National Aeronautics and Space Administration

JSC-14704

Lyndon B. Johnson Space Center
Houston, Texas 77058

"AS-BUILT" DESIGN SPECIFICATION FOR CCIT8 PROCESSOR PROGRAM

Job Order 71-475

(TIRF 78-0045)

Prepared By

Lockheed Electronics Company, Inc
Systems and Services Division
Houston, Texas

Contract NAS 9-15800

For

EARTH OBSERVATIONS DIVISION

SPACE AND LIFE SCIENCES DIRECTORATE

February 1979

LEC-13077

JSC-14704

"AS-BUILT" DESIGN SPECIFICATION FOR CCIT8 PROCESSOR PROGRAM

Job Order 71-475

(TIRF 78-0045)

PREPARED BY

R. F. Hansen

APPROVED BY

J. A. Morrow

J. A. Morrow, Supervisor
Scientific Applications Software Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

Space and Life Sciences Directorate

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
Houston, Texas

February 1979

LEC-13077

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
JSC-14704		

4. Title and Subtitle	5. Report Date
"As-Built" Design Specification for CCIT8 Processor Program	February 1979
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
R. F. Hansen Lockheed Electronics Company, Inc.	LEC-13077
	10. Work Unit No.
9. Performing Organization Name and Address Lockheed Electronics Company, Inc. 1830 NASA Road 1 Houston, Texas 77058	11. Contract or Grant No. NAS 9-15800
	13. Type of Report and Period Covered Technical Report
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Lyndon B. Johnson Space Center Houston, Texas 77058 Technical Monitor: (J. M. Sulester)	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

The program CCIT8 is a utility module of the Accuracy Assessment Software System of the Large Area Crop Inventory Experiment (LACIE). This program accesses data originating on the Classification and Mensuration Subsystem/Crop Assessment Subsystem interface tapes (CCIT's) of the LACIE System (version 8). The data items needed for subsequent accuracy assessment processing are written into three disk files. The data extracted consist of a summary of the classification stratified areal estimate, cluster-dot match, and analyst-labeled dots (types 1 and 2).

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Accuracy assessment CCIT LACIE	

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	64	

*For sale by the National Technical Information Service, Springfield, Virginia 22161

CONTENTS

TABLES

FIGURES

ABBREVIATIONS

AA	Accuracy Assessment
CAMS	Classification and Mensuration Subsystem
CAS	Crop Assessment Subsystem
CCIT	CAMS/CAS interface tape
DEC	Digital Equipment Corporation
DPR	Data processing request
DTL	Data Techniques Laboratory
DTRM	Data terminal
EOD	Earth Observations Division
ERIPS	Earth Resources Interactive Processing System
LACIE	Large Area Crop Inventory Experiment
Pixel	Picture element
SAE	Stratified areal estimate
TIRF	Transmittal Information Request Form
UIC	User identification code

1. SCOPE

This document specifies the final design for a PDP 11/45 software module called CCIT8, which manipulates and extracts data from Accuracy Assessment (AA) data base files. These files are derived from Large Area Crop Inventory Experiment (LACIE[1], version 8, Classification and Mensuration Subsystem/Crop Assessment Subsystem (CAMS/CAS) interface tapes (CCIT's) executed on the IBM 360/75. The data extracted are output into three new data base files for direct input to AA analytical programs.

2. APPLICABLE DOCUMENTS

The following documents, of the exact issue shown, form parts of the specification to the extent specified herein.

a. "As-Built" Design Specification for PDP 11/45 Accuracy Assessment System Using Disk Data File. JSC-13893 (LEC-11881), February 1978 (and references therein).

b. Implementation of CCIT6A Processor Program. Transmittal Information Request Form (TIRF) 78-0022, May 11, 1978.

c. CAM/CAS Interface Tape Interface Control Document. LACIE-00708, revision A (JSC-09866), July 1976.

d. Classification and Mensuration Subsystem (CAMS) Requirements. LACIE-00200, volume II, revision D (JSC-11330), August 1977.

e. "As-Built" Design Specification for CCIT6A Processor Program. JSC-14368 (LEC-12303), August 1978.

f. "As-Built" Specification for CCIT7 Processor Program. JSC-14554 (LEC-12518), November 1978.

g. TIRF 78-0045, September 25, 1978.

3. SYSTEM DESCRIPTION

The CCIT8 processor module accomplishes the data manipulations shown in figure 1. Basically, the CCIT data for a particular segment number, SSSS, and classification date, YYDDD, contained in file SSSSYYDDD.CCO are processed to obtain three output files required as input to existing or planned AA programs. The SSSSYYDDD.CLO file contains data needed for futur programs. The SSSSYYDDD.AI1 and SSSSYYDDD.AI2 files are required for input to existing modules SPATL and MLTCRP.

3.1 HARDWARE DESCRIPTION

The PDP 11/45, with the following peripherals, is required.

a. Card reader or user terminal

b. Line printer

c. Two disk units

3.2 MODULE DESCRIPTION

The CCIT8 module is implemented on the PDP 11/45 for time-sharing or background processing of CCIT data files into three data files: an unformatted file of character data and two formatted files of analyst-labeled dots. See the functional flow diagram (fig. 2).

The LACIE CCIT is a universal nonimaging tape containing extensive statistical and ancillary data for a series of Earth Resources Interactive Processing System (ERIPS) runs. Using the AA CCIT program, all data for a relevant segment are transferred to a Files-11 disk file named SSSSYYDDD.CCO,***, where SSSS is the segment number, YY is the year, and DDD is the day of the year. This .CCO file contains three 80-byte header records and a large number (>20) of 720-byte data records.

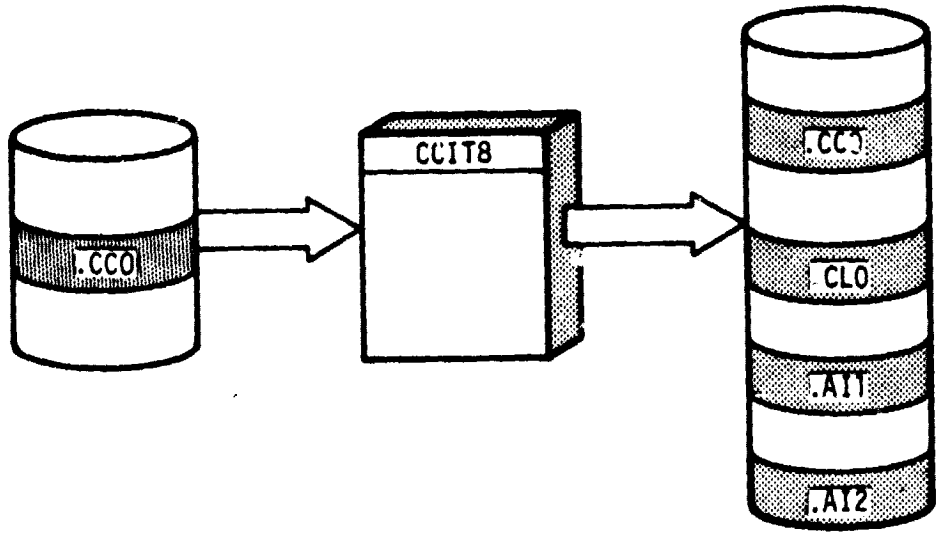

Figure 1.– Data flow of the CCIT8 processor program.

