19	=	29	λ	39	√	49	I	59	R	69	Z	79	9
0A	⊡	1A		2A	∝	3A	‡	4A	¢	5A	!	6A	∞	7A	:
0B	✳	1B	∫	2B	δ	3B	‡	4B	.	5B	$	6B	,	7B	#
0C	⧖	1C	⊃	2C	∈	3C	←	4C	<	5C	*	6C	%	7C	@
0D	|	1D	∨	2D	η	3D	×	4D	(	5D	)	6D	—	7D	'
0E	☆	1E	∼	2E	UC	3E	↑	4E	+	5E	;	6E	>	7E	=
0F	_	1F	≈	2F	LC	3F	↓	4F	|	5F	¬	6F	?	7F	"

Fig. C.3. Characters Available in Symbol Routine (IBM 360)

CALL PLOT (XPAGE, YPAGE, IC)

(XPAGE,YPAGE) is an instruction to the plot routine to generate the increments necessary to move the pen from the current position to (XPAGE, YPAGE).

IC is a signed integer used as follows:

a) If IC ends in 2, the pen is lowered before moving.
If IC ends in 3, the pen is raised before moving.

b) If

i. $|IC| < 10$, then (XPAGE, YPAGE) represents a physical page dimension.

ii. $10 \leq |IC| < 20$, then (XPAGE, YPAGE) is adjusted by the scale factors provided (see OFFSET below):

XPAGE=(XPAGE-XOFF)/XFAC and
YPAGE=(YPAGE-YOFF)/YFAC

iii. $20 \leq |IC| < 30$, then the values of the current pen location is set to (0,0) after moving.

iv. $|IC| \geq 30$, then a block address of the value of IC is written and the tape is closed. A negative IC in this case will override the closing of the tape unit. Normally, IC=999 should be used to close the tape.†

c) If IC is negative, the appropriate action is taken according to the value of IC, a block address is written, and the current pen location is set to (0,0).

A third entry is to assist in plotting optimization. This entry provides current plotter position. The calling sequence is

CALL WHERE (XP,YP,FCTR)

(XP,YP) are filled with the current pen location upon return from the PLOT routine.

(FCTR) is the current multiplying factor being used by PLOT. (See FACTOR below.)

†Always close the tape when plotting is completed. This procedure replaces the ENDPLOT call that is used to close a tape in the ANL PLOT package for the 580 and the ANL PLOT package for the 780 on the CDC 3600 Computer System.

Additional entries are available to provide factors and offsets. These are accessed as

CALL FACTOR (FCTR)

FCTR is a floating-point number used as a multiplicative factor with <u>all</u> subsequent coordinates.

CALL OFFSET (XOFF,XFAC,YOFF,YFAC)

Enters factors to be used by the plot routine when IC is 12 or 13 (see PLOT above). Initially, XOFF=YOFF=0.0, XFAC=YFAC=1.0.

4. Sample Problem

```
LEVEL 25 AUG 66   ARGONNE NATIONAL LABORATORY   OS/360  FORTRAN H   CONTROLLED RELEASE VERSION 3  DATE  66.3'

                CSAMPLE          COPYRIGHT 1965 CALIFORNIA COMPUTER PRODUCTS                SAMPL000
                C
                C THE FOLLOWING STATEMENT RESERVES A WORK REGION FOR PLOT.IT IS
                C  RECOMMENDED THAT THIS WORK REGION BE AT LEAST 2000 CHARACTERS. THIS
                C  MAY BE AS SMALL AS 120 CHARACTERS OR AS LARGE AS 180000 CHARACTERS.
                C  NOTE THAT 3 ADDITIONAL ELEMENTS ARE ASSIGNED FOR SCALE,LINE,AND AXIS
                C  ROUTINES.
                C
  ISN 0002            DIMENSION DATA(1000),X(52),Y(52),TV(5)                              SAMPL010
                C
  ISN 0003            DIMENSION BCD(80)                                                   SAMPL020
  ISN 0004            DIMENSION NA(16)                                                    SAMPL030
  ISN 0005       101  FORMAT (16I3)                                                       SAMPL040
  ISN 0006       102  FORMAT (20A4)                                                       SAMPL050
  ISN 0007            READ (5,101)(NA(I),I=1,16)                                          SAMPL060
  ISN 0008            READ (5,102)(BCD(I),I=1,80)                                         SAMPL070
  ISN 0009            CALL PLOTS  (DATA(1    ),4000,11)                                   SAMPL080
                C
                C THE FOLLOWING STATEMENT INITIALIZES THE PLOT ROUTINE.
                C
  ISN 0010            CALL PLOT(0.0,-.5,2)                                                SAMPL090
                C
                C THE FOLLOWING 4 STATEMENTS TELLS PLOTTER TO DRAW OUTLINE (8.5X11)
                C  WITH PEN DOWN (2).
                C
  ISN 0011            CALL PLOT (0.0,10.5,2)                                              SAMPL100
  ISN 0012            CALL PLOT (8.5,10.5,2)                                              SAMPL110
  ISN 0013            CALL PLOT (8.5,-.5,2)                                               SAMPL120
  ISN 0014            CALL PLOT (0.0,-.5,2)                                               SAMPL130
                C
                C THE FOLLOWING 2 STATEMENTS PRINTS LEGEND FOR SYMBOL TABLE AT TOP OF
                C  THE PAGE.
                C
  ISN 0015            CALL SYMBOL(1.0,10.3,0.10,BCD(1),0.0,10)                            SAMPL140
  ISN 0016            CALL SYMBOL(1.5,10.0,0.14,BCD(4),0.0,48)                            SAMPL150
                C
                C THE FOLLOWING 22 STATEMENTS GENERATE THE SYMBOL TABLE.
                C
  ISN 0017            W=0.2                                                               SAMPL160
  ISN 0018            DO 12 I=1,8                                                         SAMPL170
  ISN 0019            Z=9.1                                                               SAMPL180
  ISN 0020            DO 10 J=1,16                                                        SAMPL190
  ISN 0021            M=16*(I-1)+J-1                                                      SAMPL200
  ISN 0022            CALL SYMBOL(W+.2,Z,0.14,NA(I),0.0,-1)                               SAMPL210
  ISN 0023            CALL SYMBOL(W+.32,Z,0.14,NA(J),0.0,-1)                              SAMPL220
  ISN 0024            IF (M-47)  7,8,9                                                    SAMPL230
```

```
ISN 0025        7  IF (M-17)  9,6,5                                          SAMPL240
ISN 0026        5  IF (M-21)  9,4,3                                          SAMPL250
ISN 0027        3  IF (M-46)  9,2,9                                          SAMPL260
ISN 0028        2  CALL SYMBOL(W+.6,Z,0.21,BCD(16),0.0,2)                    SAMPL270
ISN 0029           GO TO 10                                                  SAMPL280
ISN 0030        4  CALL SYMBOL(W+.6,Z,0.21,BCD(17),0.0,2)                    SAMPL290
ISN 0031           GO TO 10                                                  SAMPL300
ISN 0032        6  CALL SYMBOL(W+.6,Z,0.21,BCD(18),0.0,2)                    SAMPL310
ISN 0033           GO TO 10                                                  SAMPL320
ISN 0034        8  CALL SYMBOL(W+.6,Z,0.21,BCD(19),0.0,2)                    SAMPL330
ISN 0035           GO TO 10                                                  SAMPL340
ISN 0036        9  CALL SYMBOL(W+.7,Z,0.35,M,0.0,-1)                         SAMPL350
ISN 0037       10  Z=Z-.6                                                    SAMPL360
ISN 0038       12  W=W+1.0                                                   SAMPL370
             C
             C THE FOLLOWING STATEMENT REESTABLISHES REFERENCE POINT FOR NEXT PLOT.
             C
ISN 0039           CALL PLOT (10.0,0.0,-3)                                   SAMPL380
             C
             C THE FOLLOWING 5 STATEMENTS GENERATE THE WHEEL OF LETTERS.
             C
ISN 0040           THETA = 0.0                                               SAMPL390
ISN 0041           DO 140 I = 1,8                                            SAMPL400
ISN 0042           TH = THETA * 0.017455                                     SAMPL410
ISN 0043           CALL SYMBOL(5.0+COS(TH),5.0+SIN(TH),0.14,BCD(20),THETA,19) SAMPL420
ISN 0044      140  THETA = THETA + 45.0                                      SAMPL430
             C
             C THE FOLLOWING STATEMENT REESTABLISHES REFERENCE POINT FOR NEXT PLOT.
             C
ISN 0045           CALL PLOT (10.0,0.0,-3)                                   SAMPL440
             C
             C THE FOLLOWING 25 STATEMENTS GENERATE THE 5 GRAPHS.
             C
ISN 0046           DLTX = 0.01                                               SAMPL450
             C
             C TV(I) DESCRIBES THE TYPE OF PAPER BEING USED BY INDICATING THE
             C  NUMBER OF DIVISIONS PER INCH. THIS PARAMETER IS USED IN THE SCALE
             C  AND AXIS ROUTINES.
             C
ISN 0047           TV(1) = 20.0                                              SAMPL460
ISN 0048           TV(2) = 25.0                                              SAMPL470
ISN 0049           TV(3) = 25.4                                              SAMPL480
ISN 0050           TV(4) = 10.0                                              SAMPL490
ISN 0051           TV(5) = 12.0                                              SAMPL500
ISN 0052           DO 11 I = 1,5                                             SAMPL510
             C
             C CHANGE RANGE OF X FOR EACH GRAPH.
             C
ISN 0053           DLTX = DLTX * 2.0                                         SAMPL520
ISN 0054           X(1) = DLTX                                               SAMPL530
```

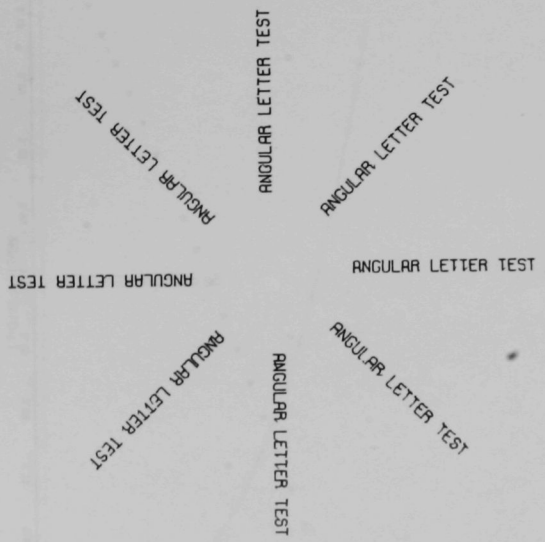
```
                C
                C EVALUATE 50 POINTS.
                C
ISN 0055              DO 21 J = 1,50                                          SAMPL540
                C
                C EQUATION TO BE EVALUATED.
                C
ISN 0056              Y(J) =   .005*X(J)**(I)+  .04*X(J)**(I-1)+  .3*X(J)**(I-2)   SAMPL550
                C
                C INCREMENT X.
                C
ISN 0057           21 X(J+1) = X(J) + DLTX                                    SAMPL560
                C
                C SCALE X VALUES,X(51)=ADJUSTED MINIMUM,X(52)=DX VALUE.
                C
ISN 0058              CALL SCALE (X,6.5,50,1,TV(I))                           SAMPL570
                C
                C SCALE Y VALUES,Y(51)=ADJUSTED MINIMUM,Y(52)=DY VALUE.
                C
ISN 0059              CALL SCALE (Y,10.0,50,1,TV(I))                          SAMPL580
                C
                C PLOT VERTICAL AXIS.
                C
ISN 0060              CALL AXIS(0.0,0.0,BCD(61),8,10.0,90.0,Y(51),Y(52),TV(I))   SAMPL590
                C
                C PLOT HORIZONTAL AXIS.
                C
ISN 0061              CALL AXIS(0.0,0.0,BCD(63),-6,6.5,0.0,X(51),X(52),TV(I))   SAMPL600
                C
                C PLOT CURVE CONNECTING POINTS (X,Y).
                C
ISN 0062              CALL LINE (X,Y,50,1,(I-3)*2,I)                          SAMPL610
                C
                C  THE NEXT 8 STATEMENTS GENERATE THE LEGEND AT TOP OF GRAPH.
                C
ISN 0063              CALL SYMBOL(1.0,9.0,0.14,BCD(25),0.0,45)                SAMPL620
ISN 0064              CALL SYMBOL(1.0,8.7,0.14,BCD(37),0.0,50)                SAMPL630
ISN 0065              CALL NUMBER ( 7.0 ,8.79,0.10,FLOAT (I-2),0.0,-1)        SAMPL640
ISN 0066              CALL NUMBER ( 6.04,8.79,0.10,FLOAT (I-1),0.0,-1)        SAMPL650
ISN 0067              CALL NUMBER ( 4.96,8.79,0.10,FLOAT (I),0.0,-1)          SAMPL660
ISN 0068              CALL SYMBOL(1.0,8.4,0.14,BCD(50),0.0,38)                SAMPL670
                C
                C PLOT FIRST X VALUE IN ARRAY.
                C
ISN 0069              CALL NUMBER ( 4.6 ,8.4 ,0.14, X(1) ,0.0, 2)             SAMPL680
                C
                C PLOT LAST X VALUE IN ARRAY.
                C
ISN 0070              CALL NUMBER ( 5.68,8.4 ,0.14, X(50),0.0, 2)             SAMPL690
                C
```

```
                C THE FOLLOWING STATEMENT REESTABLISHES REFERENCE POINT FOR NEXT PLOT.
                C
ISN 0071           11 CALL PLOT  (10.0,0.0,-3)                                  SAMPL700
                C
                C THE FOLLOWING STATEMENT CLOSES THE TAPE.
                C
ISN 0072              CALL PLOT (0,0,999)                                       SAMPL710
                C
ISN 0073              STOP                                                      SAMPL720
ISN 0074                 END                                                    SAMPL730
```

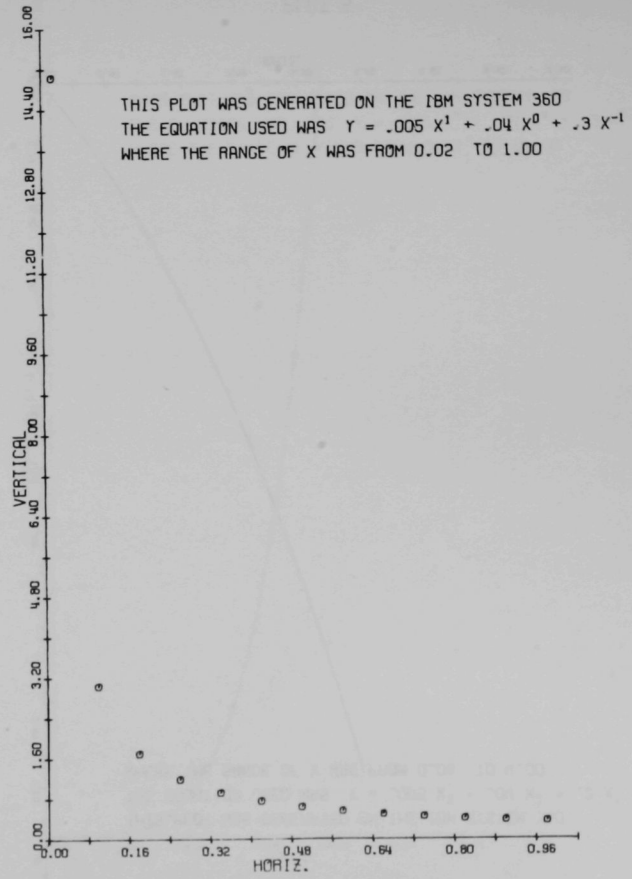
CHARACTERS AVAILABLE IN SYMBOL ROUTINE (IBM 360)

00		10	|	20	}	30	∑	40		50	&	60	—	70	0
01		11	BS	21	{	31	÷	41	A	51	J	61	/	71	1
02		12	∧	22	μ	32	≤	42	B	52	K	62	S	72	2
03		13	≡	23	π	33	≥	43	C	53	L	63	T	73	3
04		14	→	24	Φ	34	△	44	D	54	M	64	U	74	4
05		15	CR	25	Θ	35	[	45	E	55	N	65	V	75	5
06		16	≠	26	ψ	36	]	46	F	56	O	66	W	76	6
07		17	±	27	χ	37	\	47	G	57	P	67	X	77	7
08		18	_	28	ω	38	↑	48	H	58	Q	68	Y	78	8
09		19	=	29	λ	39	√	49	I	59	R	69	Z	79	9
0A		1A		2A	∝	3A	‡	4A	¢	5A	!	6A	∞	7A	:
0B		1B	∫	2B	δ	3B	‡	4B	.	5B	$	6B	,	7B	#
0C		1C	⊃	2C	∈	3C	←	4C	<	5C	*	6C	%	7C	@
0D	|	1D	∨	2D	η	3D	×	4D	(	5D	)	6D	—	7D	'
0E	☆	1E	~	2E	UC	3E	↑	4E	+	5E	;	6E	>	7E	=
0F	_	1F	≈	2F	LC	3F	↓	4F	|	5F	¬	6F	?	7F	"

