National Aeronautics and
Space Administration

JSC- 16253
NOV 1979

Lyndon B. Johnson Space Center
Houston Texas 77058

NASA CR-
160395

EARTH OBSERVATIONS DIVISION

SPACE AND LIFE SCIENCES DIRECTORATE

"AS-BUILT" DESIGN SPECIFICATION FOR CAMS DEVELOPMENT DOT DATA SYSTEM (CDDDS)

TIRF 79-0004

Job Order 76-662

Prepared By
Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15800

September 1979

LEC- 14055

JSC-16253

"AS-BUILT" DESIGN SPECIFICATION FOR CAMS DEVELOPMENT DOT DATA SYSTEM (CDDDS)

TIRF 79-0004

Job Order 76-662

Prepared By

O. Wehmanen

APPROVED BY

LEC

10-10-79

J. I. Morrow, Supervisor
Scientific Applications Software Section

NASA

J. M. Sulester
Systems and Facilities Branch

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

September 1979

LEC- 14055

CONTENTS

Section Page

FIGURES

1. INTRODUCTION

This document describes the implementation details of that part of the CAMS Development Dot Data System which has been put under formal configuration control.

This consists of a data base, eight programs which can be used to maintain the data base, and fifteen subroutines which can be used in Fortran programs to process the data in various ways. This system supports the task of developing products to aid the analyst interpreter in identifying grid dots. Since this is a development system, the ability to quickly and easily change the application software is important. The programs described here contribute to this end.

2. APPLICABLE DOCUMENTS

- TIRF 79-0004, "File Storage Problem", March 1979.

3. SYSTEM DESCRIPTION

3.1 GENERAL

The CAMS Development Dot Data System (CDDDS) has 3 parts. The first is the data file and its index. The data file consists of records containing 256 Integer *2 words, usually spectral values for one channel for 209 dots or labels for 209 dots. The index file contains segment pointers for the data file. The second part of the system is a set of programs which can be used to maintain the data files. Third is a collection of subroutines which can be called by programs to use the stored data for exploratory and development tasks.

3.2 FILE DESCRIPTION

The data files for the CDDDS consist of 512 byte records, a size chosen to minimize I/O processing time (Fortran IV - Plus User's Guide pg 4-9). In order to avoid confusion with other files, these files were given the type .ACA, by Al C. Anderson, their designer. There are two different types of .ACA files, index files and data files.

3.2.1 THE INDEX FILE

An index file consists of duplicate record pairs. Each record is organized as 256 Integer *2 words. The first record pair, record 1 and 2 contains only 2 words:

Word 1 - The number of records in the index file

Word 2 - The number of records in the data file

All other words are zero.

The subsequent identical record pairs contain segment identifiers in odd words and pointers in even words:

Word 2n-1 The segment number for a segment which has been stored in the data file (or zero).

Word 2n A record number in the data file which contains data for the segment number in word 2n-1 (or zero).

One of these files may have pointers for 128 segments. An index file typically has 10 or fewer records.

3.2.2 THE DATA FILE

Each record of the data file contains 256 Integer *2 words. 209 of the words typically contain spectral values for one channel for 209 dots or labels for the 209 dots. The other 47 words contain identifying information, data base pointers and ancillary information.

The input sources do not always contain the ancillary information, so the ancillary information may be in error.

Data file record format.

Word	
1	Segment identification
2	Acquisition year
3	Acquisition day
4	0
5-7	Record type code
8	0
9-217	Dot specific data
218	0
219	Sun angle
220	Soil line greenness estimate
221	Satellite identification
222	Latitude degrees
223	Latitude min.
224	Longitude degrees
225	Longitude min.

226-253	0
254	Delete flag
255	Pointer-back
256	Pointer-forward

3.3 PROGRAMS FOR FILE MAINTENANCE

3.3.1 LOAD LABELS FROM CARD (CDLABLD)

3.3.1.1 Linkage

This program calls GETNAM entry GETCRD and PUT.

3.3.1.2 Interface

The interface with GETNAM is through the calling arguments the interface with PUT is through the calling arguments and the common blocks ANCIL and FILNAM.

3.3.1.3 Input

Data is read from cards in the default file FØR001.DAT.

For each segment to be loaded there must be a HEADER card with format:

COL 1-6 HEADER
COL 10-13 Segment number
COL 15-20 Code name, for example TRTH78

HEADER 1663 TRTH78
The dot labels may be entered in two different formats on DOT or LINE cards.

DOT card format (like ERIPS format):
COL 1-4 DOTb (b is blank)
COL 6 Label 1
COL 8 Label 2
COL 10-80 Dot numbers separated by one or more blanks.

For example, the card DOT W 1 37 123 201 assigns label W1 to dots 37, 123, and 201.

LINE card format:

COL	1-4	LINE
COL	10-11	Line number (1,2...,11)
COL	13-14	Label for first dot of line
COL	16-17	Label for second dot of line
COL	67-68	Label for 19th dot of line

For example, the card
LINE 06 W1 A3.
Assigns label W1 to dot number 96 or line 6 col 1 and label A3 to dot 97.

The last data card should have STOP in col 1-4. If this is missing, the statement CDLABLD STOP BAD DATA? will print on the operators terminal at the end of the job.

3.3.1.4 Output

This program creates a report (fig. 3.3.1.4) and writes new records into the ACA data file.

3.3.1.5 Storage

955 Bytes

3.3.1.6 Description

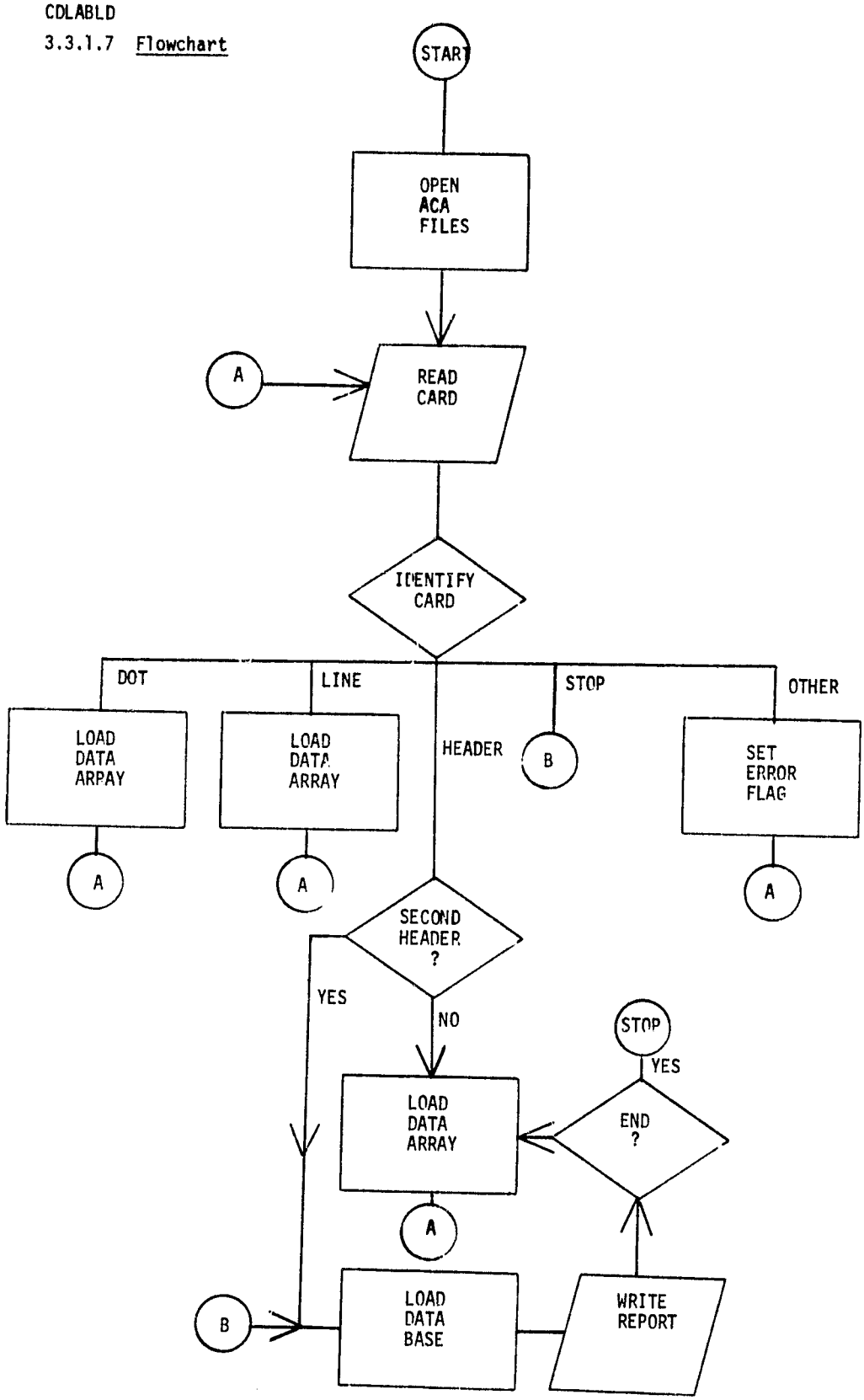
This program reads cards for one segment, loading the labels into an array. When it encounters the next HEADER card or a STOP card it calls PUT to load the data into the ACA data base.

```
CDLABLD UPDATE OF [200,7]DGTDATA.ACA
LABELS FOR SEG  1663 HAVE BEE. LOADED FROM CARD  CODE NAME CDLBLD STATUS   0
```


Figure 3.3.1.4

CDLABLD

3.3.1.7 Flowchart



```
      C CDLABLD.FTN
      C          CDLABLD  ?.A.NEUMANEN  FEB 1979
            IMPLICIT INTEGER*2 (A-L)
            IMPLICIT REAL (M-Z)
            INTEGER*2 ILB(12)
            INTEGER*2 STATUS,H,AL,DLBL(210),IN(71)
            INTEGER *4  SEG,DTE(2),TMP,GIN,IOC
            INTEGER*4 TEST(6),II(6),ATST
             BYTE BLLS,DASH,CHAR(11),INA(71),FAN(32),FIN(32),FDN(32)
             BYTE CRDSYM(6)
            EQUIVALENCE (DLBL(210),G)
            COMMON /ANCIL/CODE,DTE(2),STATUS,FILL(36)   !FOR PUT
            COMMON /FILNAM/ FDN,FIN
            EQUIVALENCE (DTE(1),DATE(1)) ,(SAT,FILL(4))
              FLAG=0
      D       WRITE(6,618)
  618         FORMAT(////,'  CDLABLD       DEBUG SWITCH ON!,/)
              BLANK=' '
             CAC=0
             KK=1
             KG =0
             CALL GETCRD(FAN,FDN,FIN,1)
             WRITE(6,6) FDN
    6        FORMAT(////'    CDLABLD UPDATE OF ',32A1)
      D     OPEN(UNIT=1,NAME='LAM.DAT',TYPE='OLD',READONLY,ERR=12)
   12          CONTINUE
             DO 19 J=1,200
              DLBL(J)=BLANK
   19        CONTINUE
      C   ANALIST INPUT DECK FROM FLOCON
             IF(FLAG .EQ. 2) GO TO 402   !A HEADER CARD IS IN INA
    1       CONTINUE
    2       READ(1,900,END=454)ATST,HL,AL,(INA(K),K=1,71)
  900       FORMAT(A4,X,A1,X,A1,X,71A1)
             IF(ATST .EQ. 'STOP') GO TO 454
      D        WRITE(6,620)ATST,AL,INA
  620        FORMAT('  ',A4,A ,1X,71A1)
            IF(ATST .EQ. 'HEAD') GO TO 401   !HEADER CARD
            IF(ATST .EQ. 'DOT ') GO TO 404   ! DOT CARD
            IF( TST .EQ. 'LINE') GO TO 405   !LINE CARD
             FLAG=5
            GO TO 1
      C               HEADER CARD INPUT
  402          FLAG=0
  401        IF (FLAG .EQ. 1)FLAG=2  !THERE IS MORE DATA, LOAD AND LOOP
             IF(FLAG .EQ. 2) GO TO 454   !GO LOAD
 4401       DECODE(15,2020,INA  SEG,CODENM
             FLAG=1
 2020       FORMAT(I4,X,6A1)
 2019       FORMAT('  ',I5,A2,I3,I3)
      D     WRITE(6,2019)SEG,CD,DTE(1),DTE(2)
            GO TO 1
      C               DOT CARD INPUT
  404       DECODE(71,2021,INA) IN
 2021       FORMAT(71I1)
      D     WRITE(6,622) ATST,IN
```


Figure 3.3.1.8 Listing for CDLABLD

```
0047     622    FØRMAT('  ',A6,2X,71I1)
0048            IX=0
0049             KK=KK+1
0050             IF(KK .GT. 40)KK=40
0051            DØ 4041 I=1,71
0052            IF(INA(I) .NE. ' ') IX=10*IX+IN(I)
0053            IF(INA(I) .EQ. ' ' .AND. IX .NE. 0) DLBL(IX)=AL+256*BL
0054    4041    IF(INA(I) .EQ. ' ') IX = 0
0055            GØ TØ 1
0056     405       DECØDE(59,2022,INA)LIN,ILB
0057    2022       FØRMAT(I2,19(X,A2))
0058               DØ 70 N=1,19
0059                  KK=K+19*(LIN-1)
0060                  DLBL(KK)=ILB(N)
0061      70       CØNTINUE
0062               GØ TØ 1
0063     454         CØNTINUE
0064                IF(SEG.EQ.0) STØP   !  SEG=0
0065                 CALL PUT(SEG,DLBL,CØDENM)
           D        TYPE 67,DLBL
0066      67        FØRMAT('  ',19A3)
0067              WRITE(6,68) SEG,CØDENM,STATUS
0068      68      FØRMAT('    LABELS FØR SEG ',I5,' HAVE BEEN LØADED FRØM CARD',
             *          '  CØDE NAME ',6A1,' STATUS',I4,/)
0069               IF(FLAG.EQ.1) STØP    'ØK'
0070               IF(FLAG.GE.3) STØP    'EAL DATA ?'
0071                 GØ TØ 12
0072     666         STØP
0073                 END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001330	364	RW,I,CON,LCL
2	$PDATA	00 024	16	RW,D,CON,LCL
3	$IDATA	000510	100	RW,D,CON,LCL
4	$VARS	001462	409	RW,D,CON,LCL
6	A%CIL	000120	40	RW,D,OVR,GBL
7	FILNAM	000100	32	RW,D,OVR,GBL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
AL	I*2	4-000716	ATST	I*4	4-001236	BL	I*2	4-001450	BLANK	I*2	4-001436	CODE	I*2	6-000000
DASH	L*1	4-00 243	FLAG	I*2	4-001454	GAG	I*2	4-001440	GIN	I*4	4-001146	I	I*4	4-001456
IDD	I*4	4-001152	IX	I*2	4-001454	J	I*2	4-001446	K	I*2	4-001452	KGN	I*2	4-001444
KK	I*2	4-001442	LIN	I*2	4-001466	M	I*2	4-000714	PLUS	L*1	4-001242	Q	R*4	4-000642
SAT	R*4	6-000016	SEG	I*4	4-001136	STATUS	I*2	6-000006	THO	I*4	4-001142			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
CHAR	L*1	4-001244	000013	5	(11)
CODENM	L*1	4-00 426	000006	3	(6)
DATE	I*2	6-000002	000004	2	(2)
DLBL	I*2	4-00 000	000e44	210	(210)
DTE	I*4	6-00 002	000030	12	(6)
FAN	L*1	4-001356	000040	16	(32)
FDN	L*1	7-000010	000040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	7-00 040	000040	16	(32)
IIN	I*4	4-001206	000030	12	(6)
ILB	I*2	4-001646	000046	19	(19)
IN	I*2	4-00 720	000216	71	(71)
INA	L*1	4-001257	000107	35	(71)
TEST	I*4	4-001156	000030	12	(6)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000174	2	**	6'	3-000000	12	1-000116	19	**
67'	**	68'	3-000104	70	**	401	1-000452	402	1-000440
404	1-000566	405	1-001014	454	1-001156	618'	**	620'	**
622'	**	666	**	900'	3-000040	2019'	**	2020'	3-000056
2021'	3-000066	2022'	3-000072	4041	**	4401	**		

FUNCTIONS AND SUBROUTINES REFERENCED

```
FORTRAN IV-PLUS V02-51          15122118   20-SEP-79          PAGE 4
CDLABLD,FTN     /TR:BLOCKS/WR

 GETCRD  PUT

TOTAL SPACE ALLOCATED = 003566   925

NO FPP INSTRUCTIONS GENERATED

CDLABLD,LP:=CDLABLD
```

3.3.2 LOAD SPECTRAL VALUES FROM CARD (CDLOAD)

3.3.2.1 Linkage

This program calls GETNAM entry GETCRD, PUT and the private subroutine HEX.

3.3.2.2 Interface

The interface with GETNAM is through the calling arguments.

The interface with PUT is through the calling arguments and through the common blocks ANCIL and FILNAM.

The interface with HEX is through the calling arguments.

3.3.2.3 Input

The input cards are read from the default file FØRC01.DAT. The first card has the code identifier (e.g SWCH1*) in col 1-6. Following is one or more decks generated by the Spectral Unload Utility on ERIPS.

3.3.2.4 Output

The output is a line printer report (fig 3.3.2.4) and new records in the ACA data file.

3.3.2.5 Storage

2599 Bytes

3.3.2.6 Description

The code name is read. Then spectral values are read, translated by HEX from hexidecimal alphanumeric to decimal integer and loaded into 4 arrays. When the arrays are full, they are loaded into the ACA data file by PUT and a new set of vlaues is read.

```
        C  CDLØAD
        C          LØADS SPECTRAL VALUES FRØM LACIE 9 CARDS
        C          I, HEX FØRMAT INTØ THE CDCUS
        C
0001            IMPLICIT INTEGER*2 (A-Z)
0002            INCLUDE 'INCLU.FTN'
      * C
      * C
      * C
0003  *           INTEGER*2 CH1(209),CH2(209),CH3(209),CH4(209)
0004  *           INTEGER*2 KH1(209),KH2(209),KH3(209),KH4(209)
0005  *           BYTE  FLAG(210),FA (32),FDN(32),FIN(32)
0006  *           INTEGER*2 SEC,DATE(2),CODE,STATUS,FILL(36)
0007  *           INTEGER*2 IAC,SLN,SAT,GAM,LAT(2),LNG(2)
0008  *             C MM1  /ANCIL/  CODE,DATE,STATUS,FILL
0009  *             C MM2  /DATA/  CH1,CH2,CH3,CH4
0010  *             C MM3  /KAUTH/  KH1,KH2,KH3,KH4
0011  *             C MM4  /FLAG/ F AG
0012  *             C MM5  /FILNAM/ FDN,FIN,FAN,KAN
0013  *             EQUIVALENCE (IA ,FILL(2)),(SLN,FILL(3))
0014  *             EQUIVALENCE (SAT,FILL(4)),(GAM,FILL(10))
0015  *             EQUIVALENCE (LAT(1),FILL(5)),(LNG(1),FILL(7))
      * C
      * C     END INCLU.FTN
      * C
0016            INTEGER*2 IN(4,6),CDNM(3)
0017            BYTE CARD(67)
        C
        C
        D        WRITE(6,99)
        D        TYPE 99
0018     99     FØ MAT(///,'     LØAD                DEBUG ØN',/)
0019            CALL GETCRD(FA ,FDN,FIN,1)
0020            WRITE(6,6) F
0021     6      FØRMAT(////'     CDL-AD UPDATE ØF ',32A1)
        D        ØPEN(UNIT=1,NAME='HEX,DAT',TYPE='ØLD',ERR=11)
0022     11         LLM=0
0023                CHK=0
0024                SEG=0
0025            READ(1,3,ERR=666,END=888) CDNM
0026     3      FØRMAT (3A2)
        C
0027     12   IF(NUM.EQ.34 .AND. CHK.EQ.1) GØ TØ 20
0028            READ(1,1,ERR=666,END=888) N,X,Y,F,CARD
0029     1      FØRMAT(I4,3I3,67A1)
0030            IF(C RD(65),NE,'A') GØ TØ 5
        C
12 0031             CHK=CHK+1
11 0032             IF(CHK.GT.1) STØP  'DECK ERRØR TØØ MANY HEADER CARDS'
10 0033             SEG=
 9 0034             IAG=X
 8 0035             SLN=Y
 7 0036             DATE(1)=2
 6 0037             D  87 I=1,3
 5 0038     87      CARD(I)=CARD(I)-'60
 4 0039             D TE(2)=10*CARD(1,+CARD(2)
 3
```


Figure 3.3.2.8

Listing for CDLOAD

```
0043                DATE(2)=10*DATE(2)+CARD(3)
0041                DO 88 I=1,6
0042                  J=10*I-5
0043                  CH1(I)=HEX(CARD(J),CARD(J+1))
0044                  CH2(I)=HEX(CARD(J+2),CARD(J+3))
0045                  CH3(I)=HEX(CARD(J+4),CARD(J+5))
0046                  CH4(I)=HEX(CARD(J+6),CARD(J+7))
0047     88         CONTINUE
0048                GO TO 12
         C
0049     5      CONTINUE
0050                NUM=NUM+1
0051                IF(NUM.GT.34) STOP  'DECK ERROR TOO MANY DATA CARDS'
0052                SET=X
0053                D1T=X
0054                D2T=Y
0055                SEG=2
0056                IF(SEG.EQ.SET.AND.D1T.EQ.DATE(1).AND.D2T.EQ.DATE(2)) GO TO 4
0057                   IF(SEG.EQ.0) GO TO 4
0058                   NUM=NUM+1
0059                   GO TO 12
0060     4          DO 44 I=1,6
0061                  J=6*SEG+I
0062                  K=10*I-9
0063                  IF(J.GT.209 .OR. J.LT.1)GO TO 88
0064                  CH1(J)=HEX(CARD(K),CARD(K+1))
0065                  CH2(J)=HEX(CARD(K+2),CARD(K+3))
0066                  CH3(J)=HEX(CARD(K+4),CARD(K+5))
0067                  CH4(J)=HEX(CARD(K+6),CARD(K+7))
0068     44         CONTINUE
         D            IF(NUM.EQ.5)TYPE 45,SET,D1T,D2T,CH1(J),CH2(J),CH3(J),CH4(J),J
0069     45           FORMAT(I8,2I3,2I4)
0070                GO TO 12
         C
0071     20         CONTINUE
         C          WRITE(6,66) SEG,DATE,CODNM
0072     66         FORMAT(' SEGMENT ',I5,'  DATE ',2I3,' HAS BEEN LOADED  ',3A2)
         D          TYPE 22,FDN,FIN
0073     22         FORMAT('   FDN=',32A1/,'      FIN=',32A1)
0074                CODNM(3)='1*'
0075                CALL PUT(SEG,CH1,CODNM)
0076                CODNM(3)='2*'
0077                CALL PUT(SEG,CH2,CODNM)
0078                CODNM(3)='3*'
0079                CALL PUT(SEG,CH3,CODNM)
0080                CODNM(3)='4*'
0081                CALL PUT(SEG,CH4,CODNM)
0082                IF(STATUS.EQ.0) WRITE(6,66) SEG,DATE,CODNM
12 0083             NUM=0
11 0084             GO TO 11
10 0085
 9 0086     666             STOP  'ERROR READING FILE 1'
 8 0087     888            WRITE(6,889)
 7 0088     889            FORMAT(///)
 6 0089                     STOP  'END OF DATA  NORMAL'
 5 0090                     END
 4
 3
```

PROGRAM SECTIONS

NUMBER	NAME		SIZE		ATTRIBUTES
1	SCODE1	001774	510		RW,I,CON,LCL
2	SPDATA	00 162	57		RW,D,CON,LCL
3	SIDATA	00 242	89		RW,D,CON,LCL
4	SVARS	00 226	75		RW,D,CON,LCL
5	STEMPS	00 004	2		RW,D,CON,LCL
6	A%CIL	000126	46		RW,D,OVR,GBL
7	DATA	00 210	836		RW,D,OVR,GBL
8	KAUTH	003216	836		RW,D,OVR,GBL
9	FLAG	00 322	105		RW,D,OVR,GBL
10	FILNAM	000112	49		RW,D,OVR,GBL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
CHK	I*2	4-000176	CODE	I*2	6-000000	D1T	I*2	4-000216	D2T	I*2	4-000220	GAM	I*2	6-000032
I	I*2	4-000210	IAG	I*2	6-000016	J	I*2	4-000212	K	I*2	4-000224	K.N	I*	10-000140
NUM	I*2	4-000174	SAT	I*2	6-000010	SEG	I*2	4-000000	SEQ	I*2	4-000222	SET	I*2	4-000214
SLN	I*2	6-000014	STATUS	I*2	6-000006	W	I*2	4-000200	X	I*2	4-000202	Y	I*	3-000204
Z	I*2	4-000206												

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
CARD	L*1	4-000070	000103	33	(67)
CH1	I*2	7-000000	000642	209	(205)
CH2	I*2	7-000642	000642	209	(205)
CH3	I*2	7-001304	000642	209	(205)
CH4	I*2	7-001346	000642	209	(205)
CPDNM	I*2	4-000262	000006	3	(3)
DATE	I*2	6-000002	000004	2	(2)
FAN	L*1	10-000100	000040	16	(32)
FDN	L*1	10-000120	000040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	10-000040	000040	16	(32)
FLAG	L*1	9-000000	000322	105	(210)
IN	I*2	4-000002	000060	24	(4,6)
KH1	I*2	8-000000	000642	209	(205)
KH2	I*2	8-000642	000642	209	(205)
KH3	I*2	8-001304	000642	209	(205)
KH4	I*2	8-001946	000642	209	(205)
LAT	I*2	6-000026	000004	2	(2)
LNG	I*2	6-000024	000004	2	(2)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS

1'	3-000042	3'	3-000036	4	1-001146	5	1-001016	6'	3-000000
11	1-000366	12	1-000150	20	1-001530	22'	**	47	**
45'	**	66'	3-000054	87	**	88	1-000770	99'	**
666	1-001720	666	1-001736	889'	3-000136				

FUNCTIØNS AND SUBRØUTINES REFERENCED

GETCRØ HEX PUT

TØTAL SPACE ALLØCATED = 012116 2539

NØ EPP INSTRUCTIØNS GENERATED

CDLOAD.FTN /TRACEBACKS/WR

```
0001          FUNCTION HEX(A,B)
0002          BYTE A,B
0003          INTEGER*2 HEX,AA,BB
0004          AA=A-"60
0005          IF(AA.EQ."20) AA=0
0006          IF(AA.GE.10)  AA=AA-7
0007          BB=B-"60
0008          IF(BB.EQ."20) BB=0
0009          IF(BB.GE.10)  BB=BB-7
0010          HEX=AA*16+BB
0011          RETURN
0012          END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	000136	47	Rw,I,CØN,LCL
3	$IDATA	00 002	1	Rw,I,CØN,LCL
4	$VARS	00 004	2	Rw,D,CØN,LCL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
HEX	I*2	1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
A	L*1	F-000002*	AA	I*2	4-000000	B	L*1	F-000004*	BB	I*2	4-000002			

TØTAL SPACE ALLØCATED = 000144 50

NØ FPP INSTRUCTIØNS GENERATED

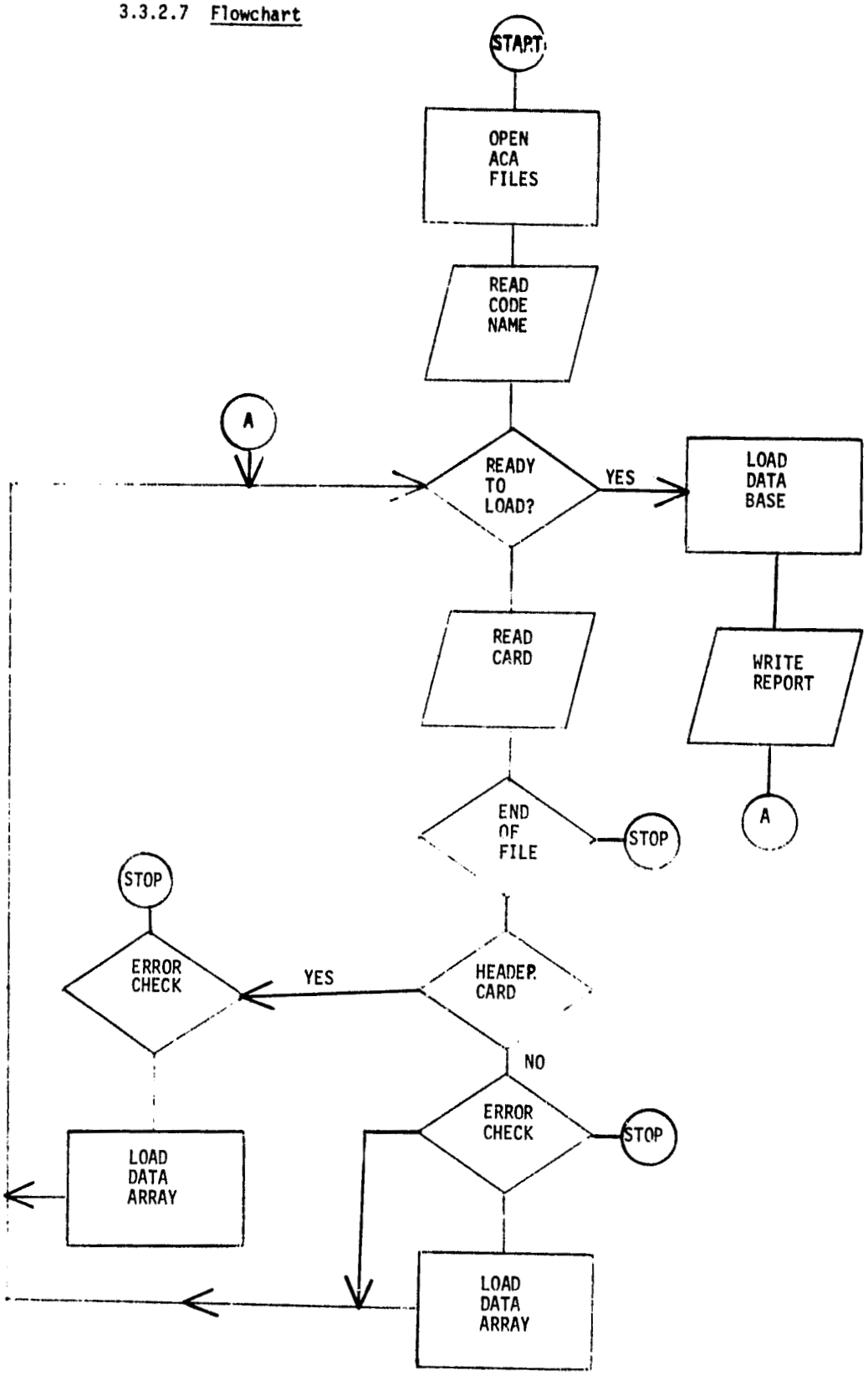
CDLØAD,LP1=CDLØAD

```
COLMAD UPDATE OF [200,71]D.TDATA,ACA
SEGMENT 1663 DATE 77123 HAS BEEN LOADED  CCST4*
```

Figure 3.3.2.4

CDLOAD

3.3.2.7 Flowchart



3.3.3 DUPLICATE FLAGGING (PURGE)

3.3.3.1 Linkages

Purge calls GETNAM entry GETCRD.