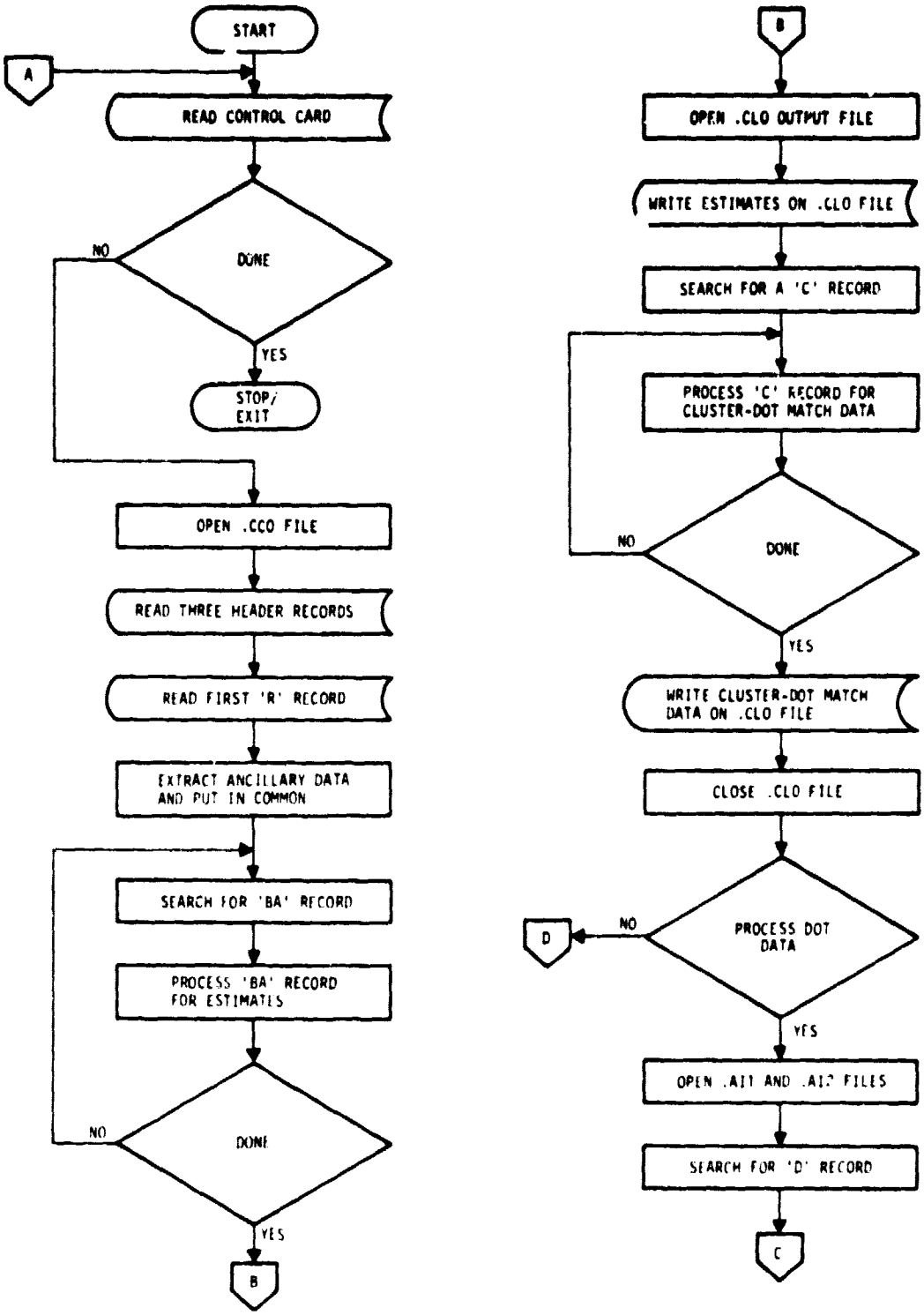

Figure 2.– Functional flow of the CCIT8 processor program.

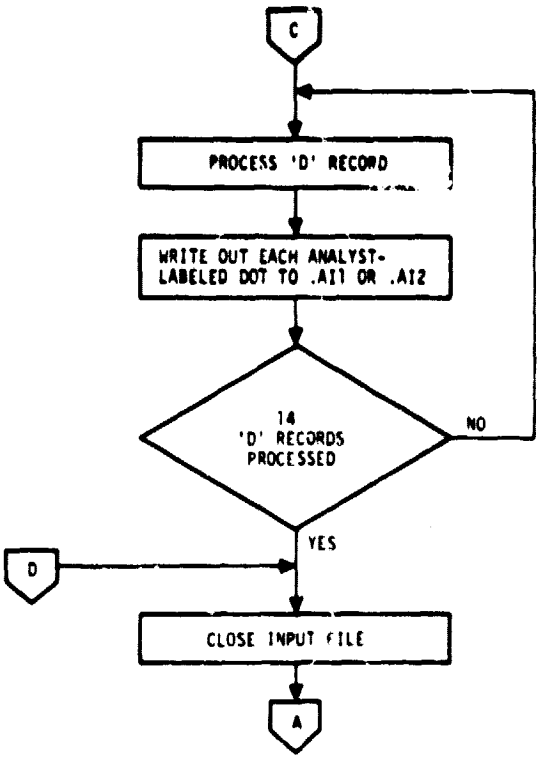

Figure 2.– Concluded.

The first step of the process is to read the name of the input .CCO file and open this file for reading. Then the three CCIT header records are read and ignored. The next record (720 bytes) is read, checked to verify that it is a recognition (R) record, and processed to extract the data processing request (DPR) number and acquisition dates used in the classification.

Next a 'BA' record is searched for and processed. The program extracts classification data on class picture element (pixel) populations, the ERIPS estimate, the stratified areal estimate (SAE), and the variance for each class. An output file SSSSYYDDD.CLO (where SSSS is the segment number and YYDDD is the classification date derived from the DPR number) is opened. The first record of the .CLO file contains an integer word giving the number of crop classes for the classification. The estimate data are written as the second record. Sixteen bytes are required for each class; a LACIE version 8 CCIT can contain data on up to 26 classes. A line printer report is output after the disk files are written.

In the next step a 'C' record is searched for and processed. The total number of clusters (Q) and cluster-dot match data are extracted as Q-groups of 12 characters. (Generally, there are more than 15 clusters, so some of these data appear in additional 'C' records.) When all the cluster-dot data are assembled into a buffer, the number of clusters is written out as the third record of file SSSSYYDDD.CLO, and the match data are written as the second record of this file. A line printer report is then output, and the SSSSYYDDD.CLO file is closed.

Then the program tests to determine if analyst-labeled dot output files are required; this is the default condition. If this condition exists, the output files SSSSYYDDD.AI1 and SSSSYYDDD.AI2 are opened, and a search is made for the first dot record. There are 14 dot records listing all 209 dots. The program examines each dot to determine if it has been labeled by the analyst. If so, it is written out (line, sample, and label) to the proper file, depending on the dot type (1 or 2). The first dot record in

each output file also contains ancillary information on the segment (number and state code), classification date, acquisition dates, data terminal (DTRM) tape number, and type of label.

When all dots have been processed, the SSSSYYDDD.AI1 and SSSSYYDDD.AI2 output files and the input file are closed. No report is output detailing the .AI file dot data because these formatted files are easily examined using the Digital Equipment Corporation (DEC) PDP-11 utility program PIP.

3.3 SOFTWARE DESCRIPTION

The CCIT8 processor program consists of 11 user-supplied routines: CCIT8 (main program), READH, READRC, HEADER, BIASC8, CLUST8, RITE8, TURNON, DOTS8, STCOD8, and PRINT8. The program makes use of a card-image-formatted file, CCIT8.DAT, for program control and the line printer and user disk for output. The following sections provide a detailed description of each of the 11 routines. The recommended task-build command file (CCIT8.CMD), used to create the load module (CCIT8.TSK), is given in table 1.

TABLE 1.— TASK-BUILDER COMMAND FILE FOR CCIT8 PROCESSOR PROGRAM

```
CCIT8,LP:/SH=CCIT8,READH,READRC,HEADER,BIASC8,CLUST8,
RITE8,TURNON,DOTS8,STCOD8,PRINT8
/
FMTBUF=132
UNITS=6
ACTFIL=6
ASG=SY:1
ASG=SY:2
ASG=SY:3
ASG=SY:5
ASG=LP:6
PRI=50
//
```

For simplicity, the definition of arrays carried in COMMON blocks, the definition of COMMON blocks, and the description of COMMON blocks are not repeated for each routine. Instead, each of these elements is described in the

routine of origin. Reference to the Interface subsections and to the compiler listings of each routine provides sufficient information to follow the data flow throughout the program.

3.3.1 MODULE CCIT8

3.3.1.1 Linkage

The CCIT8 program is the main program. It calls user subroutines READH, READRC, HEADER, BIASC8, CLUST8, DOTS8, and PRINT8. Subroutines BIASC8, DOTS8, and PRINT8 are called using multiple entry points.

3.3.1.2 Interface

Most communication with the user routines is handled via COMMON blocks. A single integer parameter is passed on call to READH, which indicates the number of CCIT header records to be read. A single integer parameter Q is passed to CLUST8 as a flag.

3.3.1.2.1 COMMON Block Buf

BUF contains a 720-byte array, A, which is used to hold one CCIT logical record for processing.

3.3.1.2.2 COMMON Block FNAME

FNAME contains a 24-byte array, FILNAM, and an integer variable, SKIP. FILNAM contains the input file name read from CCIT8.DAT. The value of SKIP determines whether the dot records are to be processed. If SKIP is nonzero, the dots are not processed.

3.3.1.2.3 COMMON Block B7

B7 contains the number of categories, NCAT, for the classification.