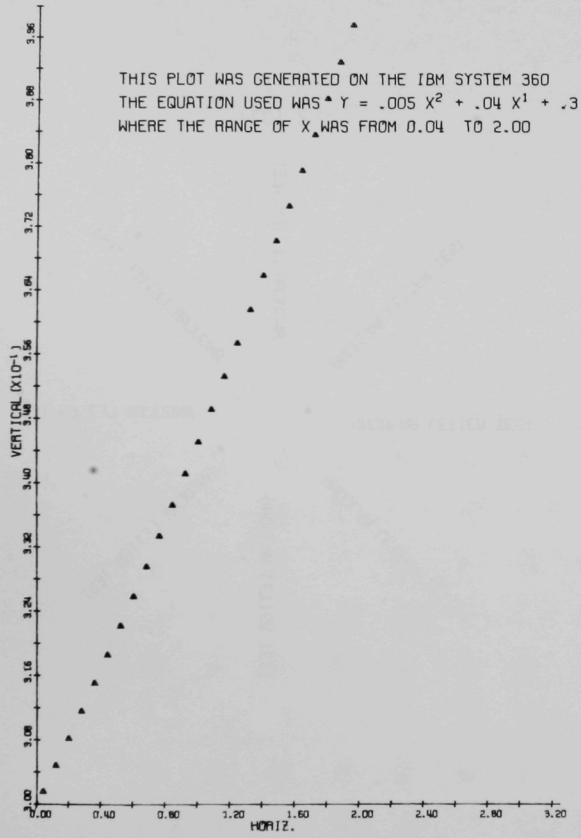
PLOT 1


ANGULAR LETTER TEST
ANGULAR LETTER TEST
ANGULAR LETTER TEST
ANGULAR LETTER TEST
ANGULAR LETTER TEST
ANGULAR LETTER TEST
ANGULAR LETTER TEST

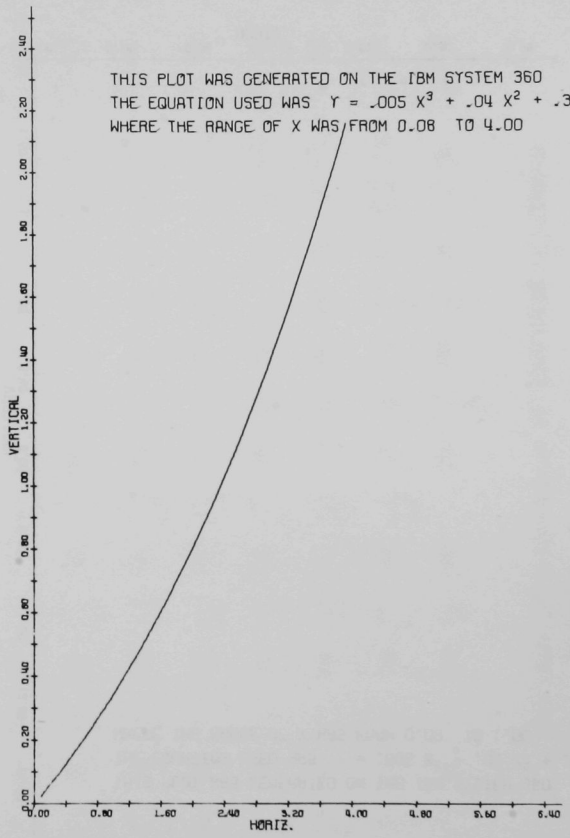
PLOT 2


THIS PLOT WAS GENERATED ON THE IBM SYSTEM 360
THE EQUATION USED WAS Y = .005 X¹ + .04 X⁰ + .3 X⁻¹
WHERE THE RANGE OF X WAS FROM 0.02 TO 1.00
VERTICAL
0.00
1.60
3.20
4.80
6.40
8.00
9.60
11.20
12.80
14.40
16.00
0.00
0.16
0.32
0.48
0.64
0.80
0.96
HORIZ.

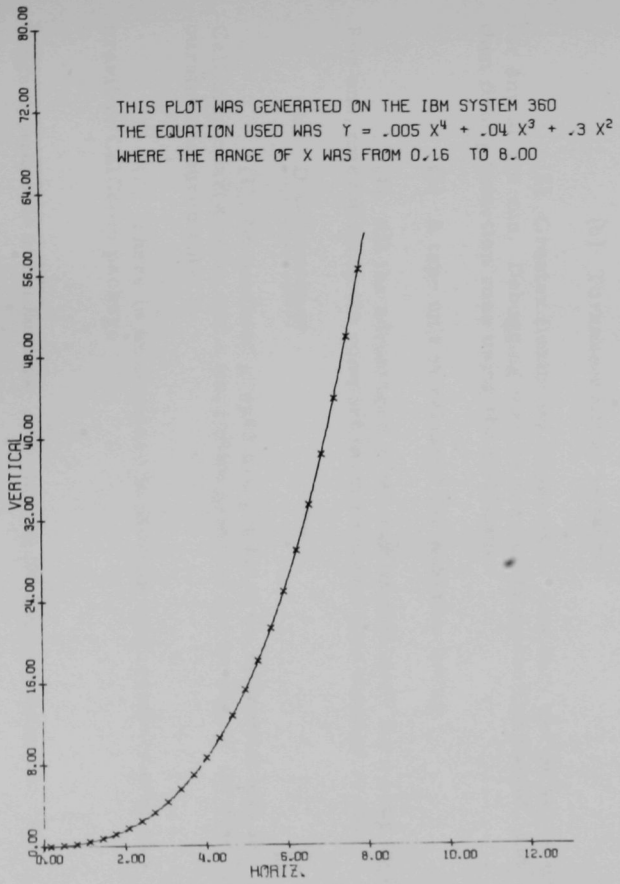
PLOT 3


THIS PLOT WAS GENERATED ON THE IBM SYSTEM 360
THE EQUATION USED WAS Y = .005 X² + .04 X¹ + .3 X⁰
WHERE THE RANGE OF X WAS FROM 0.04 TO 2.00
VERTICAL (X10⁻¹)
HORIZ.

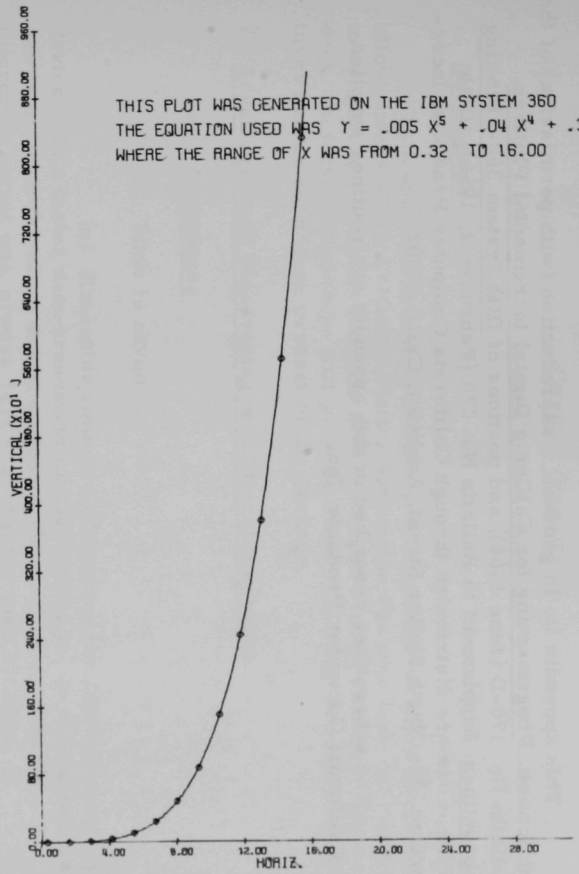
PLOT 4


THIS PLOT WAS GENERATED ON THE IBM SYSTEM 360
THE EQUATION USED WAS Y = .005 X³ + .04 X² + .3 X¹
WHERE THE RANGE OF X WAS FROM 0.08 TO 4.00
VERTICAL
HORIZ.

PLOT 5


THIS PLOT WAS GENERATED ON THE IBM SYSTEM 360
THE EQUATION USED WAS Y = .005 X^4 + .04 X^3 + .3 X^2
WHERE THE RANGE OF X WAS FROM 0.16 TO 8.00
VERTICAL
HORIZ.
0.00
8.00
16.00
24.00
32.00
40.00
48.00
56.00
64.00
72.00
80.00
2.00
4.00
6.00
8.00
10.00
12.00

PLOT 6


THIS PLOT WAS GENERATED ON THE IBM SYSTEM 360
THE EQUATION USED WAS Y = .005 X^5 + .04 X^4 + .3 X^3
WHERE THE RANGE OF X WAS FROM 0.32 TO 16.00
VERTICAL (X10¹)
HORIZ.
0.00
80.00
160.00
240.00
320.00
400.00
480.00
560.00
640.00
720.00
800.00
880.00
960.00
4.00
8.00
12.00
16.00
20.00
24.00
28.00

PLOT 7

5. Acknowledgments for Appendix C

This appendix is, in general, a reproduction (with permission) of the publication, Programming for CalComp Digital Incremental Plotters, Bulletin No. 170-D (June 1966), and portions of IBM System 360--Plotting Subroutines, Supplement to Bulletin No. 270 (February 2, 1966). Both publications are distributed through California Computer Products, Incorporated, 305 North Muller Street, Anaheim, Calif. 92803.

The subroutines described in this appendix are routines supplied by California Computer Products, Inc.

APPENDIX D

Graphic Output†

Ronald F. Krupp

1. Introduction

This appendix briefly describes a new package of subroutines which allows users of the CalComp Plotter Subroutine Package (see Appendix E) to convert to microfilm (on-line DD80A) output with no change to source deck. In fact, this package will produce either CalComp or microfilm pictures and as such is an overset of the package.

2. Advantages and Disadvantages of Microfilm Output

a. Advantages

(1) Time is saved.

(a) Execution time is faster since the DD80 is an on-line device with a faster data-transmission rate than tape. In all the test cases run to produce the same output on both devices, the execution time for microfilm output was always less than that for the CalComp.

(b) Turnaround time is faster.

(2) Greater flexibility is possible. One may use either device for any single run. Debugging may be done using microfilm output, and then final production runs using the CalComp.

(3) A tape unit is released for program usage.

(4) All the advantages of microfilm data over hard copy apply. For one, microfilm in its compact form requires less storage space.

b. Disadvantages

(1) No prelined graphs are produced on microfilm like the CalComp chart paper, and a microfilm graph is not very good for fine, accurate measurements.

(2) There is an increase in storage requirements over the previous CalComp package.

---

†Originally published for internal distribution only in ANL-AMD 3600 Newsletter 33 (March 31, 1966).

(a) The new package requires $1544_8 = 868_{10}$ additional locations (a new total of $4733_8 = 2523_{10}$ locations).

(b) The new package requires the FORTRAN library program IOP. that needs $1301_8 = 705_{10}$ memory locations. (This subroutine is always in core with any FORTRAN-coded program anyway.)

(c) If microfilm output is selected, the DD80 driver will be loaded. This driver is the subroutine required by SCOPE to handle film input-output requests, and has a current length of $3142_8 = 1634_{10}$ memory locations.

3. How to Use the New Package

a. Replace the old package with the new package.

b. To select microfilm output for a run, insert, at the beginning of the job deck, the SCOPE EQUIP control card

7/9EQUIP,ℓ=TV

where ℓ is the same logical unit number LUN used in the initialization call, CALL PLØTS(ARRAY,LENGTH,LUN).† (Remember to fill out the AMD film-processing request when submitting the job.)

c. To select CalComp output for a run, follow the standard procedures that have already been set up by the AMD Operations Group.

Note that no change to source deck is required when one wishes to run the same job to get microfilm output or CalComp output.

4. Correspondence of Plotting Area

Normal CalComp pictures are pen-drawn on chart paper where the basic plot size is

$$0 \leq X \leq 163 \text{ in.}, \quad 0 \leq Y \leq 10 \text{ in.}$$

Each frame on the microfilm represents a 10- x 10-in. plotting area, and a sequence of frames will be produced when the plot exceeds 10-in. in the X direction. Each frame in the sequence will represent a 10-in. segment in the X direction.

†See Appendix E.

5. Restrictions on Microfilm Output

a. Stacking Procedure

Since the film can only move forward, data that exceed 10 in. in the X direction for a picture are stacked into buffers. The memory locations assigned to plot package by the initialization call, CALL PLØTS(ARRAY,LENGTH,LUN)† is the first area into which data was stacked. When this area is full, the routine requests available memory from IOP. to be used as stacks. When no more memory is available, the diagnostic "DATA NØT PLØTTED, BUFFERS FULL." is written on the Standard Output tape through the Q8QERRØR routine, and the remaining information to be stacked is ignored. Once an "end of graph" (i.e., a CALL PLØT(X,Y,IND)† with $X \neq 0$ and IND negative, or CALL ENDPLØT) is encountered, the information from the stacks is plotted into microfilm and the stacks are reused for the next graph.

b. Axis Plotting

Due to a lack of plotting room, when one generates an axis, the axis label and the tick-mark labels may not be in the same relative positions as one would find them on a CalComp plot.

(1) When generating a horizontal or vertical axis, the routine will plot the axis label on the other side of the axis if there is not enough room to plot the label in its normal position. For example, the label for the axis generated by the call

CALL AXIS(0,0,6HHØRIZ.,-6,10.0,0.0,XMIN,DX)

would be above the axis rather than below where one would expect it.

(2) The tick-mark labels may not always be spaced from mark to mark and will have an appearance of drifting. When a picture requires two or more frames, the label for the last tick mark on each frame, except the last one, will be drawn under the first tick mark of the succeeding frames to eliminate duplication.

c. Partial Symbol Drawing

If the X coordinate of a symbol is such that part of the symbol is on one frame and the other part on the succeeding frame, the routine draws the symbol in exactly that manner.

---

†See Appendix E.

d. Data Outside the Available Plotting Area

The available plotting area for a picture on microfilm is defined within the boundaries

$$-0.23 \leq X \leq 163 \text{ in.}, \quad -0.23 \leq Y \leq 10 \text{ in.}$$

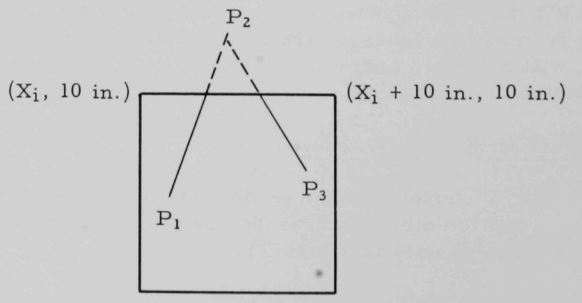
If one attemps to plot outside the picture, one of the following diagnostics is written out on the Standard Output unit through Q8QERRØR:

$X < -0.23$ in.,	"DATA NØT PLØTTED ØUT ØF RANGE AT LEFT ØF PLØTTING AREA."
$X > 163$ in.,	"X GREATER THAN 16383 E-2."
$Y < -0.23$ in.,	"DATA NØT PLØTTED ØUT ØF RANGE AT BØTTØM ØF PLØTTING AREA."
$Y > 10$ in.,	"DATA NØT PLØTTED ØUT ØF RANGE AT TØP ØF PLØTTING AREA."

Only one diagnostic, including the one for buffers full mentioned in Section a above, per picture sequence will be written.

If $X > 163$ in., the routine reduces X modulo 163 and continues plotting.

If $X < -0.23$ in., or $Y < -0.23$ in., or $Y > 10$ in. the routine draws the partial line segment to the appropriate border. The routine then keeps track of the data so that when plotting can be resumed inside the picture area, the line segment will be drawn from the appropriate point along the border. For example, to draw the vectors P_1P_2, P_2P_3, where P_2 is outside the plotting area, one can expect the following picture:



APPENDIX E

CalComp Plotting on the CDC 3600

1. Subroutine Package to Prepare Input to CalComp 580 and 780 Plotters*--Clifford LeVee and John Ohde

a. General Information

To relieve the user of the need to provide the precise magnetic-tape format needed by the CalComp 580 and 780 Plotters, a set of subroutines is provided. The routines are compatible with the FORTRAN systems and monitors on the Argonne CDC 3600.