3.3.3.2 Interface

The interface with GETNAM is through the calling arguments.

3.3.3.3 Input

The ACA index and data files as named in FN.DAT are read.

3.3.3.4 Output

Duplicate records in the data file are marked and a Line Printer report (fig. 3.3.3.4) is generated.

3.3.3.5 Storage

1795 Bytes

3.3.3.6 Description

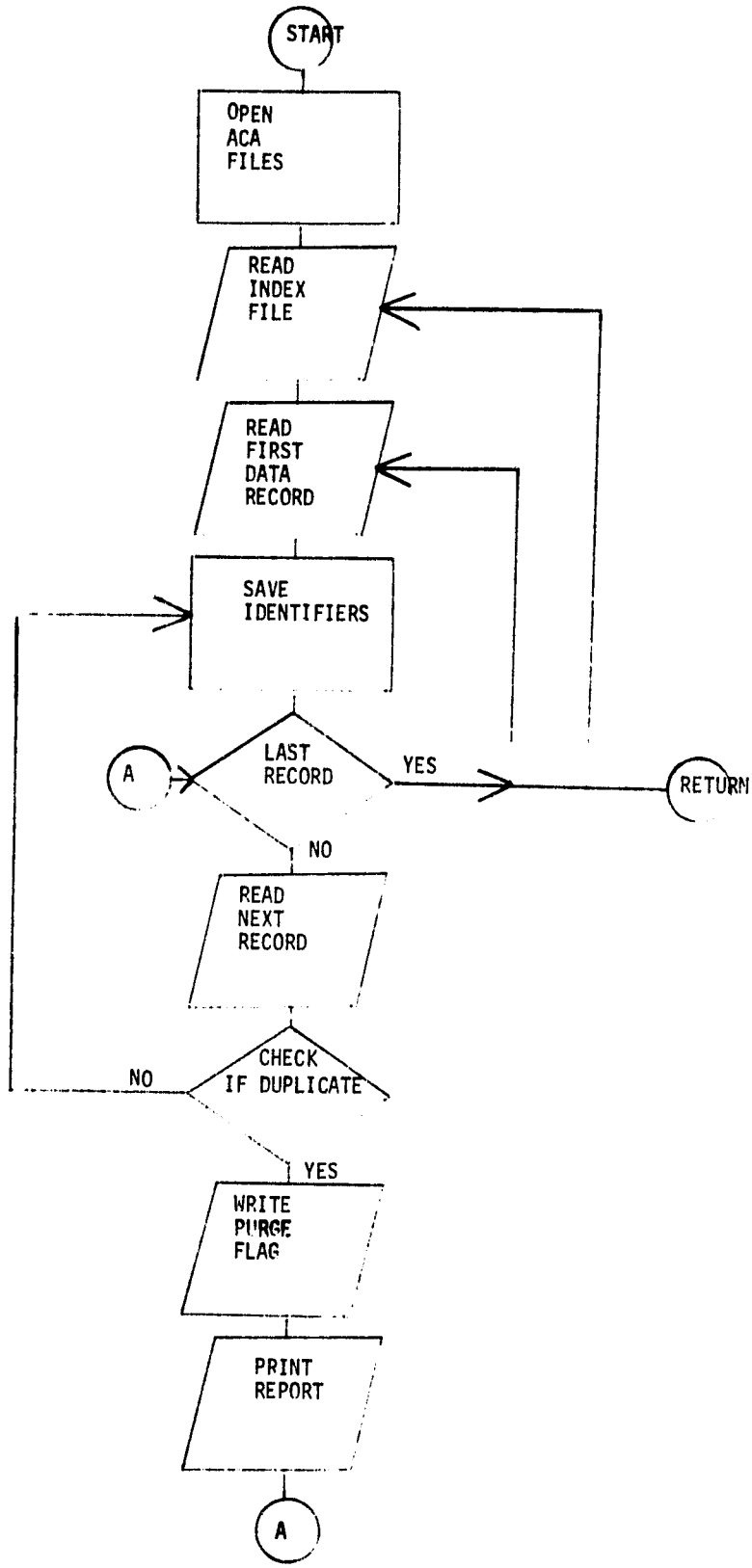
The records for each segment are searched for duplicates. When a duplicate is found, byte number 254 is set to .TRUE. Otherwise byte 254 is set to .FALSE. Other programs in the system ignore records marked .TRUE.

```
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH1* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH2* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH3* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
THE RECØRD NAMED S.CH4* FØR SEGMENT 1472 HAS BEEN MARKED FØR DELETIØN
```

Figure 3.3.3.4

PURGE

3.3.3.7 Listing



```
FØRTRAN IV-PLUS V02-51          15:08:45    24-SEP-79          PAGE 1
PURGE.FTN       /TH:BLØCKS/WE

0001            PRØGRAM PURGE
0002            IMPLICIT INTEGER*2 (A-Z)
0003            BYTE CHAR(2),FAN(32),FUN(32),FIN(32)
0004            DIMENSION RECLST(5,1"0), BUFFER(256), INDEX(256)
0005            DIMENSION MINDEX(256)
0006            DIMENSION DATE(2), NAME(3)
0007            PARAMETER T=,TRUE.
0008            PARAMETER F=,FALSE.
0009            EQUIVALENCE (CHAR(1),BUFFER(7)),(DATE(1),BUFFER(2))
0010            EQUIVALENCE (SEG,BUFFER(1)), (NAME(1),BUFFER(5))
0011            EQUIVALENCE (MINDEX(1),LINDEX)
0012             KIL=0
0013       10   CØNTINUE
0014            CALL GETCRD(FAN,FUN,FIN,1)
0015            ØPEN (UNIT=2,NAME=FIN,TYPE='ØLD',ACCESS='DIRECT',ERR=40)
0016            READ (2'1) MINDEX
0017            ØPEN (UNIT=3,NAME=FØN,TYPE='ØLD',ACCESS='DIRECT',ERR=40)
001e            DØ 40, K=3,LINDEX-1,5
0019              READ (2'K) INDEX
0020              DØ 40, I=2,256,2
0021              IF (INDEX(I),EQ,0) GØ TØ 50
0022              N = 0       ! NUMBER ØF NAMES IN RECLST
0023              READ (3'INDEX(I)) BUFFER    ! READ FIRST RECØRD
0024            IF (BUFFER(254)) GØ TØ 25
0025       20     N = N + 1
0026              RECLST(1,N) = BUFFER(5)
0027              RECLST(2,N) = BUFFER(6)
0028              RECLST(3,N) = BUFFER(7)
0029              IF (CHAR(2),'E,'*') GØ TØ 25
0030              RECLST(4,N) = BUFFER(2)
0031              RECLST(5,N) = BUFFER(3)
0032       25     CØNTINUE
0033              IF(RECNØ ,EQ, BUFFER(256)) GØ TØ 40
0034              RECNØ = BUFFER(256)
0035              IF (RECNØ,EQ,INDEX(I)) GØ TØ 40
0036              READ (3'RECNØ) BUFFER     ! READ NEXT RECØRD
0037              IF(SEG,NE,INDEX(I-1)) GØ TØ 37
0038            IF (BUFFER(254)) GØ TØ 25
0039              DØ 30, J=1,N
0040                IF (RECLST(3,J),NE,BUFFER(7)) GØ TØ 30
0041                IF (RECLST(2,J),NE,BUFFER(6)) GØ TØ 30
0042                IF (RECLST(1,J),NE,BUFFER(5)) GØ TØ 30
0043                IF (CHAR(2), E,'*') GØ TØ 35
0044                IF (DATE(2),'E,RECLST(5,J)) GØ TØ 30
0045                IF (DATE(1),'Q,RECLST(4,J)) GØ TØ 35
0046                IF (BUFFER(4),EQ,'**') GØ TØ 35
0047       30       CØNTINUE
12 0048           BUFFER(254)=F
11 0049           WRITE(3'RECNØ)  BUFFER
10 0050           GØ TØ 20
 9 0051       35  BUFFER(254) = T
 8 0052           WRITE (3'RECNØ) BUFFER
 7 0053           PRINT 100, NAME, SEG
 6 0054           GØ TØ 25
 5 0055       37  CØNTINUE
 4 0056           PRINT 101, SEG,INDEX(I-1),RECNØ,BUFFER(255),BUFFER(256)
```

Figure 3.3.3.8
Listing for PURGE

PURGE.FTN /TR:BLOCKS/WR

```
0057     101      FORMAT(' TROUBLE ',5I8)
0058              KILL=KILL+1
0059              IF(KILL.GT.100) STOP ' KILLED BY 100 POINTER ERRORS'
0060       40     CONTINUE
0061       50 CLOSE (UNIT=2)
0062          CLOSE (UNIT=3)
0063          STOP
0064     100 FORMAT (' THE RECORD NAMED ',3A2,' FOR SEGMENT ',I4,' HAS BEEN MAR
            ,KED FOR DELETION')
0065          END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	001314	350	RW,I,GBL,LCL
2	$PDATA	00 044	18	RW,D,GBL,LCL
3	$IDATA	00 276	95	RW,D,GBL,LCL
4	$VARS	005124	1322	RW,D,GBL,LCL
5	$TEMPS	00 004	2	RW,D,GBL,LCL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-005114	J	I*2	4-005122	K	I*2	4-005112	KIL	I*2	4-005110	LINDEX	I*2	4-000000
N	I*2	4-005116	RECNØ	I*2	4-005120	SEG	I*2	4-001000						

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
BUFFER	I*2	4-001000	001000	256	(256)
CHAR	L*1	4-001014	000002	1	(2)
DATE	I*2	4-001002	000004	2	(2)
FAN	L*1	4-002000	000040	16	(32)
FDN	L*1	4-002040	000040	16	(32)
FIN	L*1	4-002100	000040	16	(32)
INDEX	I*2	4-005110	001000	256	(256)
MINDEX	I*2	4-000000	001000	256	(256)
NAME	I*2	4-001010	000006	3	(3)
RECLST	I*2	4-002140	001760	500	(5,100)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
10	**	20	1-000304	25	1-000374	30	1-000652	35	1-000742
37	1-001056	40	1-001206	50	1-001260	100	3-000024	101	3-000100

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSS GETCRD ØPENS

TØTAL SPACE ALLØCATED = 007006 1795

NØ FPP INSTRUCTIØNS GENERATED

3.3.4 INTERACTIVE QUERY (QUERY)

3.3.4.1 Linkage

This program calls GETNAM.

3.3.4.2 Interface

The interface with GETNAM is through the calling arguments.

3.3.4.3 Input

This program reads the ACA index and data files. Control is from an interactive terminal.

3.3.4.4 Output

The program writes to the interactive terminal. An optional Line Printer report (fig 3.3.4.4) may be generated.

3.3.4.5 Storage

1683 Bytes

3.3.4.6 Description

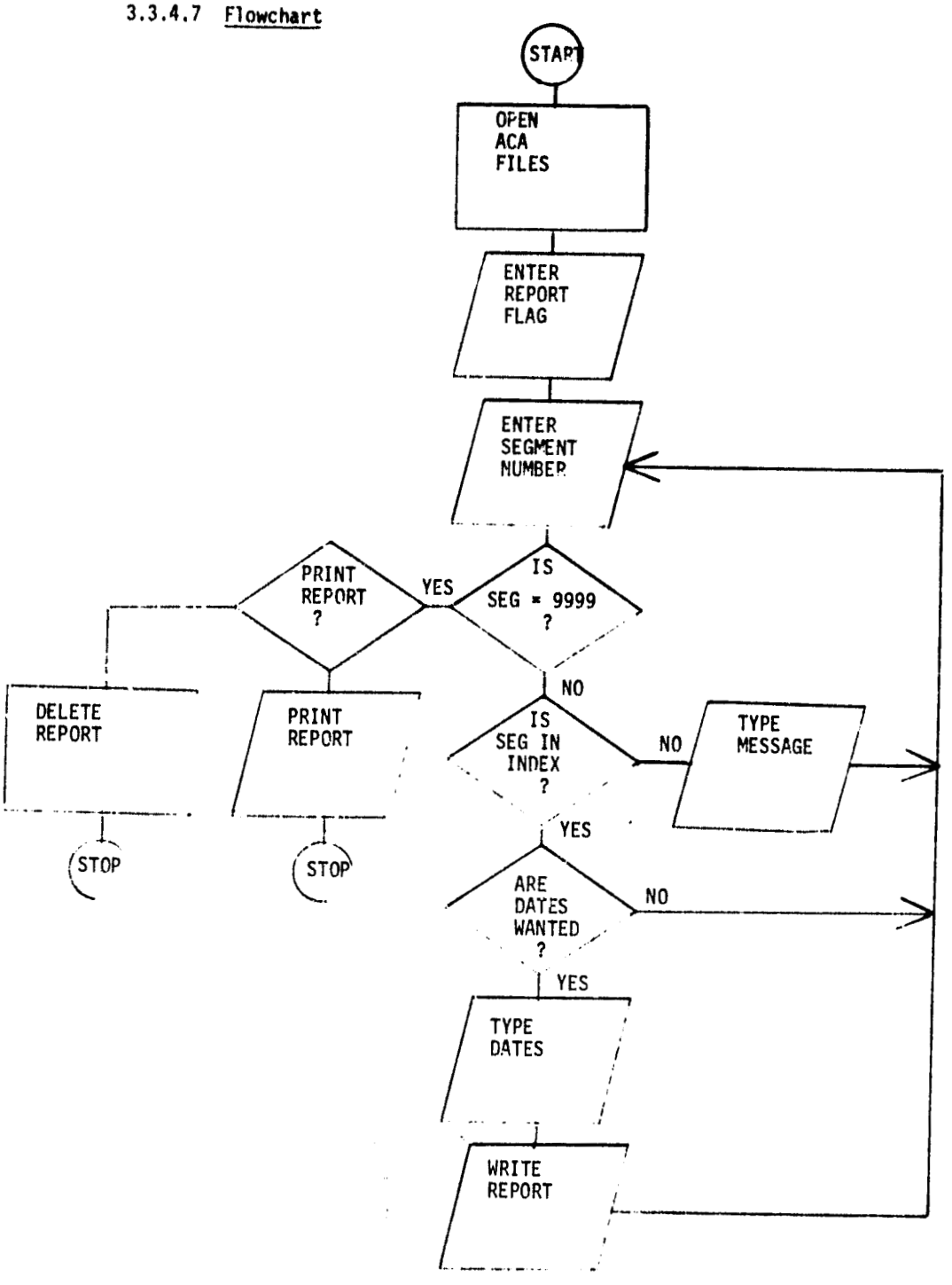
The ACA file names are requested and the ACA files are opened. The optional line printer flag is set, and a segment number is requested. If the segment is in the data base, the dates are optionally printed. Then a new segment is requested. When setment 9999 is entered, the report is optionally printed and the program stops.

SEC	DATE	CODE	DECIMAL	IAG	SLN	SAT	LAT		LONG			GAMMA	FLAG	ADDRESS
1472	78117	SWCH1=		45	-11	2	0	0	0	0	0	0	0	60
1472	78135	SWCH1=		51	-7	2	0	0	0	0	0	0	0	56
1472	78216	SWCH1=		50	-6	3	0	0	0	0	0	0	0	52
1472	78243	SWCH1=		42	-6	2	0	0	0	0	0	0	0	48
1472	78252	SWCH1=		45	-4	3	0	0	0	0	0	0	0	44
1472	78270	SWCH1=		30	-4	3	0	0	0	0	0	0	0	40
1472	0	0	TRIM/B	0	0	0	0	0	0	0	0	0	0	36

Figure 3.3.4,4

QUERY

3.3.4.7 Flowchart



```
FØRTRAN IV-PLUS V02-51          15:19:37    26-SEP-79          PAGE 1
QUERY.FTN       /TR:BLOCKS/WR

      C   QUERY INTERACTIVLY CHECKS DØTDATA,ACA FILES
            IMPLICIT INTEGER*2  A-Z)
            DIMENSION MINDEX(25 ),STUF(7),INDEX(256),BUFFER(256)
            DIMENSION  ALSØ(2),ANK(9)
            BYTE  DIC(10),FD (32),FIN(32),FAN(32),DØTDA(11),DCTIN(12)
            BYTE  IT,IP,ITST(2)
            EQUIVALENCE (MINDEX(1),LDX),(MINDEX(2),MXR)
            EQUIVALENCE (IZ, BUFFER(256)),(STUF(1),BUFFER(1))
            EQUIVALENCE (ANK(1),BUFFER(219)),(ITST(1),STUF(7))
       101  CALL GETNAM (FDN,FIN)
            WRITE(2,113)
       113  FØRMAT('     SEG    DATE  CØDE    DECIMAL',
           *       '  TAG   SEN   SAT   LAT             LØNG        ',
           *       '        GAMMA  FLAG ADDRESS')
       111       CONTINUE
            ØPEN(UNIT=3,NAME=FIN,TYPE='ØLD',ACCESS='DIRECT',ERR=300)
            READ(3'1)MINDEX
            TYPE 201,LDX,MXR
       201  FØRMAT(' ',I6,' INDEX FILES    ',I8,' DATA FILES')
            ØPEN(UNIT=4,NAME=FDN,TYPE='ØLD',ACCESS='DIRECT',ERR=300)
            TYPE 1002
      1002     FØRMAT('     INPUT F   FØR FULL REPØRT',/,
           *          '             P   FØR PARTIAL REPØRT',/,
           *          '             N   FØR NØ REPØRT')
               ACCEPT 2002, IRPT
      2002       FØRMAT(A1)
         1  CØNTINUE
            TYPE 102
       102  FØRMAT('   ENTER SEGMENT NUMBER (9999 TØ STØP)')
            ACCEPT 202,SEG
       202  FØRMAT(I4)
            IF (SEG.EQ.9999) GØ TØ 666
             ITS=0
            DØ 310,K=3,LDX-1,2
              READ(3'K) INDEX
              DØ 311 J=1,255,2
              IF(INDEX(J).EQ.0) GØ TØ 40
              IF(INDEX(J).EQ.SEG) GØ TØ 50
       311    CONTINUE
       310    CONTINUE
        40    TYPE 104, SEG
       104    FØRMAT(///'  ***  SEG',I6,'  DØES NØT APPEAR IN DØTINDEX',//)
              GØ TØ 1
        50    TYPE 106, SEG
       106    FØRMAT(' SEGMENT',I6,'  APPEARS IN THE DØTINDEX',/
           *          ' TYPE Y FØR DATES ,N FØR A NEW SEGMENT')
              ACCEPT 206, IT
       206    FØRMAT(A1)
              IF(IT.EQ.'N')GØ TØ 1
                  FIRST=INDEX(J+1)
               IZ=FIRST
         2  CØNTINUE
             ITS=ITS+1
             IF(ITS.GT.100) GØ TØ 777
                 ALSØ(2)=IZ
                 READ(4'IZ)BUFFER
```


Figure 3.3.4.8

Listing for QUERY

```
0051                    ALSØ(1)=BUFFER(254)
0052                  IF(STUF(7).EQ.'1*') TYPE 106,STUF,ANK(1),ANK(2),ANK(3),ALSØ
0053                  IF(ITST(2).NE.'*') TYPE 109,STUF,ANK(1),ANK(2),ANK(3),ALSØ
0054    109           FØRMAT(' ',I6,2I3,4A4,5I6)
0055                  IF(IRPT .EQ. 'F') WRITE(2,108) STUF,ANK,ALSØ
0056                  IF(IRPT .EQ. 'P' .AND. STUF(7) .EQ. '1*')
      *                 WRITE(2,108) STUF,ANK,ALSØ
0057                  IF(IRPT .EQ. 'P' .AND. ITST(2) .NE. '*')
      *                 WRITE(2,108) STUF,ANK,ALSØ
0058    108           FØRMAT(' ',I6,2I3,4A2,2X,9I6,2I6)
0059                  IF(IP.EQ.FIRST) GØ TØ 1
0060                  GØ TØ 2
0061    666     TYPE 110
0062            WRITE(2,108) STUF
0063    110     FØRMAT(//,'      TØ KILL LINEPRINTER ØUTPUT TYPE N')
0064            ACCEPT 210, IP
0065    210     FØRMAT(A1)
0066            IF(IP.EQ.'N'.OR.IRPT.EQ.'N') CLØSE(UNIT=2,DISP='DELETE')
0067            IF(IP.NE.'N'.AND.IRPT.NE.'N') CLØSE(UNIT=2,DISP='PRINT')
0068            CLØSE(UNIT=3)
0069            CLØSE(UNIT=4)
0070            STØP
0071    300     CLØSE (UNIT=3)
0072            CLØSE (UNIT=4)
0073            GØ TØ 101
0074    777     TYPE 667
0075    667     FØRMAT('   ******  CØUNT EXCEEDS 100  ******')
0076            GØ TØ 1
0077    999     END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	001752	501	RW,I,CØN,LCL
3	$IDATA	001246	339	RW,D,CØN,LCL
4	$VARS	003224	842	RW,D,CØN,LCL
5	$TEMPS	000002	1	RW,D,CØN,LCL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
FIRST	I*2	4-003222	IP	L*1	4-003206	IRPT	I*2	4-003210	IT	L*1	4-003205	ITS	I*2	4-003214
IZ	I*2	4-000776	J	I*2	4-003220	K	I*2	4-003216	LDX	I*2	4-001000	MXR	I*2	4-001002
SEG	I*2	4-003212												

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
ALSO	I*2	4-003000	000004	2	(2)
ANK	I*2	4-000664	000022	9	(9)
BUFFER	I*2	4-000000	001000	256	(256)
DØTDA	L*1	4-003156	000013	5	(11)
DØTIN	L*1	4-003171	000014	6	(12)
FAN	L*1	4-003116	000040	16	(32)
FDN	L*1	4-003016	000040	16	(32)
FIN	L*1	4-003056	000040	16	(32)
INDEX	I*2	4-002000	001000	256	(256)
ITST	L*1	4-000014	000002	1	(2)
MINDEX	I*2	4-000400	001000	256	(256)
STUF	I*2	4-000000	000016	7	(7)
VIC	L*1	4-003004	000012	5	(10)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000240	2	1-000674	40	1-000510	50	1-000550	101	1-000020
102'	3-000360	104'	3-000436	106'	3-000522	108'	3-000670	109'	3-000650
110'	3-000716	111	**	113'	3-000000	201'	3-000150	202'	3-000032
206'	3-000644	210'	3-000772	300	1-001666	310	**	311	**
666	1-001452	667'	3-000770	777	1-001720	999	**	1002'	3-000216
2002'	3-000354								

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSE GETNAM ØPENS

TØTAL SPACE ALLØCATED = 006446 1683

```
NØ FPP INSTRUCTIONS GENERATED

QUERY,LP1=QUERY
```

3.3.5 INITIAL FILE GENERATION (SEED)

3.3.5.1 Linkage

This program calls no subroutines.

3.3.5.2 Interface

None

3.3.5.3 Input

None

3.3.5.4 Output

Seed creates the ACA files SEEDINX.ACA and SEEDATA.ACA.

3.3.5.5 Storage

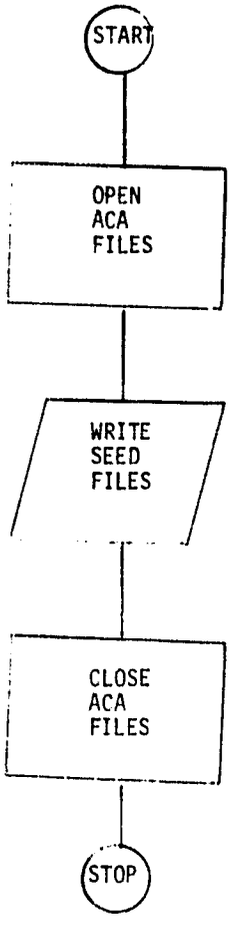
923 Bytes

3.3.5.6 Description

In order for the CDDDS programs to function there must be pre-existing ACA files. SEED creates such files with only one data record having segment number 1, which is marked for deletion.

SEED

3.3.5.7 Flowchart



```
FØRTRAN IV-PLUS V02-51          15:20:14    26-SEP-79          PAGE 1
SEED.FTN        /T5:BLØCKS/NP

      C     SEE .FTN CREATES SEED FILES FØR THE AL ANDERSØN
      C     RØUTINES FETCH AND STØRE
0001        IMPLICIT INTEGER*4 (A-Z)
0002        DIMENSION INDEX(256),BUF(256),INSEG(256)
0003        DATA INDEX/4,1,254*0/
0004        DATA INSEG/1,1,254*0/
0005        DATA B/1,77,111,'.','.',' T','ES','T ','.','.',209*1,36*2,.TRUE.,1,1/
0006        ØPEN(UNIT=2,NAME='[500,/]SEELINX,ACA',TYPE='NEW',ACCESS='DIRECT'
      C     ,RECØRDSIZE=128)
0007        ØPEN(UNIT=3,NAME='[500,/]SELDATA,ACA',TYPE='NEW',ACCESS='DIRECT'
      C     ,RECØRDSIZE=128)
0008        WRITE(2'1) INDEX
0009        WRITE(2'2) INDEX
0010        WRITE(2'3)INSEG
0011        WRITE(2'4) INSEG
0012        WRITE(3'1) BUF
0013        CLØSE(UNIT=2)
0014        CLØSE(UNIT=3)
0015        STØP
0016        END
```


3.3.5.8 Listing for SEED

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	000276	95	RW,I,CØN,LCL
2	$PDATA	00 050	20	RW,D,CØN,LCL
3	$IDATA	000120	40	RW,D,CØN,LCL
4	$VARS	003000	768	RW,D,CØN,LCL

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
BUF	I*2	4-001000	001000	256	(256)
INDEX	I*2	4-000000	001000	256	(256)
INSEG	I*2	4-002000	001000	256	(256)

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSS ØPENS

TØTAL SPACE ALLØCATED = 003466 923
NØ FPP INSTRUCTIØNS GENERATED

SEED,LP:=SEED

3.3.6 TABLE OF CONTENTS (TOC)

3.3.6.1 Linkage

This program calls GETNAM, and also its entry point GETCRD.

3.3.6.2 Interface

The interface with GETNAM is through the calling arguments.

3.3.6.3 Input

This program reads the data file and the index file. GETNAM either reads FN.DAT, or queries the interactive terminal.

3.3.6.4 Output

The output is a listing of all the files in the ACA data base (figure 3.3.6.4). This list is ordered by segment number. In debug mode, only the segment numbers are listed.

3.3.6.5 Storage

3424 Bytes

3.3.6.6 Description

The program first calls GETCRD, an entry of GETNAM, to get the index and data file names. Then it opens the index and data files. Up to 1024 segment numbers are read in and sorted. Then the data for each segment is read from the data file and written to the line printer.