3.3.1.2.4 COMMON Block CLUSTR

CLUSTR contains a 60-byte by 12-byte array, CNAME, and an integer variable, CNUM. CLUSTR provides an interface between subroutine CLUST8 and subroutine RITE8.

3.3.1.3 Input

The CCIT8 program receives all input CCIT data via subroutine READRC. Control data are provided directly by reading card images from file CCIT8.DAT.

3.3.1.4 Output

The CCIT8 program provides all output via subroutines PRINT8, RITE8, BIASC8, DOTS8, and CLUST8.

3.3.1.5 Storage

The CCIT8 program requires 1078 words of storage.

3.3.1.6 Description

The CCIT8 routine provides the control function for the program. Flow is controlled via tests on the first bytes (descriptive characters) of each logical record in the CCIT input file and by counts based upon the required number of records of a given type.

3.3.1.7 Flow Chart

The flow chart for CCIT8 is given in figure 3.

3.3.1.8 Listing

The listing for this subroutine is given in figure 4.

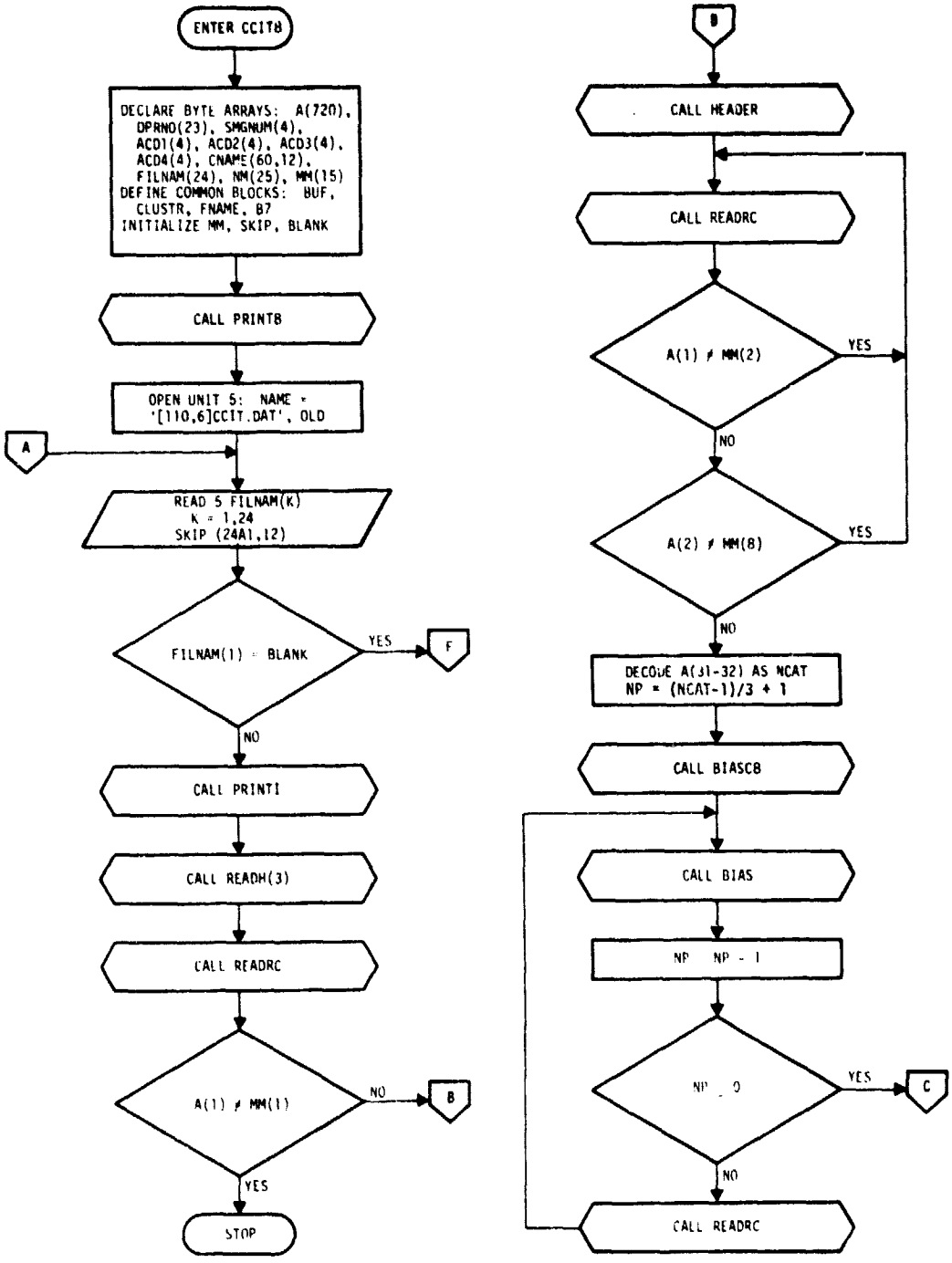

Figure 3.– Flow diagram for the CCIT8 processor program.