In the following description of the subroutine package, the normal FORTRAN naming rules apply to the formal parameter names. The lengths of the subroutines are given in decimal form.

The formal parameters X and Y will stand for the coordinate values of variables to be plotted on the horizontal (along the drum) and vertical (across the drum) axes, respectively.

All the 3600 routines use the library routines Q8QDICT., and Q8QRESID. PLØT uses SENTRY; NUMBER requires SENTRY, ENC., and Q8QERRØR.

The user is responsible for scaling his data to page dimensions for plotting (see Subroutine Scale below). The limitations on page size are such that Y must be less than 11 in., 10 is normal, and X may vary up to 1440 in.; however, the accuracy of subroutine calculations further restricts X to be less than 163.83 in. New picture references may be used to go beyond this value.

b. Specific Description of the Subroutines

The logical organization of the subroutines in the plot package is shown in Fig. E.1. The entry points, number of arrangements, and length of each routine are listed in Table E.1.

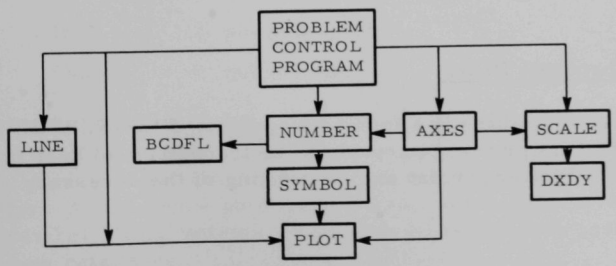


Fig. E.1

Normal Flow Diagram

*Extracted from ANL-AMD Technical Memorandum 70 (for internal distribution only), modified to remove 704 references and add 780 applicability.

TABLE E.I. Table of Subroutine Entry Points and Size

Subroutine	Entry Point	Usage	Number of Arguments	Length
PLOT[a]				345,[e] 576,[c] 775[d]
	WHERE	Return current x,y values.	2	
	FACTØR	Rescale plot	1	
	PLØTS	Initialize package	2, 3	
	PLØT	Basic plot entry.	3	
	ENDPLOT[c,d]			
	NUMBLOCK[c,d]			
SYMBØL				343,[e] 379[c,d]
	SYMBØL	Generate alphanumeric symbols.	6	
		Generate special centered symbol.	6	
NUMBER				82,[e] 104[c,d]
	NUMBER	Generate a number for label, etc.	6	
AXIS				252,[e] 318[c,d]
	AXIS	Generate an axis with marking and label.	8	
LINE				104,[e] 138[c,d]
	LINE	Draw continuous line through x,y coordinates.	4	
SCALE				103,[e] 229[c,d]
	SCALE	Scale variable to page dimensions.	6	
DXDY[b]				142[e]
	DXDY	Optimize plot size for SCALE	3	
		Package length		1371,[e] 1744,[c] 1943[d]

[a]PLOT580 in SYS20-2 and PLOT780 in SYS25-3.
[b]This routine is normally used only within the package, and not by the programmer. It exists in SYS20-1 only; its functions have been incorporated into subroutine SCALE in the SYS20-2 and SYS25-3 versions.
[c]SYS20-2 version.
[d]SYS25-3 version.
[e]SYS20-1 version.

1) Subroutine PLOT

This subroutine has four entry points: PLØTS, WHERE, FACTØR, and PLØT. Its primary purpose is the translation of data into CalComp 580 and 780 Plotter format and the writing of the necessary magnetic tape.

Entry PLØTS Initialization of package

Calling Sequence: CALL PLØTS(ARRAY,LENGTH,LUN)

ARRAY is a block of core memory which is made available to the package and must not be used by the calling program between the initialization entry and the final use of the package.

LENGTH is the size of the array.

LUN is the logical tape unit to be used by the routines ($1 \leq LUN \leq 49$).

Entry WHERE Find the current plot position

Calling Sequence: CALL WHERE(X,Y)

The routine will store the present x, y coordinate (in page dimensions) in locations X, Y. The coordinate is reduced by the factor employed by any prior use of entry point FACTOR.

Entry FACTØR Change the scale of the plot

Calling Sequence: CALL FACTØR(SIZE)

Size is a multiplying factor with respect to the x and y values. The initial value is one. This can be used to "blow up" (enlarge) a plot; for example, a picture that uses a 5-in. scale can be made on a 10-in. scale by setting SIZE to 2.0.

Entry PLØT Basic entry for plotting

Calling Sequence: CALL PLØT(X,Y,IND)

X,Y is the coordinate in page size of the next point on the plot. IND is a pen motion indicator which tells whether the values of IND are interpreted as follows:

IND = 0 or 1 no change in pen position.

IND = 2 put pen down.

IND = 3 put pen up.

If IND is negative, pen movement is made as indicated above, movement to (X,Y) is made, and this now becomes the origin point of a new picture; for example, CALL PLOT (8.5, 0.0, -3) will lift the pen, move to current X reference point plus 8.5 in., and set this to the (0,0) reference for further PLOT entries. The user is required to set the proper Y value for new reference.

2) Subroutine AXIS

Entry AXIS

Calling Sequence: CALL AXIS(X,Y,BCD,N,S,THETA,XMIN,DX)

This subroutine generates an axis from (X,Y) to (X+S*COS(THETA), Y+S*SIN(THETA)), where THETA is the angle of the axis expressed in degrees. A tick mark placed every inch along the axis is labeled with the value of the variable at that point. BCD is a label that will be centered along the axis; it is of length N, and will be placed on the right side of the axis if N is negative, and on the left side if N is positive. This label is written 1/2 in. away from the axis. XMIN is the value of the variable at (X,Y), and DX is the change in the variable per inch of axis. These values are usually determined by the use of subroutine SCALE.

Example: To generate an X axis 6.5 in. long and a Y axis 10 in. long.

CALL AXIS(0,0,6HHORIZ.,-6,6.5,0,XMIN,DX)

CALL AXIS(0,0,8HVERTICAL,8,10.0,90.0,YMIN,DY)

Requires: SIN(F), CØS(F), PLØT, NUMBER, SYMBØL.

3) Subroutine SCALE

Entry SCALE

Calling Sequence: CALL SCALE(X,N,S,XMIN,DX,K)

The purpose of this routine is to scale the values of the variable X (after finding maximum and minimum (XMIN) values) to fit actual page dimensions for plotting. The array is of length N, and the variable is in every Kth position. S is the length of the axis on which the variable will be plotted, and DX is the change in value of X per inch of S.

Requires: DXDY

4) Subroutine DXDY

Entry DXDY

Calling Sequence: CALL DXDY(XMAX,XMIN,DX)

This routine chooses the value of DX from 1, 2, 4, 5, and 8 and the appropriate power of ten that will allow the graph to fit on the page and still maintain reasonable units.

5) Subroutine LINE

Entry LINE

Calling Sequence: CALL LINE(X,Y,N,K)

This routine draws a continuous line through the coordinates in the arrays X and Y where there are N points. K is the repetition factor in a mixed array. X and Y values must be in page dimensions.

Requires: PLØT, WHERE

6) Subroutine SYMBØL

Entry SYMBØL

Calling Sequence: CALL SYMBØL(X,Y,H,BCD,THETA,N)

Two kinds of symbols may be generated by this routine: alphanumeric symbols, or special centered symbols. Since the calling sequence is the same in either case, the sign of the variable N is investigated to determine which figure or symbol is to be drawn.

Positive N specifies the alphanumeric set; negative N specifies the special set.

If the normal (alphanumeric) set is to be used, (X,Y) is interpreted as the lower left corner of a 4 x 7 grid on which the symbol is drawn. If the special set is to be used, (X,Y) becomes the approximate center of a 4 x 4 grid. To approximate point plotting, it is recommended that the special symbols be used.

H is the height of the symbol to be drawn and should be a multiple of 0.07. Spacing is automatically set to 6/7 of H. BCD is the location of the characters to be generated by the routine, or in the special set, an integer value ($0 \leq BCD \leq 14$) to determine the specific symbol. THETA is the angle in degrees at which the symbols are to be drawn. N is the number of characters in the string. Since only one special character may be drawn at a time, the values of N are restricted in this case. The routine interprets N = (-1) as a command to lift the pen before moving to (X,Y); if $N \leq (-2)$ the pen is lowered before moving to (X,Y).

Requires: SIN(F), COS(F), PLOT.

7) Subroutine NUMBER

Entry NUMBER

Calling Sequence: CALL NUMBER(X,Y,H,FPN,THETA,NHFORMAT)

This routine will plot the number at FPN beginning at (X,Y) as the lower left corner using the SYMBØL subroutine. THETA is the angle in degrees at which FPN will be plotted, and H is its height as described under Subroutine SYMBØL above.

The 3600 routine accepts a format under which FPN will be interpreted. The legal formats are: I, E, F, D, Ø, and R. No scale factors or parentheses are allowed in the format.

FØRMAT is restricted to six characters and may request no more than 32 characters of output.

Requires: SYMBØL,ENC.

2. CDC 3600 FORTRAN Programming for the CalComp 580 and 780 Plotters--George Concaildi and Roger Rempert

a. Introduction

The CalComp package of subroutines is set up assuming that all plotting is to be done in the first quadrant only. It will, therefore, plot everything in just that way. To use two, three, or four quadrants, the programmer must perform various manipulations.

Some sections of this presentation are quite general. A more detailed explanation appears in Section 2.i below.

One important note to remember: At the time of plotting a CalComp tape, the initial location of the pen on the paper is assumed to be the (0.0,0.0) point. The CalComp operator will always make sure that this initial position is one that will allow maximum Y-axis length, (10 in.) and maximum X-axis length (1440 in.). However, program restrictions limit plotting accuracy in the X-direction to no more than 163.83 in.

In all the CALL statements, we assume the FORTRAN notation for integers and floating-point numbers; I,J,K,L,M, and N are integers, and all others are real. This notation must be strictly followed.

b. Deck Arrangement

A sample deck structure for submission to the CDC-3600 Computer designed to produce CalComp plotter output is given below. The appropriate SCOPE control cards required are shown in capitals.

```
SEQUENCE CARD
JOB CARD
ACCOUNTING CARD
MOUNT CARD(S)
EQUIP CARD(S)
FORTRAN CONTROL CARD
   .
   .
   .
source decks
   .
   .
   .
SCOPE CARD
LOAD CARD
   .
   .
   .
CalComp Subroutine Package (available in Scheduling Area in
   .                        Building 221)
   .
RUN CARD
   .
   .
   .
data cards
   .
   .
   .
END OF FILE CARD
```

c. CALL PLOTS(ARRAY,LENGTH,LUN)

This statement is normally the first call made to the CalComp subroutines. Its purpose is to initialize the CalComp package.

ARRAY: This is an area of core set aside for use by the CalComp subroutines. The name given to ARRAY is determined by the programmer, but it must not be used by the calling program between initialization entry and the final use of the CalComp subroutines.

LENGTH: This gives the number of storage locations set aside for ARRAY. ARRAY must appear in a DIMENSION statement.

LUN: This gives the logical unit number the programmer wishes to assign to his CalComp tape. The logical unit number may be any integer value $1 \leq LUN \leq 49$. It is not necessary to use an EQUIP card to save the tape.

An example of the first three cards of the source program deck is

```
PROGRAM NAME
DIMENSION DATA(2000)
CALL PLOTS(DATA,2000,10)
```

d. CALL PLOT (X,Y,IND)

This CALL statement allows the programmer to do three things:

1) Move the pen from point (X0,Y0) to point (X1,Y1).

2) Regulate the vertical movement of the pen.

3) Move the origin from the initially assigned position to another location.

X,Y: The coordinates, in inches, of the next point to which the pen is moved.

IND: The value of this determines the up or down status of the pen point according to the following rules:

0 or 1	No change in pen status.
2	Pen down.
3	Pen up.

Consider two examples to show how this CALL statement is used.

Example 1. Generate a 10-in. square with an 8-in. square centered in it, as shown in Fig. E.2.

```
PROGRAM SQUARES
DIMENSION DATA(2000)
CALL PLOTS(DATA,2000,10)
CALL PLOT(10.0,0.0,2)
CALL PLOT(10.0,10.0,1)
CALL PLOT(0.0,10.0,1)
CALL PLOT(0.0,0.0,1)
CALL PLOT(1.0,1.0,3)
CALL PLOT(9.0,1.0,2)
CALL PLOT(9.0,9.0,1)
CALL PLOT(1.0,9.0,1)
CALL PLOT(1.0,1.0,1)
CALL END PLOT
END
```

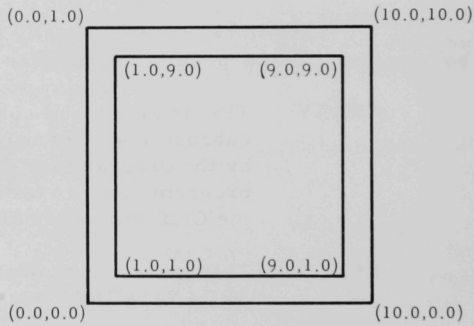


Fig. E.2. Figure Generated for Example 1

NOTE: It is important to have the CALL END PLOT statement, because it puts an end mark on the tape. Without this statement plotting may continue and whatever is recorded on the tape beyond the user's output could result in a garbled plot.

Example 2. Generate the figure from Example 1, space 2 in., and generate a 10-in. line 5 in. from the length of the paper, as shown in Fig. E.3. Here the negative sign defines the new origin.

```
PROGRAM MOVE
DIMENSION DATA(2000)
CALL PLOTS(DATA,2000,10)
        ***
CALL PLOT statements from
            Example 1
        ***
CALL PLOT(12.0,0.0,-3)
CALL PLOT(0.0,5.0,2)
CALL PLOT(10.0,5.0,1)
CALL END PLOT
END
```

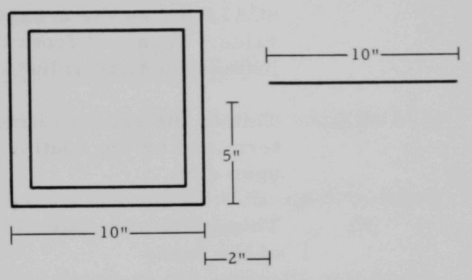


Fig. E.3. Figure Generated for Example 2

e. CALL LINE(X,Y,N,K)

X,Y: This defines the data points in the arrays† to be plotted. The programmer need not give numeric values to these variables.

N: This gives the number of data points to be plotted.

K: This is the repetition factor. It selects every Kth element of the arrays through which a continuous line will be drawn.††

This routine draws a continuous line through each data point to be plotted.

Example 3. Consider arrays X and Y each of 100 elements. To draw a continuous line through each pair of elements, the CALL statement would be

CALL LINE(X,Y,100,1).

Example 4. To draw a continous line through every other pair of elements, the statement would be

CALL LINE(X,Y,50,2).

f. CALL SCALE(X,N,S,XMIN,DX,K)

X: This defines the axis to be scaled.

N: This is the number of elements of the array.

†These arrays must be defined by a dimension statement.

††For further reference, subroutine LINE is listed in Section 2.i.1) of this appendix.

S: This is the length, in inches, of the axis on which the variable will be plotted.

XMIN: This is the minimum value of the variable. The routine SCALE scans the array X and determines this minimum value. Upon exit from the routine, XMIN contains an adjusted minimum value.

DX: This is the rate of change in X per inch of axis. It is determined by the routine and assigned to the location DX upon exit.

K: This is the repetition factor. It selects every Kth element of the array.

Examples 5 and 6 below show how the CALL SCALE routine is used.

Example 5. Consider an axis of 20.0 in., and X having a range $0 \leq X \leq 16$. The SCALE routine automatically chooses the minimum value of the variable as the origin. In this case it would choose 0.0. It will increment the scale values in the following way:

$$DX \cong (XMAX - XMIN)/S$$

$$\text{i.e., } DX \cong (16.0 - 0.0)/20.0 = 0.8.$$

CALL SCALE(X,160,20.0,XMIN,DX,1) gives the result† as shown in Fig. E.4.