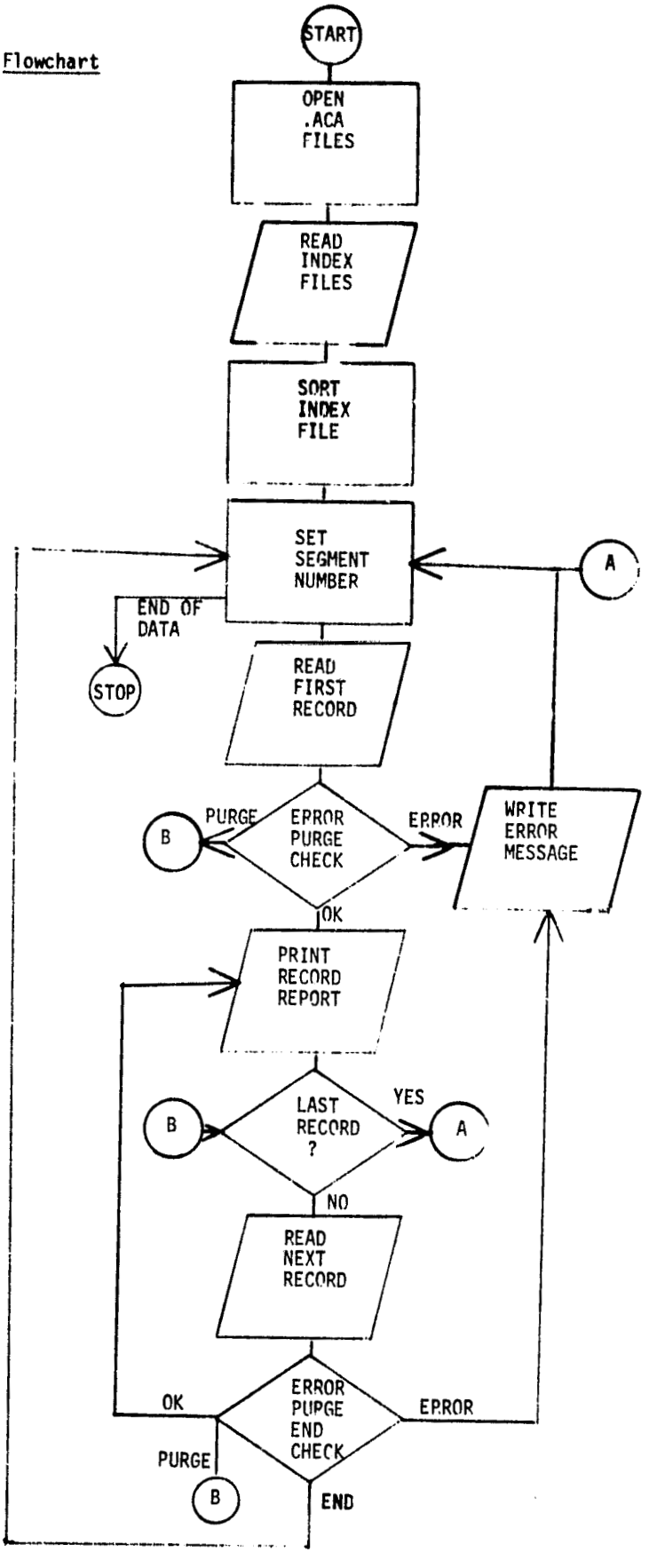
SEGMENT 1566 HAS THE FOLLOWING RECORDS IN THE DATABASE

SWCH4*	78169	2	100
TRTH78	0 0	0	80
SWC 1*	78196	3	81
SWCH2*	78196	3	82
SWCH3*	78196	3	83
SWCH4*	78196	3	84
SWC 1*	78232	3	85
SWCH2*	78232	3	86
SWCH3*	78232	3	87
SWC 4*	78232	3	88
SWCH1*	78115	2	89
SWC 2*	78115	2	90
SWCH3*	78115	2	91
SWCH4*	78115	2	92
SWCH1*	78133	2	93
SWCH2*	78133	2	94
SWCH3*	78133	2	95
SWCH4*	78133	2	96
SWCH1*	78169	2	97
SWC 2*	78169	2	98
SWC 3*	78169	2	99

Figure 3.3.6.4

TOC

3.3.6.7 Flowchart



```
FZRTRAN IV-PLUS V02-51          15117152    20-SEP-79          PAGE 1
TOC.FTN          /T :BLOCKS/WP

0001          PROGRAM TOC
0002          IMPLICIT INTEGER*2 (A-Z)
0003          BYTE NAM(6),FAM(32),FDN(32),FIN(32)
0004          DIMENSION SEGLST(1024),LPCLST(1024)
0005          DIMENSION MINDEX(256),INDEX(256)
0006          DIMENSION BUFFER(256)
0007          COMMON /FILNAM/ FDN,FIN,FAM
0008          EQUIVALENCE (MINDEX(1),LINDEX)
0009          EQUIVALENCE (NAM,BUFFER(1))
0010              CALL GETORD(FAM,FDN,FIN,1)
0011        5     CONTINUE
0012          OPEN (UNIT=2,NAME=FIN,TYPE='OLD',ERR=555,
             ,ACCESS='DIRECT',READONLY)
0013          READ (2'1) MINDEX
0014          OPEN (UNIT=3,NAME=FDN,TYPE='OLD',ERR=555,
             ,ACCESS='DIRECT')
0015          DO 1 , K=3,LINDEX+1,2
0016              READ (2'K) INDEX
0017              DO 10, J=1,255,2
0018                  SEGLST(64*K+192+(J+1)/2)=INDEX(J)
0019        10        LPCLST(64*K+192+(J+1)/2)=INDEX(J+1)
0020          DO 20, J=1,1024
0021              IF (SEGLST(J).NE.0) GO TO 21
0022                  N=J-1      ! N=NUMBER OF SEGMENTS IN DATABASE
D                   TYPE 15,N
0023        15        FORMAT('   TOC,NUMBER OF SEGMENTS =',I6)
0024              GO TO 30
0025        20 CONTINUE
0026        30 DO 50, K=2,N,1
0027              IF (SEGLST(K).GE.SEGLST(K-1)) GO TO 50
0028              SEGLT=SEGLST(K)
0029              LPCLT=LPCLST(K)
0030              SEGLST(K)=SEGLST(K-1)
0031              LPCLST(K)=LPCLST(K-1)
0032              IF (K.EQ.2) GO TO 44
0033              DO 40, M=K-1,2,-1
0034                  IF (SEGLT.GE.SEGLST(M-1)) GO TO 45
0035                  SEGLST(M)=SEGLST(M-1)
0036        40        LPCLST(M)=LPCLST(M-1)
0037        44    M=1
0038        45    SEGLST(M)=SEGLT
0039              LPCLST(M)=LPCLT
0040        50    CONTINUE
D               WRITE(6,53)  SEGLST
0041        53      FORMAT(' ',2215)
D               CLOSE(UNIT=6,DISP='PRINT')
D                     STOP
12 0042     54  DO 70, I=1,N
11 0043           ITS=0
10 0044           PRINT 1010, SEGLST(I)
 9 0045           IF(SEGLST(I).LE.2) GO TO 70
 8 0046           KK=LPCLST(I)
 7 0047           READ (2'KK) BUFFER
 6 0048           IF(SEGLST(I).NE. BUFFER(1)) GO TO 333
 5 0049      60   CONTINUE
 4 0050       IF (BUFFER(254)) GO TO 65
 3
```

Figure 3.3.6.8
Listing for TOC

```
0051            PRINT 1020, (BUFFER(J),J=5,7),BUFFER(2),BUFFER(3),BUFFER(21),KKK
0052       65     IF (BUFFER(255).EQ.LOCLST(I)) GO TO 70
0053              KKK=BUFFER(255)
0054              READ (3'KKK) BUFFER
0055              IF(SEGLST(I).NE.BUFFER(1)) GO TO 333
0056               ITS=ITS+1
0057               IF(ITS.GT.100) GO TO 70
0058              GO TO 60
0059       70      CONTINUE
0060              GO TO 71
0061      333     PRINT 3033,SEGLST(I),BUFFER(1),KKK
0062     3033     FORMAT(' SEGLST=',I6,'  BUFFER(1)=',I6,'  BAAD ERROR AT',I6)
0063              GO TO 70
0064       71     PRINT 1030
0065            CLOS  (UNIT=2)
0066            CLOSE (UNIT=3)
0067            STOP
0068      555       CALL GETNAM(FDN,FIN)
0069                GO TO 5
0070     1010 FORMAT (1H1,'  SEGMENT ',I4,' HAS THE FOLLOWING RECORDS IN THE DAT
                ,ABASE')
0071     1020 FORMAT (16X,3A2,5X,I3,I3,I6,I6)
0072     1030 FORM T(///)
0073            END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001534	430	RW,I,CØN,LCL
2	$PDATA	000004	2	RW,D,CØN,LCL
3	$IDATA	000350	116	RW,D,CØN,LCL
4	$VARS	013022	2825	RW,D,CØN,LCL
5	$TEMPS	000006	3	RW,D,CØN,LCL
6	FILNAM	000140	48	RW,D,ØVR,GBL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-013014	ITS	I*2	4-013016	J	I*2	4-013002	K	I*2	4-013000	KKK	I*2	4-013020
LINDEX	I*2	4-001000	LØCLT	I*4	4-013010	M	I*2	4-013012	N	I*2	4-013004	SEGLT	I*4	4-013006

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
BUFFER	I*2	4-000000	001000	256	(256)
FAN	L*1	6-000100	000040	16	(32)
FDN	L*1	6-000000	000040	16	(32)
FIN	L*1	6-000040	000040	16	(32)
INDEX	I*2	4-012000	001000	256	(256)
LØCLST	I*2	4-006000	004000	1024	(1024)
MINDEX	I*2	4-001000	001000	256	(256)
NAM	L*1	4-013015	000006	3	(6)
SEGLST	I*2	4-002000	004000	1024	(1024)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
5	1-000034	10	**	15'	**	20	1-000362	30	1-000404
40	**	44	1-000276	45	1-000612	50	1-000642	53'	**
54	**	60	1-001044	65	1-001176	70	1-001322	71	1-001440
333	1-001352	555	1-001512	1010'	3-000062	1020'	3-000160	1030'	3-000400
3033'	3-000300								

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØS$ GETCRD GETNAM ØPEN$

TØTAL SPACE ALLØCATED = 015300 3424

NØ FPP INSTRUCTIØNS GENERATED

TØC,LP:=TØC

3.3.7 TAPE LOAD (TPLOAD)

3.3.7.1 Linkage

TPLOAD calls the entry GETCRD of GETNAM and PUT. In addition several system routines included in QIO are called.

3.3.7.2 Interface

The interface with GETNAM is through the calling arguments.

The interface with PUT is through the calling arguments and through the common block/ANCIL/.

3.3.7.3 Input

One input card is read from the default unit 1 file FØR001.

COL 1-3 Tape drive identifier one of MT0, MT1, XT0, XT1
COL 5-7 Number of segments to be read in, e.g. 999.
COL 10-15 Code name to be used for loading, e.g SECH**.

3.3.7.4 Output

Data is transferred from the tape to the .ACA data base. At the end, a short line printer report is written (fig 3.3.7.4).

3.3.7.5 Storage

5214 Bytes

3.3.7.6 Description

TPLOAD reads the data for one acquisition from the tape using QIO routines, loads the data into the appropriate array location and PUTS the data into the data base. After all requested data has been loaded, a line printer report is created.

SPECTRAL DATA UNLOAD REPORT IPS140

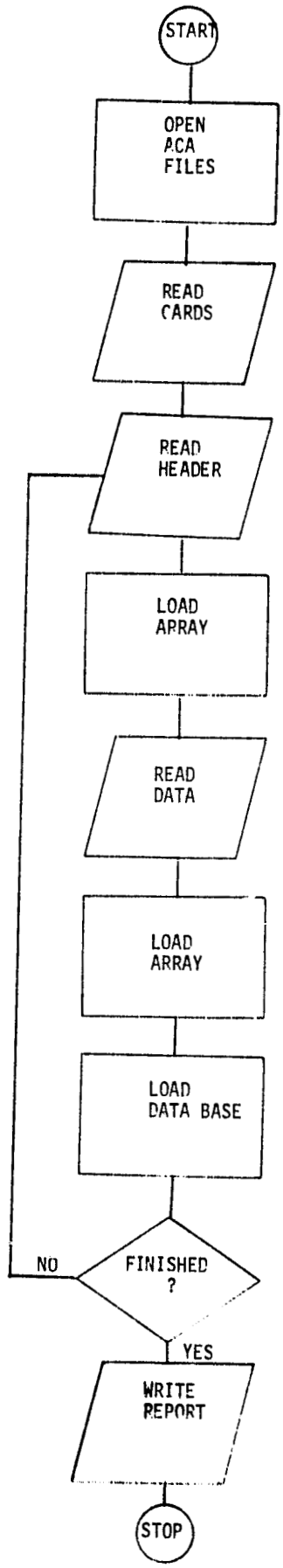
PAGE 1
DATE 8/30/7-
TIME 09126128

SEGMENT NO.	YEAR	JULIAN DATE			
112	78	169	0	55	14
112	78	187	0	54	2
112	78	196	0	56	118
112	78	205	0	52	1
112	78	224	0	48	3

Figure 3,3,7,4

TPLOAD

3.3.7.7 Flowchart



```
C     TPLØAD.FTN
      IMPLICIT INTEGER*2 (A-Z)
      DIMENSION CH1(209),CH2(209),CH3(209),CH4(209)
      DIMENSION ST(2),IPRM(6)
      DIMENSION INDX1(11),INDX2(11),E(4),CDNAME(3)
      DIMENSION AA(32),P TFIL(6,200)
      BYTE BUFF1(3060),BUFF2(1260),A(8),C(4),DILL(32),F(4)
      BYTE FDN(32),FIN(32),FAN(32)
      COMMON /ASCIL/CHLE,DATE(2),STATUX,FILL(36)
      COMMON/FILNAM/ FDN,FIN,FAN
      COMMON/STATUS/W1,W2
      EQUIVALENCE (A(5),F(1))
      EQUIVALENCE (ST,W1)
      EQUIVALENCE (DILL(1),BUFF1(2133))
      DATA INDX1/73,168,263,358,453,548,643,738,833,928,1023/
      DATA INDX2/1,20,39,58,77,96,115,134,153,172,191/
      DATA IATCH/"1400/,IRWND/"2400/,IRLB/"1000/,ISKPF/"2440/
100   FØRMAT(8A1,X,3A2)
104   FØRMAT(1H ,'  ')
105   FØRMAT(1H ,4X,'SEGMENT NØ.=',I4,4X,'YEAR=',I4,4X,
     1'JULIAN DATE=',I4)
107   FØRMAT(1H1,40X,'SPECTRAL DATA UNLØAD REPØRT',2X,3A2,2X,
C    1   24X,'PAGE ',I2)
108   FØRMAT(1H$,4X,'INPUT NUMBER ØF FILES TØ BE LØADED >')
109   FØRMAT(1H ,43X,'SEGMENT NØ.',3X,'YEAR',2X,'JULIAN DATE')
110   FØRMAT(I2)
111   FØRMAT(I4)
112   FØRMAT(1H ,46X,I4,6X,I4,6X,I4,6X,I4,6X,I4,6X,I4)
113   FØRMAT(1H ,106X,'DATE ',I2,'/',I2,'/',I2)
114   FØRMAT(1H ,106X,'TIME ',4A2)
      CALL GETCRD(FAN,FDN,FIN,1)
      WRITE(6,6) FDN
6     FØRMAT(///'    TPLØAD UPDATE ØF ',32A1)
      XX=0
      MTXTFG='MT'   ! 0 MEANS MT
      DRIVE=0   ! 0 MEANS 0    IE  MT0
      M=1
      LINCTR=50
      PAGE=0
      CALL IDATE(X,Y,Z)
      CALL TIME(E)
      READ(1,100) (A(J),J=1,8),CDNAME
C     XT1  05  SECH**     EXAMPLE  5 RECØRDS ØN XT1
      IF(A(1) .EQ. 'X') MTXTFG='XT'  ! X MEANS XT  CØL 1
      IF(A(1) .EQ. '1') MTXTFG='XT'  !  ØR  1 MEANS XT  CØL 1
      IF(A(3) .EQ. '1') DRIVE=1  !  CØL 3
      DECØDE(3,111,F) TØTFIL  !  3 DIGIT CØL 5,6,7   NØ ØF SEGS
         CALL ASNLUN(2,MTXTFG,DRIVE)
         CALL WTQIØ(IATCH,2,1,,ST)
         CALL WTQIØ(IRWND,2,1,,ST)
      DØ 20 R=1,TØTFIL
         CALL GETADR(IPRM,BUFF1)
         IPRM(2)=3060
         CALL WTQIØ(IRLB,2,1,,ST,IPRM)
C     WRITE(6,678) (BUFF1(K),K=2201,2400)
678   FØRMAT(' ',250A)
```

Figure 3.3.7.8

Listing for TPLOAD

```
FORTRAN IV-PLUS V02-51          15:21:24    20-SEP-79          PAGE 2
TPLOAD.FTN          /T :OLFCKS/WR

       C
0052   679      WRITE(6,679) BUFF1
0053            FORMAT('    ',25A")
                IF(X1 .EQ. "366) GO TO 30
       C
0054   721      WRITE(6,721) FILL
0055            FORMAT('  ',1306,/,1806)
0056            DO 677 I=1,32
0057            AA(I)=DILL(I)+"20
0058   677      CONTINUE
0059             AA(30)=BUFF1(67)-"177
0060             AA(30)=AA(30)+127
0061             IF(AA(30).LT.0) AA(30)=BUFF1(67)
0062              AA(31)=BUFF1(68)-"177
0063             AA(31)=BUFF1(68)-127
0064              IF(AA(31).LT.0)  AA(31)=BUFF1(68)
                 SEG,A=256*AA(30)*AA(31)
       C          RITE(6,721) AA
       D        TYPE 544,SEGA
0065             FILL(2)=BUFF1(2202)  !SUN ANGLE
       C        TYPE 544,FILL(2)
0066   544      FORMAT(I8)
0067             FILL(3)=BUFF1(2254)-"156      ! SOIL LINE
       C         TYPE 544,FILL(3)
0068             FILL(4)=BUFF1(2123)-"360    ! MISSION NUMBER   LANDSAT 1
       D         TYPE 544,FILL(3)
0069              FILL(5)='S'
0070             IF(BUFF1(2134)+"53.EQ."0) FILL(5)='N'
       C          TYPE 542,FILL(5)
0071   542       FORMAT('  ',A4)
0072   540        FORMAT(I3)
0073             FILL(6)=100*AA(3)+10*AA(4)+AA(5)  !LATITUDE DEGREES
       C         TYPE 544,FILL(6)
0074              FILL(7)=10*AA(6)+AA(7)  !LATITUDE  MIN.
       C           TYPE 544,FILL(7)
0075              FILL(8)='E'
0076              IF(BUFF1(2140)+"32.EQ."0) FILL(8)='W'
0077              FILL(9)=100*AA(9)+10*AA(10)+AA(11)   !LONGITUDE DEGREES
       C        TYPE 542,FILL(8)
0078             FILL(10)=10*AA(12)+AA(13)   !LONGITUDE MIN.
       C        TYPE 544,FILL(9)
       C        TYPE 544,FILL(10)
0079   543       FORMAT(A1,I3,I2,A1,I3,I2)
0080               A(1)=BUFF1(22?9)+"20
0081               A(2)=BUFF1(22?0)+"20
0082               DATE(1)=10*A( )+A(2)    !YEAR
       C           TYPE 544, DATE(1)
0083               AA(1)=BUFF1(2251)+"20
0084               AA(2)=BUFF1(2252)+"20
0085               AA(3)=BUFF1(2253)+"20
0086             DATE(2)=100*AA( )+10*AA(2)+AA(3)  !DAY OF YEAR
       C           TYPE 544,DATE(2)
0087         XX=XX+1
0088         PRTFIL(1,XX)=S-00?
0089         PRTFIL(2,XX)=DATE(1)
0090         PRTFIL(3,XX)=DATE(2)
0091         PRTFIL(4,XX)=FILL(1)
0092         PRTFIL(5,XX)=FILL(2)
```

```
              PRTFIL(6,XX)=FILL(3)
                  CALL GETADR(IPRM,BUFF2)
                  IPRM(2)=12*2
                  CALL WTQIO(IRL-,2,1,,ST,IPRM)
              IF(N1 .EQ. "366) GO TO 39
                  IPRM(1)=1
                  CALL WTQIO(ISPF,2,1,,ST,IPRM)
      39      CONTINUE
              DO 10 P=1,11
              J=INDX1(P)
              K=INDX2(P)
      1       CH1(K)=BUFF2(J)
              CH2(K)=BUFF2(J+19)
              CH3(K)=BUFF2(J+38)
              CH4(K)=BUFF2(J+57)
              J=J+1
              K=K+1
              IF(K .NE. (INDX2(P)+19)) GO TO 1
      10      CONTINUE
              CDNAME(3)='1*'
              CALL PUT(SEGNO,CH1,CDNAME)
              CDNAME(3)='2*'
              CALL PUT(SEGNO,CH2,CDNAME)
              CDNAME(3)='3*'
              CALL PUT(SEGNO,CH3,CDNAME)
              CDNAME(3)='4*'
              CALL PUT(SEGNO,CH4,CDNAME)
      20      CONTINUE
      30          CALL WTQIO(IRAND,2,1,,ST)
              DO 35 D=1,XX
              IF(LINCTR .EQ. 50) GO TO 31
      32      LINCTR=LINCTR+1
              WRITE(6,112) (PRTFIL(G,D),G=1,6)
              GO TO 3355
      31      LINCTR=0
              PAGE=PAGE+1
              WRITE(6,107) CDNAME,PAGE
              WRITE(6,113) X,Y,Z
              WRITE(6,114) (E(L),L=1,4)
              WRITE(6,109)
              GO TO 32
      3355    CONTINUE
      35      CONTINUE
              WRITE(6,104)
              STOP
              END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	002426	651	RW,I,CON,LCL
2	$PDATA	000010	4	RW,D,CON,LCL
3	$IDATA	000540	176	RW,D,CON,LCL
4	$VARS	02 006	4201	RW,D,CON,LCL
5	$TEMPS	00 014	2	RW,D,CON,LCL
6	$LCIL	000120	40	RW,D,OVR,GBL
7	FILNAM	00 140	48	RW,D,OVR,GBL
8	STATUS	000004	2	RW,D,OVR,GBL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
B	I*2	4-020570	CODE	I*2	6-000000	D	I*2	4-020600	DRIVE	I*2	4-020546	G	I*2	4-020602
IATCH	I*2	4-02 532	IRLR	I*2	4-020536	IRWND	I*2	4-020534	ISKPF	I*2	4-020540	J	I*	4-020564
K	I*2	4-02 572	L	I*2	4-020604	LINCTR	I*2	4-020552	M	I*2	4-020550	MTXTFG	I*2	4-020544
P	I*2	4-02 576	PAGE	I*2	4-020554	SEGN0	I*2	4-020574	STATUX	I*2	6-000006	T0TFIL	I*2	4-020566
W1	I*2	8-000000	W2	I*2	8-000002	X	I*2	4-020556	XX	I*2	4-020542	Y	I*	4-020560
Z	I*2	4-02 562												

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
A	L*1	4-005764	000010	4	(8)
AA	I*2	4-01 312	000100	32	(32)
BUFF1	L*1	4-000000	005764	1530	(3060)
BUFF2	L*1	4-016152	002354	636	(1260)
C	L*1	4-02 526	000004	2	(4)
CDNAME	I*2	4-011304	000006	3	(3)
CH1	I*2	4-002774	000642	209	(205)
CH2	I*2	4-00 636	000642	209	(205)
CH3	I*2	4-007530	000642	209	(205)
CH4	I*2	4-010342	000642	209	(205)
DATE	I*2	6-00 002	000004	2	(2)
DILL	L*1	4-006124	000040	16	(32)
E	I*2	4-01 274	000010	4	(4)
F	L*1	4-002770	000004	2	(4)
FAN	L*1	7-000100	000040	16	(32)
FDN	L*1	7-00 000	000040	16	(32)
FILL	I*2	6-001010	000110	36	(36)
FIN	L*1	7-00 040	000040	16	(32)
INDX1	I*2	4-011220	000026	1	(11)
INDX2	I*2	4-011246	000026	11	(11)
IPRM	I*2	4-011204	000014	6	(6)
PRTFIL	I*2	4-011412	004540	1200	(6,200)
ST	I*2	8-000000	000004	2	(2)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-001472	6'	3-000264	10	**	20	**	30	1-001734
31	1-002116	32	1-001004	35	**	100'	3-000000	104'	3-000010
105'	**	107'	3-000020	108'	**	109'	3-000104	110'	**
111'	3-000156	112'	3-000162	113'	3-000216	114'	3-000244	391	**
540'	**	542'	**	543'	**	544'	**	617	**
678'	**	679'	**	721'	**	3355	1-002352		

FUNCTIONS AND SUBROUTINES REFERENCED

ASNLUN GETADR GETCØD IDATE PUT TIME WTQ10

TØTAL SPACE ALLØCATED = 024274 5214

NØ FPP INSTRUCTIONS GENERATED

TPLØAD,LP:=TPLØAD

3.3.8 DATA BASE REBUILD (XFER)

3.3.8.1 Linakge

XFER calls GETNAM's entry point GETCRD.

3.3.8.2 Interface

The interface with GETNAM is through calling arguments only.

3.3.8.3 Input

The ACA dat file specified in FN.DAT is read.

3.3.8.4 Output

All data records in the data file except those marked for deletion (byte 254 set .TRUE.) are entered into the new data base SEEDINX.ACA, SEEDATA.ACA.

A small report is generated (fig. 3.3.8.4)

3.3.8.5 Storage

2198 Bytes

3.3.8.6 Description

The ACA data file is read directly and sequentially. If the segment is marked for deletion, (byte 254 set .TRUE.) the next record is read. Otherwise statements copied from PUT are used to enter the record into the SEED files.

If a new smaller data base is required based on factors other than duplication, line 28 of the listing is the place to insert the logic.

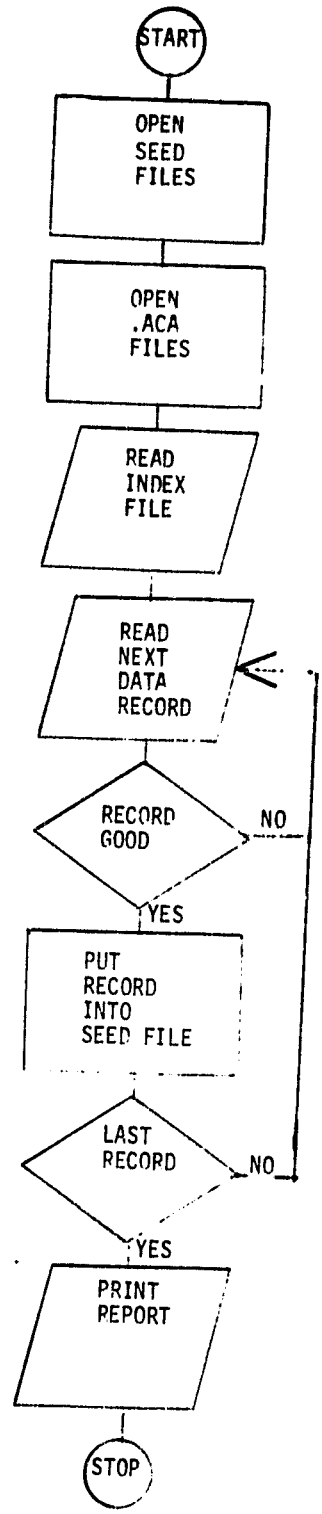
```
FILE [200,7]DMTDATA,ACA HAS BEEN XFRED
  100 RECORDS SHOULD HAVE BEEN READ
    1 RECORDS WERE DELETED
  100 RECORDS WERE SAVED
```

fig. c 3.3.X.4

Pigure 3,3,8,4

XFER

3.3.8.7 Flowchart



```
      C         XFER   MODIFIED STEED
                IMPLICIT INTEGER*2 (A-Z)
                BYTE FAN(32),FTN(32),FDN(32)
                DIMENSION INDEX(256),BUFFER(256),NNNNNN(256),MINDEX(256)
                DIMENSION NEWBUF(256)
                DIMENSION NAME(3),ARRAY(209)
                PARAMETER T=.TRUE.,F=.FALSE.
                EQUIVALENCE (MINDEX(1),LINDEX),(MINDEX(2),MAXREC)
                 EQUIVALENCE (NEWBUF(1),SEG)
                COMMON /ANCIL/CODE,DATE(2),STATUS,FILL(36)
                COMMON /FILNAM/FDN,FTN,FAN
                DATA FILL/36*0/
                DATA FAN/'X','F','E','R','.','.','D','A','T',24*0/
                DATA B/'  '/
                ØPEN (UNIT=7,NAME='[200,7]SEEDINX.ACA',TYPE='ØLD',
               *        ACCESS='DIRECT',ERR=2)
                ØPEN (UNIT=8,NAME='[200,7]SEEDATA.ACA',TYPE='ØLD',
               *        ACCESS='DIRECT',ERR=2)
                CALL GETCRD(FAN,FDN,FTN,1)
                ØPEN (UNIT=3,NAME=FTN,TYPE='ØLD',ACCESS='DIRECT',ERR=2
               *       ,READONLY)
                 READ(3'1)MINDEX
                 KKK=LINDEX(2)
      D             TYPE 310,KKK
        310        FØRMAT('  KKK ',I8)
                 CLØSE(UNIT=3)
                ØPEN (UNIT=4,NAME=FDN,TYPE='ØLD',ACCESS='DIRECT',ERR=2
               *       ,READONLY)
      D             KKK=100
      D             TYPE 301
        301        FØRMAT('  XFR DEBUG   FILES ØPEN   ')
                     KREJ=0
                DØ 1 KK=1,KKK
                 READ(4'KK) NEWBUF
                 IF(NEWBUF(254)) KREJ=KREJ+1
                IF(NEWBUF(254)) GØ TØ 1
                READ(7'1) MINDEX
                MAXREC=MAXREC+1
      D             TYPE 302 ,MAXREC
        302        FØRMAT('   MAXREC = ',I6)
      C         SEARCH THE INDEX FØR THE SEGMENT NUMBER
                DØ 20, K=3,LINDEX-1,2
                    READ (7'K) INDEX
                    DØ 20, J=1,255,2
                        IF (INDEX(J).EQ.0) GØ TØ 40
                        IF (INDEX(J).EQ.SEG) GØ TØ 50
         20         CØNTINUE
      C         ADD A PAGE TØ THE INDEX
                K=LINDEX+1
                LINDEX=K+1
                DØ 30, I=1,256
         30          INDEX(I)=0
                J=1
      C         WRITE THE FIRST RECØRD FØR A SEGMENT NUMBER
         40     INDEX(J)=SEG
```