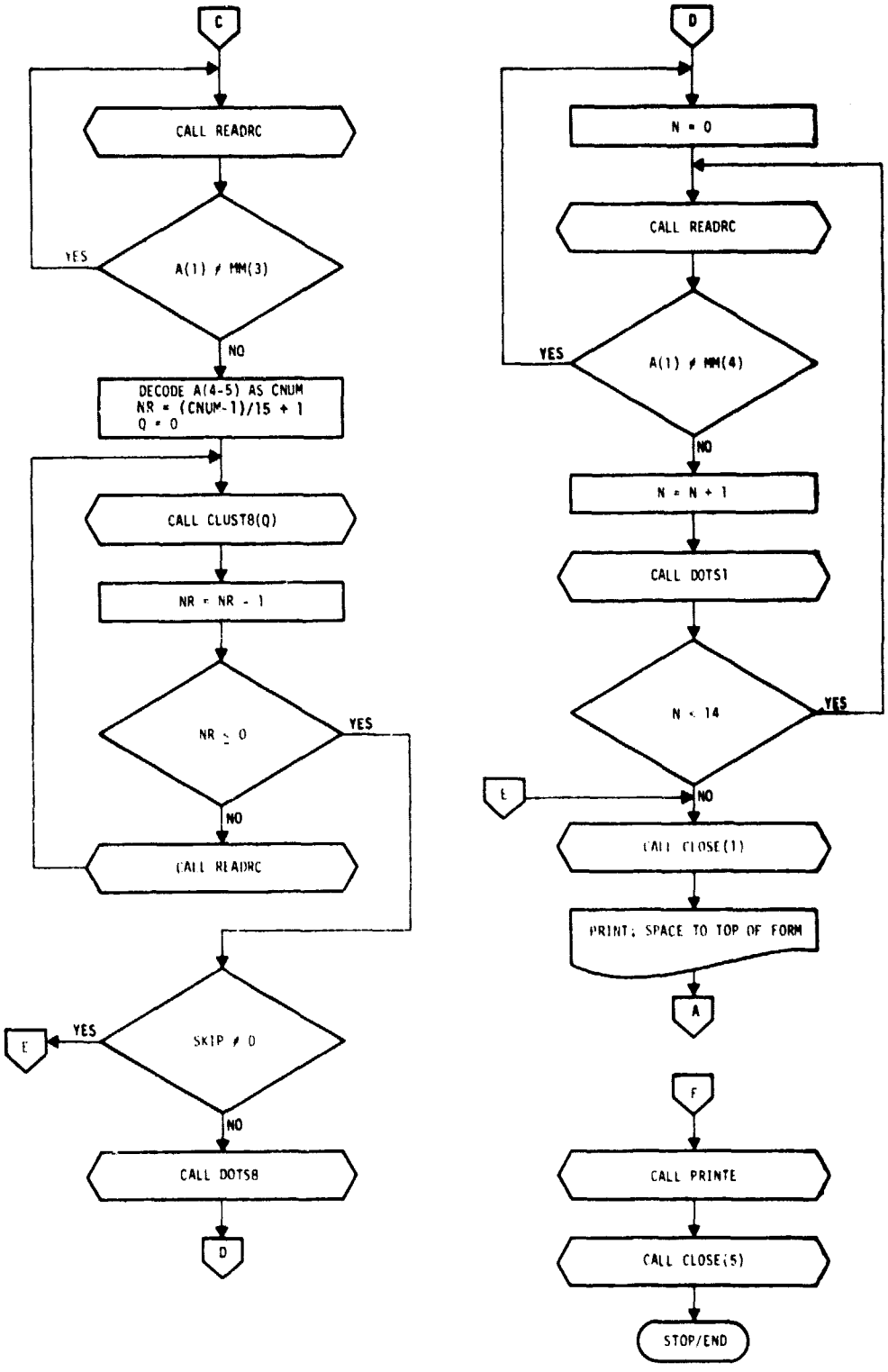

Figure 3. -- Concluded.

```
            PROGRAM CCIT8
            IMPLICIT INTEGER(A-Z)
            BYTE A(720),PRNT(23),SMONTH(4),ACD1(4),ACD2(4),ACD3(4),ACD4(4),
           1 CNAME(80,12),FILNAM(24),MM(25),MH(15),BLANK
      C***************************************************************
      C     MAIN OF CCIT8 PROCESSOR - READ A CARD IMAGES FROM [110,6]CCIT8.DAT
      C     WHICH GIVES THE FILE NAME OF THE LACIE AA CCIT FILE ON DISC
      C     IN THE FORM [xxx,xx]SSSSYYDDD.CCI
      C     A(720) IS THE BUFFER WHICH CONTAINS ONE CCIT8 DATA RECORD
      C***************************************************************
            COMMON/B/A
            COMMON/B2/CAT
            COMMON/CLUSTR/CNAME,CNUM
            COMMON/FNAME/FILNAM,SKIP
            DATA MM/1H0,1H0,1HC,1H0,1H0,1H1,1H2,1HA,7*1H /
            DATA SKIP/0/
            DATA BLANK/1H /
            CALL PRINTS
      C     READ INPUT FILNAME AND SKIP FACTOR
            OPEN(UNIT=5,NAME='[110,6]CCIT8.DAT',TYPE='OLD',READONLY,
           1 ACCESS='SEQUENTIAL',FORM='FORMATTED')
      200   READ(5,101)(FILNAM(K),K=1,24),SKIP
            IF(FILNAM(1).EQ.BLANK) GO TO 200
            CALL PRINTI
      C     READ 3 HEADER RECORDS OF 80 BYTES EACH
            CALL READH(3)
      C     READ THE FIRST ... RECORD AND PROCESS FOR CPR NUMBER,ACQUISITION
      C     DATES ETC
            CALL READRC
            IF(A(1).NE.MM(1)) STOP
            CALL HEADER
      C     READ AND PROCESS BIAS CORRECTION SUMMARY RECORD
      C     AS FIRST RECORD OF FILE SSSSYYDDD.CLD
      210   CALL READRC
            IF(A(1).NE.MM(2)) GO TO 210
            IF(A(2).NE.MM(8)) GO TO 210
            DECODE(2,100,A(31))NCAT
            NP = (NCAT-1)/3 + 1
            CALL BIASC0
      245   CALL BIAS
            NP = NP - 1
            IF( NP.LE.0) GO TO 220
            CALL READRC
            GO TO 245
      C     READ FIRST CLUSTER RECORD
      C     THE, ONE SUBROUTINE WRITER' IS CALLED WHICH WRITES THE NUMBER
      C     OF CLUSTERS) AND THE CLUSTER-DOT MATCH DATA OUT AS TWO RECORDS
      C     OF FILE SSSSYYDDD.CLD
      220   CALL READRC
            IF(A(1).NE.MM(3)) GO TO 220
            DECODE(2,100,A(4))CNUM
            NR = (CNUM-1)/5 + 1
            C = 0
      225   CALL CLUSTR(C)
            NR = NR -1
            IF(NR.LE.0) GO TO 225
```


Figure 4.— Listing for the CCIT8 Processor Program.

```
[illegible]                                   13:55:22    05-JAN-79            PAGE 2
[illegible]           /[illegible]CKS/[illegible]

                      CALL [illegible]
                      GO TO 235
      C               IF [illegible] DATA IS NOT NEEDED SKIP TO END OF PROGAM
        225           IF(S[illegible].E.0) GO TO 250
      C               INITIALIZE THE [illegible] OUTPUT FILES AND COUNTERS
                      CALL [illegible]
      C               READ [illegible]TS RECORDS
        230           [illegible]
      C               LOOK FOR [illegible] (20S [illegible]TS) RECORDS
        2[illegible]  CALL [illegible]
                      IF(A[illegible]).[illegible]E.[illegible](4)) GO TO 230
                      [illegible] + 1
      C               PROCESS A [illegible] RECORD
                      CALL [illegible]
                      IF([illegible].14) GO TO 240
      C               CLOSE INPUT FILE
        250           CALL CLOSE(1)
                      PRINT 102
                      GO TO [illegible]
        2[illegible]  CALL [illegible]
                      CALL CLOSE(5)
                      STOP
        1[illegible]  FORMAT(I2)
        1[illegible]  FORMAT(24A1,I2)
        102           FORMAT(1H1)
                      END
```

Figure 4.— Concluded.

3.3.2 SUBROUTINE READH

3.3.2.1 Linkage

Subroutine READH calls subroutine TURNON.

3.3.2.2 Interface

READH interfaces with TURNON via an integer parameter (passed on call) giving the logical unit number to be opened and via COMMON block NAME containing the name of the file to be opened. COMMON block FNAME (see section 3.3.1.2.2) interfaces CCIT8 with READH. COMMON block BUF provides no true interfacing function for this routine.

3.3.2.2.1 COMMON Block NAME

NAME contains a 25-byte array, NM, which contains the complete name of a file to be opened by subroutine TURNON. NAME also interfaces several subroutines with subroutine PRINT8. The last byte of array NM should contain the null (0) character.

3.3.2.3 Input

Header records from the CCIT input file are input.

3.3.2.4 Output

The only output is a read error message to the line printer.

3.3.2.5 Storage

READH requires 514 words of storage.

3.3.2.6 Description

Subroutine READH spaces past the three 80-byte CCIT header records, and the CCIT file name is written into the NM array. Subroutine TURNON opens the file on unit 1, the three records are read, and READH returns to CCIT8.

3.3.2.7 Flow Chart

The flow diagram for subroutine READH is given in figure 5.

3.3.2.8 Listing

The listing for this subroutine is given in figure 6.

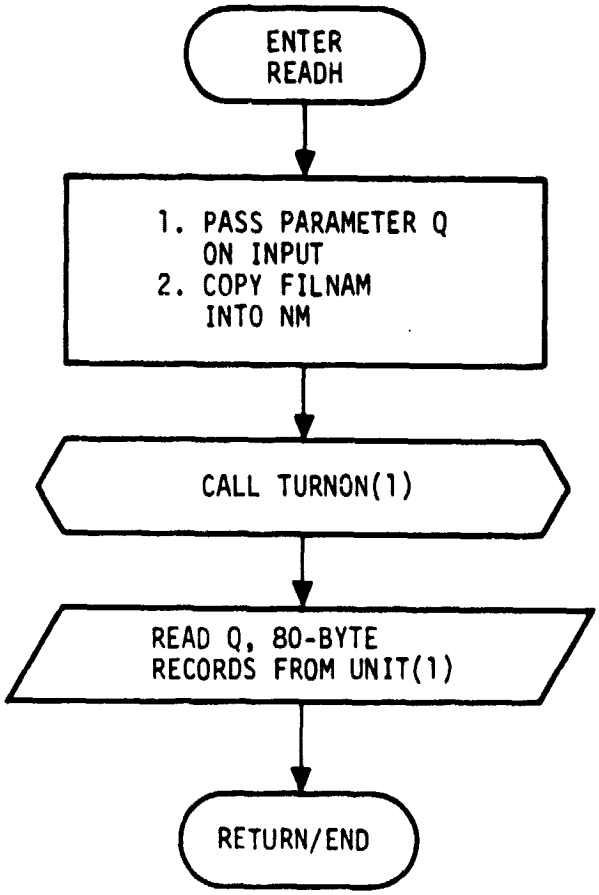

Figure 5.-- Flow diagram for subroutine READH.

```
FORTRAN IV-PLUS V02-51          13:52:11    09-JAN-79          PAGE 1
READH.FTN         /TR:ALL/WR

            SUBROUTINE READH(I)
            IMPLICIT INTEGER(A-Z)
      C     READ CCIT HEADER RECORDS FROM DISC FOR FILE FILNAM
            BYTE  M(25),FILNAM(24),A(720)
            COMMON /BUF/A
            COMMON /NAME/M
            COMMON/FNAME/FILNAM,SKIP
            DO 10 JJ=1,24
      10    M(JJ) = FILNAM(JJ)
            M(25) = 0
            CALL TURN(1)
            DO 1 J=1,8
      1     READ(1,ERR=99,END=99)(A(K),K=1,89)
            RETURN
      99    PRINT 200,J
      200   FORMAT(1H ,' READ ERROR IN READH; ITERATION ',I2,//)
            STOP
            END
```

Figure 6.— Listing for Subroutine READH.