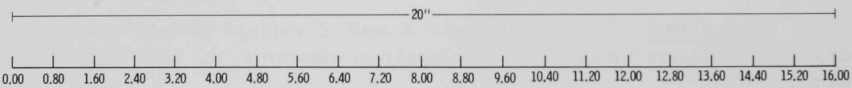


Fig. E.4. Figure Generated for Example 5

The SCALE subroutine also scales according to the following rule. The scale will be based on 1, 2, 4, 5, 8 times an appropriate power of 10, so as to have a range large enough to allow the graph to fit the chosen axis.

Example 6. Consider an array X, which consists of 249 points with a range $0 \leq X \leq 325$, to be plotted on a 10-in. axis. Then 4*10**2 is calculated as the scale range. Therefore, CALL SCALE(X,249,10.0,XMIN, DX,1) gives the result in Fig. E.5.

†The calculation of DX as shown here is only a rough estimate. For the exact adjusted value, see Section 2.i.3) of this appendix.

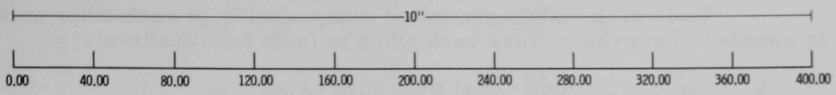
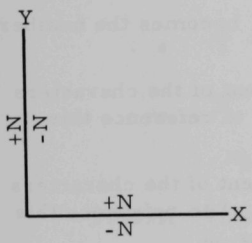


Fig. E.5. Figure Generated for Example 6

g. CALL AXIS(X,Y,BCD,N,S,THETA,XMIN,DX)

X,Y: These are the coordinates, in inches, of the initial point of the axis.

BCD: This is a regular Hollerith expression with which to label the axis. This label is centered on the axis.



N: This is the length of the Hollerith statement. For the X-axis, a negative N will put the label below the axis and a positive N puts it above the axis. For the Y-axis, a negative N puts the label to the right of the axis and a positive N puts it to the left, as shown in the diagram.

S: This is the length of the axis, in inches.

THETA: This gives the angle, in degrees, with respect to the length of the paper, at which the axis is drawn.

XMIN: This is the minimum value of the array, normally determined by the SCALE routine.

DX: This gives the change in the variable per inch of axis, normally determined by the SCALE routine.

Example 7. Generate a set of axes to form the first quadrant, where

$0 \leq X \leq 100$ On a 10-in. axis.

$0 \leq Y \leq 100$ On a 10-in. axis.

This is shown in Fig. E.6. The statements to do this are

```
CALL AXIS(0.0,0.0,8HX VALUES,-8,
          10.0,0.0,XMIN,DX)
CALL AXIS(0.0,0.0,8HY VALUES,8,
          10.0,90.0,YMIN,DY)
```

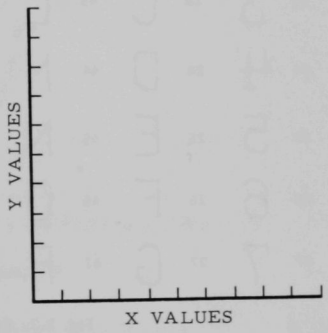


Fig.E.6. Figure Generated for Example 7

NOTE: When an axis is drawn, tick marks are placed along its length on every 1-in. interval.

Since each axis is generated independently of each other axis, it is simple to have them cross each other to form four quadrants.

h. CALL SYMBOL(X,Y,H,BCD,THETA,N)

X,Y: These are the coordinates of the initial point of the symbol to be plotted. X and Y are in inches.

H: This gives the height of the characters to be printed. This value must be expressed as a multiple of 0.07 in. The width of each symbol is taken as (4/7)*H and the spacing between characters of the symbol as (6/7)*H.

BCD&N: BCD can be one of three expressions:

a. A Hollerith statement, where N becomes the number of characters in the statement.†

b. BCD may be the bioctal equivalent of the characters listed in Fig. E.7. N must be 1 to reference this "regular" set of characters.††

c. BCD may be the integer equivalent of the characters listed in Fig. E.8. N must be a -1 to reference this "special" set of characters.††

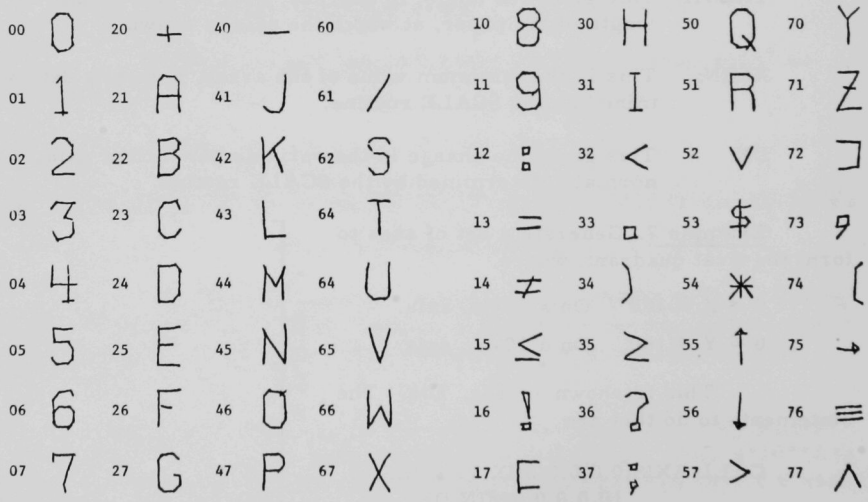


Fig. E.7. Regular Character Set; Bioctal Equivalence

† The coordinates X,Y are the coordinates of the lower left-hand comer of an imaginary rectangle surrounding the first character in the Hollerith message.

†† The coordinates X,Y are the coordinates of the center of the symbol.

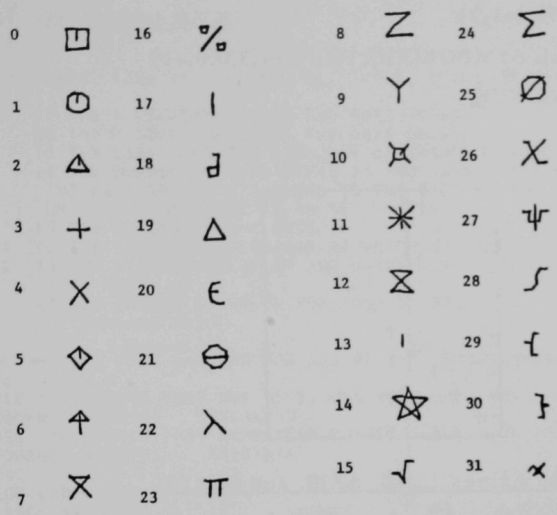


Fig. E.8. Special Character Set; Integer Equivalence

THETA: This is the angle, in degrees, at which the symbol given by BCD is plotted.

Examples 8 and 9 will illustrate the three cases described above.

Example 8. Generate the title Y=X**2-6.0, starting at (4.0,7.0).

```
CALL SYMBOL(4.0,7.0,0.14,10HY=X**2-6.0,0.0,10)
```

The resulting plot is shown in Fig. E.9.

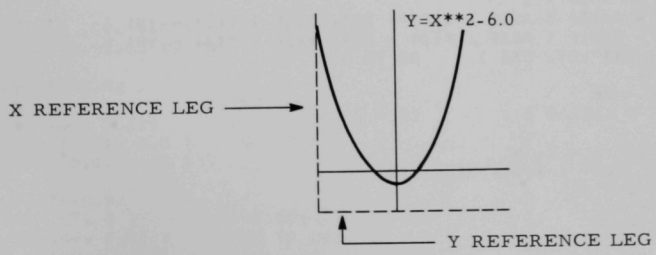


Fig. E.9. Figure Generated for Example 8

Example 9. Generate symbols to represent data points for the line Y=X. Assuming the 11 elements from arrays X(I) and Y(I) as stored in memory, the following statements will generate the desired result:

```
      DO 30 I=1,11
      CALL SYMBOL(X(I),Y(I),0.14,3,0.0,-1)
   30CONTINUE
```

The resulting plot is shown in Fig. E.10.

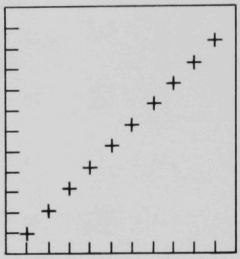


Fig. E.10

Figure Generated for Example 9

i. Subroutines LINE, AXIS, and SCALE

1) Subroutine LINE

```
      SUBROUTINE LINE  ( X, Y, NN, K1)
C
C        X       IS THE ARRAY TO BE PLOTTED AGAINST THE HORIZONTAL AXIS
C        Y       IS THE ARRAY TO BE PLOTTED AGAINST THE VERTICAL AXIS
C        NN      IS THE NUMBER OF DATA POINTS
C        K1      IS THE REPEAT CYCLE OF A MIXED ARRAY (NORMALLY ONE)
C
      DIMENSION X(1), Y(1)
C
      CALL WHERE ( XN, YN )
      I3 = 3
      N = NN
      NT = N - 1
      K = K1
      KK = XABSF ( K )
      J = KK * NT + 1
      DX1 = MAX1F ( ABSF ( X(1) - XN), ABSF ( Y(1) - YN) )
      DX2 = MAX1F ( ABSF ( X(J) - XN), ABSF ( Y(J) - YN) )
      IF ( DX1 .GT. DX2 )     GO TO 9
      J = 1
      KK = - KK
      IF ( K .LT. 0 )          I3 = 2
C
    9 DO  10  I = 1, N
       CALL PLOT ( X(J), Y(J), I3)
       J = J - KK
       I3 = 2
   10 CONTINUE
      END
```

2) Subroutine AXIS

```
C                                                                       SYSA0250
      SUBROUTINE AXIS  (X1, Y1, IB, NC, S1, THETA, YMIN1, DY1)          AXIS0010
C                                                                       AXIS0020
C        X1      IS THE X COORDINATE FOR THE AXIS ORIGIN                AXIS0030
C        Y1      IS THE Y COORDINATE FOR THE AXIS ORIGIN                AXIS0040
C        IB      IS THE LABEL FOR THE AXIS                              AXIS0050
C        NC      IS THE NUMBER OF CHARACTERS IN THE LABEL, AND ALSO     AXIS0060
C                 THE POINTER (BY ITS SIGN) TO THE SIDE OF THE AXIS     AXIS0070
C                 ON WHICH THE LABEL IS TO BE WRITTEN                   AXIS0080
C        S1      IS THE LENGTH OF THE AXIS                              AXIS0090
C        THETA   IS THE ANGLE (IN DEGREES) AT WHICH THE AXIS IS DRAWN   AXIS0100
C        YMIN1   IS THE MINIMUM VALUE OF THE DATA TO BE PLOTTED ON      AXIS0110
C                 THIS AXIS                                             AXIS0120
C        DY1     IS THE CHANGE IN VALUE PER INCH OF THE DATA            AXIS0130
C                                                                       AXIS0140
C                                                                       SYSA0010
C     THIS ROUTINE HAS BEEN CHANGED FOR SYS 21 THE DD80A COMPATABILITY  SYSA0020
C      PACKAGE                                                          SYSA0030
C         THOSE STATEMENTS THAT ARE ADDITIONS WILL HAVE THE             SYSA0040
C          SEQUENCE NUMBERS    SYSAXXXX                                 SYSA0050
C         THOSE STATEMENTS THAT HAVE BEEN CHANGED WILL HAVE THE         SYSA0060
C          SEQUENCE NUMBERS    AXISXXX*                                 SYSA0070
C                                                                       SYSA0080
      DIMENSION LABEL (2)                                               AXIS0150
      DATA (LABEL =10H(X10      ))    , (ANGLE = 17724357506504518),    AXIS0160
     X      (BIGMAX = 9999.99), ( NFMT = 2HI6), (LFMT = 4HF7.2)         AXIS0170
C                                                                       SYSA0090
      DATA (IFF=0),(TL=0.1),(TE=0.2143),(C1=0.2),(C2=0.05),(MV=0)       SYSA0100
     X(HN=0.1),(C3=0.36),(TD=1.0)                                       SYSA0110
C                                                                       AXIS0180
      IF ( IFF .LT. 0 )       GO TO 100                                 SYSA0120
      IFF=-1$CALL DDCCKEY ( ID )                                        SYSA0130
      IF ( ID .LT. 0 )       GO TO 110                                  SYSA0140
      TL=0.06$TE=1.20$C1=0.16$C2=0.07$HN=0.14$MV=-1                     SYSA0150
  100 IF ( ID .LT. 0 )       GO TO 110                                  SYSA0160
      IF ( THETA .EQ. 0 )       GO TO 101                               SYSA0170
      MV=0                                                              SYSA0180
      IF ( THETA .EQ. 90.0 )       GO TO 105                            SYSA0190
      TD=1.0$TE=0.0$GO TO 109                                           SYSA0200
C                                                                       SYSA0210
  101 IF ( X1+S1 .LE. 10.0 )   GO TO 102                                SYSA0220
      MV=-1$SIZEA=10.0-X1$SIZES=S1-SIZEA$N=SIZEA+0.5$TD=1.0$TE=1.20     SYSA0230
      GO TO 104                                                         SYSA0240
  102 MV=0                                                              SYSA0260
      IF ( X1+S1 .GT. 9.0 )     GO TO 103                               SYSA0270
      TD=1.0$TE=0.92$GO TO 104                                          SYSA0280
C                                                                       SYSA0290
  103 TD=0.92$TE=0.92                                                   SYSA0300
  104 IF ( NC .GE. 0 )       GO TO 109                                  SYSA0310
      IF ( Y1 .GE. 0.21)       GO TO 109                               SYSA0320
  105 IF ( Y1+S1 .GT. 9.0 )     GO TO 106                              SYSA0330
      TD=1.0$TE=0.0$GO TO 107                                           SYSA0340
C                                                                       SYSA0350
  106 TD=0.92$TE=0.92                                                   SYSA0360
  107 IF ( NC .LT. 0 )       GO TO 109                                  SYSA0370
      IF ( X1 .GE. 0.21 )       GO TO 109                               SYSA0380
  108 C3=-0.11$GO TO 110                                                SYSA0390
  109 C3=0.35                                                           SYSA0400
C                                                                       SYSA0410
  110 NAC = XABSF (NC)                                                  AXIS019*
C                                                                       SYSA0420
      SIGN = 1.0                                                        AXIS0200
      IF ( NC .LT. 0 )         SIGN = -1.0                              AXIS0210
```

```
      TH = THETA * ANGLE                                                AXIS0220
      S = S1                                                            AXIS0230
      CTH = COSF (TH)                                                   AXIS0240
      STH = SINF (TH)                                                   AXIS0250
      DY = DY1                                                          AXIS0260
      YMIN = YMIN1                                                      AXIS0270
      X = X1                                                            AXIS0280
      XB = X                                                            AXIS0290
      Y = Y1                                                            AXIS0300
      YB = Y                                                            AXIS0310
C                                                                       SYSA0430
      IF ( MV .GE. 0 )       N=S+0.5                                    AXIS032*
C                                                                       SYSA0440
      TN = N                                                            AXIS0330
C                                                                       AXIS0340
C       DRAW AXIS AND TIC MARKS EVERY INCH                              AXIS0350
C                                                                       AXIS0360
C                                                                       SYSA0450
      XA = XB - ( TL   * SIGN ) * STH                                   AXIS037*
      YA = YB + ( TL   * SIGN ) * CTH                                   AXIS038*
C                                                                       SYSA0460
        CALL PLOT (XA, YA, 3)                                           AXIS0381
      DO  10  I = 1, N                                                  AXIS0390
        CALL PLOT ( XB, YB, 2)                                          AXIS0400
       XC = XB + CTH                                                    AXIS0410
       YC = YB + STH                                                    AXIS0420
        CALL PLOT ( XC, YC)                                             AXIS0430
       XA = XA + CTH                                                    AXIS0440
       YA = YA + STH                                                    AXIS0450
C                                                                       SYSA0470
        CALL PLOT ( XA, YA, 3)                                          AXIS046*
C                                                                       SYSA0480
       XB = XC                                                          AXIS0470
       YB = YC                                                          AXIS0480
   10 CONTINUE                                                          AXIS0490
C                                                                       SYSA0490
      CALL PLOT (XB, YB, 2)                                             SYSA0500
C                                                                       SYSA0510
C                                                                       AXIS0500
      ABSV = YMIN + ( DY * TN)                                          AXIS0510
      IXP  = 0                                                          AXIS0520
      IF ( DY .EQ. 0 )          GO TO 25                                AXIS0530
   15 IF ( DY .LT. 100.0)       GO TO 20                                AXIS0540
      DY = DY / 10.0                                                    AXIS0550
      ABSV = ABSV / 10.0                                                AXIS0560
      IXP = IXP + 1                                                     AXIS0570
      GO TO 15                                                          AXIS0580
   20 IF ( DY .GE. 0.01 )       GO TO 25                                AXIS0590
      DY = DY * 10.0                                                    AXIS0600
      ABSV = ABSV * 10.0                                                AXIS0610
      IXP = IXP - 1                                                     AXIS0620
      GO TO 20                                                          AXIS0630
C                                                                       SYSA0520
   25 XA = XB - ( ( SIGN * C1  ) - C2  ) * STH - TE      * CTH          AXIS064*
      YA = YB + ( ( SIGN * C1  ) - C2  ) * CTH - TE      * STH          AXIS065*
C                                                                       SYSA0530
      BIGNO = MAX1F ( ABSV, YMIN, -YMIN )                               AXIS0660
   30 IF ( BIGNO .LE. BIGMAX )  GO TO 40                                AXIS0670
      DY = DY / 10.0                                                    AXIS0680
      ABSV = ABSV /10.0                                                 AXIS0690
      IXP = IXP + 1                                                     AXIS0700
      BIGNO = BIGNO / 10.0                                              AXIS0710
      GO TO 30                                                          AXIS0720
C                                                                       SYSA0540
   40 IF ( MV .GE. 0 )       GO TO 111                                  SYSA0550
      CALL DSAXIS ( Y1,SIZES,THETA,ABSV,DY,NC)$ABSV=ABSV-DY             SYSA0560
  111 N = N + 1 + MV                                                    AXIS073*
C                                                                       SYSA0570
```

```
      DO  50  I = 1, N                                                  AXIS0740
C                                                                       SYSA0580
        CALL NUMBER ( XA, YA, HN , ABSV, THETA, LFMT)                   AXIS075*
C                                                                       SYSA0590
       ABSV = ABSV - DY                                                 AXIS0760
C                                                                       SYSA0600
       XA = XA - CTH*TD                                                 AXIS077*
       YA = YA - STH*TD                                                 AXIS078*
C                                                                       SYSA0610
   50 CONTINUE                                                          AXIS0790
C                                                                       AXIS0800
C       DRAW THE LABEL                                                  AXIS0810
C                                                                       AXIS0820
      TNC = NAC +7                                                      AXIS0830
C                                                                       SYSA0620
      XA = X + ( ( S / 2.0 ) - ( 0.06 * TNC ) ) * CTH - STH *           AXIS0840
     1          (- 0.07 + SIGN * C3    )                               AXIS085*
      YA = Y + ( ( S / 2.0 ) - ( 0.06 * TNC ) ) * STH + CTH *           AXIS0860
     1          (- 0.07 + SIGN * C3    )                               AXIS087*
C                                                                       SYSA0630
        CALL SYMBOL ( XA, YA, 0.14, IB, THETA, NAC)                     AXIS0880
C                                                                       AXIS0890
      IF (IXP .EQ. 0)          RETURN                                  AXIS0900
C                                                                       AXIS0910
      XA = XA + ( CTH * ( TNC - 6.0 ) * 0.12 )                          AXIS0920
      YA = YA + ( STH * ( TNC - 6.0 ) * 0.12)                           AXIS0930
        CALL SYMBOL ( XA, YA, 0.14, LABEL, THETA, 10)                   AXIS0940
      XA = XA + 0.60 * CTH - 0.07 * STH                                 AXIS0950
      YA = YA + 0.48 * STH + 0.07 * CTH                                 AXIS0960
        CALL NUMBER ( XA, YA, 0.1, IXP, THETA, NFMT)                    AXIS0970
      END                                                               AXIS0990
```