Figure 3.3.8.8

Listing for XFER

```
D                 TYPE 303,SEG
      303         FORMAT('       SEG =',I6,'  AT 40')
            INDEX(J+1)=MAXREC
            NEWBUF(255)=MAXREC
            NEWBUF(256)=MAXREC
            WRITE (8'MAXREC) NEWBUF
D                 TYPE 311,MAXREC
      311         FORMAT('    MAXREC A   ',I8)
            GO TO 60
C           WRITE ANOTHER RECORD FOR THE SEGMENT NUMBER
       50   NEXT=INDEX(J+1)
D                 TYPE 304,SEG
      304         FORMAT('       SEG IS ',I6,'  AT 50')
            READ (8'NEXT) BUFFER

D                 TYPE 312,MAXREC
      312         FORMAT('    MAXREC C   ',I8)
            LAST=BUFFER(255)
            NEWBUF(255)=LAST
            NEWBUF(256)=NEXT
            WRITE (8'MAXREC) NEWBUF
            READ (8'LAST) NNNNNN
            NNNNNN(256)=MAXREC
            WRITE (8'LAST) NNNNNN
            BUFFER(255)=MAXREC
            WRITE (8'NEXT) BUFFER
            INDEX(J+1)=MAXREC
       60   WRITE (7'1) MINDEX
D                 TYPE 306,MAXREC
      306         FORMAT('                     AT 60 MAXREC=',I8)
            WRITE (7'2) MINDEX
            WRITE (7'K) INDEX
            WRITE (7'K+1) INDEX
        1       CONTINUE
D                 TYPE 307,MAXREC
      307         FORMAT   ('   1      CONTINUE  MAXREC=',I8)
        2       CONTINUE
            CLOSE (UNIT=7)
            CLOSE (UNIT=8)
            CLOSE (UNIT=4)
            CLOSE (UNIT=3)
            WRITE(6,100) FDN,KKK,KREJ,MAXREC
      100   FORMAT(////,' FILE ',32A1,' HAS BEEN XFRED',/
     *              ,I6,' RECORDS SHOULD HAVE BEEN READ',/
     *              ,I5,' RECORDS WERE DELETED',/
     *              ,I6,' RECORDS WERE SAVED')
            STOP
            END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	001550	436	RW,I,LCL,LCL
2	$PDATA	00 054	22	RW,D,LCL,LCL
3	$IDATA	00 452	149	RW,D,LCL,LCL
4	$VARS	005672	1501	RW,D,LCL,LCL
5	$TEMPS	00 004	2	RW,D,LCL,LCL
6	ANCIL	000120	40	RW,D,OVR,GBL
7	FILNAM	00 140	48	RW,D,OVR,GBL

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
B	I*2	4-005650	CØDE	I*2	6-000000	I	I*2	4-005664	J	I*2	4-005662	K	I*2	4-005660
KK	I*2	4-005656	KKK	I*2	4-005652	KREJ	I*2	4-005654	LAST	I*2	4-005670	LINDEX	I*	4-001000
MAXREC	I*2	4-001002	NEXT	I*2	4-005666	SEG	I*2	4-000000	STATUS	I*2	6-000006			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
ARRAY	I*2	4-005006	000642	209	(209)
BUFFER	I*2	4-003006	001000	256	(256)
DATE	I*2	5-00 002	000004	2	(2)
FAR	L*1	7-000100	000040	16	(32)
FDB	L*1	7-000000	100040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	7-000080	000040	16	(32)
INDEX	I*2	4-002000	001000	256	(256)
MINDEX	I*2	4-001000	001000	256	(256)
NAME	I*2	4-002000	000006	3	(3)
NEABLE	I*2	4-000000	001000	256	(256)
NNNNN	I*2	4-004000	001000	256	(256)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-001356	2	1-001414	20	**	30	**	40	1-000352
50	1-000555	60	1-001200	100'	3-000000	301'	**	302'	**
303'	**	304'	**	306'	**	307'	**	310'	**
311'	**	312'	**						

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSS GETCRD ØPENS

TØTAL SPACE ALLØCATED = 010454 2198

NØ FPP INSTRUCTIONS GENERATED

3.4 SUBROUTINES FOR LANDSAT DATA MANIPULATION

3.4.1 SATELLITE CALIBRATION (CALIB(SAT,NUM))

3.4.1.1 Linkage

Subroutine CALIB does not call any other subroutine.

3.4.1.2 Interface

The calling arguments are the satellite identifier (SAT) and the number of points to be processed, (NUM).

4-channel LANDSAT data is passed through the common block /DATA/.

3.4.1.3 Input

N.A.

3.4.1.4 Output

N.A.

3.4.1.5 Storage

1025 Bytes

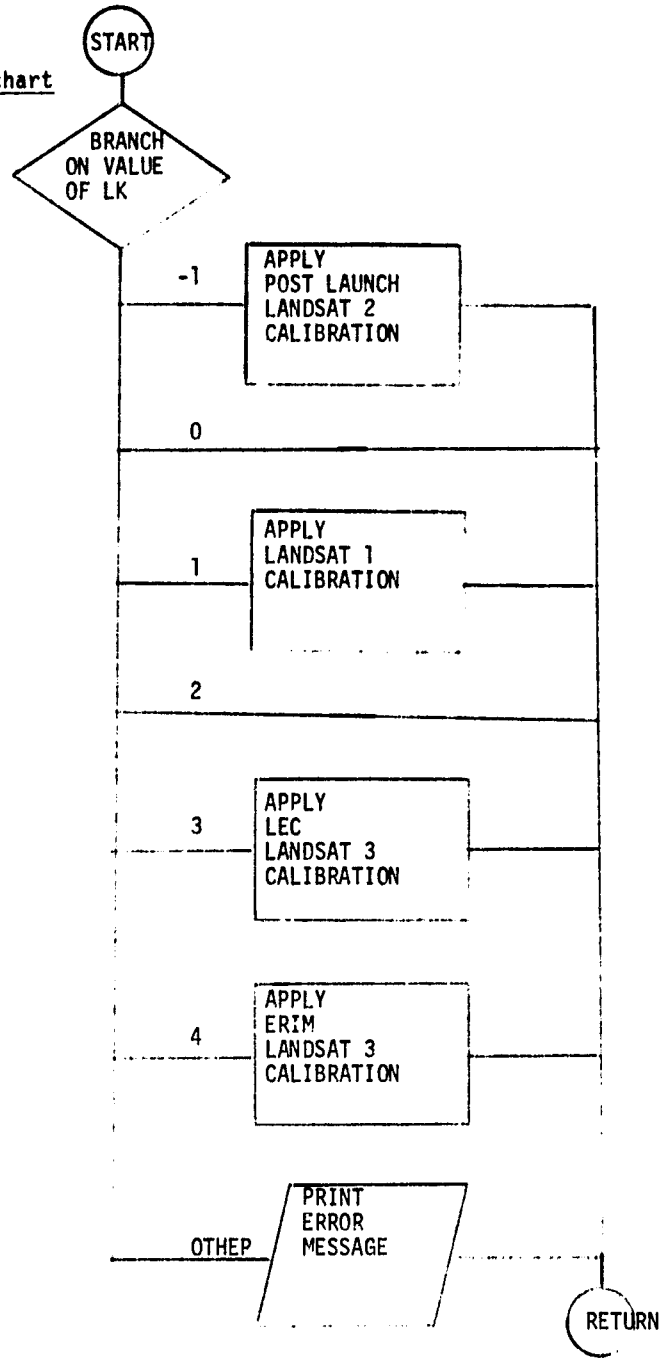
3.4.1.6 Description

The program recalibrates the contents of the common block /DATA/ to the LANDSAT 2 pre-launch calibration.

Desired recalibration	Sat identifier
LANDSAT 2 Post launch	-1
No correction	0
LANDSAT 1	1
LANDSAT 2 Prelaunch	2
LANDSAT 3 (Lockheed version)	3
LANDSAT 3 (ERIM version)	4

CALIB

3.4.1.7 Flowchart



```
FØRTRAN IV-PLUS V02-51          15:25:35    20-SEP-79          PAGE 1
CALIB.FTN       /TR:FLECKS/KR

      C   CALIB APPLIES CALIBRATION CONSTANTS TØ DØT DATA
      C        LK=-1  FØR PRELAUNCH LANDSAT 2
      C        LK= 0  FØR NØ CØRRECTIØN
      C        LK= 1  FØR LANDSAT 1 CALIBRATIØN
      C        LK= 2  FØR PRELAUNCH LANDSAT 2 (NØ CØRRECTIØN)
      C        LK= 3  FØR LØC L NDSAT 2 CALIBRATIØN
      C        LK= 4  FØR GRIM LANDSAT 3 CALIBRATIØN
0001          SUB ROUTINE CALIB(LK,N)
0002          IMPLICIT INTEGER*2 (A-S)
0003          DIMENSIØN CH1(209),CH2(209),CH3(209),CH4(209)
0004          REAL CALFAC(4,4),CALBAS(4,4)
0005          CØMMØN /DATA/ CH1,CH2,CH3,CH4
0006          DATA CALFAC/1.040,1.000,1.090,0.820,
     *                    1.275,1.141,1.098,0.948,
     *                    1.151,1.230,1.246,1.062,
     *                    1.137,1.173,1.247,1.126/
0007          DATA CALBAS/-5.700, 1.190,-2.910, 3.010,
     *                    -1.445,-2.712,-2.950, 0.446,
     *                     0.0  , 0.0  , 0.0  , 0.0  ,
     *                     0.0  , 0.0  , 0.0  , 0.0  /
      D         TYPE 1,LK
      D         TYPE 2,CALBAS(1,LK),CALBAS(2,LK),CALFAC(1,LK),CALFAC(2,LK)
0008    2       FØRMAT(' CALIB BIAS CHECK',4F8.3)
0009    1       FØRMAT(' CALIB    SAT=',I4)
0010          IF(LK.EQ.2) RETURN  ! LACIE PRE-LAUNCH CALIBRATIØN
0011          IF(LK .EQ. 0) RETURN    ! 0 MEANS SATELLITE ID NØT FØUND
0012          IF(LK .EQ. -1) LK=2     ! -1 MEANS LANDSAT 2 PØST LAUNCH CALIBRATIØN
0013            I (LK.LT.1 .ØR. LK.GT.4) WRITE(6,1) LK
0014            IF(LK.LT.1 .ØR. LK.GT.4) RETURN
0015          DØ  00 I=1,N
0016          CH1(I)=0.5+CH1(I)*CALFAC(1,LK)+CALBAS(1,LK)
0017          CH2(I)=0.5+CH2(I)*CALFAC(2,LK)+CALBAS(2,LK)
0018          CH3(I)=0.5+CH3(I)*CALFAC(3,LK)+CALBAS(3,LK)
0019          CH4(I)=0.5+CH4(I)*CALFAC(4,LK)+CALBAS(4,LK)
0020    100   CØNTINUE
0021          RETURN
0022          EN
```

Figure 3.4.1.8
Listing for CALIB

CALIB,FTN /TR:BLØCKS/WR

PROGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	SCØDE1	000342	113	RW,I,GBL,LCL
3	%IDATA	00 0026	11	RW,D,GBL,LCL
4	SVARS	000202	65	RW,D,GBL,LCL
5	SIFABS	000002	1	RW,D,GBL,LCL
6	DATA	00 0210	136	RW,D,REL,GBL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
CALIB		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-000200	LK	I*2	4-000002*	N	I*2	F-000094*						

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
CALBAS	R*4	4-000100	000100	32	(4,4)
CALFAC	R*4	4-00 000	000100	32	(4,4)
CH1	I*2	6-00 000	000642	209	(205)
CH2	I*2	6-000642	000642	209	(205)
CH3	I*2	6-00 504	000642	209	(205)
CH4	I*2	6-002346	000642	209	(205)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1'	3-000000	2'	**	100	**				

TØTAL SPACE ALLØCATED = 004002 1025

CALIB,LP:=CALIB

3.4.2 DATE CONVERSION (DDATE(YEAR, MONTH,DAY,YY,DDD,MK))

3.4.2.1 Linkage

DDATE calls no other program.

3.4.2.2 Interface

All information is passed through the calling arguments, e.g. YEAR = 1979, MONTH = 'JAN' DAY = 1, YY = 79, DDD = 001.

3.4.2.3 Input

None

3.4.2.4 Output

None

3.4.2.5 Storage

437 Bytes

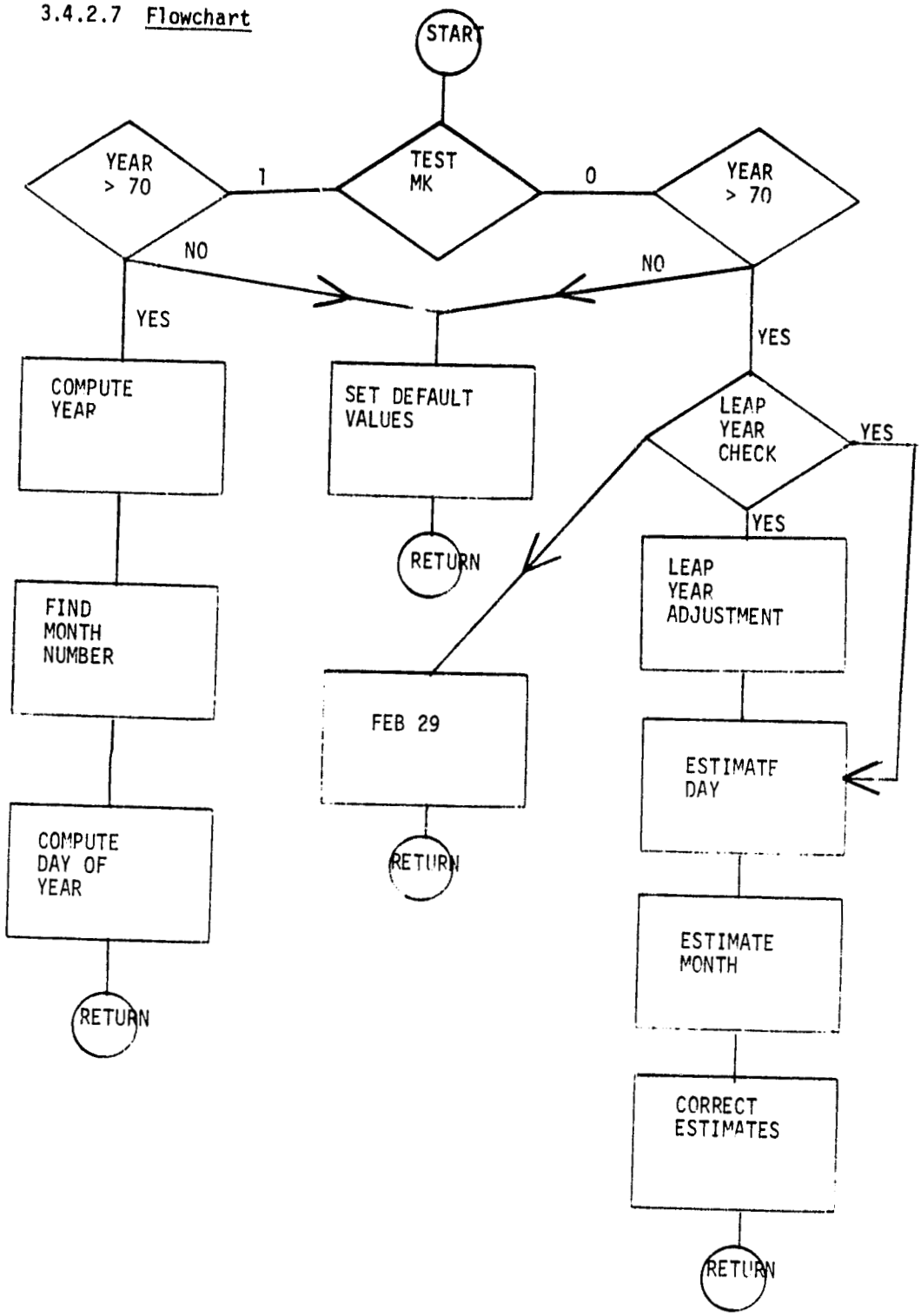
3.4.2.6 Description

For MK = 0, DDATE converts Julian dates to calendar dates, i.e.
79001 1979, Jan, 1
For MK = 1 DDATE converts calendar dates to Julian dates, i.e, 1979, Jan, 1 79001

The permitted values for MONTH are 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC'.

DDATE

3.4.2.7 Flowchart



```
FØRTRAN IV-PLUS V02-53          15:28:12    20-SEP-79          PAGE 1
DDATE.FTN       /TH:HL:CKS/HR

      C   DDATE CONVERTS FROM AND TØ YEAR,MONTH,DAY / YYDDD
0001          SUBROUTINE DDATE(YEAR,MONTH,DAY,YY,DDD,MK)
      C               MK=0  YYDDD TØ YEAR,MONTH,DAY
      C               MK=1  YEAR,MONTH,DAY TØ YYDDD
0002          INTEGER*2 MK
0003          INTEGER*4 YEAR,MONTH,DAY
0004          INTEGER*2 YY,DDD
0005          INTEGER*4 MØ(13),FNAGL(12),BUG(13)
0006          DATA MØ/'JAN','FEB','MAR','APR','MAY','JUN','JUL'
     *              ,'AUG','SEP','ØCT','NØV','DEC','ERR'/
0007          DATA BUG/0,-1,-1,-2,-2,-3,-3,-4,-4,-5/
0008          DATA FNAGL/31,28,31,30,31,30,31,31,30,31,30,31/
      C
0009          IF(MK.EQ.1)GØ TØ 1
0010          IF(YY .LT. 70) GØ TØ 888
0011          IF(MK.NE.0)RETURN
0012            LY=MØD(YY,4)
0013            IF(LY.EQ.0.AND.DDD.EQ.59)GØ TØ 2
0014            IF(LY.EQ.0.AND.DDD.GT.59)DDD=DDD-1
0015            DAY=MØD(DDD,30)
0016            MØNTH=1+(DDD/30)
0017            DAY=DAY+BUG(MONTH)
0018            IF(DAY.LE.0)MONTH=MONTH-1
0019            IF(DAY.LE.0)DAY=DAY+FNAGL(MØNTH)
0020            IF(DAY.EQ.29.AND.MONTH.EQ.2)GØ TØ 3
0021              IF(MONTH.LE.12.AND.MONTH.GE.1)MONTH=MØ(MONTH)
0022            YEAR=1900+YY
0023            IF(LY.EQ.0.AND.DDD.GE.59)DDD=DDD+1
0024            RETURN
0025      2     MONTH=MØ(2)
0026            DAY=29
0027            RETURN
0028      3     MONTH=MØ(3)
0029            DAY=1
0030            RETURN
      C
0031      1       IF(YEAR .LT. 1970) GØ TØ 888
0032            LY=MØD(YEAR,4)
0033             YY=MØD(YEAR,100)
0034             MOT=0
0035            DØ 4 I=1,12
0036            IF(MONTH.EQ.MØ(I))MØT=I
0037      4     CONTINUE
0038          IF ( MOT.EQ.0)MOT=13
0039             MONTH=MØ(MOT)
0040             IF(LY.EQ.0.AND.MOT.GT.2) DAY=DAY+1
0041            IF(MOT.EQ.1) DDD=DAY
0042            IF(MOT.GT.1) DDD=(MOT-1)*30-BUG(MOT)+DAY
0043             IF(LY.EQ.0.AND.DDD.GT.59) DAY=DAY-1
0044            RETURN
0045    888       DAY=0
0046              DDD=0
0047            YEAR=1970
0048            YY=70
0049            MONTH='ERR'
0050             RETURN
```

Figure 3.4.2.8
Listing for DDATE

DDATE.FTN /TR:IBLPCKS/WR

0051 E:D

DDATE,FTN /TR:BLOCKS/WR

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001270	348	RW,I,CON,LCL
2	$PDATA	00 010	4	RW,D,CON,LCL
3	$IDATA	000014	6	RW,D,CON,LCL
4	$VARS	00 236	79	RW,D,CON,LCL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
DDATE		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
DAY	I*4	F-000006*	DDD	I*2	F-000012*	I	I*2	4-000234	LY	I*2	4-000230	MM	I*2	F-000014*
MONTH	I*4	F-000004*	MMT	I*2	4-000232	YEAR	I*4	F-000002*	YY	I*2	F-000010*			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
BUF	I*4	4-000144	000064	26	(13)
FNAGL	I*4	4-000064	000060	24	(12)
MM	I*4	4-000000	000064	26	(13)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000626	2	1-000932	3	1-000570	4	**	888	1-001210

FUNCTIONS AND SUBROUTINES REFERENCED

SJMOD

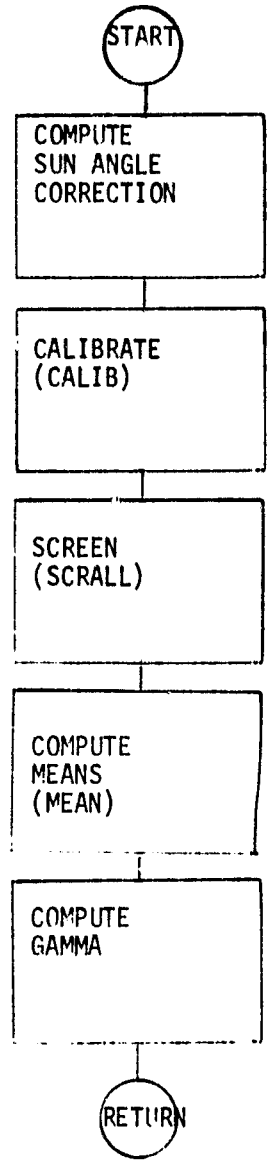
TOTAL SPACE ALLOCATED = 001552. 437

NO FPP INSTRUCTIONS GENERATED .

DDATE,LP:=DDATE

GAMMA

3.4.3.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15126153    20-SEP-79          PAGE 1
GAMMA.FTN       /TR:BLOCKS/WR

        C  GAMMA.FTN
        C
0001            SUBROUTINE GAMMA
        C   GAMMA COMPUTES THE XSTAR OPTICAL DEPTH PARAMETER GAMMA
        C       IAG IS SUN ELEVATION ANGLE IN DEGREES
        C       GMA IS THE HAZE PARAMETER OUTPUT
0002            REAL AA,BB,CC,DD,GAMMA,X(4),Z(4),ANG,S
0003            IN LUDE 'INCLU.FTN.
      * C
      * C
      * C
0004  *         INTEGER*2 CH1(209),CH2(209),CH3(209),CH4(209)
0005  *         INTEGER*2 KH1(209),KH2(209),KH3(209),KH4(209)
0006  *         BYTE  FLAG(210),FA (32),FIN(32),FIN(32)
0007  *         INTEGER*2 SEC,DATE(2),CODE,STATUS,FILL(36)
0008  *         INTEGER*2 IAG,SLN, AT,GAM,LAT(2),LNG(2)
0009  *           COMMON /ANCIL/ CODE,DATE,STATUS,FILL
0010  *           COMMON /DATA/  CH1,CH2,CH3,CH4
0011  *           COMMON /KAUTH/ KH1,KH2,KH3,KH4
0012  *           COMMON /FLAG/ F AG
0013  *           COMMON /FILMAN/ FUN,FIN,FAN,KAN
0014  *           EQUIVALENCE (IAG,FILL(2)),(SLN,FILL(3))
0015  *           EQUIVALENCE (SAT,FILL(4)),(GAM,FILL(10))
0016  *           EQUIVALENCE (LA (1),FILL(5)),(LNG(1),FILL(7))
      * C
      * C     END INCLU.FTN
      * C
        C.....................
        C.....................
        D       WRITE(6,16) IAG,SAT
0017     16     FORMAT('     GAMMA   IAG=',I5,'  SAT=',I5)
0018            IF(IAG .LT. 15 .OR. IAG .GT. 75) GO TO 333
0019            IF(SAT .LT. -2 .OR. SAT .GT. 3)  GO TO 333
0020       1    ANG= AG *3.14159 /180.0
0021            CH=. 7713/SIN(ANG)
0022            CALL CALIB(SAT,209)
0023            CALL SCALL(209)
0024            CALL MEAN(CH1,FLAG,209,NN,X(1),S)
0025            CALL MEAN(CH2,FLAG,209,NN,X(2),S)
0026            CALL MEAN(CH3,FLAG,209,NN,X(3),S)
0027            CALL MEAN(CH4,FLAG,209,NN,X(4),S)
        D       WRITE(6,17) X
0028     17     FORMAT('  GAMMA MEANS  ',4E8.3)
0029            Z(1)=CH*X(1)
0030            Z(2)=CH*X(2)
0031            Z(3)=CH*X(3)
0032            Z(4)=CH*X(4)
12 0033         AA=1.6078*(Z(1)**51,9)*(=0,8995)
11 0034         AA=AA+1.5910*(Z(2)**6 ,2)*0,4263
10 0035         AA=AA+0,8358*(Z(3)**8 ,2)*0,0759
 9 0036         AA=AA-0,5971*(Z(4)**35,9)*0,0408
 8 0037         BB=-1,2480*(Z(1)**61,9)*0,8995
 7 0038         BB=BB+1,0405*(Z(2)**6 ,2)*0,4263
 6 0039         BB=B +0,9142*(Z(3)**83,2)*0,0759
 5 0040         BB=B -0,7734*(Z(4)**35,9)*0,0408
 4 0041         CC=-Z(1)*0,8995+Z(2)*0,4263+Z(3)*0,0759-Z(4)*0,0408+11,2082
 3
```