18

3.3.3 SUBROUTINE READRC

3.3.3.1 Linkage

Subroutine READRC is called by CCIT8.

3.3.3.2 Interface

Subroutine READRC interfaces with CCIT8 via COMMON block BUF (see section 3.3.1.2.1).

3.3.3.3 Input

One data record read from the CCIT disk file is input.

3.3.3.4 Output

A read operation error message is output to the line printer.

3.3.3.5 Storage

This subroutine requires 442 words of storage.

3.3.3.6 Description

READRC reads one 720-byte logical data record from the CCIT input file into a buffer array, A.

3.3.3.7 Flow Chart

The flow diagram for subroutine READRC is given in figure 7.

3.3.3.8 Listing

The listing for this subroutine is given in figure 8.

A

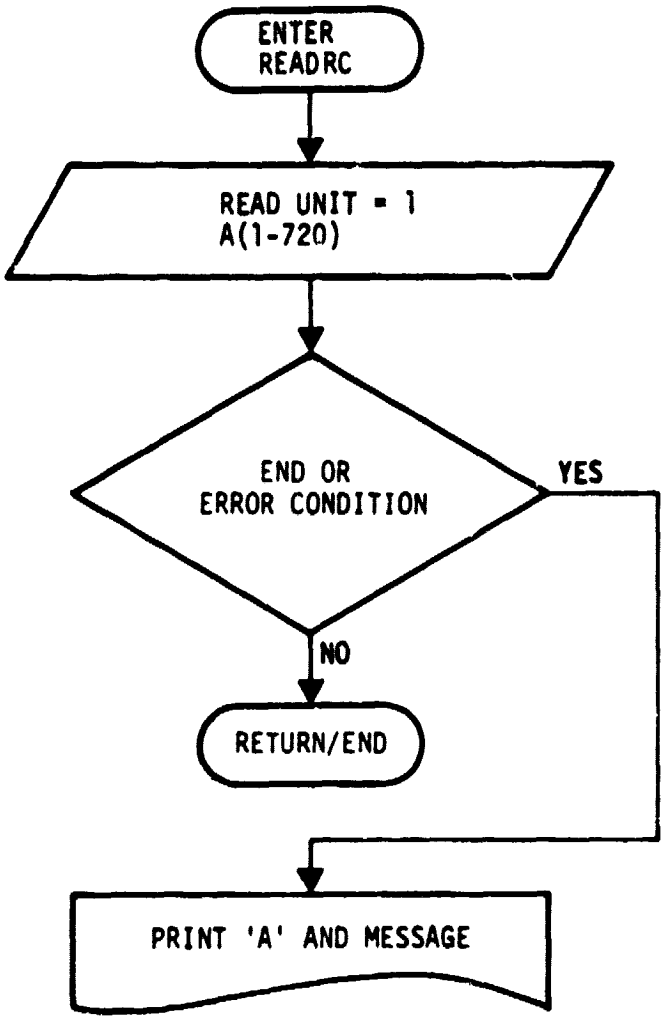

Figure 7.– Flow diagram for subroutine READRC.

```
FORTRAN IV-PLUS V02-51          07144149     C5-MAY-78           PAGE 1
READRC.FTN       /TR:BLOCKS/WR

0001             SUBROUTINE READRC
0002             IMPLICIT INTEGER(A-Z)
           C     READ A CCIT DATA RECORD OF 720 BYTES INTO THE BUFFER A
0003             BYTE A(720)
0004             COMMON/BUF/A
0005             READ(1,ERR=99,END=99)(A(K),K=1,720)
0006             RETURN
0007        99   PRINT 100,A
0008        100  FORMAT(1H ,'ERROR IN READRC',//,' BUFFER CONTAINS',//,
               *1H ,6(120A1))
0009             END
```


Figure 8.— Listing for subroutine READRC.

3.3.4 SUBROUTINE HEADER

3.3.4.1 Linkage

Subroutine HEADER is called by CCIT8.

3.3.4.2 Interface

HEADER interfaces with CCIT8 via COMMON blocks BUF (see section 3.3.1.2.1) and DOTS and interfaces with PRINT8 (entry PRINTH) via COMMON block DOTS.

3.3.4.2.1 COMMON Block DOTS

DOTS contains a 23-byte array (DPRNO) that is used to hold the ERIPS DPR number, four 4-byte arrays (ACD1, ACD2, ACD3, and ACD4) that are used to store the acquisition dates used for the ERIPS run, and a 4-byte array (SMGNUM) that is used to store the LACIE segment number.

3.3.4.3 Input

Buffer array A is input.

3.3.4.4 Output

The outputs are the LACIE segment number, acquisition dates used, and ERIPS DPR number.

3.3.4.5 Storage

This subroutine requires 588 words of storage.

3.3.4.6 Description

Subroutine HEADER selects byte data from the 'R' record of a CCIT (contained in buffer array A) and stores it into arrays in COMMON block DOTS. The data selected are the LACIE segment number [SMGNUM(1-4)], acquisition dates used for the ERIPS run [ACD1,ACD2,ACD3,ACD4], and ERIPS DPR number [DPRNO(1-23)]. The DPR number is printed in a message via a call to entry PRINTH of PRINT8.

3.3.4.7 Flow Chart

The flow diagram for subroutine HEADER is given in figure 9.

3.3.4.8 Listing

The listing for this subroutine is given in figure 10.