3) Subroutine SCALE

```
      SUBROUTINE SCALE (X, N1, S1, XMIN1, DX1, K1)                      SCALE010
C                                                                       SCALE020
C        X       IS THE NAME OF THE ARRAY TO BE SCALED                  SCALE030
C        N1      IS THE NUMBER OF DATA POINTS IN THE ARRAY              SCALE040
C        S1      IS THE LENGTH OVER WHICH THE DATA IS TO BE SCALED      SCALE050
C        XMIN1   IS THE LOCATION IN WHICH THE ADJUSTED MINIMUM VALUE OF SCALE060
C                 X WILL BE STORED                                      SCALE070
C        DX1     IS THE LOCATION IN WHICH THE ADJUSTED CHANGE IN VALUE  SCALE080
C                 PER INCH OF X WILL BE STORED                          SCALE090
C        K       IS THE REPEAT CYCLE OF A MIXED ARRAY (NORMALLY ONE)    SCALE100
C                                                                       SCALE110
      DIMENSION X(1)                                                    SCALE120
C                                                                       SCALE130
      S = S1                                                            SCALE140
      K = K1                                                            SCALE150
      NP = K * N1                                                       SCALE160
      DX = 0                                                            SCALE170
      XMAX = X(1)                                                       SCALE180
      XMIN = XMAX                                                       SCALE190
      DO  10  I = 1, NP, K                                              SCALE200
       IF ( XMAX .LT. X(I) )  XMAX = X(I)                               SCALE210
       IF ( XMIN .GT. X(I) )  XMIN = X(I)                               SCALE220
   10 CONTINUE                                                          SCALE230
C                                                                       SCALE240
      TX = ( XMAX - XMIN ) / S                                          SCALE250
      IF ( TX .GT. 0 )  GO TO 30                                        SCALE260
      DX = 1.0                                                          SCALE270
      XMIN = XMIN - 0.5                                                 SCALE280
      GO TO 65                                                          SCALE290
C                                                                       SCALE300
```

```
   30 IDX = ALOG10 ( TX )                                          SCALE310
      IXMN = XMIN * ( 10.0 ** (-IDX) )                             SCALE320
      IF ( XMIN .LT. 0 )       IXMN = IXMN - 1                     SCALE330
      IF ( XMIN . EQ. 0 )       GO TO 35                           SCALE340
      XMIN = IXMN * ( 10.0 ** IDX )                                SCALE350
   35 TX = ALOG10 ( (XMAX - XMIN) / S )                            SCALE360
      IDX = TX                                                     SCALE370
      XMAX = IDX                                                   SCALE380
      TX = 10.0 ** ( TX - XMAX )                                   SCALE390
      XMAX = 1.0                                                   SCALE400
C                                                                  SCALE410
   40 IF ( TX - 1.0 ) 45, 60, 50                                   SCALE430
   45 TX = TX * 10.0                                               SCALE430
      IDX = IDX - 1                                                SCALE440
      GO TO 40                                                     SCALE450
C                                                                  SCALE460
   50 XMAX = 2.0                                                   SCALE470
      IF ( TX .LE. 2.0 )       GO TO 60                            SCALE480
      XMAX = 4.0                                                   SCALE490
      IF ( TX .LE. 4.0 )       GO TO 60                            SCALE500
      XMAX = 5.0                                                   SCALE510
      IF ( TX .LE. 5.0 )       GO TO 60                            SCALE520
      XMAX = 8.0                                                   SCALE530
      IF ( TX .LE. 8.0 )       GO TO 60                            SCALE540
      XMAX = 10.0                                                  SCALE550
   60 DX = XMAX * ( 10.0 ** IDX )                                  SCALE560
C                                                                  SCALE570
   65 DO  70  I = 1, NP, K                                         SCALE580
       X(I) = ( X(I) - XMIN ) / DX                                 SCALE590
   70 CONTINUE                                                     SCALE600
C                                                                  SCALE610
  100 DX1 = DX                                                     SCALE620
      XMIN1 = XMIN                                                 SCALE630
      END                                                          SCALE640
```

j. Program GRAPH

This program plots the curve Y=2*X for a first-quadrant set of axes. The CALL PLOTS routine may appear anywhere in the program, except that it must come before all the other CalComp CALL statements. A plot of this is shown in Fig. E.11.

```
      PROGRAM GRAPH
      DIMENSION DATA(1024),X(200),Y(200),Z(200)
   95 READ 1,N,(X(I),I=1,N)
    1 FORMAT(I5/(5F10.0))
      IF(EOF,60)90,95
   90 PRINT 5
    5 FORMAT(55X,*X VALUE*)
      PRINT 6,(X(I),I=1,N)
    6 FORMAT(30X/(5F10.0))
      CALL PLOTS(DATA,1024,10)
      DO 3 I=1,N
      Y(I)=2.0*X(I)
      Z(I)=Y(I)
    3 CONTINUE
      PRINT 7
    7 FORMAT(55X,*Z VALUE*)
      PRINT 8,(Z(I),I=1,N)
    8 FORMAT(30X/(5F10.0))
      CALL SCALE(X,100,10.0,XMIN,DX,1)
      CALL SCALE(Y,100,10.0,YMIN,DY,1)
      CALL AXIS(0.0,0.0,7HX VALUE,-7,10.0,0.0,XMIN,DX)
      CALL AXIS(0.0,0.0,7HY VALUE,7,10.0,90.0,YMIN,DY)
      CALL LINE(X,Y,100,1)
      CALL END PLOT
      END
```

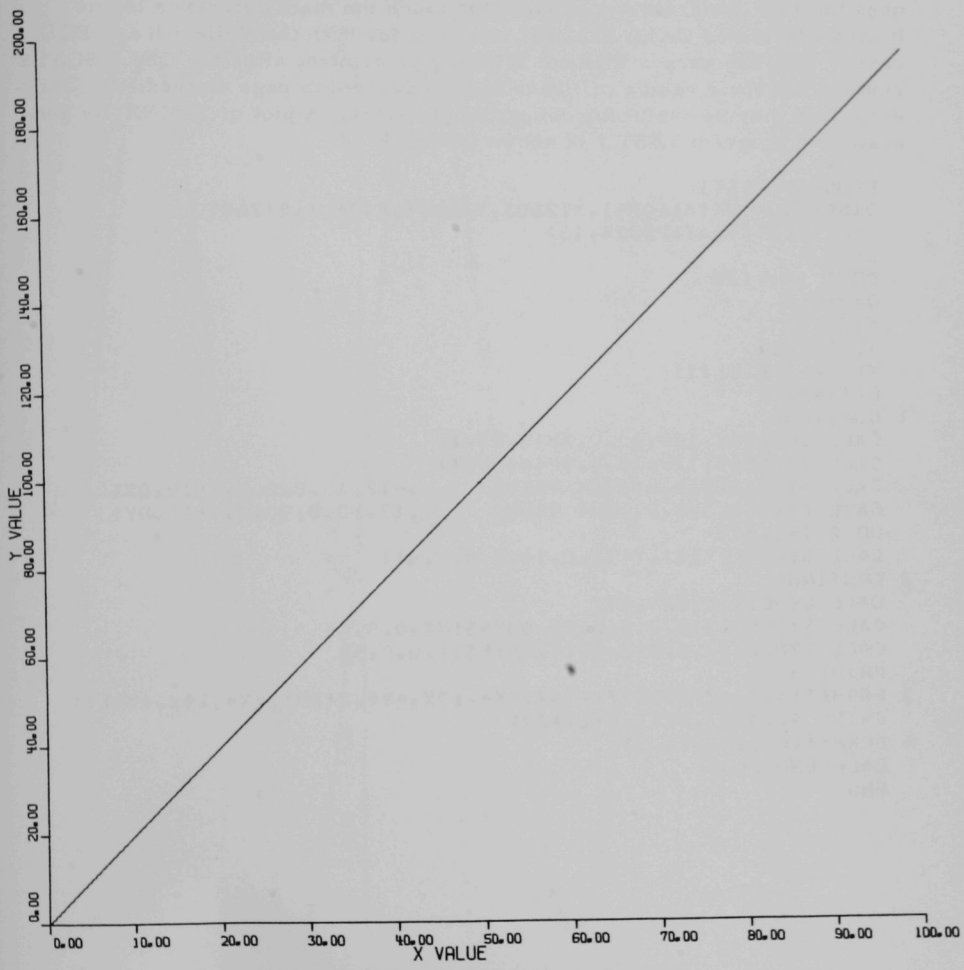


Fig. E.11. Plot Generated by Program GRAPH

k. Program TEST 1

This program illustrates the plotting of the curve Y=SINX. It uses the four-quadrant set of axes that touch the main reference frame. It also illustrates CALL SYMBOL routines for both the Hollerith and BCD cases. The two arrays X(I) and Y(I) may be printed, after the CALL SCALE routine, but their values will have been converted to page dimensions. However, this may be useful for debugging purposes. A plot of Y=SINX, as generated by program TEST 1 is shown in Fig. E.12.

```
      PROGRAM TEST1
      DIMENSION DATA(1024),X(260),Y(260),A(260),B(260)
      CALL PLOTS(DATA,1024,10)
      Q=-6.4
      DO 1 I=1,129
      Q=Q+0.1
      X(I)=Q
      A(I)=X(I)
      Y(I)=SINF(X(I))
      B(I)=Y(I)
    1 CONTINUE
      CALL SCALE(X,129,10.0,XMIN,DX,1)
      CALL SCALE(Y,129,10.0,YMIN,DY,1)
      CALL AXIS(0.0,5.0,12HX VALUE      ,-12,10.0,0.0,XMIN,DX)
      CALL AXIS(3.5,0.0,12HY VALUE      ,12,10.0,90.0,YMIN,DY)
      DO 2 I=1,129
      CALL SYMBOL(X(I),Y(I),0.14,3,0.0,-1)
    2 CONTINUE
      CALL LINE(X,Y,129,1)
      CALL SYMBOL(2.0,2.5,0.14,6HY=SINX,0.0,6)
      CALL SYMBOL(2.0,3.0,0.14,5HTEST1,0.0,5)
      PRINT 3
    3 FORMAT(55X,*Y=SINX*//(14X,*X*,19X,*Y*,2(19X,*X*,19X,*Y*)))
      PRINT 4,(A(I),B(I),I=1,129)
    4 FORMAT(6(10X,F10.5))
      CALL END PLOT
      END
```