Figure 3.4.3.8
Listing for GAMMA

```
GAMMA.FTN       /TH:OLPCKS/WR

0042            DD=BF*BB-2.0*AA*CC
0043            IF(DD .LT. 0.0) DD=0.0
0044            GAMA=(-BB-SQRT(DD))/AA
      D         WRITE(6,2020)GAMA
0045            GAMMA=GAMA*1000.+300.
0046      2020  FORMAT(//,'     GAMMA =',F6.4,//)
0047            RETURN
0048      333   TYPE *,IAC,SAT,NN
0049      8     FORMAT('   PROBLEMS   IAC=',I6,'   SAT=',I6,'   NN=',I6,
     *              /,'   S TO STOP    CR TO CONTINUE',10X,'GAMMA')
0050            ACCEPT 9,IITT
0051      9     FORMAT(A1)
0052            IF(IITT.EQ.'S') STOP
0053            GO TO 1
0054            END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001134	302	RW,I,GBL,LCL
2	$PDATA	000124	42	RW,D,GBL,LCL
3	$IDATA	00 256	79	RW,D,GBL,LCL
4	$VARS	000104	35	RW,D,GBL,LCL
5	$TEMPS	00 030	12	RW,D,GBL,LCL
6	ANCIL	000120	40	RW,D,OVR,GBL
7	DATA	002210	836	RW,D,OVR,GBL
8	KAUTH	002210	836	RW,D,OVR,GBL
9	FLAG	000322	105	RW,D,OVR,GBL
10	FILNAM	00 142	49	RW,D,OVR,GBL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
GAMMA		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
AA	R*4	4-000000	ANG	R*4	4-000064	BB	R*4	4-000004	CC	R*4	4-000010	CODE	I*2	6-000000
CR	R*4	4-000076	DD	R*4	4-000014	GAM	I*2	6-000032	GAMA	R*4	4-000020	IRG	I*4	6-000012
IITT	I*2	4-000104	KAN	I*4	10-000140	NN	I*2	4-000102	S	R*4	4-000070	SAT	I*2	6-000016
SEG	I*2	4-000074	SLN	I*4	6-000014	STATUS	I*2	6-000006						

ARRAYS

	NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
	CH1	I*2	7-000000	000642	209	(209)
	CH2	I*2	7-000642	000642	209	(209)
	CH3	I*2	7-001504	000642	209	(209)
	CH4	I*2	7-002346	000642	209	(209)
	DATE	I*2	6-000002	000004	2	(2)
	FAN	L*1	10-000100	000040	16	(32)
	FDN	L*1	10-000000	000040	16	(32)
	FILL	I*2	6-000010	000010	16	(36)
	FIN	L*1	10-000040	000040	16	(32)
	FLAG	L*1	9-000000	000322	105	(210)
	KH1	I*2	8-000000	000642	209	(209)
12	KH2	I*2	8-000642	000642	209	(209)
11	KH3	I*2	8-001504	000642	209	(209)
10	KH4	I*2	8-002346	000642	209	(209)
9	LAT	I*2	6-000020	000004	2	(2)
8	LNG	I*2	6-000024	000004	2	(2)
7	X	R*4	4-000024	000020	8	(4)
6	Z	R*4	4-000044	000020	8	(4)
5						
4						
3						

GAMMA,FTN   /TR:BLOCKS/WR

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000072	2'	3-000000	9'	3-000130	16'	**	17'	**
333	1-001004	2020'	**						

FUNCTIONS AND SUBROUTINES REFERENCED

CALIB   REAL   SCNALL $SIN $SQRT

TOTAL SPACE ALLOCATED = 011130 2336

GAMMA,LP:=GAMMA

3.4.4 DATA ACCESS GET(SEG,ARRAY,CODE)

3.4.4.1 Linkage

In case of error, GET calls subroutine GETNAM through its entry GETCRD.

3.4.4.2 Interface

GET interfaces with the calling routine through the calling arguments, and the common blocks /ANCIL/, /FILNAM/, and /DATA/.

The interface with GETCRD is through the calling arguments.

3.4.4.3 Input

NA

3.4.4.4 Output

When a segment is not in the data base, a line of output is sent to the line printer.

3.4.4.5 Storage

2385 Bytes

3.4.4.6 Description

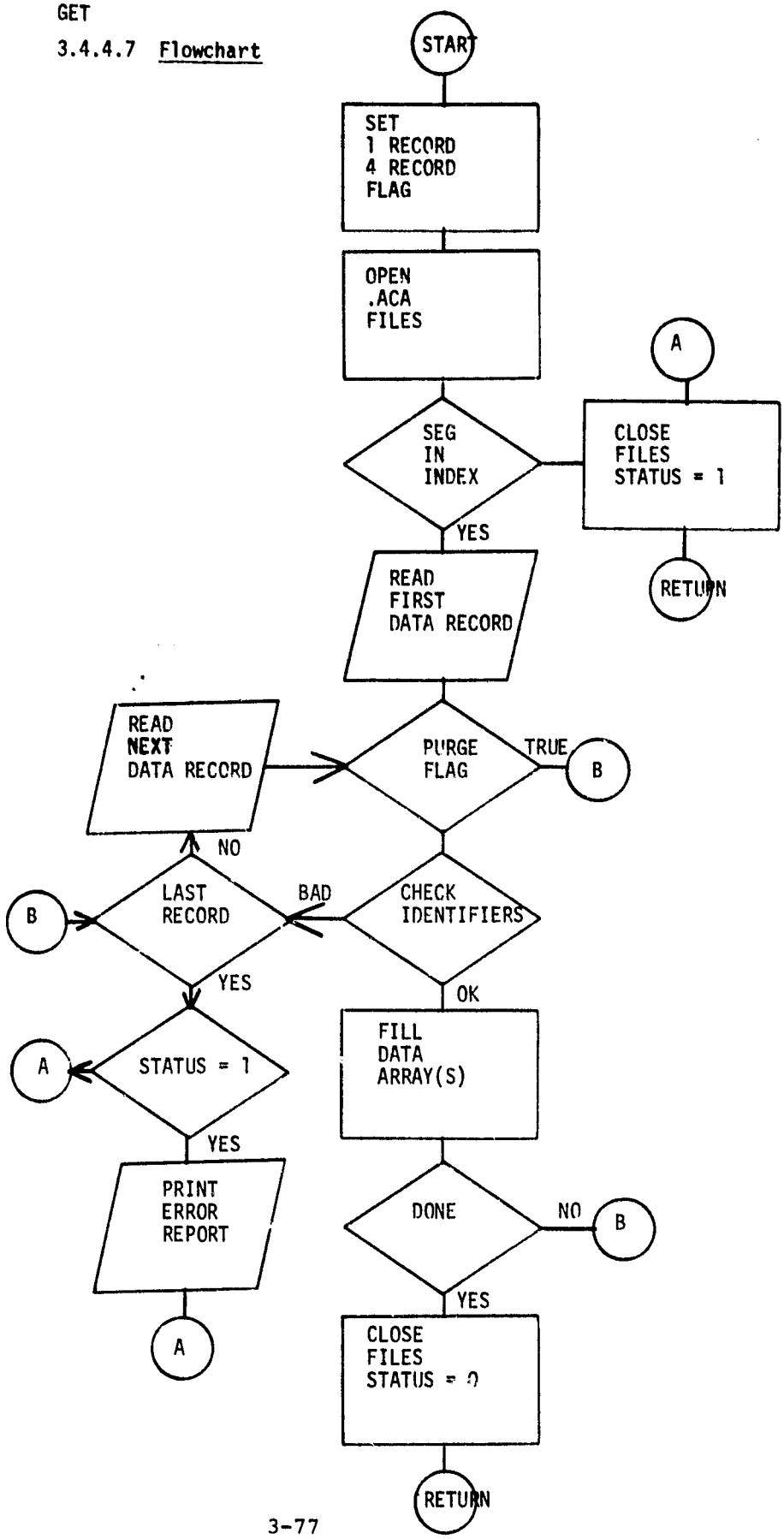
GET first opens the ACA files. In case of error opening the file, GETNAM is called and it tries again. With a second failure it returns.

After the files are opened, the index file is searched for the segment. If the segment is found, the data file for that segment is searched for the correct code name. If the sixth byte of the code name is '*', the date is also checked. If the fifth byte is '*', four records are loaded into the common /DATA/. Otherwise the dot data is passed through the argument ARRAY. The common block /ANCIL/ is used for ancillary data.

If data is not found, status is set to 1. If status is already 1, a message is printed.

GET

3.4.4.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15:23:11    20-SEP-79          PAGE 1
GET.FTN          /T 1BLOCKS/WR

      SUBROUTINE GET(SEG,ARRAY,NAME)
C
C     GET USED TO BE FETCH
C     GET INTERACTIVELY GET THE VIC FROM YOU (IF NECESSARY)
C     GET ALSO DOES 4 CHANNELS AT A TIME
C
      IMPLICIT INTEGER*2 (A-Z)
      BYTE NUM(4),FAN(32),FDN(32),FIN(32),NAME(6), CHAR(6)
      DIMENSION INDEX(256),ARRAY(209), BUFFER(256)
      DIMENSION LINDEX(256),CH(209,4)
      DIMENSION CH1(209), CH2(209),CH3(209),CH4(209)
      EQUIVALENCE (CHAR(1), BUFFER(5))
      EQUIVALENCE (LINDEX(1),LINDEX)
      EQUIVALENCE (CH1,CH(1,1)),(CH2,CH(1,2))
      EQUIVALENCE (CH3,CH(1,3)),(CH4,CH(1,4))
      COMMON /DATA/ CH1,CH2,CH3,CH4
      COMMON /ANCIL/CODE, DATE(2), STATUS, FILL(36)
      COMMON /FILNAM/ FDN,FIN,FAN,KAN
      DATA NUM/'1','2','3','4'/
       FLAG=0
       ERT C
       IF(NAME(5).EQ.'*') FLAG=1
   10 OPEN (UNIT=7,NAME=FIN,TYPE='OLD',ERR=300,
     ,ACCESS='DIRECT',READONLY)
      DO 11 I=1,209
      DO 11 J=1,4
   11 CH(I,J)=0
      READ (7'1) LINDEX
D        TYPE 13,LINDEX,SEG,DATE,NAME
   13 . FORMAT('   LINDEX.,I4,I6,2I4,6A2)
      OPEN (UNIT=8,NAME=FD ,TYPE='OLD',ERR=300,
     ,ACCESS='DIRECT',READONLY)
      BADAT=0
      DO 20, K=3,LINDEX-1,2
         READ (7'K) INDEX
         DO 20, J=1,255,2
            IF (INDEX(J).EQ.0) GO TO 30
            IF (INDEX(J).EQ.SEG) GO TO 40
   20    CONTINUE
   30 IF (STATUS .EQ. 1) PRINT 200, SEG
      GO TO 130
   40 READ (8'INDEX(J+1)) BUFFER
D        TYPE 230,(BUFFER(M),M=1,7),CHAR
  230    FORMAT('   BUFFER    ',3I6,4A3,6A2)
   45 IF (BUFFER(254)) GO TO 110
   50 DO 60, K=1,4
         IF (NAME(K).NE.CHAR(K)) GO TO 110
   60    CONTINUE
         IF(FLAG.EQ.0.AND.CHAR(5).NE.NAME(5)) GO TO 110
         IF(CHAR(6).NE.'*')GO TO 70
         LL=CHAR(5)-'0'
         IF(LL.LT.1.OR.LL.GT.4) GO TO 110
      IF (DATE(2).NE.BUFFER(3)) GO TO 100
      IF (DATE(1).NE.BUFFER(2)) GO TO 100
   70 CODE = BUFFER(8)
      DO 75  KK=1,36
```

Figure 3.4.4.8

Listing for GET

```
 75          FILL(KK)=BUFFER(217+KK)
          IF(FLAG .EQ. 0)GØ TØ 86
         DØ 8 , I=1,209
   80       CH(I,LL)=BUFFER(I+8)
         IF(CH(1,1).EQ.0.ØR.CH(1,2).EQ.0.ØR.
     *      CH(1,3).EQ.0.ØR.CH(1,4).EQ.0)  GØ TØ 110
 88      CLØSE (UNIT=7)
         CLØSE (UNIT=8)
         STATUS=0
         RETURN
 86        CØNTINUE
           DØ 87 L=1,209
 87        ARRAY(L)=BUFFER(L+8)
           GØ TØ 88
 300        IF(ERT.NE.0) GØ TØ 130
          CALL GETCRD (FAN,FLN,FIN,KAN)
            E T=1
          GØ TØ 10
   90 IF (DATATE.EQ.1) GØ TØ 120
      IF (STATUS.EQ.1) PRINT 210, NAME, SEG
      GØ TØ 130
  100 DATATE = 1
  110 IF ( BUFFER(256).EQ.INDEX(J+1)) GØ TØ 90
      READ (8'BUFFER(256)) BUFFER
D             TYPE 230,(BUFFER(MM),MM=1,7),CHAR
      GØ TØ 45
  120 IF (STATUS.EQ.1) PRINT 220, NAME, SEG, DATE,BUFFER(2),BUFFER(3)
  130 CLØSE (UNIT=7)
      CLØSE (UNIT=8)
      STATUS = 1
      RETURN
  200 FØRMAT (' ',' SEGMENT ',I4,' HAS NØ DATA IN THE DATABASE'/)
  210 FØRMAT (' ',' THE ARRAY NAME, ',6A1,', DØES NØT EXIST FØR SEGMENT
     ,',I4/)
  220 FØRMAT (' ',' THE ARRAY NAME, ',6A1,', IS PRESENT IN THE DATABASE
     ,FØR SEGMENT ',I4,', BUT NØT FØR THE DATE ',I2,X,I3/,
     ,'       THE CØNTENTS ØF THE BUFFER ARE ',2I4,//)
      END
```

PROGRAM SECTIONS

NUMBER	NAME		SIZE	ATTRIBUTES
1	$CODE1	001730	492	RW,I,GBL,LCL
3	LIDATA	000670	188	RW,D,GBL,LCL
4	$VARS	003026	779	RW,D,GBL,LCL
5	$TEMPS	000002	1	RW,D,GBL,LCL
6	DATA	003210	636	RW,D,OVR,GBL
7	ASCII	00 120	40	RW,D,OVR,GBL
8	FILNAM	00 142	49	RW,D,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
GET		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
BADATE	I*2	4-003014	CODE	I*2	7-000000	ERT	I*2	4-003006	FLAG	I*2	4-003004	I	I*2	4-003010
J	I*2	4-003012	K	I*2	4-003016	KAN	I*2	8-000140	KK	I*2	4-003022	L	I*2	4-003024
LINDEX	I*2	4-00 000	LL	I*2	4-003020	SEG	I*2	F-000002*	STATUS	I*2	7-000006			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
ARRAY	I*2	F-000004*	000642	209	(209)
BUFFER	I*2	4-00 000	001000	256	(256)
CH	I*2	6-00 000	003210	836	(209,4)
CHAR	L*1	4-001010	000006	3	(6)
CH1	I*2	6-00 000	000642	209	(209)
CH2	I*2	6-000642	000642	209	(209)
CH3	I*2	6-001504	000642	209	(209)
CH4	I*2	6-00 346	000642	209	(209)
DATE	I*2	7-000002	000004	2	(2)
FAN	L*1	8-000100	000040	16	(32)
FDN	L*1	8-00 000	000040	16	(32)
FILL	I*2	7-000010	000110	36	(36)
FIN	L*1	8-00 040	000040	16	(32)
INDEX	I*2	4-00 004	001000	256	(256)
MINDEX	I*2	4-000000	001000	256	(256)
NAME	L*1	F-00 006*	000006	3	(6)
NUM	L*1	4-002000	000004	2	(4)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS

10	1-000110	11	**	13'	**	20	**	30	1-000450
40	1-000522	45	1-000574	50	**	60	**	7"	1-001016
75	**	80	**	86	1-001250	87	**	8?	1-001310
90	1-001366	100	1-001456	110	1-001472	120	1-001560	130	1-001666
200'	3-000000	210'	3-000060	220'	3-000152	230'	**	3-0	1-001?24

FUNCTIONS AND SUBROUTINES REFERENCED

CLASS GETCRD OPENS

TOTAL SPACE ALLOCATED = 011242 2365

NO FPP INSTRUCTIONS GENERATED

GET,LP1=GET

3.4.5 ACA FILE NAME INPUT (GETNAM(FDN,FIN))

3.4.5.1 Linkage

GETNAM does not call other subroutines.

3.4.5.2 Interface

GETNAM is interactive. The interface with the calling program is only through the arguments.

The entry GETCRD(FAN,FDN,FIN,KQ) is not interactive except when an error is encountered.

3.4.5.3 Input

See figure 3.4.5.4 for an example of interactive file definition.

3.4.5.4 Output

See figure 3.4.5.4 for an example of interactive file definition.

3.4.5.5 Storage

655 Bytes

3.4.5.6 Description

GETNAM establishes the ACA data file name FDN and the ACA index file FIN.

Interactively if you enter [a,b] FDN and FIN are defined.

```
FDN = [a,b]DOTDATA.ACA.
FIN = [a,b]DOTINDEX.ACA.
```

If you enter FN, file [200,7] FN.DAT is opened and FDN and FIN are read. If Z is entered FND and FIN are obtained interactively.

```
$
  INPUT UIC OF DOTDATA FILE        FN, S OR Z  GETNAM
[200,7]


$
  INPUT UIC OF DOTDATA FILE        FN, S OR Z  GETNAM
FN


$
  INPUT UIC OF DOTDATA FILE        FN, S OR Z  GETNAM
S
GENPLT  --- STOP


$
  INPUT UIC OF DOTDATA FILE        FN, S OR Z  GETNAM
Z
 ENTER 32 CHARACTER DATA FILE NAME  END=@
[200,7]DDATA.ACA@
 ENTER 32 CHARACTER INDEX FILE NAME   END=@
[200,7]DINDX.ACA@

[200,7]DDATA.ACA@
[200,7]DINDX.ACA@
 CR TO ACCEPT ,S TO STOP ,A TO TRY AGAIN
A
 ENTER 32 CHARACTER DATA FILE NAME  END=@
[200,7]DOTDATA.ACA@
 ENTER 32 CHARACTER INDEX FILE NAME   END=@
[200,7]DOTINDEX.ACA@

[200,7]DOTDATA.ACA@
[200,7]DOTINDEX.ACA@
CR TO ACCEPT ,S TO STOP ,A TO TRY AGAIN
```


Figure 3.4.5.4

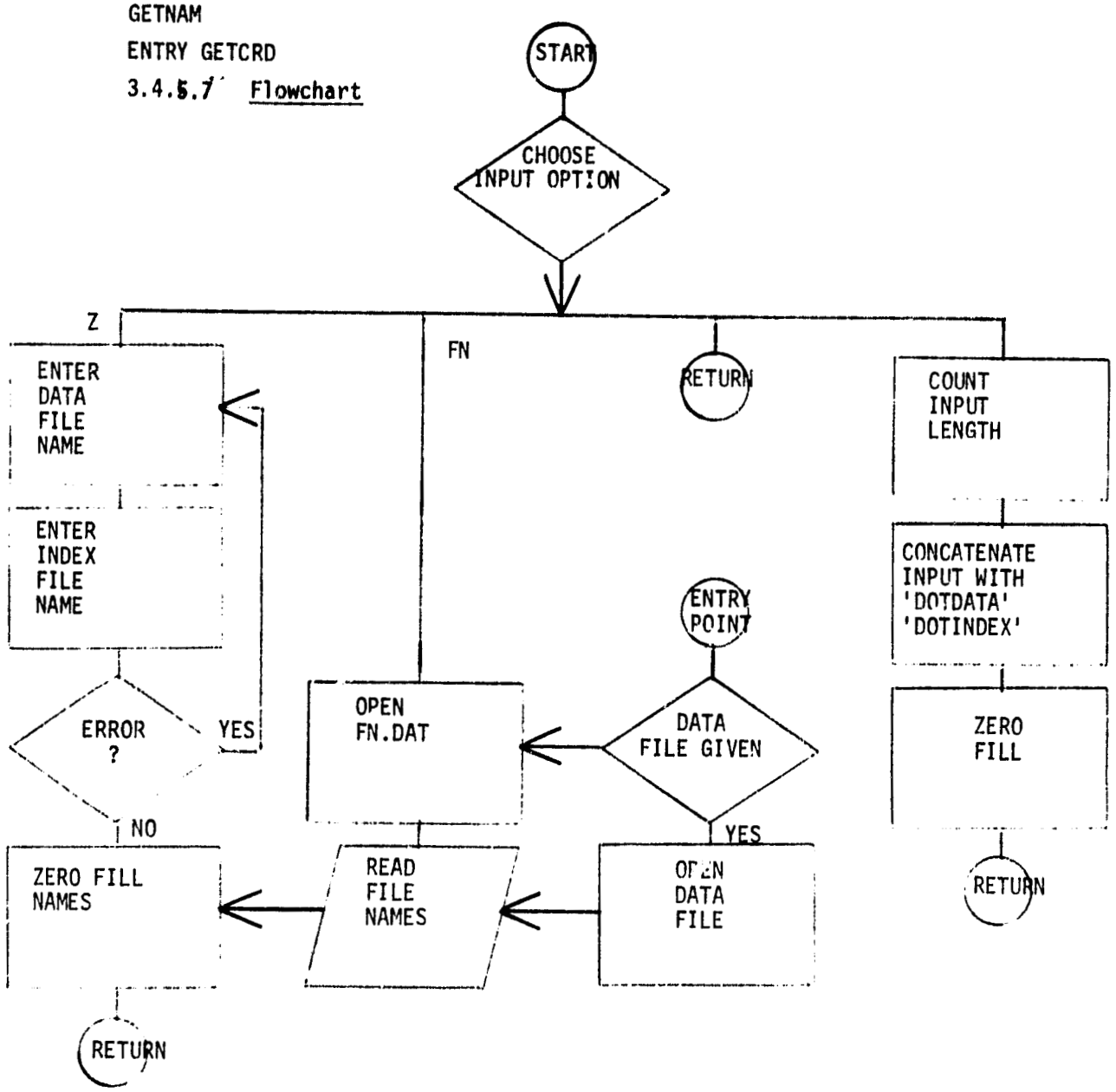
For entry GETCRD, the file name in FAN of length KQ-1, is opened and FDN and FIN are read. If KQ = 1, [200,7] FN.DAT is opened and FDN and FIN are read.

In every case, the arrays FIN and FDN are zero filled before control is returned to the calling program.

GETNAM

ENTRY GETCRD

3.4.5.7 Flowchart



```
C     GETNAM INTERACTIVLY GETS FILE UIC FØR DØTDATA.ACA
          SUBROUTINE GETNAM(FDN,FIN)
C     FDN IS THE DATA FILE NAME ([XXX,XXX]DØTDATA.ACA000)
C     FIN IS THE INDEX FILE NAME ([XXX,XXX]DØTINDEX.ACA000)
          IMPLICIT INTEGER*2 (A-Z)
          BYTE UIC(10),FDN(32),FIN(32),FAN(32),DØTDA(11),DØTIN(12)
          BYTE IT,IP
          DATA DØTDA/'D','Ø','T','D','A','T','A','.','A','C','A'/
          DATA DØTIN/'D','Ø','T','I','N','D','E','X','.','A','C','A'/
          NN=0
 99       TYPE 100
 100      FØRMAT('   INPUT UIC ØF DØTDATA FILE       FN, S ØR Z  GETNAM')
          ACCEPT 200,UIC
 200      FØRMAT(10A1)
            IF(UIC(1).EQ.'Z')GØ TØ 600
            IF(UIC(1).EQ.'F') GØ TØ 404
             IF(UIC(1).EQ.'S')STØP
          DØ 301 L=1,10
          IF(UIC(L).EQ.' '.AND.NN.EQ.0) NN=L
 301      CØNTINUE
          DØ 302 I=1,NN
          FDN(I)=UIC(I)
          FIN(I)=UIC(I)
 302      CØNTINUE
          DØ 303 I=NN,NN+10
          FDN(I)=DØTDA(I-NN+1)
          FIN(I)=DØTIN(I-NN+1)
 303      CØNTINUE
          DØ 304 I=NN+11,32
          FDN(I)=0
          FIN(I)=0
 304      CØNTINUE
          FIN(NN+11)=DØTIN(12)
           RETURN
 600        CØNTINUE
            TYPE 610
 610        FØRMAT('  ENTER 32 CHARACTER DATA FILE NAME  END=@')
            ACCEPT 620, FDN
 620        FØRMAT(32A1)
            TYPE 611
 611        FØRMAT('  ENTER 32 CHARACTER INDEX FILE NAME   END=@')
            ACCEPT 620,FIN
            TYPE 612,FDN,FIN
 612        FØRMAT(/,' ',32A1,/,' ',32A1,/
     *       ,  'ØR TØ ACCEPT ,S TØ STØP ,A TØ TRY AGAIN')
            ACCEPT 200,TX
            IF(TX.EQ.'S') STØP
            IF(TX.EQ.'A') GØ TØ 600
 650          CØNTINUE
            ZED=0
            ZAD=0
             DØ 613 K=1,32
              IF(FIN(K).EQ.'@') ZED=99
              IF(ZED.EQ.99)FIN(K)=0
               IF(FDN(K).EQ.'@') ZAD=99
 613           IF(ZAD.EQ.99)FDN(K)=0
```

Figure 3.4.5.8

Listing for GETNAM

```
GETNAM,FTN      /TR:BLOCKS/HR

0053                RETURN
        C
0054                ENTRY GETCRD(FAN,FDN,FIN,KG)
        D           WRITE(6,444) FAN
0055                IF(A. .EQ. 1) GO TO 403
0056                D. 401 II=KU,32
0057        401     FAN(II)=0
        D           WRITE(6,444) FAN
0058        444     FORMAT(' ',32A4,//)
0059                OPEN(UNIT=2,NAME=FAN,TYPE='OLD',
                *       ACCESS='SEQUENTIAL',ERR=99)
0060        402     READ(2,420,ERR=666,END=666)FIN
0061                READ(2,420,ERR=666,END=666)FIN
0062                CLOSE(UNIT=2)
0063        420     FORMAT(32A1)
0064                GO TO 650
0065        665     WRITE(6,667),FAN,FDN,FIN
0066                CLOSE(UNIT=5)
0067        667     FORMAT(' FILE NAME ERROR   GETCRD (GETNAM)',3(/32A1))
0068                RETURN
0069        403     OPEN(UNIT=2,NAME='[200,7]FA,LAT',TYPE='OLD',
                *       ACCESS='SEQUENTIAL',ERR=99,READONLY)
0070                GO TO 402
0071                END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CØDE1	001552	437	Rw,I,CØN,LCL
2	$PDATA	001016	7	Rw,D,CØN,LCL
3	$IDATA	00 560	184	Rw,D,CØN,LCL
4	$VARS	000064	26	Rw,D,CØN,LCL
5	$TEMPS	00 002	1	Rw,D,CØN,LCL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
GETCMD		1-001062	GETNAM		1-000000									

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-000050	II	I*2	4-000062	IP	L*1	4-000042	IT	L*1	4-000041	K	I*2	4-000060
KQ	I*2	F-00[illegible]	L	I*2	4-000046	NN	I*2	4-000044	TX	I*2	4-000052	ZAD	I*2	4-000056
ZED	I*2	4-00 054												

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
DØTDA	L*1	4-000012	000013	5	(11)
DØTIN	L*1	4-000025	000014	6	(12)
FAL	L*1	F-00 002*	000040	16	(32)
FDV	L*1	3-000460*	000040	16	(32)
FIL	L*1	3-000452*	000040	16	(32)
UIC	L*1	4-00 000	000012	5	(10)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
99	1-000044	100'	3-000000	200'	3-000070	301	**	302	**
303	**	304	**	401	**	402	1-001216	403	1-001240
420'	3-000330	444'	**	600	1-000452	610'	3-000074	611'	3-000156
612'	3-000256	613	**	620'	3-000152	650	1-000716	696	1-001344
667'	3-000334								

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSE ØPEN

TØTAL SPACE ALLØCATED = 002436 635

```
GETNAM,FTN      /TH:BLØCKS/WR

N2 FPP INSTRUCTIONS GENERATED

GETNAM,LP:=GETNAM
```

3.4.6 HAZE CORRECTION (HAZCOR)

3.4.6.1 Linkage

HAZCOR calls KAUTH.

3.4.6.2 Interface

HAZCOR interfaces with the calling program through the common blocks /ANCIL/, /DATA/, and /KAUTH/.

The interface with KAUTH is through the common block /KAUTH/.

3.4.6.3 Input

None

3.4.6.4 Output

None

3.4.6.5 Storage

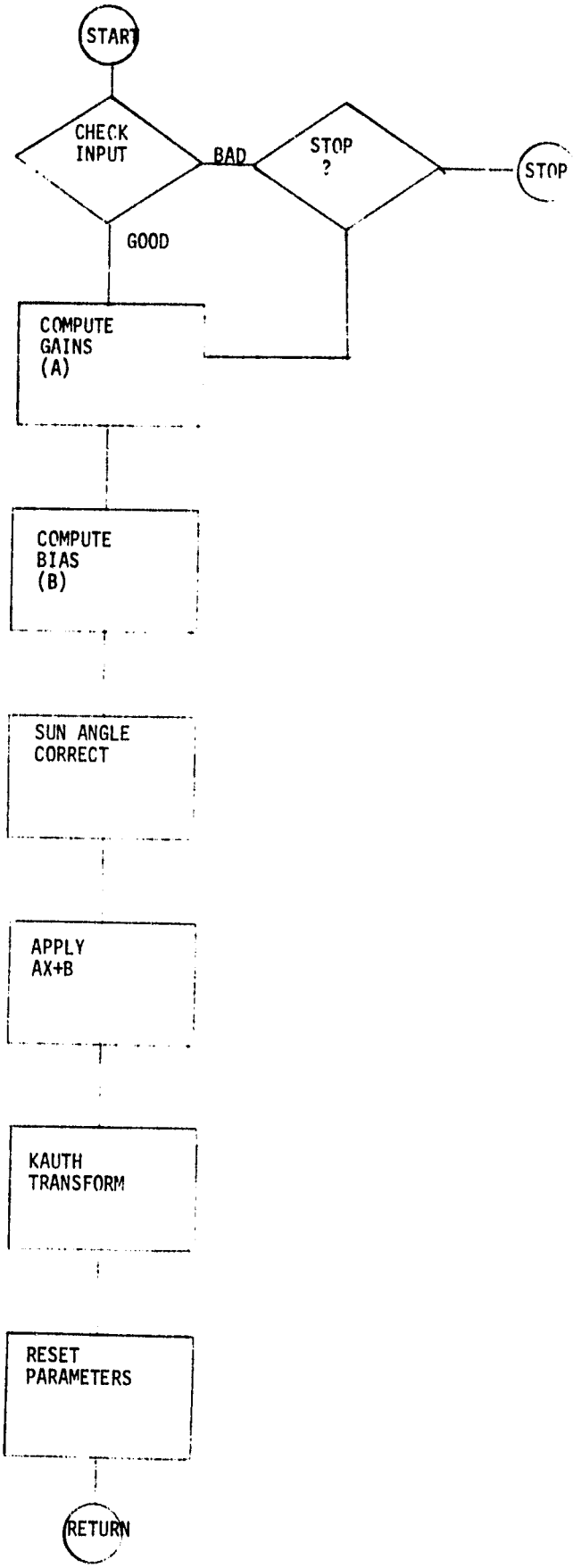
2245 Bytes

3.4.6.6 Description

HAZCOR takes the value of GAM from the common block /ANCIL/, computes correction terms and applies them to the contents of /DATA/. Then KAUTH transforms the corrected values stored in DATA, updating /KAUTH/.

HAZCOR

3.4.6.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15:27:12    20-SEP-79          PAGE 1
HAZCOR.FTN      /T :BLOCKS/WR

        C  HAZCOR.FTN
        C
0001            SUBROUTINE HAZCOR
0002            REAL A(4),B(4),GAMMA,COR,ANGRAD
0003            INCLUDE 'INCLU.FTN'
      *  C
      *  C
      *  C
0004  *         INTEGER*2 CH1(209),CH2(209),CH3(209),CH4(209)
0005  *         INTEGER*2 KH1(209),KH2(209),KH3(209),KH4(209)
0006  *         BYTE  FLAG(210),FAN(32),FDN(32),FIN(32)
0007  *         INTEGER*2 SEG,DATE 2),CODE,STATUS,FILL(36)
0008  *         INTEGER*2 IAG,SLN,SAT,GAM,LAT(2),LNG(2)
0009  *            COMMON /ANCIL/  CODE,DATE,STATUS,FILL
0010  *            COMMON /DATA/  CH1,CH2,CH3,CH4
0011  *            COMMON /KAUTH/ KH1,KH2,KH3,KH4
0012  *            COMMON /FLAG/ FLAG
0013  *            COMMON /FILNAM/ FDN,FIN,FAN,KAN
0014  *            EQUIVALENCE (IAG,FILL(2)),(SLN,FILL(3))
0015  *            EQUIVALENCE (SAT,FILL(4)),(GAM,FILL(10))
0016  *            EQUIVALENCE (LAT(1),FILL(5)),(LNG(1),FILL(7))
      *  C
      *  C     END INCLU.FTN
      *  C
        C
        D       TYPE 100
0017     100    FORMAT(' HAZCOR DEBUG ON')
0018            GAMMA=(GAM-300)/10.0
0019            IF(GAMMA.GT.0.7 .OR. GAMMA.LT.-0.7) GO TO 333
0020            IF(IAG.LT.15 .OR. IAG.GT.75)        GO TO 333
0021       1    ANGRAD=IAG*3.1416/180
0022            COR=.77713/SIN(ANGRAD)
0023            A(1)=EXP(-1.2340*GAMMA)
0024            A(2)=EXP(-1.0910*GAMMA)
0025            A(3)=EXP(-0.8358*GAMMA)
0026            A(4)=EXP(-0.5961*GAMMA)
0027            B(1)=61.0*(1-A(1))*COR
0028            B(2)=66.2*(1-A(2))*COR
0029            B(3)=83.2*(1-A(3))*COR
0030            B(4)=35.9*(1-A(4))*COR
0031            A(1)=A(1)*COR
0032            A(2)=A(2)*COR
0033            A(3)=A(3)*COR
0034            A(4)=A(4)*COR
        D       TYPE 102,A,B
0035     102    FORMAT(' HAZCOR A AND B',/,8F6.3)
        C
12      C
11 0036         DO 2 K=1,209
10 0037            CH1(K)=CH1(K)*A(1)+B(1)
 9 0038            CH2(K)=CH2(K)*A(2)+B(2)
 8 0039            CH3(K)=CH3(K)*A(3)+B(3)
 7 0040            CH4(K)=CH4(K)*A(4)+B(4)
 6 0041       2    CONTINUE
 5 0042            CALL KAUTH(209)
 4 0043            SAT=-2
```