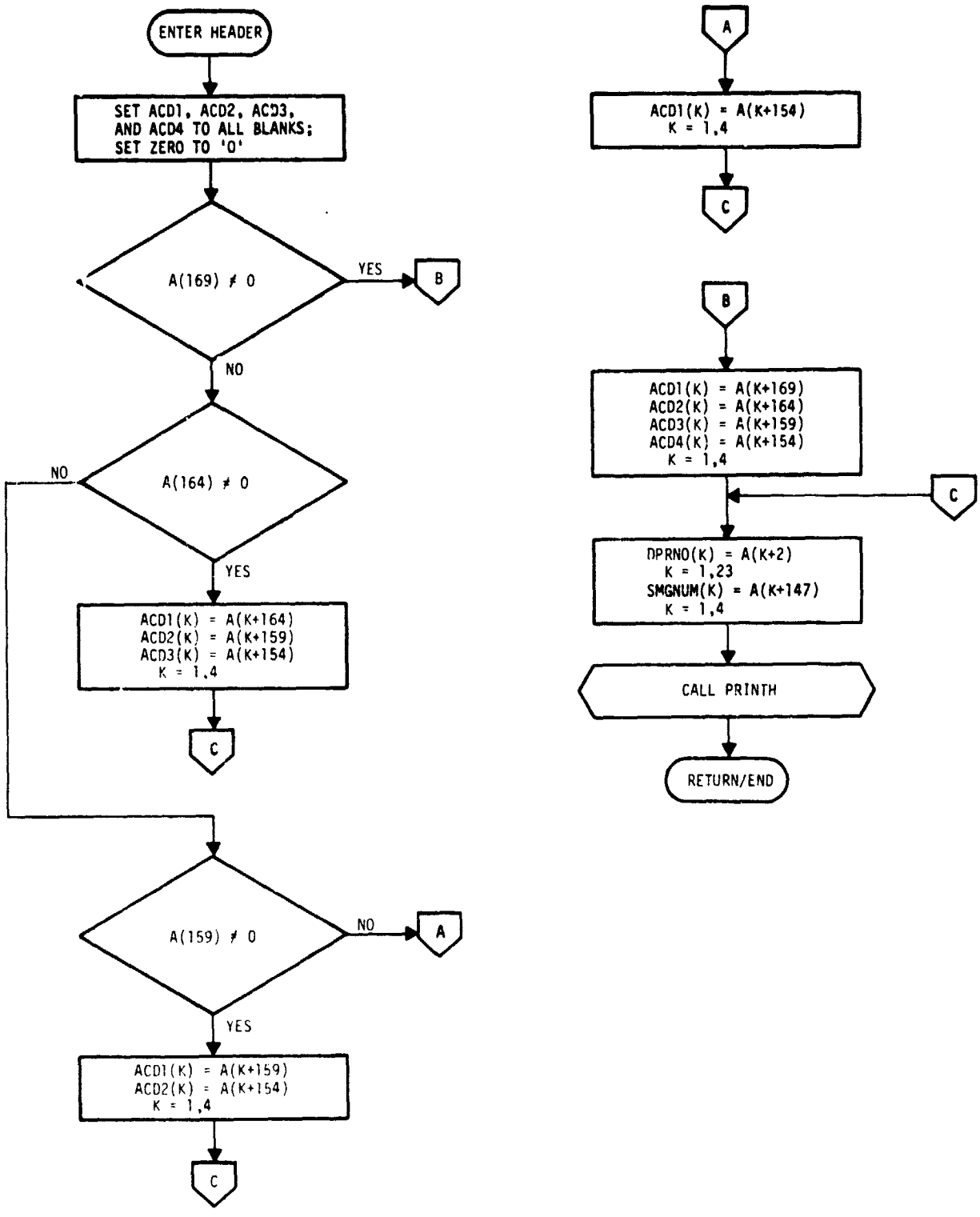

Figure 9.— Flow diagram for subroutine HEADER.

```
[illegible]                  13:52:25    05-JAN-79          PAGE 1
[illegible]

      SUBROUTINE HEADER
C***************************************************************
C     THIS SUBROUTINE OBTAINS THE DPR NUMBER ,AND ACQUISITION
C     DATES FROM AN IRT RECORD.
C     THE ACQUISITION DATES ARE ORDERED BY LATEST DATE FIRST
C     TO CONFORM TO OTHER AA SOFTWARE MODULE CONVENTIONS
C***************************************************************
      IMPLICIT INTEGER(A-Z)
      BYTE A(720),ACD1(4),ACD2(4),ACD3(4),ACD4(4),
     1 DPRNO(23),SMGNUM(4)
      BYTE ZERO(1),BLANK(1)
      COMMON /BUF/A
      COMMON /CDTS/DPRNO,ACD1,ACD2,ACD3,ACD4,SMGNUM
      DATA ZERO/1H0/
      DATA BLANK/1H /
C     CHECK TO SEE HOW MANY ACQUISITIONS WERE USED
C     ASSUMED TO BE IN THE ORDER OF EARLIEST FIRST
C     WITH ASCII ZERO FILL
C     BLANK OUT ACQUISITION DATE STOREAGE UNITS
      DO 2 J=1,4
      ACD1(J) = BLANK(1)
      ACD2(J) = BLANK(1)
      ACD3(J) = BLANK(1)
      ACD4(J) = BLANK(1)
    2 CONTINUE
      IF(A(159).NE.ZERO(1)) GO TO 10
      IF(A(164).NE.ZERO(1)) GO TO 20
      IF(A(159).NE.ZERO(1)) GO TO 30
      DO 4 K=1,4
    4 ACD1(K) = A(K+154)
      GO TO 40
   30 DO 5 K=1,4
      ACD1(K) = A(K+159)
    5 ACD2(K) = A(K+154)
      GO TO 40
   20 DO 6 K=1,4
      ACD1(K) = A(K+164)
      ACD2(K) = A(K+159)
    6 ACD3(K) = A(K+154)
      GO TO 40
   10 DO 7 K=1,4
      ACD1(K) = A(K+169)
      ACD2(K) = A(K+164)
      ACD3(K) = A(K+159)
    7 ACD4(K) = A(K+154)
C     WRITE THE DPR NUMBER
   40 DO 3 K=1,23
    3 DPRNO(K) = A(K+2)
C     WRITE SEGMENT NUMBER
      DO 1 K=1,4
    1 SMGNUM(K) = A(K+147)
C     PRINT MESSAGE ON PRINTER
      CALL PRINTM
      RETURN
      END
```

Figure 10.– Listing for Subroutine HEADER.

25

3.3.5 SUBROUTINE BIASC8

3.3.5.1 Linkage

BIASC8 is called once by the CCIT8 program. It calls subroutine TURNON once. An additional entry point, BIAS, is called one or more times by CCIT8.

3.3.5.2 Interface

BIASC interfaces with CCIT8 via COMMON blocks BUF (see section 3.3.1.2.1) and B7 (see section 3.3.1.2.3), and with TURNON via COMMON block NAME (see section 3.3.2.2.1).

3.3.5.3 Input

The input is the Bias Correction Data from the CCIT "BA" record.

3.3.5.4 Output

BIASC8 writes two unformatted records onto unit 3. The data contained in this record are detailed in the appendix.

3.3.5.5 Storage

This subroutine requires 874 words of storage.

3.3.5.6 Description

BIASC8 codes the output file name as SSSSYYDDD.CLO, where SSSS is the segment number and YYDDD is the classification date. Unit 3 is opened for output via a call to subroutine TURNON. Then a record containing the integer NCAT (the number of categories or classes) is written on unit 3, and 12 times NCAT bytes of character data are output as the second record.

3.3.5.7 Flow Chart

The flow diagram for subroutine BIASC8 is given in figure 11.

3.3.5.8 Listing

The listing for this subroutine is given in figure 12.