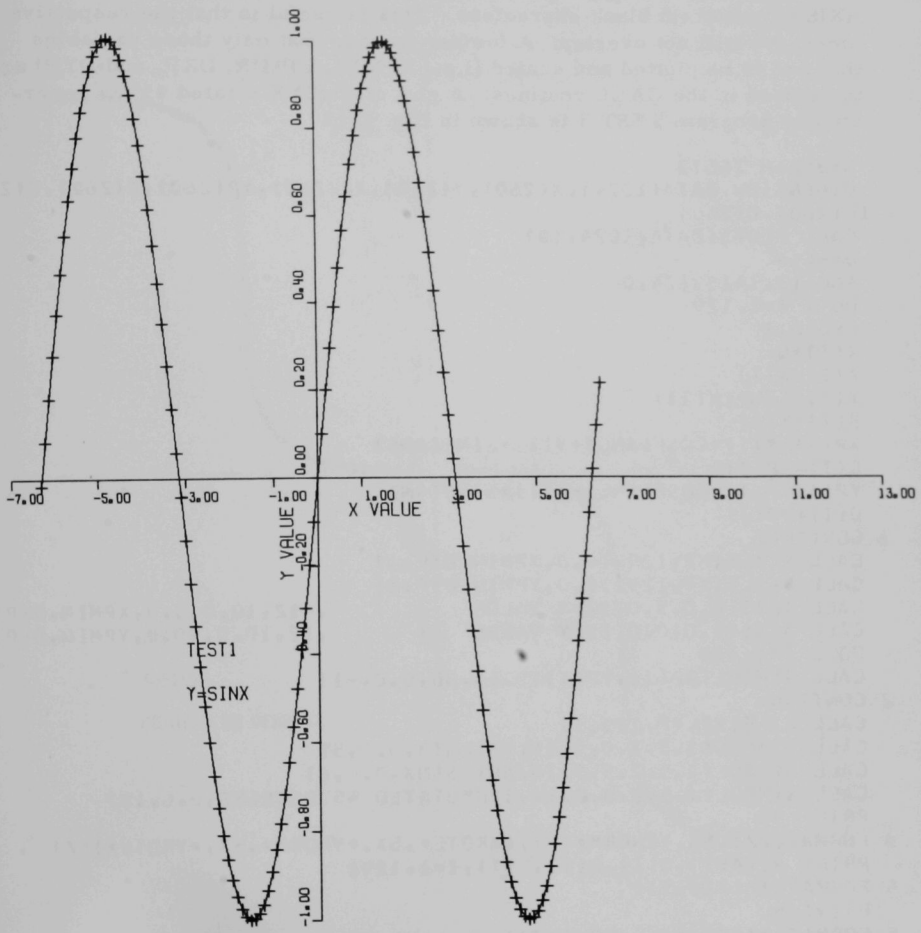


Fig. E.12. Plot Generated by Program Test 1

1. Program TEST 3

This program is similar to program TEST 1 except that the curve is rotated. The program shows that the Hollerith message in CALL AXIS may contain blank characters. This is useful in that the respective messages will not overlap. A further point is that only those variables that are to be plotted and scaled (i.e., XPMIN, YPMIN, DXP, and DYP) are to be used in the CALL routines. A plot of Y=SINX rotated 45° as generated by program TEST 3 is shown in Fig. E.13.

```
      PROGRAM TEST3
      DIMENSION DATA(1024),X(260),Y(260),XP(260),YP(260),A(260),B(260)
     1C(260),D(260)
      CALL PLOTS(DATA,1024,10)
      Q=-6.4
      ANG=(3.14159)/4.0
      DO 1 I=1,129
      Q=Q+0.1
      X(I)=Q
      A(I)=X(I)
      Y(I)=SINF(X(I))
      B(I)=Y(I)
      XP(I)=X(I)*COSF(ANG)+Y(I)*SINF(ANG)
      C(I)=XP(I)
      YP(I)=Y(I)*COSF(ANG)-X(I)*SINF(ANG)
      D(I)=YP(I)
    1 CONTINUE
      CALL SCALE(XP,129,10.0,XPMIN,DXP,1)
      CALL SCALE(YP,129,10.0,YPMIN,DYP,1)
      CALL AXIS(0.0,5.0,12HX VALUE            ,-12,10.0,0.0,XPMIN,DXP)
      CALL AXIS(5.0,0.0,12HY VALUE            ,12,10.0,90.0,YPMIN,DYP)
      DO 2 I=1,129
      CALL SYMBOL(XP(I),YP(I),0.14,3B,0.0,-1)
    2 CONTINUE
      CALL LINE(XP,YP,129,1)
      CALL SYMBOL(1.5,3.0,0.14,5HTEST3,0.0,5)
      CALL SYMBOL(1.5,2.5,0.14,6HY=SINX,0.0,6)
      CALL SYMBOL(1.5,2.0,0.14,18HROTATED 45 DEGREES,0.0,18)
      PRINT 3
    3 FORMAT(//3(5X,*XNORM*,5X,*XROTE*,5X,*YNORM*,5X,*YROTE*)//)
      PRINT 4,(A(I),C(I),B(I),D(I),I=1,129)
    4 FORMAT(12(F10.5))
      PRINT 5
    5 FORMAT(//55X,*ALL DONE*//)
      CALL END PLOT
      END
```

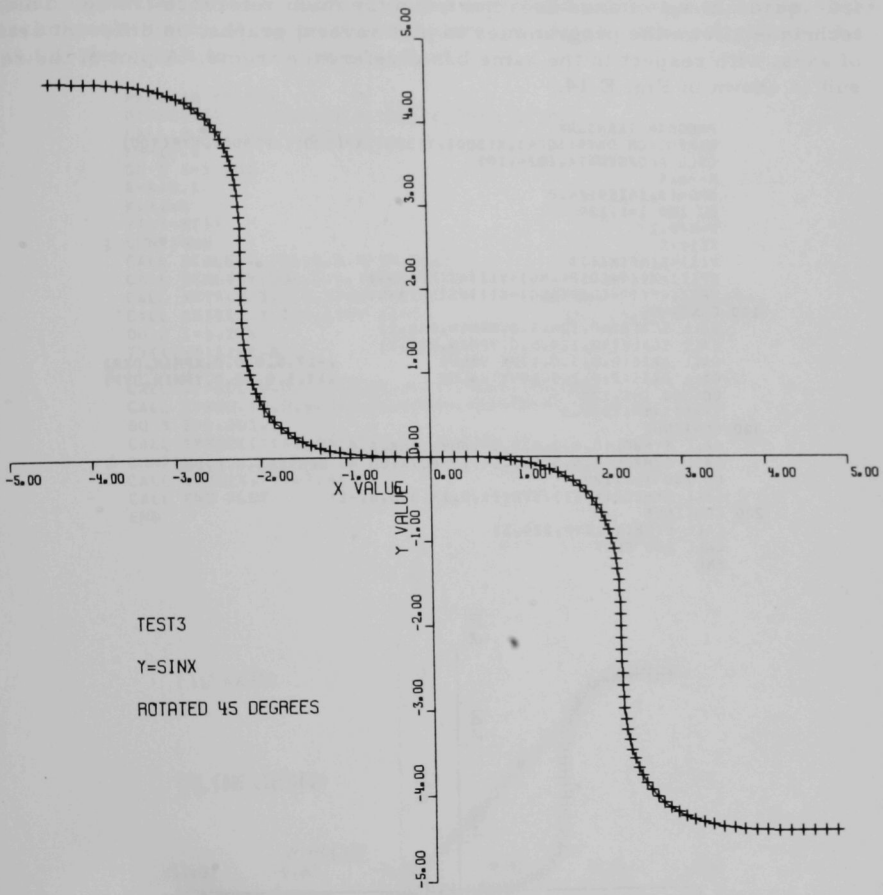


Fig. E.13. Plot Generated by Program TEST 3

m. Program TRANSGRF

This program also generates a rotated sine curve. However, the important difference between this program and the previous one is that the four-quadrant set of axes does not touch the main reference frame. This technique allows the programmer to plot several graphs, on different sets of axes, with respect to the same basic reference frame. A plot of the result is shown in Fig. E.14.

```
      PROGRAM TRANSGRF
      DIMENSION DATA(1024),X(300),Y(300),XP(300),YP(300),TYP(300)
      CALL PLOTS(DATA,1024,10)
      R=-6.4
      ANG=(3.14159)/4.0
      DO 100 I=1,129
      R=R+0.1
      X(I)=R
      Y(I)=SINF(X(I))
      XP(I)=X(I)*COSF(ANG)+Y(I)*SINF(ANG)
      YP(I)=Y(I)*COSF(ANG)-X(I)*SINF(ANG)
  100 CONTINUE
      CALL SCALE(XP,129,5.0,XPMIN,DXP,1)
      CALL SCALE(YP,129,5.0,YPMIN,DYP,1)
      CALL AXIS(0.0,5.0,17HX VALUE            ,-17,5.0,0.0,XPMIN,DXP)
      CALL AXIS(2.5,2.5,17HY VALUE            ,17,5.0,90.0,YPMIN,DYP)
      DO 300 I=1,129
      TYP(I)=YP(I)+2.5
  300 CONTINUE
      CALL SYMBOL(0.5,3.5,0.14,6HY=SINX,0.0,6)
      CALL SYMBOL(0.5,3.0,0.14,18HROTATED 45 DEGREES,0.0,18)
      DO 200 I=1,129
      CALL SYMBOL(XP(I),TYP(I),0.14,3,0.0,-1)
  200 CONTINUE
      CALL LINE(XP,TYP,129,1)
      CALL END PLOT
      END
```

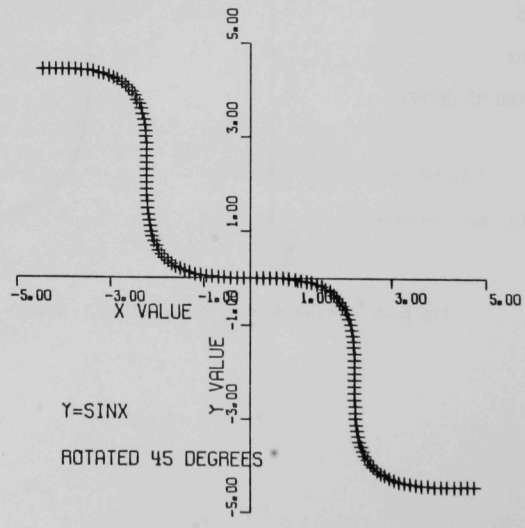


Fig. E.14. Plot Generated by Program TRANSGRF

n. Program TRLINE (TEST 2)

This program is a plot of a straight line on a four-quadrant set of axes that does not touch the main reference frame. However, this program shows the value of plotting only every third point of each array. This technique is useful if the points of the arrays are very close together. A plot of the result is shown in Fig. E.15.

```
      PROGRAM TRLINE
      DIMENSION DATA(3000),X(700),Y(700),TY(700)
      CALL PLOTS(DATA,1024,10)
      R=-10.1
      DO 1 I=1,201
      R=R+0.1
      X(I)=R
      Y(I)=X(I)
    1 CONTINUE
      CALL SCALE(X,201,5.0,XMIN,DX,3)
      CALL SCALE(Y,201,5.0,YMIN,DY,3)
      CALL AXIS(0.0,5.0,17HX VALUE            ,17,5.0,0.0,XMIN,DX)
      CALL AXIS(2.2,2.8,17HY VALUE            ,-17,5.0,90.0,YMIN,DY)
      DO 2 I=1,201
      TY(I)=Y(I)+2.8
    2 CONTINUE
      CALL SYMBOL(0.0,7.0,0.14,9HY(I)=X(I),0.0,9)
      CALL SYMBOL(0.0,6.0,0.14,13HTRLINE(TEST2),0.0,13)
      DO 3 I=1,201,3
      CALL SYMBOL(X(I),TY(I),0.14,3,0.0,-1)
    3 CONTINUE
      CALL LINE(X,TY,67,3)
      CALL END PLOT
      END
```

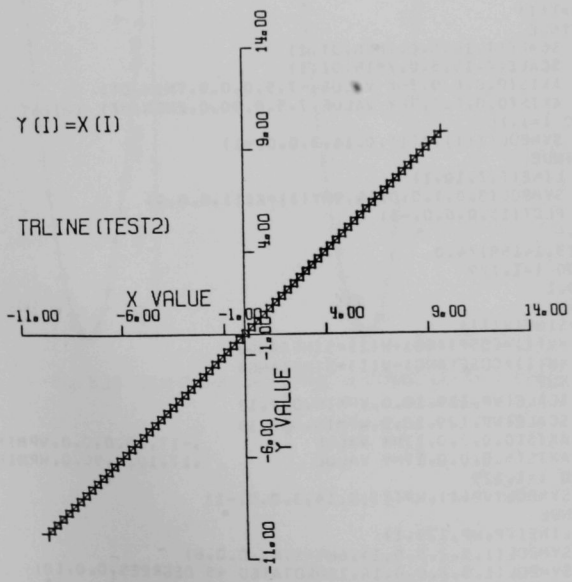


Fig. E.15. Plot Generated by Program TRLINE

o. Program COMBGRAF

This program illustrates the plotting of three individual curves, Y=SINX, Y=X, and Y=SINX rotated 45° from the negative X-axis. Note the change of the basis reference frame via the CALL PLOT routine. It shows that each plot is independent of the others once the CALL PLOT routine has set the new reference frame. The resulting plots are shown in Figs. E.16-E.18.

```
      PROGRAM COMBGRAF
      DIMENSION DATA(1024),X(260),Y(260),T(100),Z(100),V(260),W(260),
     1VP(260),WP(260)
      CALL PLOTS(DATA,1024,10)
      Q=-6.4
      DO 1 I=1,129
      Q=Q+0.1
      X(I)=Q
      Y(I)=SINF(X(I))
    1 CONTINUE
      CALL SCALE(X,129,10.0,XMIN,DX,1)
      CALL SCALE(Y,129,10.0,YMIN,DY,1)
      CALL AXIS(0.0,5.0,17HX VALUE            ,-17,10.0,0.0,XMIN,DX)
      CALL AXIS(3.5,0.0,17HY VALUE            ,17,10.0,90.0,YMIN,DY)
      DO 2 I=1,129
      CALL SYMBOL(X(I),Y(I),0.14,3,0.0,-1)
    2 CONTINUE
      CALL LINE(X,Y,129,1)
      CALL SYMBOL(2.0,2.5,0.14,6HY=SINX,0.0,6)
      CALL PLOT(15.0,0.0,-3)
      R=-0.5
      DO 10 I=1,10
      R=R+0.5
      T(I)=R
      Z(I)=T(I)
   10 CONTINUE
      CALL SCALE(T,10,5.0,TMIN,DT,1)
      CALL SCALE(Z,10,5.0,ZMIN,DZ,1)
      CALL AXIS(0.0,0.0,7HX VALUE,-7,5.0,0.0,TMIN,DT)
      CALL AXIS(0.0,0.0,7HY VALUE,7,5.0,90.0,ZMIN,DZ)
      DO 20 I=1,10
      CALL SYMBOL(T(I),Z(I),0.14,3,0.0,-1)
   20 CONTINUE
      CALL LINE(T,Z,10,1)
      CALL SYMBOL(3.0,1.0,0.14,9HY(I)=X(I),0.0,9)
      CALL PLOT(15.0,0.0,-3)
      S=-6.4
      ANG=(3.14159)/4.0
      DO 100 I=1,129
      S=S+0.1
      V(I)=S
      W(I)=SINF(V(I))
      VP(I)=V(I)*COSF(ANG)+W(I)*SINF(ANG)
      WP(I)=W(I)*COSF(ANG)-V(I)*SINF(ANG)
  100 CONTINUE
      CALL SCALE(VP,129,10.0,VPMIN,DVP,1)
      CALL SCALE(WP,129,10.0,WPMIN,DWP,1)
      CALL AXIS(0.0,5.0,17HX VALUE            ,-17,10.0,0.0,VPMIN,DVP)
      CALL AXIS(5.0,0.0,17HY VALUE            ,17,10.0,90.0,WPMIN,DWP)
      DO 200 I=1,129
      CALL SYMBOL(VP(I),WP(I),0.14,3,0.0,-1)
  200 CONTINUE
      CALL LINE(VP,WP,129,1)
      CALL SYMBOL(1.5,2.5,0.14,6HY=SINX,0.0,6)
      CALL SYMBOL(1.5,2.0,0.14,18HROTATED 45 DEGREES,0.0,18)
      CALL END PLOT
      END
```