Figure 3.4.6.8

Listing for HAZCOR

```
              IAG=51
              GAM=300
              RETURN
      333     TYPE 200, GAM,GAMMA,IAG,SAT
      200     FORMAT('      HAZCOR PROBLEMS',/,
             *      '       GAM=',I6,/,
             *      '       GAMM =',F6.4,/,
             *      '       IAG=',I6,/,
             *      '       SAT=',I6,/,
             *      '       CR TO CONTINUE      S TO STOP')
              ACCEPT 201,ITT
      201     FORMAT(A1)
              IF(ITT .EQ. 'S') STOP
              GO TO 1
              END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001014	262	RW,I,CON,LCL
2	$PDATA	00 064	26	RW,D,CON,LCL
3	$IDATA	00 202	65	RW,D,CON,LCL
4	$VARS	000064	26	RW,D,CON,LCL
6	ANCIL	00 120	40	RW,D,OVR,GBL
7	DATA	003210	836	RW,D,OVR,GBL
8	KAUTH	003210	836	RW,D,OVR,GBL
9	FLAG	00 322	105	RW,D,OVR,GBL
10	FILNAM	000142	49	RW,D,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
HAZCOR		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
ANGRAD	R*4	4-000050	CODE	I*2	6-000000	COR	R*4	4-000044	GAM	I*2	6-000032	GAMMA	R*4	4-000040
IAG	I*2	6-000012	IIT	I*2	4-000060	ITT	I*2	4-000062	K	I*2	4-000056	KAN	I*2	10-000140
SAT	I*2	6-000016	SEG	I*2	4-000054	SLN	I*2	6-000014	STATUS	I*2	6-000006			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
A	R*4	4-000000	000020	8	(4)
B	R*4	4-000020	000020	8	(4)
CH1	I*2	7-000000	000642	209	(209)
CH2	I*2	7-000642	000642	209	(209)
CH3	I*2	7-001504	000642	209	(209)
CH4	I*2	7-002346	000642	209	(209)
DATE	I*2	6-000002	000004	2	(2)
FAN	L*1	10-000100	000040	16	(32)
FDN	L*1	10-000000	000040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	10-000040	000040	16	(32)
FLAG	L*1	9-000000	000322	105	(210)
KH1	I*2	8-000000	000642	209	(209)
KH2	I*2	8-000642	000642	209	(209)
KH3	I*2	8-001504	000642	209	(209)
KH4	I*2	8-002346	000642	209	(209)
LAT	I*2	6-000020	000004	2	(2)
LNG	I*2	6-000024	000004	2	(2)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000126	2	**	100'	**	102'	**	200'	3-000000
201'	3-000172	333	1-000654						

FUNCTIONS AND SUBROUTINES REFERENCED

KAUTH $EXP $SIN

TOTAL SPACE ALLOCATED = 010612 2245

HAZCAR,LP:=HAZCAR

3.4.7 INTERACTIVE SEGMENT CHOICE (INTRAC(SEG,DATE,CNAM)

3.4.7.1 Linkage

INTRAC calls GETNAM.

3.4.7.2 Interface

The interface with the calling program is through the arguments and the common block /FILNAM/.

The interface with GETNAM is through the arguments.

3.4.7.3 Input

INTRAC is interactive. See figure 3.4.7.4 for examples.

3.4.7.4 Output

INTRAC is interactive. See figure 3.4.7.4 for examples.

There is an optional line printer report.

3.4.7.5 Storage

1566 Bytes

3.4.7.6 Description

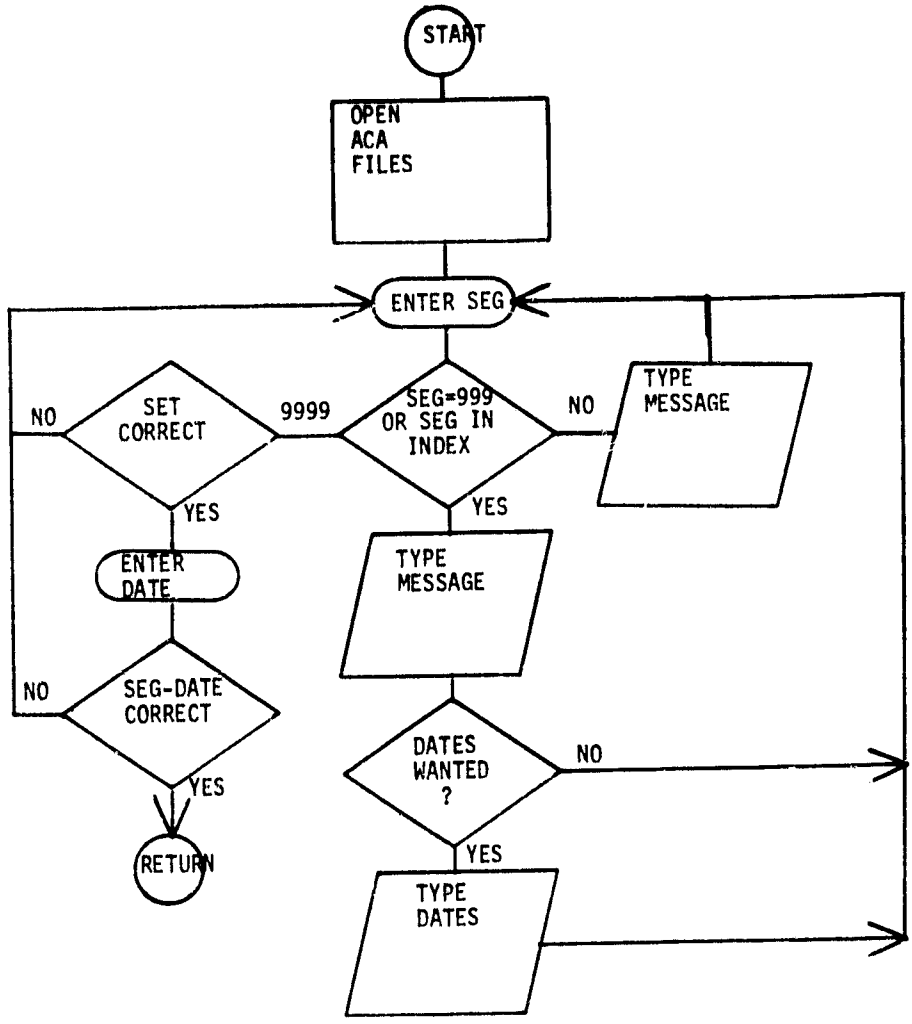
INTRAC makes it possible to interactively choose segment acquisitions for further processing.

INTRAC first opens the ACA files. If this fails, it interactively gets the file names through GETNAM and opens the ACA files. It then asks for a segment number. If it gets 9999 it returns. If it gets a segment in the data base, the acquisitions are optionally displayed. If the segment is not in the data base the program asks for a new segment.

```
$
 TO STOP TYPE S                    INTRAC

  ENTER SEGMENT NUMBER (9999 TO PROCESS)
1566
SEGMENT  1566  APPEARS IN THE DOTINDEX
TYPE Y FOR DATES ,N FOR A NEW SEGMENT
Y
     1566   0   0  TRTH78          0     5994
     1566 78196  SWCH1◆           0     2948
     1566 78232  SWCH1◆           0     2944
     1566 78115  SWCH1◆           0     2940
     1566 78133  SWCH1◆           0     2936
     1566 78169  SWCH1◆           0     2932
  ENTER SEGMENT NUMBER (9999 TO PROCESS)
9999



    SEGMENT IS  1566
    FOR NEW SEGMENT TYPE X,    CR TO CONTINUE
                                              INTRAC

    ENTER DATE                  INTRAC
78232
   ENTER Y TO PROCESS  1566  78232
Y
```


Figure 3.4.7.4

3.4.7.7 Flowchart



```
        C INTRAC INTERACTIVLY INPUTS DATA NAMES FØR  GET
0001              SUBROUTINE INTRAC(SEG,DATE,CNAM1)
0002            IMPLICIT INTEGER*2 (A-Z)
0003            INT GER*2 SEG,SIG,DATE(2),CNAM1(3)
0004            DIMENSION INDEX(256),STUF(7),INDEX(256),BUFFER(256)
0005            DIMENSION  ALS%(2)
0006            BYTE JIC(10),FD%(32),FIN(32),FAN(32),DØTDA(11),DØTIN(12)
0007            BYTE IT,IP,IQ,TT(2)
0008            EQUIVALENCE (INDEX(1),LDX),(MINDEX(2),MXR)
0009            EQUIVALENCE (IZ,BUFFER(256)),(STUF(1),BUFFER(1))
0010            EQUIVALENCE(STUF(7),TT(1))
0011            COMMON /FILNAM/ FD%,FIN,FAN,MAN
0012            DATA DATDA/'D','A','T','D','A','T','A',',','A','C','A'/
0013            DATA DØTI/'D','ø',,'T','I','N','D','E','X',',','A','C','A'/
0014      10      CONTINUE
0015              TYPE 11
0016      11      FORMAT('  T  STØP TYPE S                INTRAC')
0017              ACCEPT 210,IQ
0018              IF (IQ.EQ.'S')STØP
0019              IF(FIN(1),EQ,0) GØ TØ 300
0020            ØPE (UNIT=3,NAME=FIN,TYPE='OLD',ACCESS='DIRECT',ERR=300)
0021            READ(3'1)MINDEX
0022            ØPE (UNIT=4,NAME=FD%,TYPE='OLD',ACCESS='DIRECT',ERR=300)
0023      1     CONTINUE
0024            TYPE 102
0025      102   FORMAT('   ENTER SEGMENT NUMBER (9999 TØ PRØCESS)')
0026            ACCEPT 202,SIG
0027      202   FORMAT(I4)
0028            IF (SIG.EQ.9999) GØ TØ 666
0029            IF( IG,EQ,9998) STØP
0030             SEG=SIG
0031            DØ 310,K=3,LDX=1,2
0032               READ(3'K) INDEX
0033                DØ 311 J=1,255,2
0034               IF(INDEX(J),EQ,0) GØ TØ 40
0035               IF(INDEX(J),EQ,SEG) GØ TØ 50
0036      311      CONTINUE
0037      310     CONTINUE
0038       40     TYPE 104, SEG
0039      104     FØRMAT(///'  ***  SEG',I6,'  DØES NØT APPEAR IN DØTINDEX',//)
0040            GØ TØ 1
0041       50     TYPE 106, SEG
0042      106     FØRMAT(' SEGMENT',I6,'  APPEARS IN THE DØTINDEX',/
     *                ' TYPE Y FØR DATES ,A FØR A NEW SEGMENT')
0043              ACCEPT 206, IT
0044      206     FØRMAT(A1)
0045              IF(IT.EQ.'Y')GØ TØ 1
12 0046                 FIRST=INDEX(Y+1)
11 0047               IZ=FIRST
10 0048               IERR=0
 9 0049                IEND=0
 8 0050     2    CØNTINUE
 7 0051             IF(IEND.EQ.1) GØ TØ 1
 6 0052                ALS%(2)=IZ
 5 0053               READ(4'IZ)BUFFER
 4 0054               IF(IZ,EQ,FIN(T))IEND=1
 3
```


Figure 3.4.7.8
Listing for INTRAC

```
0055                    ALS2(1)=SUFFLE(254)
          D             WRITE(2,108) STUF,ALS0,I4,FIRST
0056                    IF(TT(2) ,EQ, '0') TYPE 108,STUF,ALS0
0057                    IF(STUF(7), E,'1') GØ TØ 2
0058                    IERR=IERR+1
0059                    IF(IERR.LE.50)S  TØ 1
0060                     TYPE 106,STUF,ALS2
0061                    IF(I2.EQ.FIRST) G  TØ 1
0062                     CHAR1(1)=STUF(5)
0063                     CHAR1(2)=STUF(6)
0064                     CHAR1(3)=STUF(7)
0065       108          FØRMAT('   ',I6,2I3,4A2,4I8)
0066                    GØ TØ 2
0067         666      TYPE 110,SEG
          D           WRITE(2,106) STUF    !LAST LINE PUSHER
0068         110      FØRMAT(//,
                 C           /,'     SEGMENT IS',I6,
                 C           /,'     FØR NEW SEGMENT TYPE X,   CR TØ CONTINUE',
                 C           /,30X,'                INTRAC')
          D          TYPE 177
0069         177      FØRMAT('      TØ KILL LINEPRINTER ØUTPUT TYPE N')
0070                  ACCEPT 210, IP
0071         210      FØRMAT(A1)
0072                   IF(IP.EQ.'X') GØ TØ 1
0073                   TYPE 112
0074         112       FØRMAT('     ENTER DATE                INTRAC')
0075                    CCEPT 212,DATE
0076         212       FØRMAT(I2,I3)
0077                     YPE 114,SEG,DATE
0078         114       FØRMAT('    ENTER Y TØ PRØCESS',I6,I4,I3)
0079                   ACCEPT 210,IQ
0080                   IF(IQ.NE.'Y') GØ TØ 1
          D            IF(IP.EQ.'N') CLØSE(UNIT=2,DISP='DELETE')
          D            I (IP.NE.'N') CLØSE(UNIT=2,DISP='PRINT')
0081                   CL SE(UNIT=3)
0082                   CLØSE(UNIT=4)
0083                   RETURN
0084         300       CALL GETNAM(FDN,FIN)
0085                   G  TØ 10
0086                     END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001606	451	RW,I,LCL,LCL
3	$IDATA	00 036	271	RW,D,LCL,LCL
4	$VARS	003004	794	RW,D,LCL,LCL
5	$STORS	00 002	1	RW,D,LCL,LCL
6	FILNAM	000142	49	RW,D,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
INTRAC		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
FIRST	I*2	4-003056	IEND	I*2	4-003062	IERR	I*2	4-003060	IP	L*1	4-003050	IQ	L*1	4-003051
IT	L*1	4-003047	IZ	I*2	4-000776	J	I*2	4-003054	K	I*2	4-003052	KSW	I*4	6-000140
LDX	I*2	4-001000	MXP	I*2	4-001002	SEG	I*2	F-000002*	SIG	I*2	4-002000			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
ALSW	I*2	4-003002	000004	2	(2)
BUFFER	I*2	4-001000	001000	256	(256)
CHAN1	I*2	F-000006*	000006	3	(3)
DATE	I*2	F-000004*	000004	2	(2)
DATDA	L*1	4-003220	000013	5	(11)
DATIM	L*1	4-003033	000014	6	(12)
FAL	L*1	6-00 100	000040	16	(32)
FDV	L*1	6-000000	000040	16	(32)
FIL	L*1	6-00 040	000040	6	(32)
INDEX	I*2	4-002002	001000	256	(256)
MTINDEX	I*2	4-001002	001000	256	(256)
STOF	I*2	4-000000	000016	7	(7)
TT	L*1	4-000014	000002	1	(2)
UIC	L*1	4-00 006	000012	5	(10)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000252	2	1-000736	10	1-000062	11'	3-000000	40	1-000342
50	1-000602	102'	3-000150	104'	3-000130	106'	3-000214	1-6'	3-000142
110'	3-000364	112'	3-000530	114'	3-000604	177'	**	202'	3-000164
206'	3-000356	210'	3-000524	212'	3-000576	300	1-001560	310	**
311	**	666	1-001254						

FUNCTIØNS AND SUBRØUTINES REFERENCED

CLØSE GETNAM ØPEN

TØTAL SPACE ALLØCATED = 004674 15^6

NØ FPP INSTRUCTIØNS GENERATED

INTRAC,LP1=I^TRAC

3.4.8 KAUTH TRANSFORMATION (KAUTH(N))

3.4.8.1 Linkage

KAUTH does not call any other subroutine.

3.4.8.2 Interface

KAUTH transforms the first N sets of channel values in the common block /DATA/ and puts the result into the common block /KAUTH/.

3.4.8.3 Input

None

3.4.8.4 Output

None

3.4.8.5 Storage

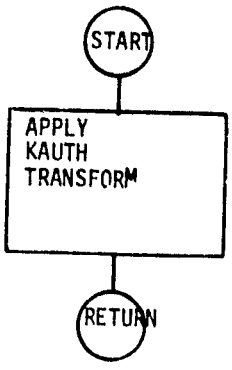
1824 Bytes

3.4.8.6 Description

KAUTH applies the LANDSAT 2 version of the Kauth-Thomas transformation to the 4 channel data in /DATA/ and stores the 4 channel result in /KAUTH/.

KAUTH

3.4.8.7 Flowchart



```
FØRTRAN IV-PLUS V02-51          15:25:40    26-SEP-79          PAGE 1
KAUTH.FTN        /TR:BLOCKS/WR

      C  KAUTH COMPUTES THE KAUTH THOMAS TRANSFØRMATION
            SUBROUTINE KAUTH(N)
            IMPLICIT INTEGER*2 (A-S)
            DIMENSION CH1(209),CH2(209),CH3(209),CH4(209)
            DIMENSION KH1(209),KH2(209),KH3(209),KH4(209)
            REAL  KH(4)
            REAL  R(4,4)
            COMMON /DATA/ CH1,CH2,CH3,CH4
            COMMON /KAUTH/ KH1,KH2,KH3,KH4
              DATA  R  /.332,.603,.675,.262,
          C              -.283,-.660,.577,.388,
          C              -.900,.428,.076,-.040,
          C              -.016,.13 ,-.452,.882/
            DO 100 I=1,N
            DO 101 J=1,4
            KH(J)=30.+CH1(I)*R(1,J)+CH2(I)*R(2,J)+CH3(I)*R(3,J)+CH4(I)*R(4,J)
      101   CONTINUE
            KH1(I)=KH(1)-29.5
            KH2(I)=KH(2)-29.5
            IF( KH2(I) .LT. 0) KH2(I)=KH2(I)-1
            KH3(I)=KH(3)-29.5
           IF(KH3(I) .LT. 0) KH3(I)=KH3(I)-1
            KH4(I)=KH(4)-29.5
      100   CONTINUE
             RETURN
             END
```


Figure 3.4.8.8

Listing for KAUTH

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	000322	105	RW,I,CON,LCL
4	$VARS	00 124	42	RW,D,CON,LCL
5	$TEMPS	00 012	5	RW,D,CON,LCL
6	DATA	003211	836	RW,D,OVR,GBL
7	KAUTH	00 210	836	RW,D,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
KAUTH		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-000120	J	I*2	4-000122	N	I*2	F-000002*						

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
CH1	I*2	6-000000	000642	209	(209)
CH2	I*2	6-000642	000642	209	(209)
CH3	I*2	6-001504	000642	209	(209)
CH4	I*2	6-002346	000642	209	(209)
KH	R*4	4-000000	000020	8	(4)
KH1	I*2	7-000000	000642	209	(209)
KH2	I*2	7-000642	000642	209	(209)
KH3	I*2	7-001504	000642	209	(209)
KH4	I*2	7-002346	000642	209	(209)
R	R*4	4-000020	000100	32	(4,4)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
100	**	101	**						

TOTAL SPACE ALLOCATED = 007100 1824

KAUTH,LP:=KAUTH

3.4.9 MEAN VALUE COMPUTATION (MEAN(IX,FLAG,N,NN,SBAR,STDEV))

3.4.9.1 Linkage

No other subroutines are called.

3.4.9.2 Interface

All values are passed through the calling arguments.

3.4.9.3 Input

None

3.4.9.4 Output

None

3.4.9.5 Storage

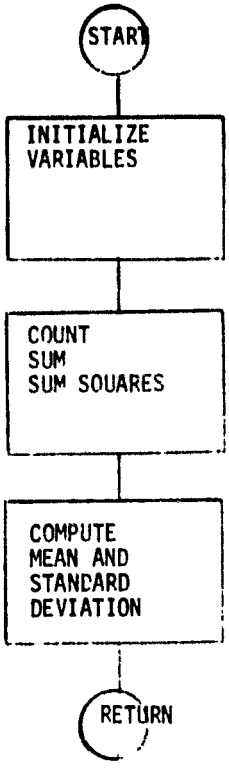
566 Bytes

3.4.9.6 Description

Mean computes the mean and standard deviation of the first N values of IX for which the value of FLAG is 1. NN is the number of values with flag = 1. XBAR is the mean. STDEV is the standard deviation.

MEAN

3.4.9.7 Flowchart



```
C       MEAN COMPUTES MEAN AND STANDARD DEVIATION FOR X(I)
                SUBROUTINE MEAN(IX,FLAG,N,NN,XBAR,STDEV)  !MEAN FOR FLAG(I)=1
                BYTE FLAG(209)
                INTEGER*2  IX(209),N,NN
                REAL  S,SS
                REAL  X(209),XBAR,STDEV,VARI,XNN

D               TYPE 808
      808       FORMAT('  MEAN')
                 S=0.
                 SS=0.
                 NN=0
      444       DO 10 I=1,N
                X(I)=FLOAT(IX(I))
       10       CONTINUE
C             ENTRY RMEAN(X,FLAG,N,NN,XBAR,STDEV)  !FOR REAL ARRAYS
                DO 1 I=1,N
                IF(FLAG(I).NE.1) GØ TØ 1
                NN=NN+1
                S=S+X(I)
                SS=SS+X(I)**2
        1       CONTINUE
              IF(NN.EQ.0)RETURN
                XBAR=S/NN
                VARI=(SS-XBAR*S)/(NN-1)
                STDEV=SQRT(ABS(VARI))
D               TYPE 809,NN,XBAR,STDEV
      809       FORMAT('    ',I4,F8.2,F8.2)
                RETURN
                END
```


Figure 3.4.9.8

Listing for MEAN

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	000430	136	RW,I,CON,LCL
3	$IDATA	000024	10	RW,D,CON,LCL
4	$VARS	001526	427	RW,D,CON,LCL
5	$TEMPS	000002	1	RW,D,CON,LCL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
MEAN		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-001524	J	I*2	F-000006*	NN	I*2	F-000010*	S	R*4	4-000000	S2	R*4	4-000004
STDEV	R*4	F-000014*	VARI	R*4	4-001516	XBAR	R*4	F-000012*	XNN	R*4	4-001520			

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
FLAG	L*1	F-000034*	000321	104	(209)
IX	I*2	F-000002*	000642	209	(209)
X	R*4	4-000010	001504	416	(209)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000260	10	**	444	**	808'	**	809'	**

FUNCTIONS AND SUBRØUTINES REFERENCED

$SORT

TØTAL SPACE ALLØCATED = 002154 566

```
12 MEAN,LP1=MEAN
11
10
 9
 8
 7
 6
 5
 4
 3
```

3.4.10 SPECTRAL PLOT (PLOT(IX,IY,CHAR,N))

3.4.10.1 Linkage

PLOT does not call any other subroutine.

3.4.10.2 Interface

All data is passed through the calling arguments.

3.4.10.3 Input

None

3.4.10.4 Output

A spectral plot is written to unit 8 and printed, (figure 3.4.10.4).

3.4.10.5 Storage

3215 Bytes

3.4.10.6 Description

The first N characters in the BYTE array CHAR are plotted at the points (IX,IY) for $20 \leq 120$, $-5 \leq IY \leq 50$.

Out of range points are truncated and plotted.

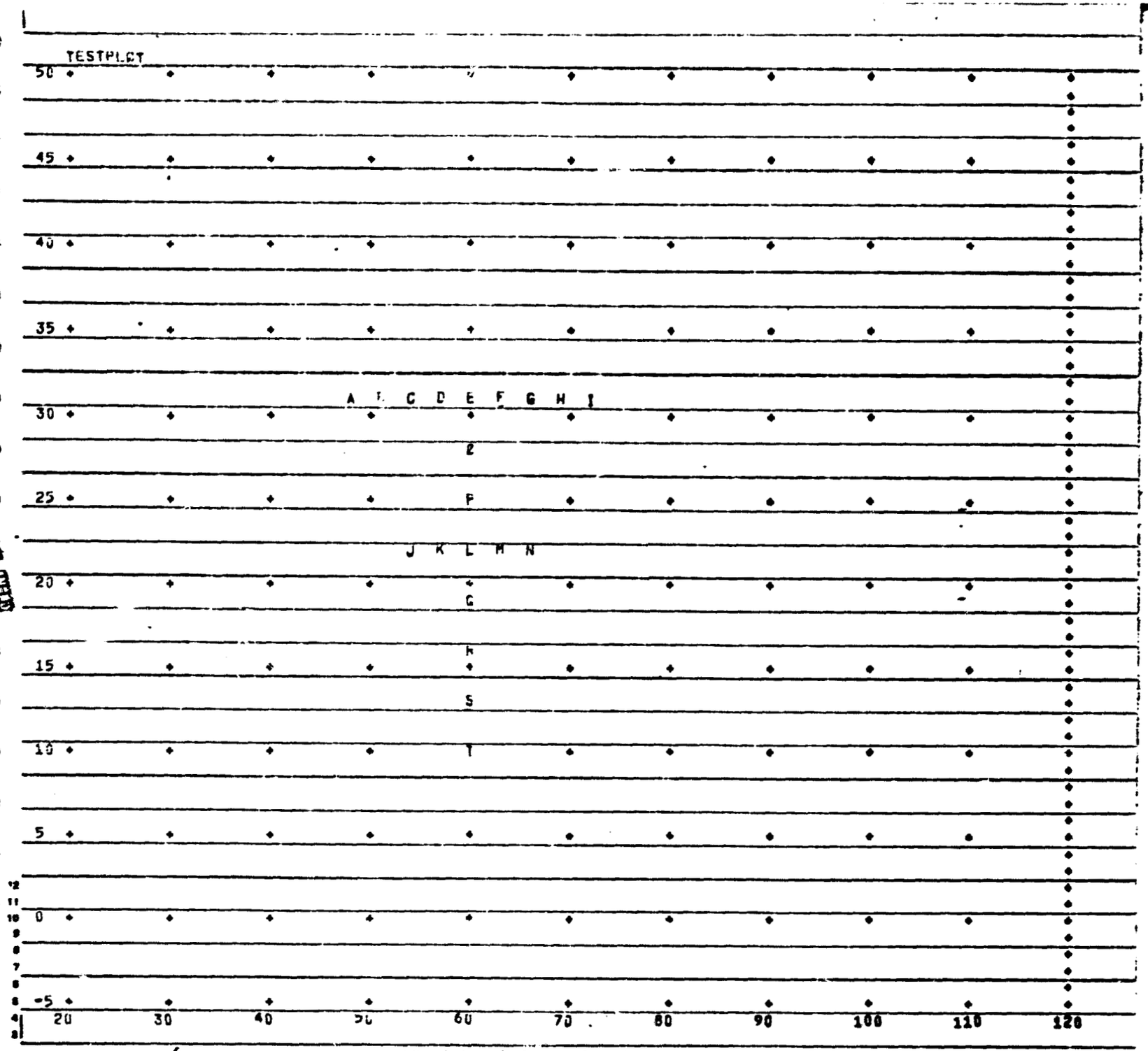
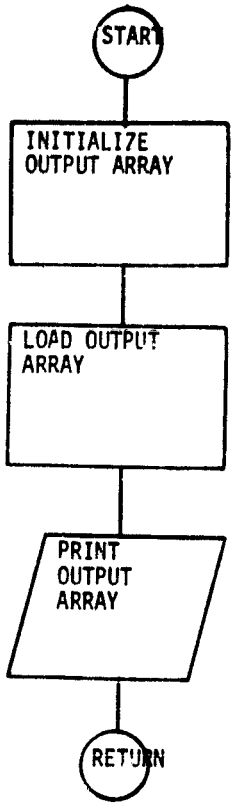


 Figure 3,4,10.4

PLOT

3.4.10.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15123:45    26-SEP-79          PAGE 1
PLOT.FTN        /TR:BLOCKS/WR

0001              SUBROUTINE PLOT(IX,IY,CHAR,N)
0002                    IMPLICIT INTEGER*2 (A-Z)
0003            INTEGER*4 SCALE(26)
0004             BYTE MAP(101,56),CHAR(1)
0005              INTEGER*2 IX(1),IY(1)
0006            DATA SCALE/'-5','0','5','10','15','20','25','30','35','40','45',
               1'50','55','60',' ','70',' ','80',' ','90',' ','100',
               2' ','110',' ','120'/

0007              TYPE 808
0008       808    FORMAT(' PLOT')
0009            DO 9  X=1,101                    !SET
0010            DO 90 Y=1,56
0011         90 MAP(X,Y)=' '
0012            DO 100 Y=1,101,10
0013            DO 100 Y=1,56,5
0014        100 MAP(X,Y)='+'
0015            DO 1 5 Y=1,56
0016        105 MAP(101,Y)='+'

0017            DO 200 I=1,N                     ! PLOT
0018               X=IX(I)-19
0019               Y=57-(IY(I)+6)
0020            IF(X.GT.101)X=101
0021            IF(X.LT.1)X=1
0022            IF(Y.GT.56)Y=56
0023            IF(Y.LT.1)Y=1
0024            IF(CHAR(I) .EQ. 0) CHAR(I)='0'
0025        200 MAP(X,Y)=CHAR(I)