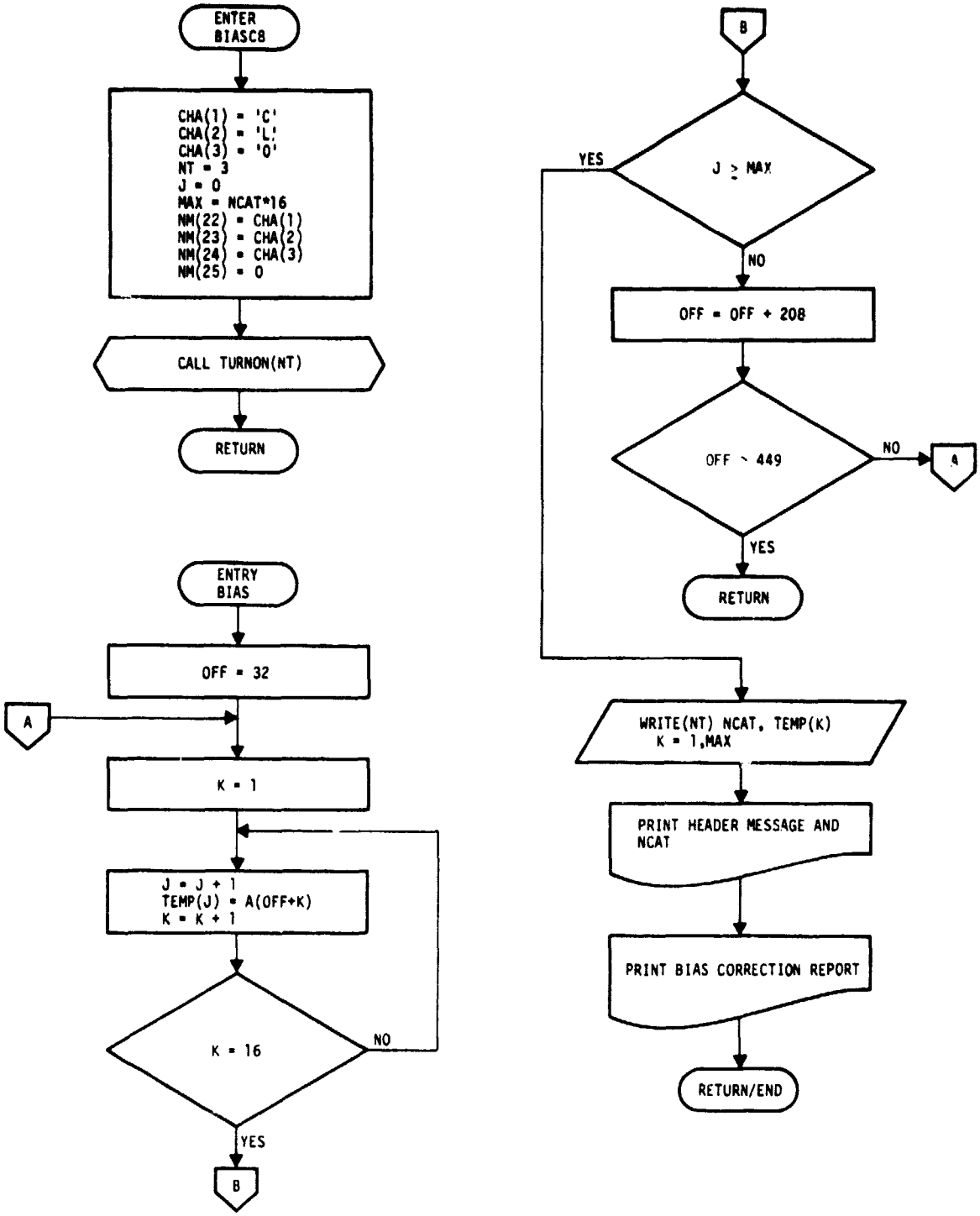

Figure 11.— Flow diagram for subroutine BIASC8.

```
FORTRAN IV-PLUS V02-51        13:52:41     05-JAN-79          PAGE 1
BIASC8.FTN      /TR:ALL/WR
          2
                SUBROUTINE BIASC8
                IMPLICIT INTEGER(A-Z)
          C*****************************************************
          C     USES THE BIAS CORRECTION (I.E., STRATIFIED AREAL
          C     ESTIMATE DATA FROM A CCIT 'BA' RECORD
          C     TO FORM THE FIRST RECORD OF THE
          C     SSSSYYDDD.CLO OUTPUT FILE, LEAVES FILE OPEN FOR
          C     WRITING OF TWO MORE RECORDS) NUMBER OF CLUSTERS,
          C     AND THE ALPHANUMERIC CLUSTER-CAT MATCH TABLE
          C     BY SUBROUTINE CLUSTB
          C*****************************************************
                COMMON/27/NCAT
                BYTE NM(25),SHGNUM(4),DPRN2(23),ACD1(4),ACD2(4),ACD3(4),
              1 ACD4(4),CHA(3),TEMP(416)
                BYTE A(720)
                COMMON/BUF/A
                COMMON/NAME/NM
                COMMON/DOTS/DPRN2,ACD1,ACD2,ACD3,ACD4,SHGNUM
                DATA CHA/1HC,1HL,1HO/
          C*****************************************************
          C     THIS PART OPENS OUTPUT FILE AND SETS UP THE
          C     PROCESSING ONLY -- ENTRY BIAS IS CALLED ONCE
          C     FOR EACH 'BA' RECORD PROCESSED
          C*****************************************************
                NT = 3
                J = 0
                MAX = NCAT*16
          C     COMPLETE NAME FOR OUTPUT FILE IN FORM SSSSYYDDD.CLO
                NM(22) = CHA(1)
                NM(23) = CHA(2)
                NM(24) = CHA(3)
                NM(25) = 0
          C     OPEN OUTPUT FILE - UNFORMATTED
                CALL TURNON(NT)
                RETURN
          C**********************************************************
          C     ENTER HERE ONCE FOR EACH 'BA' RECORD PROCESSED.
          C     WHEN ALL CLASSES ("CATEGORIES") HAVE BEEN WRITTEN
          CC    INTO BUFFER TEMP, WRITE TO DISK AND PRINT REPORT
          C**********************************************************
                ENTRY BIAS
                OFF = 32
          2     DO 1 K=1,16
                J = J + 1
          1     TEMP(J) = A(OFF+K)
                IF(J.GE.MAX) GO TO 10
                OFF = OFF + 208
                IF(OFF.GT.449) RETURN
                GO TO 2
          C**********************************************************
          C     WRITE DISK RECORD AND PRINTER REPORT
          C**********************************************************
          10    WRITE(NT),NCAT
                WRITE(NT),(TEMP(K),K=1,MAX)
                WRITE(6,200)NCAT
```

Figure 12.— Listing for subroutine BIASC8.

```
FORTRAN IV PLUS V02-51          13152141   09-JAN-79          PAGE 2
BIAS08,FTN      /TR:ALL/-R

0031                WRITE(6,300)(TEMP(K),K=1,MAX)
0032                RETURN
0033          200   FORMAT(1H ,15X,'BIAS CORRECTION ',//,15X,'FOR ',I2,' CATAGORIES')
0034          300   FORMAT(1H ,26(9X,'CATAGORY LABEL ',A1,'  POPULATION ',9A1,//,
                1   10X,'UNCORRECTED ',1A1,',',2A1,' CORRECTED ',1A1,',',2A1,
                2   ' VARIANCE ',2A1,',',2A1,//))
0035                END
```

Figure 12.– Concluded.

29

3.3.6 SUBROUTINE CLUST8

3.3.6.1 Linkage

Subroutine CLUST8 is called one to four times by CCIT8. CLUST8 calls RITE8 for data output and calls the PRINTC entry of PRINT8 for an output message.

3.3.6.2 Interface

CLUST8 interfaces with CCIT8 via COMMON block BUF (see section 3.3.1.2.1) and passed parameter Q, with RITE8 via COMMON block CLUSTR (see section 3.3.1.2.4), and with PRINT8 via passing parameter RCNUM on call.

3.3.6.3 Input

The input is the cluster information from the CCIT 'C' records.

3.3.6.4 Output

The total number of clusters, the number of clusters on the record; for each cluster, the cluster name and dot name are output.

3.3.6.5 Storage

This subroutine requires 856 words of storage.

3.3.6.6 Description

CLUST8 processes the CCIT 'C' records to provide the total number of clusters and the identity of the analyst-labeled (type 1) dot used to name each cluster.

On the first record processed, CLUST8 decodes bytes 4 and 5 to obtain the total number of clusters (CNUM) and bytes 6 and 7 to obtain the number of clusters contained on the record [RCNUM(1-15)]. Then for each cluster, the 12 bytes representing the cluster name (6 bytes) and dot name (6 bytes) used in labeling the cluster are copied into the array CNAME. PRINT8 is called via entry PRINTC to print a message containing the parameter RCNUM. When CNUM sets of data are written into CNAME, CLUST8 calls subroutine RITE8 for output.

3.3.6.7 Flow Chart

The flow diagram for subroutine CLUST8 is given in figure 13.

3.3.6.8 Listing

The listing for this subroutine is given figure 14.