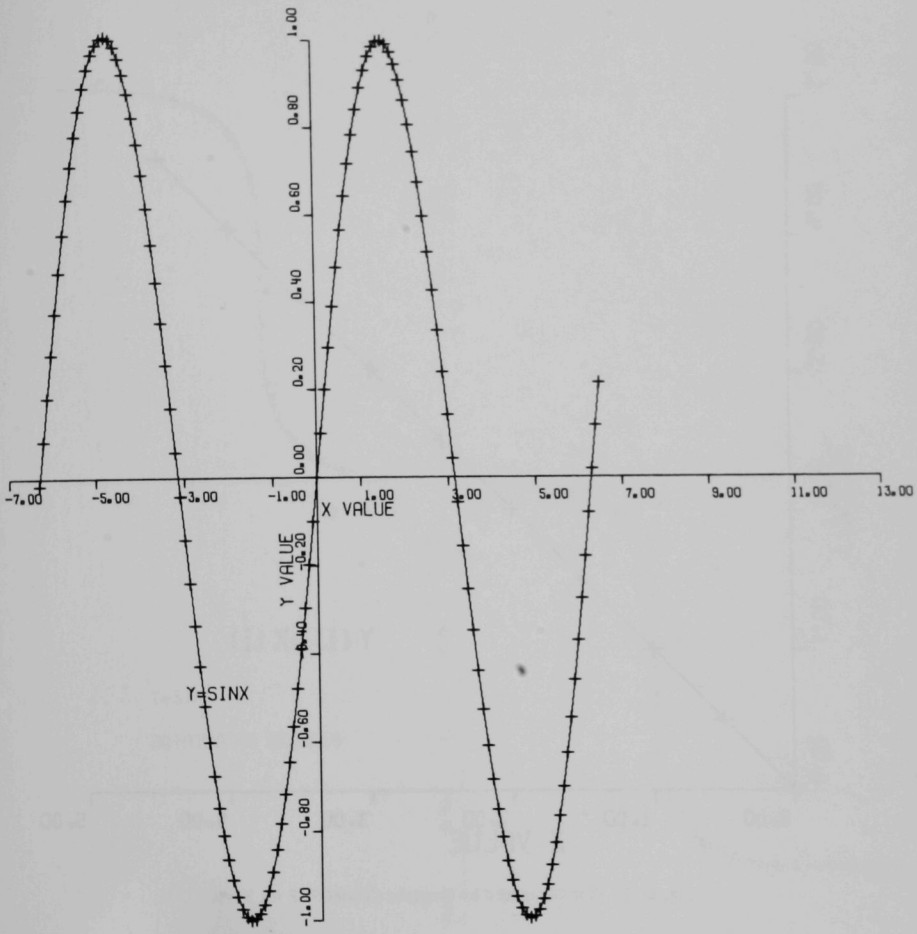


Fig. E.16. Plot Generated by Program COMBGRAF for Y = SINX

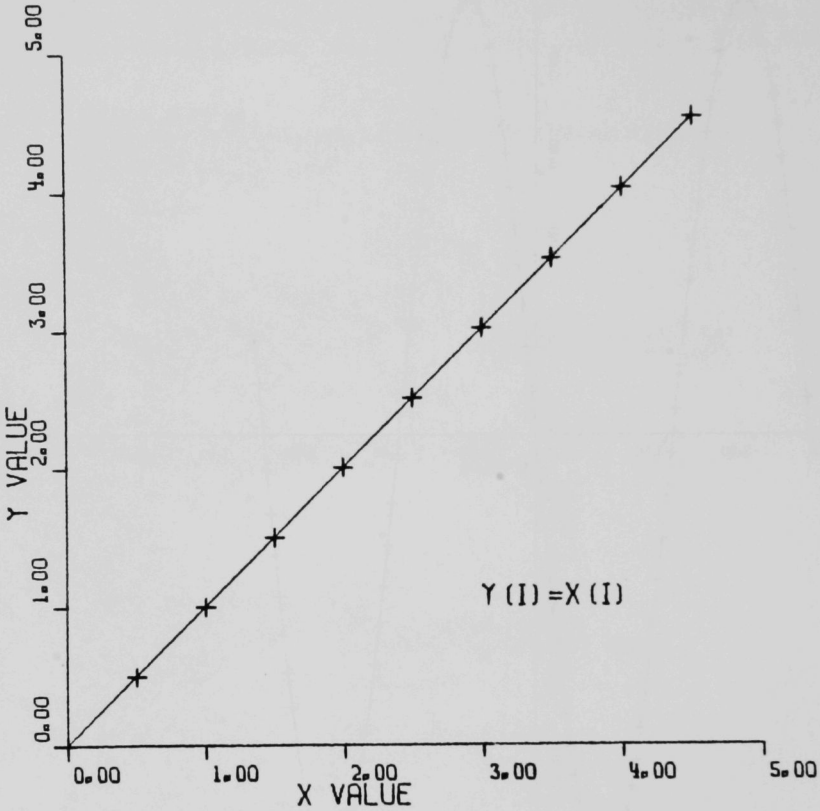


Fig. E.17. Plot Generated by Program COMBGRAF for Y = X

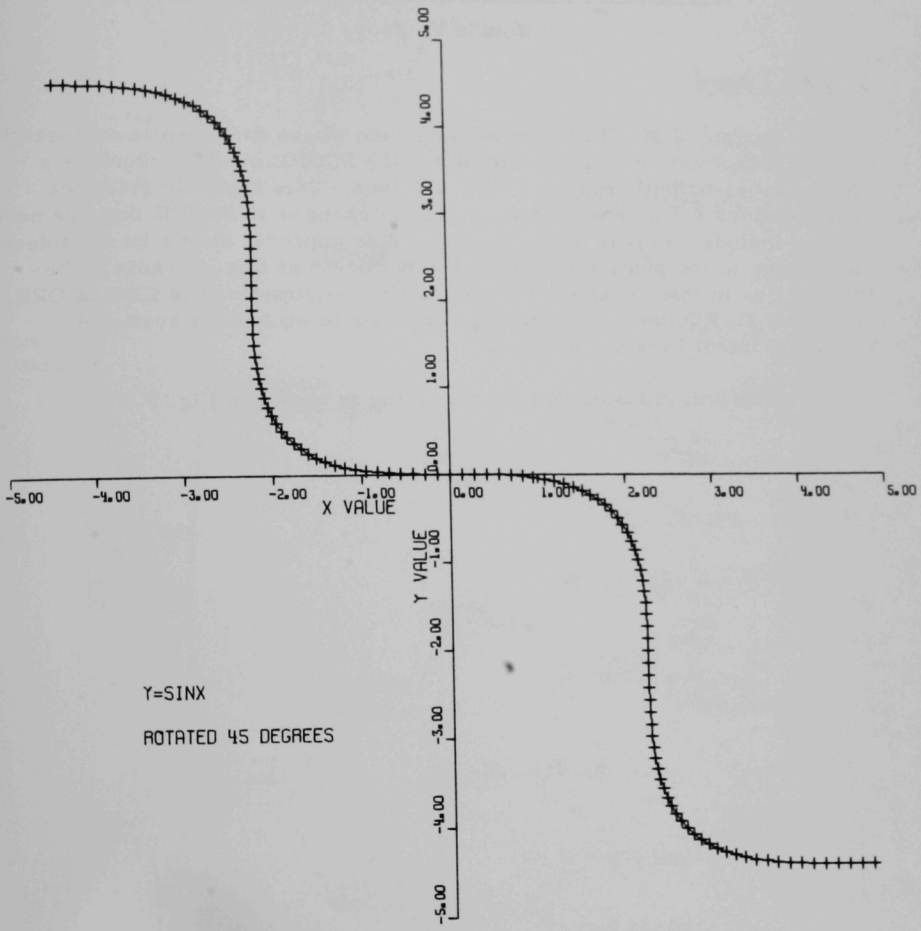


Fig. E.18. Plot Generated by Program COMBGRAF for Y = SINX Rotated 45°

APPENDIX F

Specifications for Input/Output Drivers for the SCOPE Monitor on the CONTROL DATA 3600†

Ronald F. Krupp

1. Introduction

For any CDC-3600 peripheral device whose data format or operations are not compatible with magnetic tapes, the SCOPE monitor requires a special input-output†† routine called a driver. This appendix gives the specifications for a general driver and the changes to SCOPE that are necessary to include a driver in the system. This appendix should be considered an addition to the publications listed in Section 5 of this appendix. The information in this appendix reflects driver requirements in CDC SCOPE, version 6.1. Equipment numbers given refer to equipment available through Control Data Corporation.

A diagram of SCOPE I/O processing is shown in Fig. F.1.

---

†Originally published for internal distribution only as ANL-AMD Technical Memorandum No. 119 (Feb 1966).

††The standard abbreviation, I/O, is used for input-output throughout this report.

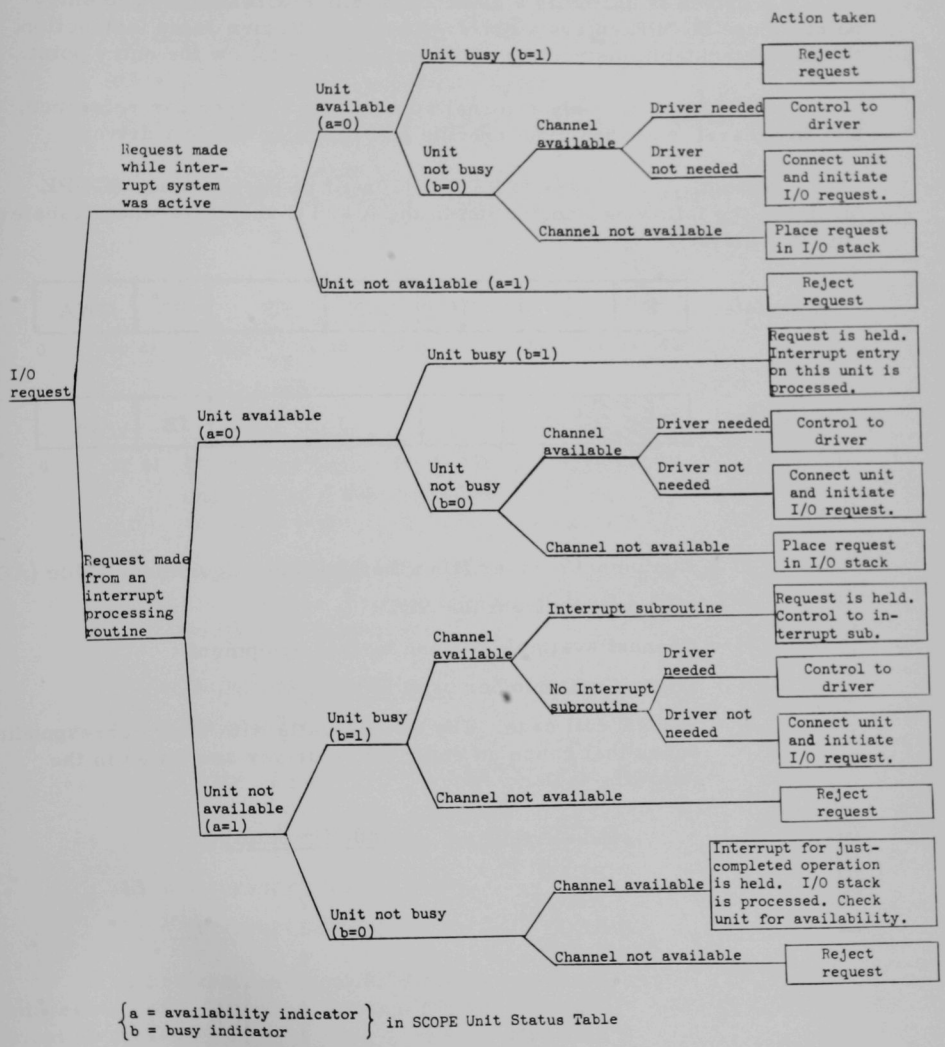


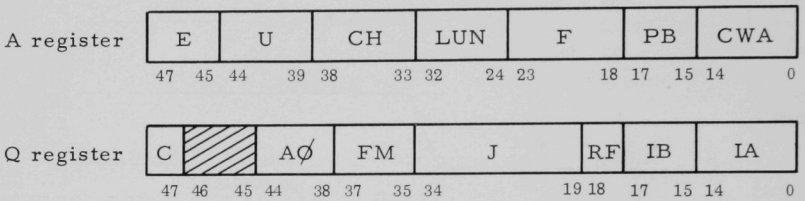
Fig. F.1. Diagram of SCOPE I/O Processing

2. Driver Specifications

A driver is normally a closed subroutine with exactly one entry point. Since SCOPE enters a driver via a Bank Return Jump instruction, the first executable instruction must immediately follow the entry point.

SENTRY is the only external symbol that a driver can reference, but, in general, one should not execute SCOPE calls within a driver.

Normally, no access to RESIDENT will be needed since SCOPE furnishes the following information in the A and Q registers when transferring to a driver:



Here

E = Equipment number from the Available Equipment Table (AET)

U = Unit number from the AET

CH = Channel available for use by this equipment

LUN = Logical unit number used in the call

F = SCOPE call code. The SCOPE calls with their corresponding codes that cause an entry into a driver are listed in the following table:

SCOPE call	Code (in octal)
READ	1
WRITE	2
REØT	3
WEØT	4
BSPR	5
BSPF	6
REWIND	7
UNLØAD	10
SKIP	11
ERASE	12
MARKEF	13

PB = Program bank from which the request was given

CWA = Control word address

C = 1, connect needed

AØ = AET ordinal of the physical unit

FM = Format bits from the Running Hardware Table (RHT)

The format bits are set by executing a SCOPE MODE request. The MODE request mnemonics and the corresponding format-bit settings are given below:

BIN	XX0
BCD	XX1
LØ	01X
HI	10X
HY	11X

Initially, the format bits are 000, which is interpreted by SCOPE as BINARY with density set manually at the tape unit. Once a density MODE call, LØ, HI or HY, has been executed there is no way to return to a 00X format.

The format bits may be used as special signals to a driver, for example, to pack the data received from a device into a particular form.

J = Reject increment

RF = Reject flag

If the request is rejected, then for

$$RF = \begin{cases} 0, \text{ control is given to the reject address} \\ 1, \text{ the job is abandoned} \end{cases}$$

IB = Interrupt address bank

IA = Interrupt address.

Further references to RESIDENT can be made by using absolute addresses. If a driver produces a Bounds Fault, the Bounds Fault interrupt must be cleared.

SCOPE enters a driver in Interrupt Mode, i.e., with the interrupt system deactivated; therefore a driver will not be interrupted. A driver processes a SCOPE request by performing the following tasks:

a. Initializing
b. Connecting the Equipment
c. Processing the User's Request
d. Translating the Data for Core-to-Device Transmission
e. Checking the I/O Control Word
f. Modifying the Interrupt Table (ITAB)
g. Readying the Equipment
h. Initiating the I/O Operation
i. Returning Control to SCOPE
j. Reject Processing
k. Error Processing
l. External Equipment Interrupt Processing
m. Aborting a Job

These tasks are described in the following paragraphs.

a. Initializing

All index registers the driver uses (and the D register, if it is used) must be saved and restored.

If the channel, equipment, and unit numbers associated with the peripheral device are fixed (i.e., will never change), they may be assembled into the I/O instructions. Otherwise, the driver must use the parameters given in the A register to set up the I/O instructions.

b. Connecting the Equipment

The driver should normally connect the equipment.† When a Connect instruction is executed, most equipment will send a reply or reject signal back to the data channel. Some devices (for example, the 3692 Typewriter) are incapable of responding; for equipment of this type, the driver must assume successful connection. The procedure for handling rejects is described under Reject Processing, paragraph j below.

A reject on a Connect instruction will occur under one or more of the following conditions:

1) Channel Busy. The selected channel is currently performing a READ or WRITE.

2) False Reference. The referenced equipment or unit is not attached to that channel or does not have its power turned on.

3) Unit Unavailable. The unit is currently being used by another source. This can only occur if the equipment is multichanneled.

---

†If the SCOPE call is not a request for data transmission, the driver may be programmed to set the Status Reply Bits arbitrarily without connecting the equipment (see Returning Control to SCOPE, paragraph i below). If the equipment is connected and control is returned while the channel is busy, SCOPE will set the Status Reply Bits from the dynamic channel status.

c. Processing the User's Request

Since the SCOPE calls are merely signals to the driver indicating that a particular operation is to be performed, one need not adhere to the literal meaning of the call, but it is advisable to treat the READ request as a device-to-core transmission of data and a WRITE request as a core-to-device transmission.

Some calls may be used to request special operations in which no data are transmitted; for example, a driver written for a cathode-ray-tube camera might interpret the MARKEF call as a request to advance the film one frame.

All calls that are not used for data transmission or special operation request may be ignored (see Returning Control to SCOPE, paragraph i below), or they may be treated as illegal requests to cause the job to be aborted.

d. Translating the Data for Core-to-Device Transmission

For some equipment, a driver may have to translate the data into a special format.

A buffer area internal to the driver may be used to hold the translated data during transmission. Using an internal buffer will require that:

1) The number of words transmitted under one SCOPE call be limited by the number of words that the buffer can hold, or

2) The driver retain control until all of the data has been translated and transmitted, or

3) The driver modify the Interrupt Table to regain control at the end of each transmission until all the data have been translated and transmitted (see Modifying the Interrupt Table (ITAB), paragraph f below).

Another method of translation used in core-to-device transmission is to store the translated data into the original data cells and write from these locations. If this method is used, the data will be in the translated form when control is returned to the user, unless the driver retains control or modifies the Interrupt Table to regain control at the end of the transmission to restore the data to the original form.

e. Checking the I/O Control Word

The I/O Control Word may have to be restricted not only in word count, as mentioned above, but in the types of Control Words that can be used. For example, an IØTR Control Word cannot be used when reading from the 3632/828 Disk File since the Disk File sends an End of Record signal to the data channel after reading each 32-word sector; only one sector would be read, no matter how large the word count.

When a Control Word violates a restriction, the driver can do any one of the following things:

1) Abort the job.
2) Change the Control Word.
3) Reject the request (see Reject Processing, paragraph j below).
4) Treat the request as an error (see Error Processing, paragraph k below).

If the Control Word in the request is used, the driver must store the Control Word address in the I/O instruction.

f. Modifying the Interrupt Table (ITAB)

A driver may need to regain control at the end of the I/O operation to do further processing, e.g., translating the data read into a particular format. By storing a Bank Return Jump instruction to the driver's interrupt routine in ITAB+16+C, control will be transferred to the driver's interrupt routine at the end of the I/O operation. ITAB is a location in RESIDENT, and C is the channel number to which the equipment is attached. Before transferring into the interrupt routine, SCOPE clears the "normal" and "abnormal" end-of-operation interrupts and places the Control Word in A and the Status in Q.