0026            DO 250 J=1,56                    ! OUTPUT
0027            K=21
0028            M=J+4
0029            IF(MOD(M,5),EQ,0)K=13-M/5
0030        250 WRITE(8,903)SCALE(K),(MAP(L,J),L=1,101)
0031            WRITE(8,904)(SCALE(J),J=6,26,2)

0032              CLOSE(UNIT=8,DISP='PRINT')
0033        999 RETURN
0034        902       FORMAT('1')
0035        903 FORMAT(1H ,A3,101A1)
0036        904 FORMAT(3X,11(A3,7X)//)
0037                    END
```


Figure 3.4.10.8

Listing for PLOT

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001106	291	RW,I,CON,LCL
3	$IDATA	00 0110	36	RW,I,CON,LCL
4	$VARS	013216	2887	RW,I,CON,LCL
5	$TEMPS	00 0002	1	RW,I,CON,LCL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
PLOT		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-013204	J	I*2	4-013206	K	I*2	4-013210	L	I*2	4-013214	M	I*2	4-013212
N	I*2	F-00 010*	X	I*2	4-013200	Y	I*2	4-013202						

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
CHAR	L*1	F-000036*	000001	0	(1)
IX	I*2	F-00 002*	000002	1	(1)
IY	I*2	F-000004*	000002	1	(1)
MAP	L*1	4-000150	013033	2628	(101,50)
SCALE	I*4	4-000000	000150	52	(26)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
90	**	100	**	105	**	200	**	250	**
808'	3-000000	902'	**	903'	3-000012	904'	3-000022	999	**

FUNCTIONS AND SUBROUTINES REFERENCED

CLOSE

TOTAL SPACE ALLOCATED = 014436 3215

NO FPP INSTRUCTIONS GENERATED

PLOT,LP:=PLOT

3.4.11 STORE NEW DATA (PUT(SEG,ARRAY,NAME))

3.4.11.1 Linkage

GETNAM may be called.

3.4.11.2 Interface

The interface with the calling program is through the calling arguments and the common block /ANCIL/.

3.4.11.3 Input

None

3.4.11.4 Output

None

3.4.11.5 Storage

2022 Bytes

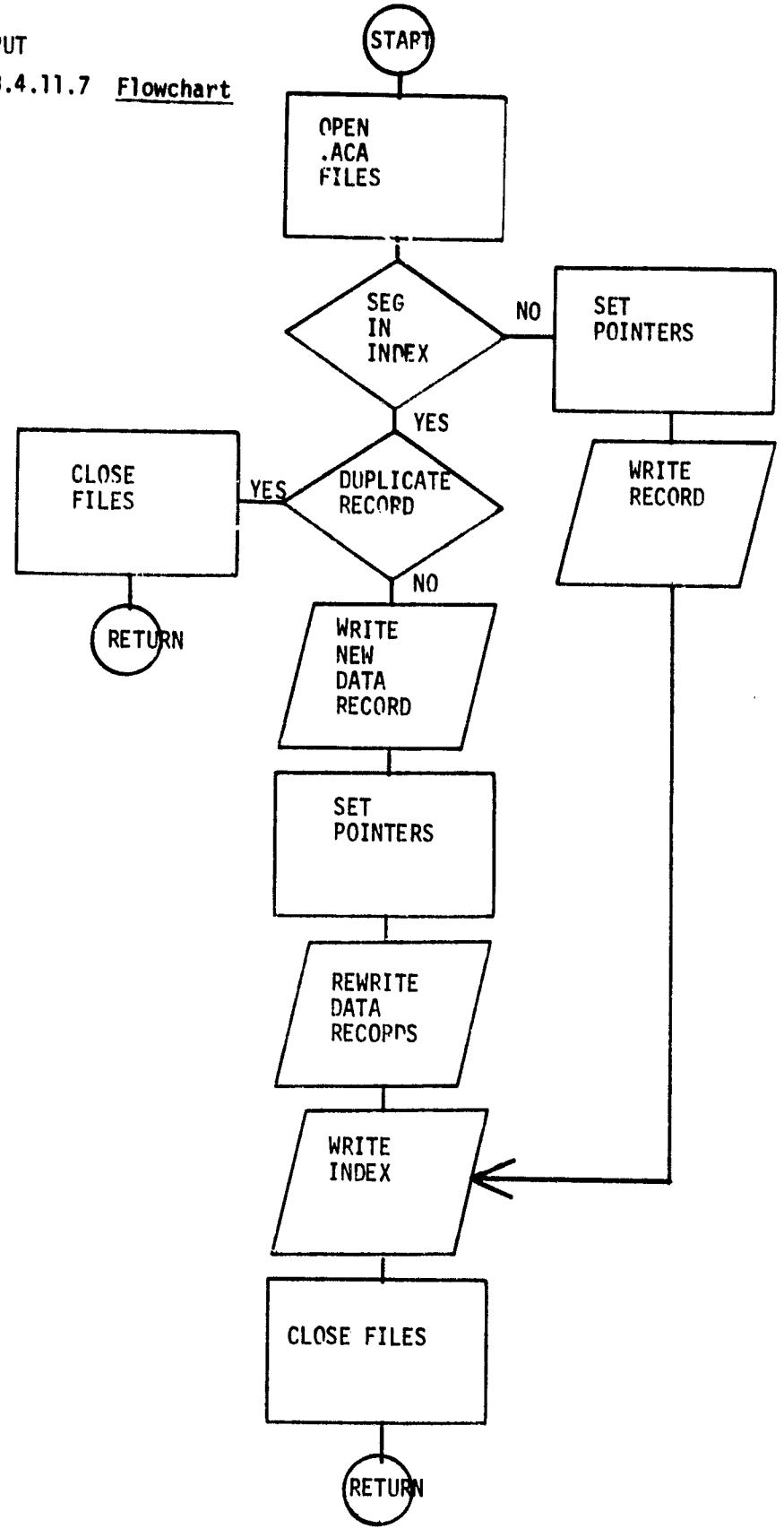
3.4.11.6 Description

Put opens the ACA files, then looks for a previous entry for segment SEG. If no such entry has been made, a new entry is made. If segment SEG does occur, PUT checks that the current data is not duplicated, and if it is not, the current data is added to the data base.

ARRAY is the dot data. NAME is the identifying code name. DATE and FILL are passed through the common block ANCIL.

PUT

3.4.11.7 Flowchart



```
      SUBROUTINE PUT(SEG,ARRAY,NAME)
      IMPLICIT INTEGER*2 ( -Z)
      BYTE FDN(32),FIN(32), CHAR(2)
      DIMENSION INDEX(256),BUFFER(256),NNNNNN(256),MINDEX(256)
      DIMENSION BUF(256)
      DIMENSION NAME(3),ARRAY(209)
      PARAMETER T=.TRUE.,F=.FALSE.
      EQUIVALENCE (MINDEX(1),LINDEX),(MINDEX(2),MAXREC)
C
      EQUIVALENCE( CHAR, BUF(7) )
C
      COMMON /ANCIL/CODE,DATE(2),STATUS,FILL(36)
      COMMON /FILNAM/ FDN,FIN
      DATA FILL/36*0/
      DATA ../'  '/
C
   10 CONTINUE
      FIN(32)=0
D        TYPE 301,FIN,FDN
      OPEN (UNIT=7,NAME=FIN,TYPE='OLD',ACCESS='DIRECT',ERR=300)
      READ (7'1) MINDEX
      FDN(32)=0
      OPEN (UNIT=8,NAME=FDN,TYPE='OLD',ACCESS='DIRECT',ERR=300)
C     SEARCH THE INDEX FOR THE SEGMENT NUMBER
      DO 20, K=3,LINDEX-1,2
          READ (7'K) INDEX
          DO 20, J=1,255,2
              IF (INDEX(J).EQ.0) GO TO 40
              IF (INDEX(J).EQ.SEG) GO TO 50
   20 .   CONTINUE
C     ADD A PAGE TO THE INDEX
      K=LINDEX+1
      LINDEX=K+1
      DO 30, I=1,256
   30     INDEX(I)=0
      J=1
C     WRITE THE FIRST RECORD FOR A SEGMENT NUMBER
   40 CONTINUE
      MAXREC=MAXREC+1
      INDEX(J)=SEG
      INDEX(J+1)=MAXREC
      WRITE (8'MAXREC)
     *SEG,DATE,6,NAME,CODE,ARRAY,FILL,F,MAXREC,MAXREC
      GO TO 60
C     WRITE ANOTHER RECORD FOR THE SEGMENT NUMBER
   50 NEXT=INDEX(J+1)
      READ (8'NEXT) BUFFER
      LAST=BUFFER(255)
      DO 55 M=1,256
   55 BUF(M) = BUFFER(M)
C
      IFLAG = 0
C
      ICNT = 1
   60 CONTINUE
      IF(NAME(3) .NE. BUF(7) ) GO TO 100
```

Figure 3.4.11.8

Lisiting for PUT

```
0046          IF(NAME(2) .NE. BUF(3) ) GO TO 100
0047          IF(NAME(1) .NE.BUF(5) ) GO TO 100
0048          IF(CHAR(2) .NE. '*'       )GO TO 200
0049          IF(DATE(2) .NE. BUF(3) ) GO TO 100
0050          IF(DATE(1) .EQ. BUF(4) ) GO TO 200
0051          IF(BUF(4) .EQ. '**' ) GO TO 200
0052      100 CONTINUE
          C
0053          IF(BUF(256) .EQ. NEXT ) GO TO 120
0054          ICOUT=ICOUT + 1
0055          RECNO = BUF(256)
0056          IF(IFLAG .EQ. 1) GO TO 120
0057          IF(BUF(256) .EQ. BUF(255) ) IFLAG=1
0058          READ'R'RECNO) BUF
0059          GO TO 60
0060      120 CONTINUE
          C
0061          MAXREC=MAXREC+1
0062          WRIT  (3'MAXREC)
             *SEG,DATE,B,NAME,CODE,ARRAY,FILL,F,LAST,NEXT
0063          READ (3'LAST) NUNNN
0064          NUNNN(256)=MAXREC
0065          IF(LAST .EQ. NEXT) NUNNN(255) = MAXREC
0066          WRIT  (3'LAST) NUNNN
0067          BUFFER(255)=MAXREC
0068          IF(LAST .NE. NEXT) WRITE(3'NEXT) BUFFER
0069          INDEX(J+1)=MAXREC
0070       60 WRITE (7'1) INDEX
0071          WRITE (7'2) INDEX
0072          WRIT  (7'K) INDEX
0073          WRIT  (7'K+1) INDEX
0074      200 CONTINUE
0075          CLOS  (UNIT=7)
0076          CLOSE (UNIT=8)
0077          STATUS=0
0078          RETURN
0079      300     TYPE 301,FIN,FDN
0080      301     FORMAT(//,' FILE IS  ',32A1,
                 *          //,' FDN IS   ',32A1)
0081          CLOSE(UNIT=7)
0082          CLOSE(UNIT=8)
0083          CALL GETNAM(FDN,FIN)
0084          GO TO 10
0085          END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	002144	562	RW,I,GBL,LCL
2	$PDATA	00 002	1	RW,D,GBL,LCL
3	$IDATA	000300	96	RW,I,GBL,LCL
4	$VARS	00 024	1200	RW,I,GBL,LCL
5	$TEMPS	000002	1	RW,I,GBL,LCL
6	A$CII	00 120	40	RW,D,OVR,GBL
7	FILNAM	00'100	32	RW,I,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
PUT		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
B	I*2	4-005000	CODE	I*2	6-000000	I	I*2	4-005006	ICNT	I*2	4-005020	IFLAG	I*2	4-005016
J	I*2	4-00 004	K	I*2	4-005002	LAST	I*2	4-005012	LINDEX	I*2	4-001000	M	I*	4-005014
MAXREC	I*2	4-001002	NEXT	I*2	4-005010	RECNO	I*2	4-005022	SEG	I*2	F-000002*	STATUS	I*2	0-000006

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
ARRAY	I*2	F-000004*	000642	209	(209)
BUF	I*2	4-001000	001000	256	(256)
BUFFER	I*2	4-002000	001000	256	(256)
CHAR	L*1	4-000014	000002	1	(2)
DATE	I*2	6-001002	000004	2	(2)
FDN	L*1	7-000000	000040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	7-000040	000040	16	(32)
INDEX	I*2	4-00 000	001000	256	(256)
MINDEX	I*2	4-001000	001000	256	(256)
NAME	I*2	F-000006*	000006	3	(3)
NNNNNN	I*2	4-004000	001000	256	(256)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
10	1-000060	20	**	30	**	40	1-000416	50	1-000024
55	**	60	1-001610	80	1-000764	100	1-001102	140	1-001220
200	1-001776	300	1-002036	301'	3-000000				

FUNCTIONS AND SUBROUTINES REFERENCED

CLOSS GETNAM ZPERR$

TOTAL SPACE ALLOCATED = 007714 2022

NO FPP INSTRUCTIONS GENERATED

PUT,LP:=PUT

3.4.12 SCREEN ALL DOTS (SCRALL(N))

3.4.12.1 Linkage

SCRALL does not call any other subroutine.

3.4.12.2 Interface

SCRALL interfaces with the calling routine through the calling argument and the common blocks /ANCIL/, /DATA/, /KAUTH/, /FLAG/.

3.4.12.3 Input

None

3.4.12.4 Output

None

3.4.12.5 Storage

2579 Bytes

3.4.12.6 Description

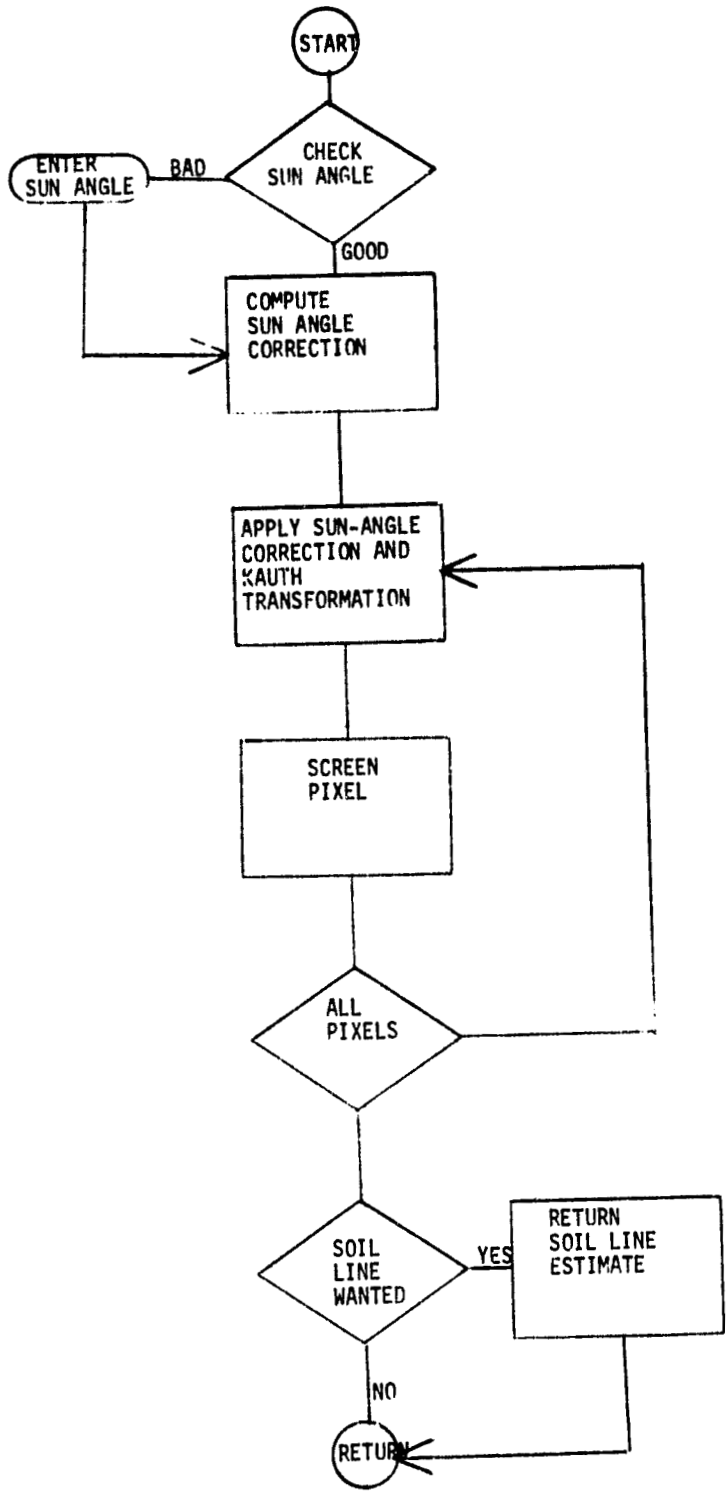
SCRALL is an implementation of the SCREEN algorithm developed for LACIE by ERIM.

The data are transformed to KAUTH coordinates and stored in /KAUTH/. The sun angle correction factor is applied and the data are screened. The screening result for each pixel is stored in the BYTE array FLAG. A soil line estimate is made, and if SLN = 99 it is returned to the calling program.

SCRALL must be compiled with the CO:50 switch.

SCRALL

3.4.12.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15124150    20-SEP-79          PAGE 1
SCRALL.FTN        /TI:BLOCKS/WR

              C  SUBROUTINE SCRALL KAUTH TRANSFORMS AND SCREENS CH DATA
              C
              C    COMPILE WITH THE CK150 SWITCH,
              C
0001                SUBROUTINE SCRALL(N)
0002                REAL R(4,4),ZZ,C,Z(4,19),RR(4,4),Y(4),CC(19)
0003                INTEGER II(19),JJ(19),LL(21),X(4),T
0004                INTEGER*2  SSN
0005                  INCLUDE 'INCLU.FTN'
           *  C
           *  C
           *  C
0006 *                INTEGER*2 CH1(209),CH2(209),CH3(209),CH4(209)
0007 *                INTEGER*2 KH1(209),KH2(209),KH3(209),KH4(209)
0008 *                BYTE  FLAG(210),FAN(32),FON(32),FIN(32)
0009 *                INTEGER*2 SEC,DATE(2),CODE,STATUS,FILL(36)
0010 *                INTEGER*2 IAG,SLN,SAT,GAM,LAT(2),LNG(2)
0011 *                    COMMON /ANCIL/ CODE,DATE,STATUS,FILL
0012 *                    COMMON /DATA/  CH1,CH2,CH3,CH4
0013 *                    COMMON /KAUTH/ KH1,KH2,KH3,KH4
0014 *                    COMMON /FLAG/ FLAG
0015 *                    COMMON /FILNAM/ FON,FIN,FAN,KAN
0016 *                    EQUIVALENCE (IAG,FILL(2)),(SLN,FILL(3))
0017 *                    EQUIVALENCE (SAT,FILL(4)),(GAM,FILL(10))
0018 *                    EQUIVALENCE (LAT(1),FILL(5)),(LNG(1),FILL(7))
           *  C
           *  C    END INCLU.FTN
           *  C
0019                DATA RR/.33231,.60310,.67581,.26279,
                  C        -.28317,-.06006,.57735,.30633,
                  C        -.89952,.42830,.07592,-.04080,
                  C        -.01594,.1306 ,-.45187,.68232/
0020                DATA Z/0,0,0,0,0,0,1,0,
                  C     0,0,0,0,0,0,-1,0'
                  C     -.03375,0,0,1,0,0,0,
                  C     -.1875,0,0,-1,0,0,0,
                  C     -0.1,-1,0,0,0,0,0,
                  C    0.555556,1,0,0,0,0,0,
                  C    -0.33333333,1,0,0,0,0,0,
                  C     1,0,0,0,0,0,0,0,
                  C     -.1,0,0,-1,0,0,0,
                  C     1,0,0,0,0,0,0,0,
                  C     -0.142857,0,0,-1,0,0,0,
                  C     -1,0,0,0,0,0,0,0,
                  C       -.5675,-1,0,0,0,0,0,
                  C        0,0,0,0,0,0,-1,0,
                  C       0,0,-1,0,0,0,-1,0,
                  C      -.5,-1,0,-1,0,-0.5,
                  C       -.4,1,0,0,0,0,0,
                  C     -0,4,1,0,-0,6,-0.5,
                  C       -1,0,0,4,0,0,0,0/
0021                DATA  CC/16.,12.,-4.,14.,20.,156.,-8.,100.,7.5,69.,3.25,
                  C -75.,0.5,-1.5,4.5,-10.,-12.2,-9.,-37.75/
0022                DATA  II/2,3,4,5,6,7,8,10,10,12,12,18,18,18,18,18,60,20,20/
0023                DATA  JJ/60,60,60,60,60,60,60,9,60,11,60,13,14,15,16,17,21,
                  C   19,60/
```


Figure 3.4.12.8

Listing for SCRALL

```
0024              DATA LL/47,46,45,44,43,42,41,34,33,32,31,26,25,24,23,22,21,
             *         13,12,1,11/
          C          4* GARBLED: 34,33 CLOUD: 32,31 HAZE
          C          2* WATER: 1* SHADOW, 11 OVER WATER: 1 GOOD.
          C
          D            TYPE 808
0025      808       FORMAT('  SCRALL')
0026                  SSN=30
          D          WRITE(8,993) CH1,CH2,CH3,CH4
0027      993        FORMAT('  ',1914)
0028      600      IF(IAG.LT.15.OR.IAG.GT.60)TYPE 101,SEG,DATE
0029      101      FORMAT('     INPUT SUNANGLE FOR',I6,2I3,'   SCRALL')
0030               IF(IAG.LT.15.OR.IAG.GT.60) ACCEPT 201 ,IAG
0031      201      FORMAT(I2)
0032                 ZZ=IAG/57.3
0033                 IF(ZZ.LT.0.1) ZZ=0.907
0034                  C=0.77713/SIN(ZZ)
          D           TYPE 990,IAG,C
          D           WRITE(5,990) IAG,C
0035                IF (C.LT.0.1) TYPE 990,IAG,C
0036                IF (C.LT.0.1) RETURN
0037      990         FORMAT('  C  ',I4,F10.4)
0038                   DO 100 K=1,N
0039                DO 5 I=1,4
0040      5       Y(I)=0
0041                X(1)=CH1(K)
0042                X(2)=CH2(K)
0043                X(3)=CH3(K)
0044                X(4)=CH4(K)
          D           IF(K.EQ.1) TYPE  995,X
0045      995         FORMAT('    X ',4I5)
0046              DO 6 I=1,4
0047              DO 6 J=1,4
0048      6       Y(I)=Y(I)+X(J)*RR(J,I)
0049                KH1(K)=Y(1)+.5
0050                KH2(K)=Y(2)+.5
0051               IF(KH2(K).LT.SSN) SSN=KH2(K)     ! SOIL LINE ESTIMATE
0052               KH3(K)=Y(3)+.5
0053               KH4(K)=Y(4)+.5
          D        IF(K.EQ.1)TYPE 995,KH1(K),KH2(K),KH3(K),KH4(K)
0054      98         DO 66 I=1,4
0055      66        Y(I)=Y(I)*C              ! SUN ANGLE CORRECTION
0056              L=1
0057      1       CONTINUE
          D          IF(K.LT.3) TYPE 996,L,T,U,CC(L)
          D          IF(K.LT.20) WRITE(8,996) L,T,L,CC(L),Y
0058      996        FORMAT('  ',I4,I3,5F10.4,F10.4)
0059              IF(L.LE.21)T=LL(L)
0060              IF(L .GE. 20) GO TO 99
0061              U=0.
0062              DO 7 I=1,4
0063      7       U=U+Y(I)*Z(I,L)
0064              IF(U .LT. CC(L)) GO TO 3
0065              L=JJ(L)
0066              GO T  1
0067      3       CONTINUE
```

```
0064            L=II(L)
0069            GO T  1
0070      99    FLAG (I)=T
0071     100    CONTINUE
0072            IF(SL..,EQ.99)SL..=SSN
          U       WRITE(8,993)KH1,KH2,KH3,LH4
          D       TYPE 991,FLAG
0073     991        FORMAT(' FLAG',19I3)
0074            RETURN
0075            END
```

PRØGRAM SECTIØNS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001230	332	RW,I,LCØN,LCL
2	$PDATA	000020	8	RW,D,LCØN,LCL
3	$IDATA	000134	34	RW,D,LCØN,LCL
4	$VARS	001246	338	RW,D,LCØN,LCL
5	$TEMPS	000002	1	RW,D,LCØN,LCL
6	ANCIL	000120	40	RW,D,REVE,GBL
7	DATA	003210	836	RW,D,REVE,GBL
8	HALTH	003210	836	RW,D,REVE,GBL
9	FLAG	000522	105	RW,D,REVE,GBL
10	FILNAM	000142	49	RW,D,REVE,GBL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
SCRALL		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
C	R*4	4-000104	CODE	I*2	6-000000	GAM	I*2	6-000032	I	I*2	4-001232	IAG	I*2	6-000012
J	I*2	4-001234	K	I*	4-001230	KAN	I*2	10-000140	L	I*2	4-001236	N	I*2	6-000020
SAT	I*2	6-000016	SEG	I*2	4-001226	SLN	I*2	6-000D14	SSN	I*2	4-001224	STATUS	I*2	6-000006
T	I*2	4-001222	U	R*.	4-001240	ZZ	R*4	4-000100						

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIØNS
CC	R*4	4-000710	000114	38	(19)
CH1	I*2	7-000000	000642	209	(205)
CH2	I*2	7-000642	000642	209	(205)
CH3	I*2	7-001504	000642	209	(205)
CH4	I*2	7-002346	000642	209	(205)
DATE	I*2	6-000002	000004	2	(2)
FAN	L*1	10-000100	000040	16	(32)
FDN	L*1	10-000060	000040	16	(32)
FILL	I*2	6-000010	000110	36	(36)
FIN	L*1	10-000040	000040	16	(32)
FLAG	L*1	9-000001	000522	105	(210)
II	I*2	4-001024	000046	19	(19)
JJ	I*2	4-001072	000046	19	(19)
KH1	I*2	8-000000	000642	209	(205)
KH2	I*2	8-000642	000642	209	(205)
KH3	I*2	8-001504	000642	209	(205)
KH4	I*2	8-002346	000642	209	(205)
LAT	I*2	6-000020	000004	2	(2)
LL	I*2	4-001140	000052	21	(21)

FORTRAN IV-PLUS V02-51 15124:50 26-SEP-79 PAGE 5
SCRALL.FTN /TR:0L0CKS/WI

LNG	I*2	6-000024	000004	2	(2)
R	R*4	4-00 100	000170	32	(4,4)
RR	R*4	4-00 570	000190	32	(4,4)
X	I*2	4-001212	000010	4	(4)
Y	R*4	4-000670	000020	8	(4)
Z	R*4	4-000110	000460	152	(4,19)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000726	3	1-001124	5	**	6	**	7	**
66	**	98	**	99	1-001150	100	**	101'	3-000000
201'	3-000052	600	**	808'	**	990'	3-000056	991'	**
993'	**	995'	**	996'	**				

FUNCTIONS AND SUBROUTINES REFERENCED

$SIN

TOTAL SPACE ALLOCATED = 012046 2579

SCRALL,LP:=SCRALL/C0150

3.4.13 DATA SORT (SORT(X,Y,P,N))

3.4.13.1 Linkage

Sort does not call any other routine.

3.4.13.2 Interface

All data is passed through the calling arguments.

3.4.13.3 Input

None

3.4.13.4 Output

None

3.4.13.5 Storage

159 Bytes

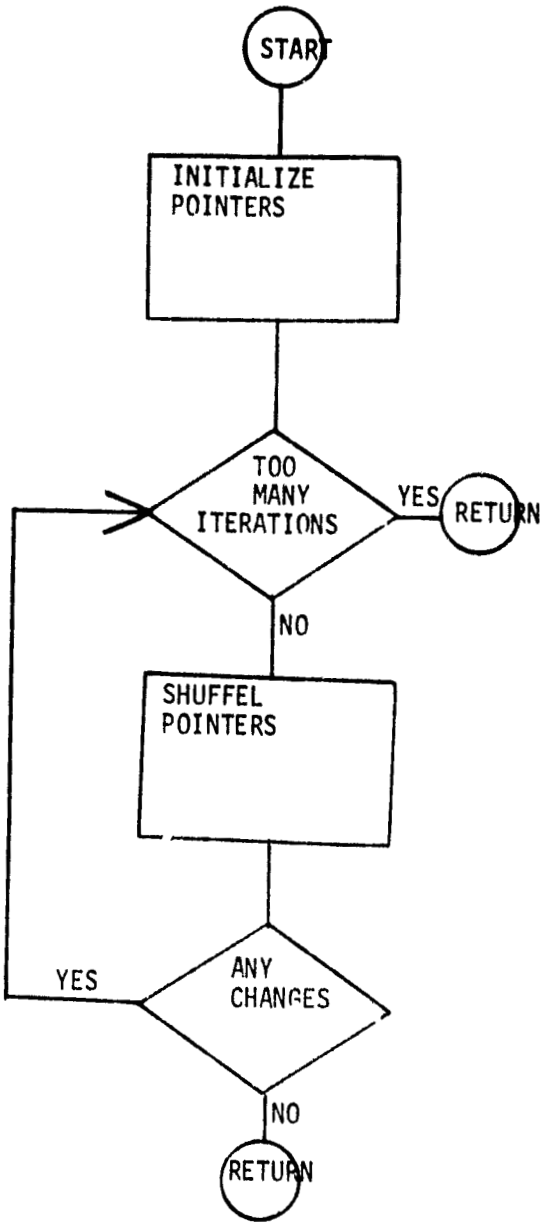
3.4.13.6 Descriptio

Sort defines the pointer array P so that $X(P(1)) \leq X(P(2)) \leq \cdots \leq X(P(N))$.

In case $X(P(K)) = X(P(K+1))$ then $Y(P(K)) \leq (P(K+1))$.

SORT

3.4.13.7 Flowchart