When processing of the interrupt is completed, the driver normally returns to SCOPE through the interrupt routine's entry point. However, if the user has an interrupt routine (i.e., IA≠0), the driver must enter this interrupt routine via a Bank Return Jump instruction with the Control Word in A and the Status in Q. After the user finishes processing the interrupt, control will be returned to the driver at the instruction immediately following the Bank Return Jump instruction. The driver then returns to SCOPE through the interrupt routine's entry point.

g. Readying the Equipment

The driver readies the equipment by executing the necessary External Functions. Usually the equipment is readied only for the calls that request data transmission or device manipulation.

If the driver returns control to SCOPE before the I/O operation is completed, the normal and abnormal end of operation interrupts should be selected. Otherwise, these interrupt selections should be released; i.e., the particular conditions should not interrupt the computer systems.

When an External Function instruction is executed, most equipment will send a reply or reject signal back to the data channel. For equipment that is not capable of returning a response, the driver must assume that an External Function is completed successfully.

A reject on an External Function will occur under one or more of the following conditions:

1) No unit or equipment is connected.

   The referenced device is not connected to the system and cannot recognize a Function instruction.

2) The code is illegal.

   The External Function code cannot be interpreted by the specified device.

3) The equipment is busy or not ready.

   The device cannot perform the operation without damaging the equipment or losing data.

4) The channel is busy.

   The selected channel is currently performing a READ or WRITE operation.

h. Initiating the I/O Operation

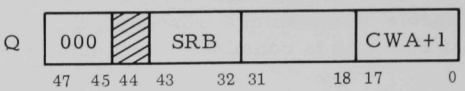
Executing a Begin Read or Begin Write instruction starts the I/O operation, and most equipment will send a reply or reject signal back to the data channel. For a device that is not capable of returning this response, the driver must assume the I/O has been started. A reject occurs when the channel is busy.

i. Returning Control to SCOPE

If control is to be returned to SCOPE immediately following the initiation of the I/O operation, the driver should set bits 47 and 46 of the Q register to "11" (the rest of the Q and A registers are irrelevant and need not be set to a particular value). Control is then transferred to SCOPE through the driver's entry point.

If the driver retains control until the transmission is complete, the equipment Status must be taken, the Status Reply Bits set as necessary, and the A and Q registers set in the following manner before returning to SCOPE:

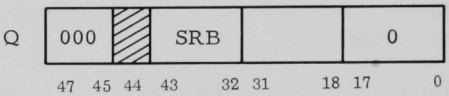
A = last control word in the chain



where SRB = Status Reply Bits, and CWA = address of last control word.

For those calls that request special operation, the driver should set the A and Q registers in the following manner before returning to SCOPE:

A = 0



where SRB = Status Reply Bits.

For some devices, a driver may need to change the Status Reply Bits. In particular, bit 1 is interpreted by SCOPE as an equipment busy bit,

$$\text{bit } 1 = \begin{cases} 0, \text{ not busy} \\ 1, \text{ busy} \end{cases}.$$

SCOPE may check this bit without checking the type of equipment involved.

A driver may use the Status Reply Bits to pass information back to the user's routine, e.g., to indicate the specific cause for a reject or an error in transmission.

j. Reject Processing

1) Internal Reject

For most equipment, an Internal Reject is generated in the following cases:

a) In the execution of Connect and External Function instructions, the equipment or unit referenced is not attached to the specified channel or does not have its power on.

b) In the execution of Read or Write instructions, the 3602 specified by the upper octal digit of the channel designator is not attached to the system.

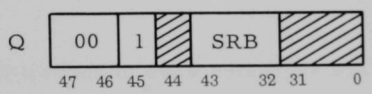
c) Failure in the data channel prevented generation of an External Reject signal.

When an Internal Reject occurs on equipment of this type, the driver should abort the job.

Some devices, for example the DD80, cause an Internal Reject for any reject condition. When an Internal Reject occurs on equipment of this type, the driver should clear the Internal Reject bit in the Interrupt Register and process the reject as an External Reject.

2) External Reject

When an External Reject occurs, the driver should copy the Status, change the Status Reply Bits as necessary, set the Q register in the following manner (the A register is irrelevant and need not be changed):



where SRB = Status Reply Bits, and return control to SCOPE through the driver's entry point.

A driver may retry an operation that has caused an External Reject.

k. Error Processing

When an I/O error occurs during transmission, the driver should do one of the following:

1) Retry the operation a finite number of times. (If unsuccessful, do 2, 3, or 4 below.)

2) Treat the error as a normal end of I/O operation, set the Status Reply Bits to indicate an error, and let the user process the error.

3) Reject the request, set the Status Reply Bits to indicate an error, and let the user process the error.

4) Abort the job.

1. External Equipment Interrupt Processing

Some devices (for example, the 3692 Typewriter), when not connected to a data channel are capable of interrupting the 3600. An interrupt of this type is referred to as an External Equipment Interrupt.

When an External Equipment Interrupt occurs, SCOPE transfers control, via a Bank Return Jump instruction, to the interrupt routine whose entry point is given in the EXTAB table (see SCOPE EQAIOC Table Entries, Section 3.b below).

If a driver for equipment of this type is assembled as part of RESIDENT, the entry point of its interrupt routine should be assembled into EXTAB. Otherwise, the installation may define one of the unused SCOPE calls as a signal to plant an address into EXTAB, and the user must execute this call at the start of his job.

The interrupt must be cleared in the interrupt routine, and when interrupt processing is completed, control must be returned to SCOPE through the interrupt routine's entry point.

m. Aborting a Job

The following are suggested ways to abort a job from a driver:

If the driver is not part of RESIDENT, a diagnostic should be written on the Standard Output Unit to indicate the cause for termination. (Although WRITE calls should not be used within a driver, the execution of a WRITE at this time will not cause any trouble as control will not be returned to the main program.) The driver then may execute an illegal SENTRY call (e.g., call code = -0), so that SCOPE can terminate the job abnormally.

If the driver is part of RESIDENT, a Return Jump to ABNERR (RTJ ABNERR) may be executed; ABNERR is a symbol internal to RESIDENT. The word immediately following this Return Jump to ABNERR must be defined as an Entry Point, e.g., EP. Then, when the driver is assembled into RESIDENT, the following sequence of code must be assembled in the RDPLIST in the SCOPE routine BOOT:

```
  EXT    EP
+ 0      EP
  BCD    2,M
```

where EP is the Entry Point mentioned above, and M is a two-word BCD message. M will be written as the Recovery Dump diagnostic after the abnormal termination (RTJ ABNERR) is executed.

3. Changes to SCOPE

a. Hardware Mnemonics and Hardware Codes

A mnemonic, consisting of two BCD characters, the first of which is alphabetic, must be defined by the octal code assigned to the hardware type. For example, assembling the following instruction into EQAIOC

```
TV    EQU    13B
```

will define TV to be the hardware code type 13_8 (the code assigned to Argonne's DD80). The following table lists Argonne's hardware types with their corresponding mnemonics and octal codes.

Hardware Type	Mnemonic	Octal Code
Magnetic Tape Unit	MT	01
(Reserved for tape-like equipment)	-	02-03
Card Reader	CR	04
Card Punch	CP	05
Line Printer	LP	06
Paper Tape Station	PT	07
Typewriter	TY	10
Disk File	DF	11
Drum	DR	12
CRT Display (DD80)	TV	13
Plotter	PL	14
(Not assigned)		15-17
3682 Satellite	SL	20
Equipment associated with Satellite 1	SA	21
Equipment associated with Satellite 2	SB	22
.	.	.
.	.	.
.	.	.
Equipment associated with Satellite 6	SF	26
(Not assigned)	27-57	
ASI-210	CH	61
ASI-2100	PH	62
3692 Typewriter	RS	63
(Reserved for use by individual installations)		64-77

Types 21-26 are defined within the SATELLITE system.

b. SCOPE EQAIOC Table Entries

1) QHDCTB Table

The hardware mnemonic is entered in the QHDCTB table. This table is ordered by hardware type, four entries per word. For example, the third word in the table would be

BCD 1,DFDRTVPL

where DF = Disk File, DR = Drum, TV = DD80, and PL = Plotter.

2) Available Equipment Table (AET)

The AET table is ordered by hardware type, and an entry is made by using the following CØMPASS Macro in the corresponding position of the AET.

AET (HARD,IØS,SATD,EXT,CØN,SC,AVAIL,E,UU,D)

where HARD = hardware mnemonic

IØS =
- IØ, unit is accessible to SCOPE, and capable of doing input and output operations.
- I, unit is accessible to SCOPE, and can do input only.
- Ø, unit is accessible to SCOPE, and can do output only.
- IOS, unit is accessible only to Satellite, and capable of doing input and output operations.
- IS, unit is accessible only to Satellite, and can do input only.
- ØS, unit is accessible only to Satellite, and can do output only.

SATD = A six-bit octal code defining the Satellite driver ordinal. The equipment and the corresponding octal codes are listed below:

Equipment	Octal Code
3624 Magnetic Tapes	01
3641 Card Reader	04
3649 Card Reader	05
088 Card Reader (through 1610)	06
167-1 Card Reader	07
167-2 Card Reader	10

Equipment	Octal Code
177,1614 Card Reader	11
3644 Card Punch	12
170,523 Card Punch (through 1610)	13
3655/3659 Line Printer	14
166 Printer	15
1612 Printer	16
3692 Paper Tape Station	17
350/BRPE-11 Paper Tape	20
3692/731 Typewriter	21
161 Typewriter	22
165 Plotter	23
1619 Disk File	24
36XX Disk File	25
36XX Drum	26
36XX CRT Display	27
3682 Satellite	30
36XX Plotter	31
162/163 Magnetic Tape Controller	32

EXT = External Equipment Interrupt bit = { 0, no External Equipment Interrupt possible; 1, unit can interrupt }

CØN = Symbolic name of the corresponding CRLIST entry (see 3) below)

SC Depends upon the hardware type:

a) For hardware types (21-26), the SC field contains the hardware code.

b) For hardware types capable of producing an External Equipment Interrupt, the SC field contains the channel number through which this interrupt may occur.

b. SCOPE EQAIOC Table Entries

1) QHDCTB Table

The hardware mnemonic is entered in the QHDCTB table. This table is ordered by hardware type, four entries per word. For example, the third word in the table would be

BCD 1,DFDRTVPL

where DF = Disk File, DR = Drum, TV = DD80, and PL = Plotter.

2) Available Equipment Table (AET)

The AET table is ordered by hardware type, and an entry is made by using the following CØMPASS Macro in the corresponding position of the AET.

AET (HARD,IØS,SATD,EXT,CØN,SC,AVAIL,E,UU,D)

where HARD = hardware mnemonic

IØS =
- IØ, unit is accessible to SCOPE, and capable of doing input and output operations.
- I, unit is accessible to SCOPE, and can do input only.
- Ø, unit is accessible to SCOPE, and can do output only.
- IOS, unit is accessible only to Satellite, and capable of doing input and output operations.
- IS, unit is accessible only to Satellite, and can do input only.
- ØS, unit is accessible only to Satellite, and can do output only.

SATD = A six-bit octal code defining the Satellite driver ordinal. The equipment and the corresponding octal codes are listed below:

Equipment	Octal Code
3624 Magnetic Tapes	01
3641 Card Reader	04
3649 Card Reader	05
088 Card Reader (through 1610)	06
167-1 Card Reader	07
167-2 Card Reader	10

Equipment	Octal Code
177,1614 Card Reader	11
3644 Card Punch	12
170,523 Card Punch (through 1610)	13
3655/3659 Line Printer	14
166 Printer	15
1612 Printer	16
3692 Paper Tape Station	17
350/BRPE-11 Paper Tape	20
3692/731 Typewriter	21
161 Typewriter	22
165 Plotter	23
1619 Disk File	24
36XX Disk File	25
36XX Drum	26
36XX CRT Display	27
3682 Satellite	30
36XX Plotter	31
162/163 Magnetic Tape Controller	32

EXT = External Equipment Interrupt bit = { 0, no External Equipment Interrupt possible; 1, unit can interrupt }

CØN = Symbolic name of the corresponding CRLIST entry (see 3) below)

SC Depends upon the hardware type:

a) For hardware types (21-26), the SC field contains the hardware code.

b) For hardware types capable of producing an External Equipment Interrupt, the SC field contains the channel number through which this interrupt may occur.

c) For the cases not covered in a) and b) above, the SC field is initially set to 40B and is used to denote the channel through which the specified equipment is currently connected.

$$\text{AVAIL} = \text{Availability indicator} = \begin{cases} 0, \text{ unassigned, available} \\ 1, \text{ unassigned, down} \\ 2, \text{ assigned to SCOPE} \\ 3, \text{ assigned to the Satellite} \end{cases}$$

E = Equipment Number

UU = Unit Number

D = Symbolic name of the Driver Name Table entry.

For example, the AET entry for the DD80 is

AET (TV,Ø,27,O,CCRT,40B,0,6,0,DRDD80).

3) Controller List Table (CRLIST)

The CRLIST table is not ordered; an entry may be made by inserting the following Macro at the end of the table:

CØNTRØL (NAME,E,S_1,C_1,S_2,C_2,S_3,C_3,S_4,C_4,S_5,C_5)

where NAME is the symbolic name that will be assigned to the location field of this entry. C_i is a channel on which the unit can be connected by

$$\text{computer system } S_i,\ S_i = \begin{cases} \text{SS, SCOPE system} \\ \text{S1, Satellite system} \end{cases}.$$

For example, the Macro used for the DD80 could be

CØNTRØL (CCRT,6,SS,2)

which means the equipment is number 6 on channel 2 in the SCOPE system; CCRT points to the DD80 entry in the AET (see 2 above).

4) Driver Name Table (DNT)

The DNT is not ordered; an entry may be made by inserting the following Macro at the end of the table:

DRIVER (NAME, LØC)

where NAME is the symbolic name that will be assigned to the location field of this entry. If the driver is not part of RESIDENT, NAME must also be the

entry-point name of the driver, and the LØC field must be deleted. LØC is the entry-point name of the driver, if the driver is part of RESIDENT. For example, the entry for the DD80 driver could be either

DRIVER (DRDD80)

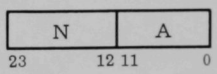
if the driver is not part of RESIDENT, or

DRIVER (DRCRT,DRDD80)

if the driver is part of RESIDENT.

5) Clear Interrupt Table (CLINTAB)

The CLINTAB is ordered by hardware type, two entries per word. The 24-bit entry contains the codes to clear normal and abnormal end-of-operation interrupts and should be given in the following form:



where N and A are External Functions as follows:

N = clear normal end-of-operation interrupt code;

A = clear abnormal end-of-operation interrupt code.

For example, the half-word in CLINTAB for a 3649 card reader would be

VFD Ø12/23, Ø12/35

where the External Functions are 23_8 (CLEAR NORMAL INTERRUPT) and 25_8 (CLEAR ABNORMAL INTERRUPT).

6) External Equipment Interrupt Table (EXTAB)

An entry into EXTAB is made only for an equipment capable of producing an External Equipment Interrupt. EXTAB is not ordered in any particular manner and an entry is made by inserting the following Macro at the end of the table:

XTAB (HARD,BANK,LØC)

where

HARD = hardware mnemonic,

BANK = program bank of interrupt routine,

and

LØC = entry-point name of the interrupt routine.

c. Placing the Driver in the System

A driver may be assembled into RESIDENT as part of EQAIOC; otherwise, the driver is first assembled to obtain a binary deck. Then by executing a Prepare Library job with the binary deck, a new SCOPE Master Library Tape may be produced with the driver inserted into the Driver File.

4. Loading a Driver at Run Time

By placing an EQUIP card, EQUIP,ℓ=h, at the beginning of the job deck, the user assigns the equipment indicated by the hardware mnemonic h to the logical unit ℓ. During the processing of this EQUIP card, SCOPE looks at the AET entry associated with the hardware to see if a driver is required. If one is needed, the corresponding DNT entry is checked to see whether the driver is in memory. If the driver is not in memory, SCOPE makes a loader call and the driver is loaded into core into the available area having the highest address.

5. Bibliography for Appendix F

a. Control Data 3600 Computer System, SCOPE/Reference Manual (June 1965), Publication No. 60053300, Rev. B.

b. Control Data 3600 Computer System, Reference Manual (Sept 1964), Publication No. 60024300.

c. External Reference Specifications, 3600 SCOPE, Control Data Applications Software Document (Nov. 20, 1964).

X1

ARGONNE NATIONAL LAB WEST
3 4444 00011296 1