```
FORTRAN IV-PLUS V02-51          15128126    20-SEP-79          PAGE 1
SORT.FTN        /TS:BLOCKS/WR

        C       SORT.FTN    SORTS OUT FILES
        C       X(I)  ARRAY TO BE SORTED
        C       Y(I)  SECONDARY SORT VARIABLE
        C       P(J)  POINTER ARRAY
        C                X(P(1)) IS SMALLEST
        C       N     NUMBER OF ELEMENTS
        C
0001                SUBROUTINE SORT(X,Y,P,N)
0002              INTEGER*2  X(N),Y(N),P(N)
0003              DO 1 I=1,N
0004     1        P(I)=I
0005              K=0
0006     2        CONTINUE
0007              K=K+1
0008              KF=0
0009              IF(K.GT.N)RETURN
0010              DO 3 I=2,N
0011              IF(X(P(I-1)).LT.X(P(I))) GO TO 3
0012              IF(X(P(I-1)).GT.X(P(I))) GO TO 4
0013              IF(Y(P(I-1)).LE.Y(P(I))) GO TO 3
0014     4        KF=1
0015              KN=P(I-1)
0016              P(I-1)=P(I)
0017              P(I)=KN
0018     3        CONTINUE
0019              IF(KF.EQ.1) GO TO 2
0020              RETURN
0021              END
```


Figure 3.4.13.8
Listing for SORT

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	000404	130	RW,I,GBL,LCL
3	$IDATA	00 060	24	RW,D,GBL,LCL
4	$VARS	00-010	4	RW,D,GBL,LCL
5	$TEMPS	000002	1	RW,D,GBL,LCL

ENTRY PØINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
SØRT		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-000000	K	I*2	4-000002	KF	I*2	4-000004	N	I*2	F-000010*	NN	I*2	3-000006

ARRAYS

NAME	TYPE	ADDRESS	SIZE	DIMENSIØNS
P	I*2	F-000006*	**	(*)
X	I*2	F-00 002*	**	(*)
Y	I*2	F-00 004*	**	(*)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	**	2	1-000150	3	1-000350	4	1-000304		

TØTAL SPACE ALLØCATED = 000476 159

NØ FPP INSTRUCTIØNS GENERATED

SØRT,LP1=SØRT

3.4.14 TRAJECTORY PLOT (TRPLT(IX,JY,NT,ND,NB))

3.4.14.1 Linkage

TRPLT calls no other subroutines.

3.4.14.2 Interface

The interface with the calling routine is through the calling arguments and the common block /TRAP/.

3.4.14.3 Input

None

3.4.14.4 Output

TRPLT creates line printer plots. (fig 6.4.14.4)

3.4.14.5 Storage

14,445 Bytes

3.4.14.6 Description

TRPLT creates plots of size IX by JY, packed several to the page with NB dots per plot, each dot plotted NT times for a total of ND (209) dots. The plot format is given by the array PLT.

Current (Aug 1979) practice is 2 dots per plot, up to 16 times per plot, 2 plots per page. The array PLT is read from [200,7] GRINT.DAT.
The points
(T(I),X(I,N)) and
(T(I),Y(I,N)) are plotted
with symbol CHAR(I,N). The number of N's per plot is NB. There may be several plots per page.

Changing the array PLT will radically change the output.

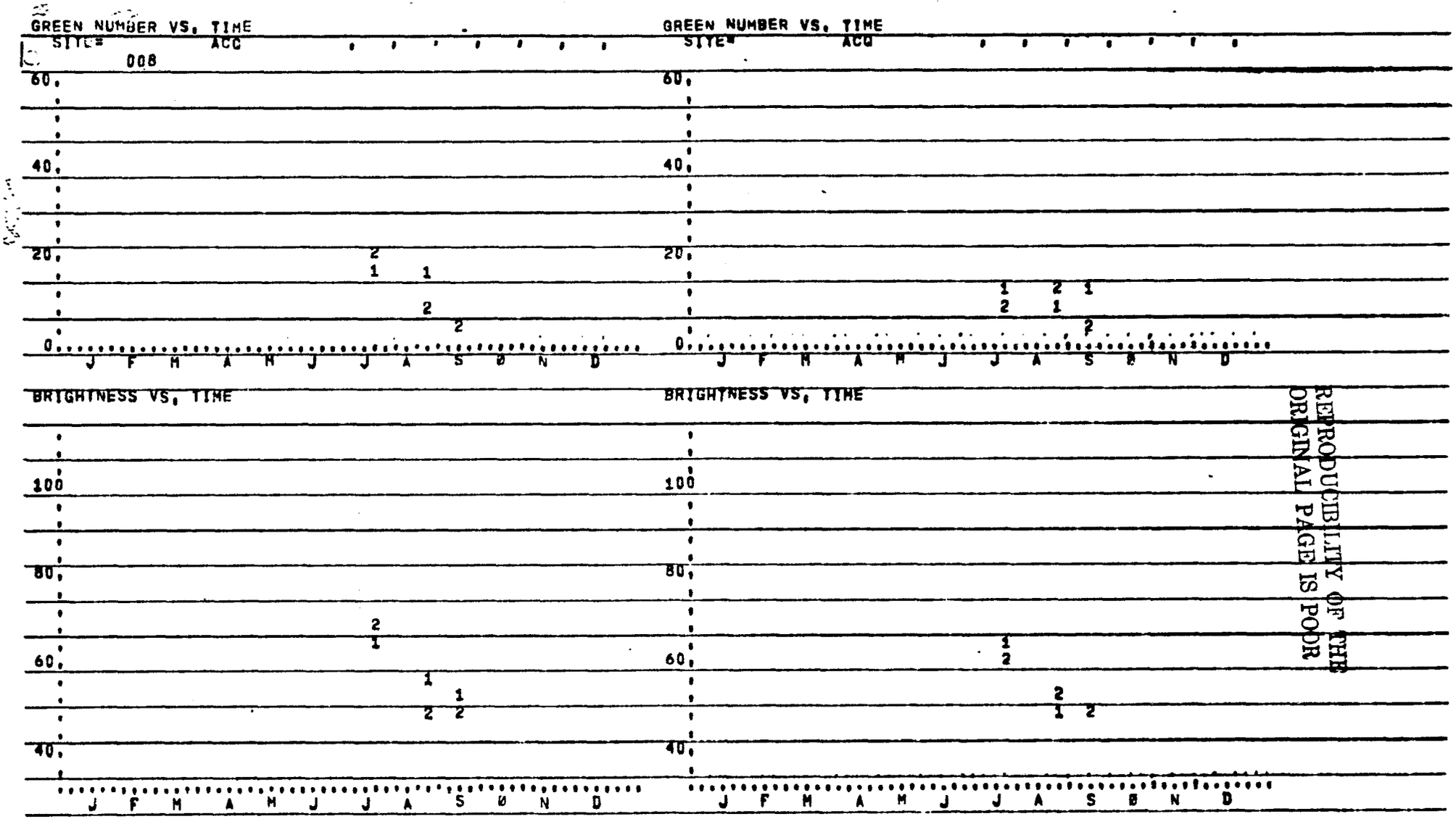
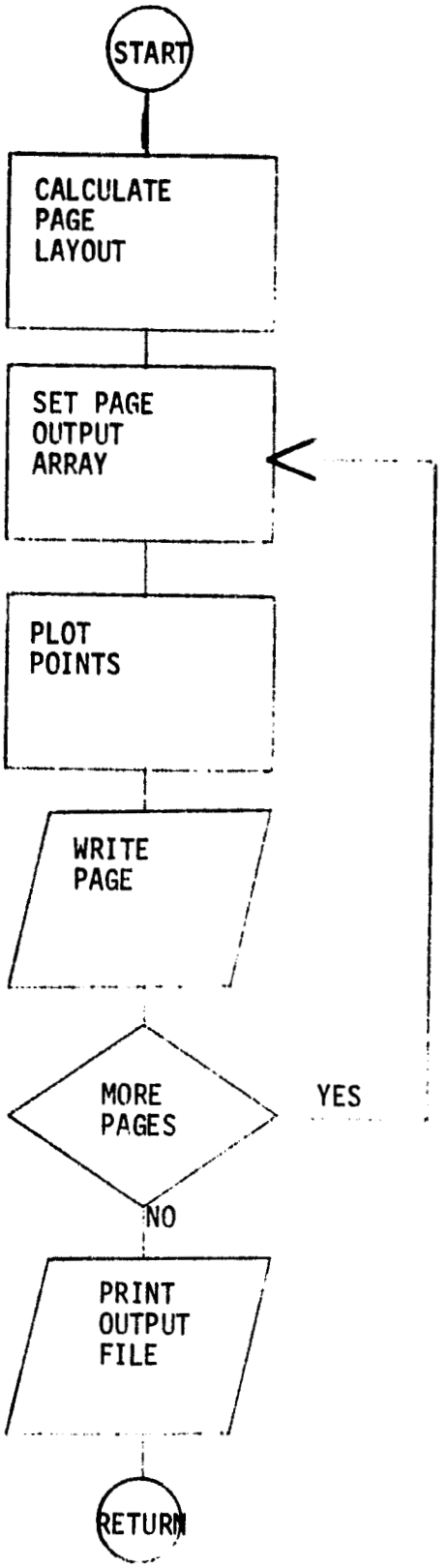


Figure 3.4.14.4

TRPLT

3.4.14.7 Flowchart



```
FØRTRAN IV-PLUS V02-51          15:24:12    20-SEP-79          PAGE 1
TRPLT.FTN       /T :8LØCKS/WR

        C     TRPLT.FTN IS A PLØTTING RØUTINE BEST FØR TRAJECTØRIES
        C
0001                SUBRØUTINE TRPLT(IX,JY,NT,NE,NB)
        C
        C     MULTIPLE PLØTS ØF SIZE IX BY JY
        C     NT    UMBER ØF PØINTS PER PLØT
        C     ND    UMBER ØF DATS (PLØTS)
        C
0002                IMPLICIT INTEGER*2 (A-Z)
0003                BYTE PAGE(121,67),PLT(60,45)
0004                BYTE CHAR(16,209)
0005                INTEGER*2 X(16,209),Y(16,209),T(16)
0006                INTEGER*2 LINE(34),LDØT(2,209)
        C
0007                CØMMØN /TWAP/ X,Y,T,PLT,CHAR,LINE,LDØT
        D            TYPE 1
0008     1          FØRMAT(' TRPLT')
        C
0009            IN=120/IX     ! NUMBER ØF PLØTS ACRØSS
0010            JN= 66/JY     ! NUMBER ØF PLØTS ØF
0011            KN=IN*JN      ! NUMBER ØF PLØTS PER PAGE
0012            NP=ND/(KN*NB)+1     ! NUMBER ØF PAGES
0013            K =0          ! NUMBER ØF CURRENT DØT
        D        TYPE 95,IN,IX,JN,JY,KN,NP,NB,NT
0014     95     FØRMAT(' ',8I5)
        D        WRITE(6,96) PLT
0015     96     FØRMAT('1',60A1)
        D        WRITE(6,97) CHAR
0016     97     FØRMAT('1',194A1)
        C
0017            DØ 99 L2=1,56
0018              PAGE(1,L2)=' '
0019            DØ 99 L1=2,121
0020     99     PAGE(L1,L2)=' '
0021            PAGE(13,3)='0'
0022            PAGE(12,3)='0'
0023            PAGE(11,3)='0'
0024            PAGE(64,3)=' '
0025            PAGE(65,3)=' '
0026            PAGE(66,3)=' '
0027            PAGE(67,3)=' '
        C
0028            DØ 100  P=1,NP
0029             PAGE(1,1)='1'
0030               DØ 101 II=1,IN
0031               DØ 101 JJ=1,JN
0032               DØ 101 II=1,IX
12 0033            DØ 101 JJ=1,JY
11 0034            IF(JJ.EQ.3.AND.II.LT.14) GØ TØ 101
10 0035               I=IX*(IL-1)+II*1
 9 0036               J=JY*(JL-1)+JJ
 8 0037               PAGE(I,J)=PLT(II,JJ)
 7 0038               IF(PAGE(I,J).EQ.0)PAGE(I,J)=' '
 6 0039     101     CØNTINUE
 5 0040            DØ 102 MT=1,NT
 4 0041               IF(T(MT).LT.1)  T(MT)=1
```

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR

Figure 3.4.14.8
Listing for TRAJ

```
109         IF(T(KT).GT.IX) T(KT)=IX
        DØ 110 IL=1,IM
        DØ 110 JL=1,JY
        DØ 110 JB=1,NB
        K=K+1
         IF(K.GT.209) GØ TØ 120
        IF( PAGE(13,3).EQ.'9' .AND.
     *      PAGE(12,3).EQ.'9' ) PAGE(11,3)=PAGE(11,3)+1
        IF(PAGE(13,3).EQ.'9')PAGE(12,3)=PAGE(12,3)+1
        IF(PAGE(12,3).GT.'9')PAGE(12,3)='0'
        PAGE(13,3)=PAGE(13,3)+1
        IF(PAGE(13,3).GT.'9') PAGE(13,3)='0'
        DØ 110  KT=1,NT
        IF(X(KT,K).LT.1)  X(KT,K)=1
        IF(X(KT,K).GT.JY) X(KT,K)=JY
        IF(Y(KT,K).LT.1)  Y(KT,K)=1
        IF(Y(KT,K).GT.JY) Y(KT,K)=JY
        PX=X(KT,K)+(JL-1)*JY
        PY=Y(KT,K)+(JL-1)*JY
        PT=T(KT)   +(IL-1)*IX
        IF(CHAR(KT,K).EQ.'*')CHAR(KT,K)="60+JB
        IF(CHAR(KT,K).EQ.0) CHAR(KT,K)='B'
        PAGE(PT,PX)=CHAR(KT,K)
        PAGE(PT,PY)=CHAR(KT,K)
110     CONTINUE
120     CONTINUE
C       WRITE(6,301)
301     FØRMAT('1')
D       WRITE(6,299) LINE,LDØT(P,1),LDØT(P,2)
299     FØRMAT('  SEG ',2I4,' DATES ',3213,/'    DØT ',I3,' TØ DØT ',I3)
        WRITE(6,300) PAGE
300     FØRMAT(121A1)
100     CØNTINUE
           CLØSE(UNIT=6,DISP='PRINT')
           RETURN
        END
```

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	$CODE1	001670	476	RW,I,CON,LCL
3	$IDATA	000026	11	RW,I,CON,LCL
4	$VARS	013566	3771	RW,D,CON,LCL
5	$TEMPS	000022	9	RW,D,CON,LCL
6	TRAP	047604	10179	RW,D,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
TRPLT		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
I	I*2	4-016546	II	I*2	4-016542	IL	I*2	4-016536	IN	I*2	4-016516	IX	I*2	F-000002*
J	I*2	4-016550	JR	I*2	4-016554	JJ	I*2	4-016544	JL	I*2	4-016540	JN	I*	4-016520
JY	I*2	F-000004*	K	I*2	4-016526	KN	I*2	4-016522	KT	I*2	4-016556	L1	I*2	4-016532
L2	I*2	4-016552	MT	I*2	4-016558	NB	I*2	F-000012*	ND	I*2	F-000010*	NP	I*2	4-016524
NT	I*2	F-000006*	P	I*	4-016534	PT	I*2	4-016564	PX	I*2	4-016560	PY	I*	4-016562

ARRAYS

NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
CHAR	L*1	6-037354	006420	1672	(16,209)
LDPT	I*2	6-046100	001504	418	(2,209)
LINE	I*2	6-045774	000104	34	(34)
PAGE	L*1	4-000000	016516	3791	(121,62)
PLT	L*1	6-032140	005214	1332	(60,45)
T	I*2	6-032100	000040	16	(16)
X	I*2	6-000000	015040	3344	(16,209)
Y	I*2	6-015040	015040	3344	(16,209)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1'	**	95'	**	96'	**	97'	**	99	**
100	**	101	1-000060	109	**	110	**	150	1-001574
299'	**	300'	3-000000	301'	**				

FUNCTIONS AND SUBROUTINES REFERENCED

CLOSS

TOTAL SPACE ALLOCATED = 070332 14445

NO FPP INSTRUCTIONS GENERATED

TRPLT,LP:=TRPLT

3.4.15 VEGETATION INDEX (VIN(VEG,ID,N))

3.4.15.1 Linkage

VIN does not call any other program.

3.4.15.2 Interface

The interface with the calling arguments and the common block /DATA/.

3.4.15.3 Input

None

3.4.15.4 Output

None

3.4.15.5 Storage

4160 Bytes

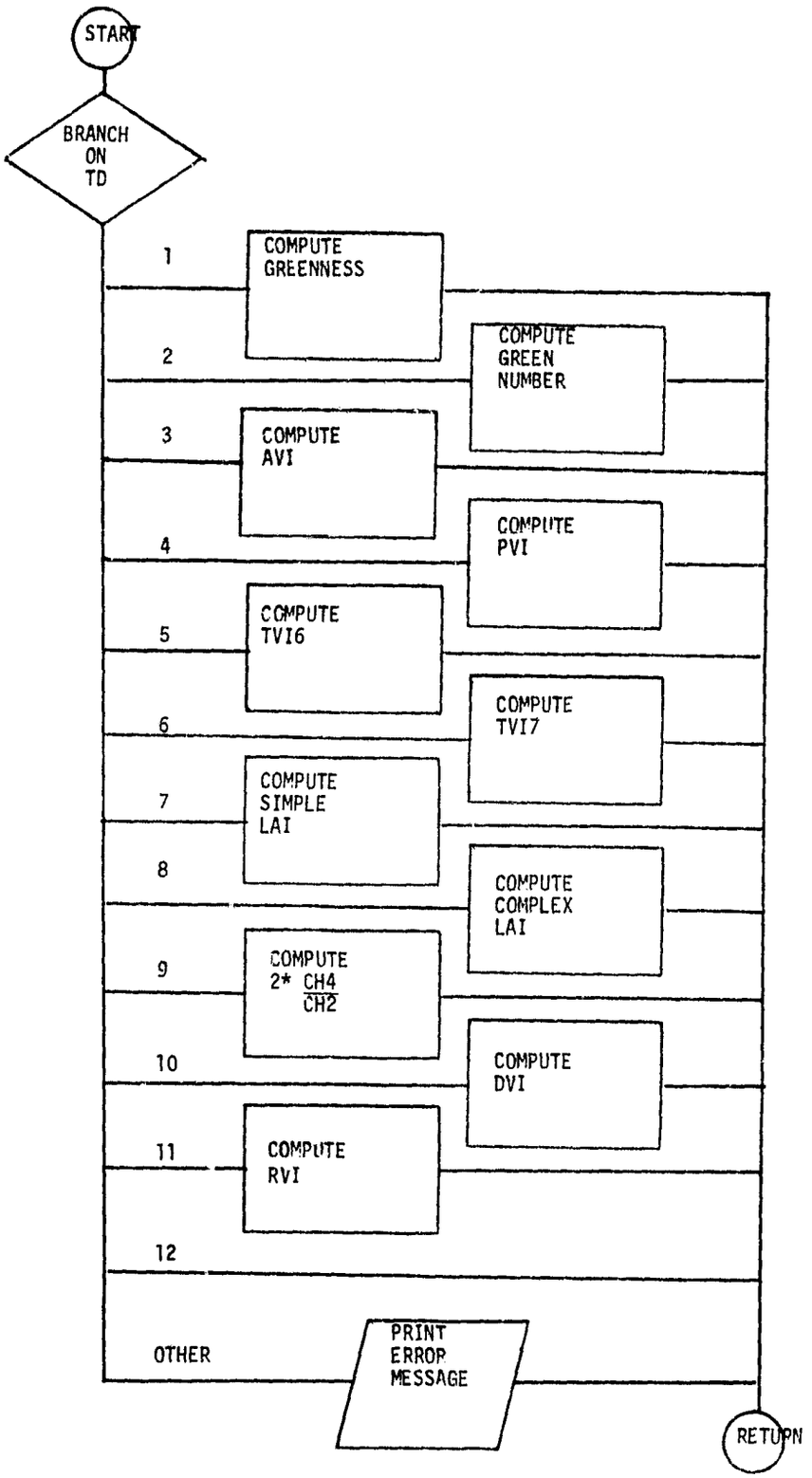
3.4.15.6 Description

VIN computes the vegetation index specified by ID for the first N points of /DATA/ and puts the result in VEG. The identifiers are

1 Greenness
2 Green Number
3 Ashburn Vegetation Index
4 Perpendicular Vegetation Index
5 Transformed Vegetation Index 6 TVI6
6 Transformed Vegetation Index 7 TVI7
7 Kannemasu's simple leaf area index
8 Kannemasu's complex leaf area index
9 Berkely ratio 2·CH4/CH2
10 DVI
11 RVI
12 Unused

VIN

3.4.15.7 Flowchart



```
        C VIN.FTN
0001          SUBROUTINE VIN(VEG,ID,N)
        C
        C     1    GREENNESS
        C     2    GREEN NUMBER
        C     3    ASHBURN VEGITATIVE INDEX
        C     4    PERPENDICULAR VEGITATION INDEX
        C     5    TRANSFERRED VEGITATION INDEX 6   TVI6
        C     6    TRANSFERRED VEGITATION INDEX 7   TVI7
        C     7    LEAF AREA INDEX   SIMPLE FORM
        C     8    LEAF AREA INDEX   COMPLEX FORM
        C     9    BERKELY    2*CH4/CH2
        C    10    DVI
        C    11    RVI
        C    12    UNUSED
        C
0002            INCLUDE 'INCLU.FTN'
      * C
      * C
      * C
0003  *        INTEGER*2 CH1(209),CH2(209),CH3(209),CH4(209)
0004  *        INTEGER*2 KH1(209),KH2(209),KH3(209),KH4(209)
0005  *        BYTE  FLAG(210),FAN(32),FDN(32),FIN(32)
0006  *        INTEGER*2 SEG,DATE(2),CODE,STATUS,FILL(36)
0007  *        INTEGER*2 IAG,SLN, AT,GAM,LAT(2),LNG(2)
0008  *          COMMON /ANCIL/ CODE,DATE,STATUS,FILL
0009  *          COMMON /DATA/  CH1,CH2,CH3,CH4
0010  *          COMMON /KAUTH/ KH1,KH2,KH3,KH4
0011  *          COMMON /FLAG/ FLAG
0012  *          COMMON /FIL/ FDN,FIN,FAN,KAN
0013  *          EQUIVALENCE (IAG,FILL(2)),(SLN,FILL(3))
0014  *          EQUIVALENCE (SA ,FILL(4)),(GAM,FILL(10))
0015  *          EQUIVALENCE (LA (1),FILL(5)),(LNG(1),FILL(7))
      * C
      * C     END INCLU.FTN
      * C
0016            REAL  VEG(1)
0017            REAL C1(209),C2(209),C3(209),C4(209)
        C
0018           IF(ID.LT.1 .OR. ID.GT.12) GO TO 333
0019           DO 13 L=1,N
0020           C1(L)=FLOAT(CH1(L))
0021           C2(L)=FLOAT(CH2(L))
0022           C3(L)=FLOAT(CH3(L))
0023           C4(L)=FLOAT(CH4(L))
0024           VEG(L)=0.
0025     13    CONTINUE
0026           GO TO (1,2,3,4,5,6,7,8,9,10,11,12) ID
        C
0027      1    CONTINUE
0028           DO 15 I=1,N
0029             VEG(I)=-.283*C1(I)-.660*C2(I)+.577*C3(I)+.388*C4(I)
0030     15    CONTINUE
0031           RETURN
0032      2    CONTINUE
0033           DO 25 I=1,N
```


Figure 3.4.15.8
Listing for VIN

```
0034                    VEG(I)=-,283*C1(I)-,660*C2(I)+,577*C3(I)+,388*C4(I)
0035                    VEG(I)=VEG(I)-SLN
0036        20      C3 TINUE
0037                RETURN
0038        3       C2 TINUE
0039                DO 30 I=1,N
0040                  VEG(I)=2*C4(I)-C2(I)
0041                  IF(VEG(I) .LT. 0.) VEG(I)=0.
0042        30      C0 TINUE
0043                RETURN
0044        4       C0 TINUE
0045                DO 40 I=1,N
0046                  VEG(I)=ABS(,38425*(2,4*C4(I)-C2(I)))
0047        40      C0 TINUE
0048                RETURN
0049        5       C2 TINUE
0050                DO 50 I=1,N
0051                  VEG(I)=SQRT(ABS((C3(I)-C2(I))/(C3(I)+C2(I))+0,5))
0052        50      C0 TINUE
0053                RETURN
0054        6       C0 TINUE
0055                DO 60 I=1,N
0056                  VEG(I)=SQRT(ABS((2*C4(I)-C2(I))/(2*C4(I)+C2(I))+0,5))
0057        60      C0 TINUE
0058                RETURN
0059        7       C0 TINUE
0060                DO  0 I=1,N
0061                  VEG(I)=1,653*C1(I)/C2(I)-1,658*C1(I)/C3(I)+,093
0062        70      CONTINUE
0063                RETURN
0064        8       C0 TINUE
0065                DO 80 I=1,N
0066                  VEG(I)=2,677 - 3,694*C1(I)/C2(I)
                    *                - 2,309*C1(I)/C3(I)
                    *                + 2,875*C1(I)/C4(I)
                    *                + 0,043*C2(I)/C3(I)
                    *                - 1,346*C2(I)/C4(I)
                    *          +3,017*((C1(I)/C2(I))-(C1(I)/(2*C4(I))))*C1(I)/C2(I)
0067        80      C2 TINUE
0068                RETURN
0069        9       C0 TINUE
0070                DO 90 I=1,N
0071                  VEG(I)=2*C4(I)/C2(I)
0072        90      C0 TINUE
0073                RETURN
0074        10      C2 TINUE
0075                DO 100 I=1,N
0076                  VEG(I)=C2(I)-2,4*C4(I)
12 0077     100     C0 TINUE
11 0078             RETURN
10 0079     11      C2 TINUE
 9 0080             DO 110 I=1,N
 8 0081               VEG(I)=C2(I)/C4(I)
 7 0082     110     C2 TINUE
 6 0083             RETURN
 5 0084     12      C2 TINUE
 4
```

```
              RETURN
      333     WRITE(5,334) ID
      334     FORMAT('    BAD ID =',I8,'       SUBROUTINE VIN')
              VEC(1)=9999.
              RETURN
              END
```

VIN,FTN /T:1BLOCKS/WR

PROGRAM SECTIONS

NUMBER	NAME	SIZE		ATTRIBUTES
1	BLNK01	007104	546	RW,I,CON,LCL
2	XPDATA	00 132	45	RW,I,CON,LCL
3	FIDATA	00 062	25	RW,I,CON,LCL
4	SWORS	00[illegible]26	1675	RW,I,CON,LCL
5	ITEMPS	000006	3	RW,I,CON,LCL
6	AUCIL	000120	40	RW,I,OVR,GBL
7	DATA	00 210	836	RW,I,OVR,GBL
8	KANIS	00[illegible]210	836	RW,I,OVR,GBL
9	FLAG	000322	105	RW,I,OVR,GBL
10	FILNAM	00 142	49	RW,I,OVR,GBL

ENTRY POINTS

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
VIN		1-000000												

VARIABLES

NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS	NAME	TYPE	ADDRESS
CODE	I*2	6-000000	GAM	I*2	6-000032	I	I*2	4-006424	IAG	I*2	6-000012	ID	I*2	F-000004*
KAN	I*2	10-000140	L	I*2	4-000422	N	I*2	F-000006*	SAT	I*2	6-000016	SEG	I*2	4-000000
SLI	I*2	6-000014	STATUS	I*2	6-000006									

ARRAYS

	NAME	TYPE	ADDRESS	SIZE		DIMENSIONS
	CH1	I*2	7-000000	000642	209	(209)
	CH2	I*2	7-000642	000642	209	(209)
	CH3	I*2	7-001304	000642	209	(209)
	CH4	I*2	7-001946	000642	209	(209)
	C1	R*4	4-000[illegible]12	001504	418	(209)
	C2	R*4	4-001516	001504	418	(209)
	C3	R*4	4-003212	001504	418	(209)
	C4	R*4	4-004716	001504	418	(209)
	DATE	I*2	6-000052	000004	2	(2)
	FAN	L*1	10-00[illegible]140	000040	16	(32)
	FDN	L*1	10-000060	000040	16	(32)
	FILL	I*2	6-00[illegible]112	000118	36	(36)
12	FIN	L*1	10-000040	000040	16	(32)
11	FLAG	L*1	9-000000	000322	105	(210)
10	KH1	I*2	8-00[illegible]000	000642	209	(209)
9	KH2	I*2	8-001682	000642	209	(209)
8	KH3	I*2	8-003[illegible]24	000642	209	(209)
7	KH4	I*2	8-003366	000642	209	(209)
6	LAT	I*2	6-000022	000004	2	(2)
5	LNG	I*2	6-000024	000004	2	(2)

VIN,FTN /TABLE2CAS/WP

VEG R*4 F=000002* 000004 2 (1)

LABELS

LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS	LABEL	ADDRESS
1	1-000214	2	1-000340	3	1-000500	4	1-000600	5	1-000700
6	1-001032	7	1-001166	8	1-001300	9	1-001552	10	1-001740
11	1-001736	12	1-002024	13	**	15	**	20	**
30	**	40	**	50	**	60	**	70	**
80	**	90	**	100	**	110	**	333	1-002026
334'	3-000300								

FUNCTIØNS AND SUBRØUTINES REFERENCED

$SQRT

TØTAL SPACE ALLØCATED = 020200 4100

VIN,LP:=VIN