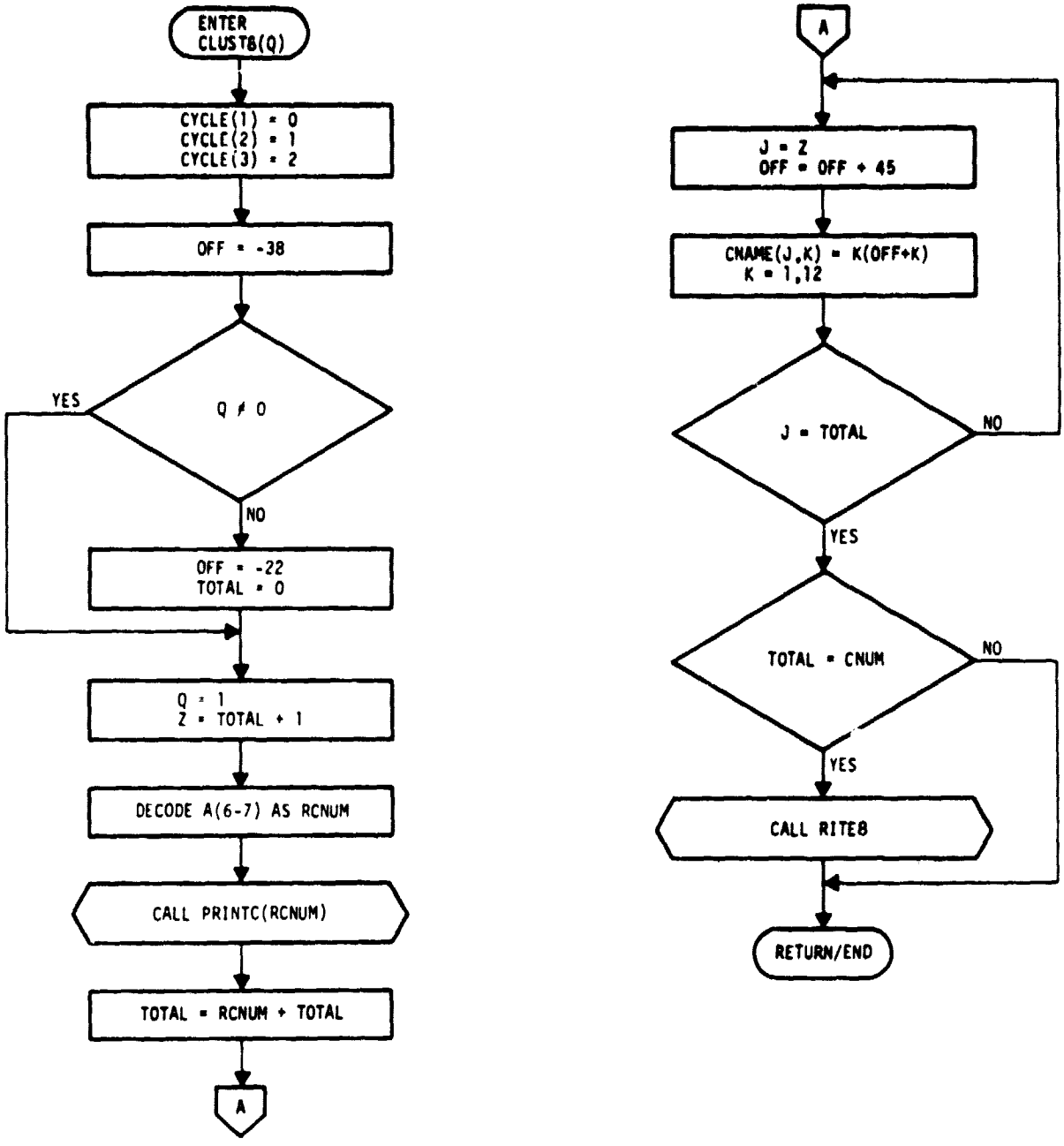

Figure 13.– Flow diagram for subroutine CLUST8.

```
FORTRAN IV-PLUS V02-51         13:52:34    05-JAN-79          PAGE 1
CLUST8.FTN        /TR:BLOCKS/WR

0001             SUBROUTINE CLUST8(O)
0002             IMPLICIT INTEGER(A-Z)
           C     READS CCIT 100 RECORDS AND OUTPUTS CLUSTER LABEL AND CLUSTER-
           C     ID BATCH DATA TO A DISC FILE NAMED SSSSYYDDD.CLO.
0003             BYTE A(720),CYCLE(3),CNAME(60,12)
           C**************************************************************
           C     CNAME IS A STORAGE ARRAY TO PERMIT WRITING THE DATA IN ONE
           C     RECORD TO DISC IN A FILE NAMED SSSSYYDDD.CLO
           C     CNUM IS THE TOTAL NUMBER OF CLUSTERS
           C     OFF IS THE BYTE OFFSET WITHIN A 'C' RECORD
           C     RCNUM IS THE NO. OF CLUSTERS IN THE CURRENT RECORD
           C**************************************************************
0004             COMMON/BUF/A
0005             COMMON/CLUST8/CNAME,CNUM
0006             DATA CYCLE/1H0,1H1,1H2/
0007             OFF = -38
0008             IF(O.LE.0) GO TO 5
0009             OFF = -22
0010             TOTAL = 0
0011       5     O = 1
0012             Z = TOTAL + 1
0013             DECODE(2,100,A(6))RCNUM
0014             CALL PRINTC(RCNUM)
0015             TOTAL = RCNUM + TOTAL
0016             DO 1 J=Z,TOTAL
0017             OFF = OFF + 45
0018             DO 1 K=1,12
0019       1     CNAME(J,K) = A(OFF+K)
           C
           C     CHECK IF DONE
           C
           C     CALL RITEEN FOR DISK WRITE AND PRINTED REPORT
           C
0020             IF(TOTAL.EQ.CNUM) CALL RITEE
0021             RETURN
0022       100   FORMAT(I2)
0023             END
```

Figure 14.– Listing for Subroutine CLUST8.

3.3.7 SUBROUTINE RITE8

3.3.7.1 Linkage

Subroutine RITE8 is called once by subroutine CLUST8.

3.3.7.2 Interface

Subroutine RITE8 interfaces with CLUST8 via COMMON block CLUSTR (see section 3.3.1.2.4).

3.3.7.3 Input

The input to RITE8 is cluster name and cluster name.

3.3.7.4 Output

RITE8 writes two records onto a previously opened file (unit 3). This unit is opened in subroutine BIASC8 as an unformatted FORTRAN disk file. In addition, RITE8 writes a line printer report of the cluster-dot match data for AA evaluation.

3.3.7.5 Storage

This subroutine requires 488 words of storage.

3.3.7.6 Description

RITE8 writes two records onto unit 3. The first record is a single integer, CNUM. The second record consists of the array CNAME as CNUM 12-byte elements. The output file is closed via a call to the system routine CLOSE.

3.3.7.7 Flow Chart

The flow diagram for subroutine RITE8 is given in figure 15.

3.3.7.8 Listing

The listing for this subroutine is given in figure 16.

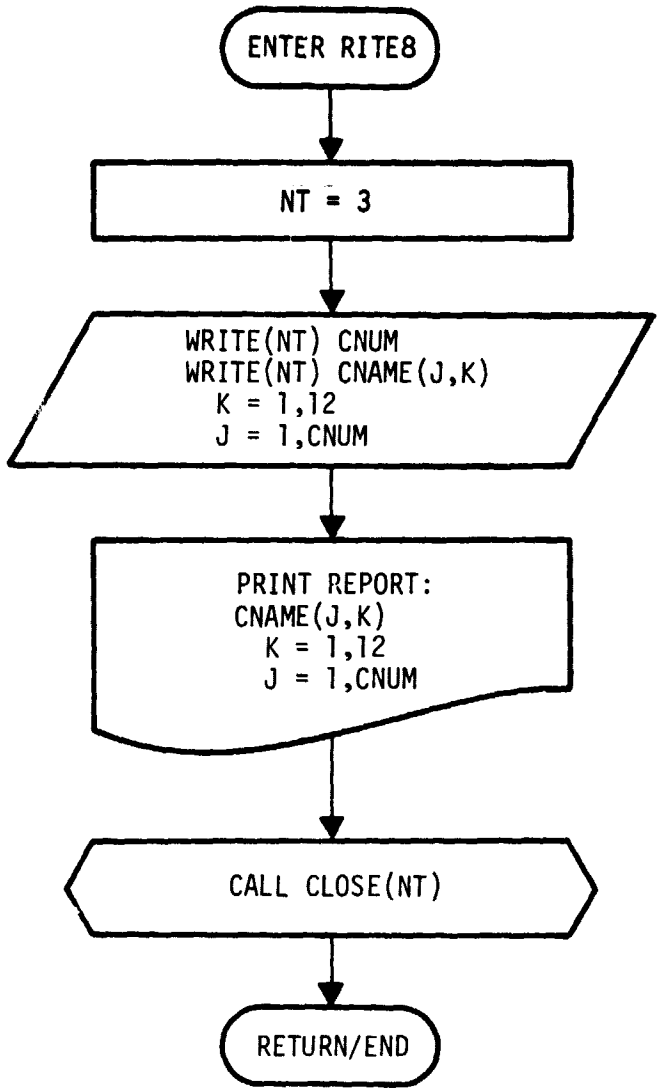

Figure 15.— Flow diagram for subroutine RITE8.

```
FORTRAN IV-PLUS V02-51          13153108    05-JAN-79          PAGE 1
RITE8.FTN       /FORMAT:L/-R

0001            SUBROUTINE RITE8
0002            IMPLICIT INTEGER(A-Z)
                C       WRITES AN UNFORMATTED FILE OF CLUSTER NAME AND CLUSTER # AT DBT MATCH
                C       DATA OUTP DISC FOR AA PROCESSING...LACIE8 VERSION
                C       ALSO PRINTS REPORT FOR AA INTERNAL USE
                C       OUTPUT WAS ADDED IN SUBROUTINE BIASC
0003            BYTE CNAME(61,12)
0004            COMMON/CLUSTR/CNAME,CNUM
0005            CNT = 3
                C       WRITE TWO RECORDS,  FIRST,THE NUMBER OF CLUSTERS, THEN, THE ALPHANUMERIC
                C       DATA...12 CHARACTERS PER CLUSTER
0006            WRITE(1'')CNUM
0007            WRITE(1'')((CNAME(J,K),K=1,12),J=1,CNUM)
0008            WRITE(6,100)((CNAME(J,K),K=1,12),J=1,CNUM)
                C       CLOSE OUTPUT FILE
0009            CALL CLOSE(1'')
0010            RETURN
0011      100   FORMAT(1HC,15X,'CLUSTER NAME     DBT NAME',//,
                1     5(15X,6A1,10X,6A1,/))
0012            END
```


Figure 16.— Listing for Subroutine RITE8.

3.3.8 SUBROUTINE TURNON

3.3.8.1 Linkage

Subroutine TURNON is called by subroutines READH, BIASC8, and DOTS8.

3.3.8.2 Interface

TURNON interfaces with its calling routines via COMMON block NAME (see section 3.3.2.2.1) and a passed parameter, NT.

3.3.8.3 Input

The input to TURNON is unit number and file name.

3.3.8.4 Output

TURNON has no output.

3.3.8.5 Storage

This subroutine requires 171 words of storage.

3.3.8.6 Description

TURNON opens a file with the file name contained in byte array NM. If NT equals 1, the input file is opened as UNIT equals 1. If NT equals 2 to 6, an unformatted file is opened as unit NT. If NT is greater than 6, a formatted file is opened as unit (NT - 6). Prior to opening the file, the routine prints a message containing the passed unit number parameter, NT, and the file name, NM.

3.3.8.7 Flow Chart

The flow diagram for subroutine TURNON is given in figure 17.

3.3.8.8 Listing

The listing for this subroutine is given in figure 18.

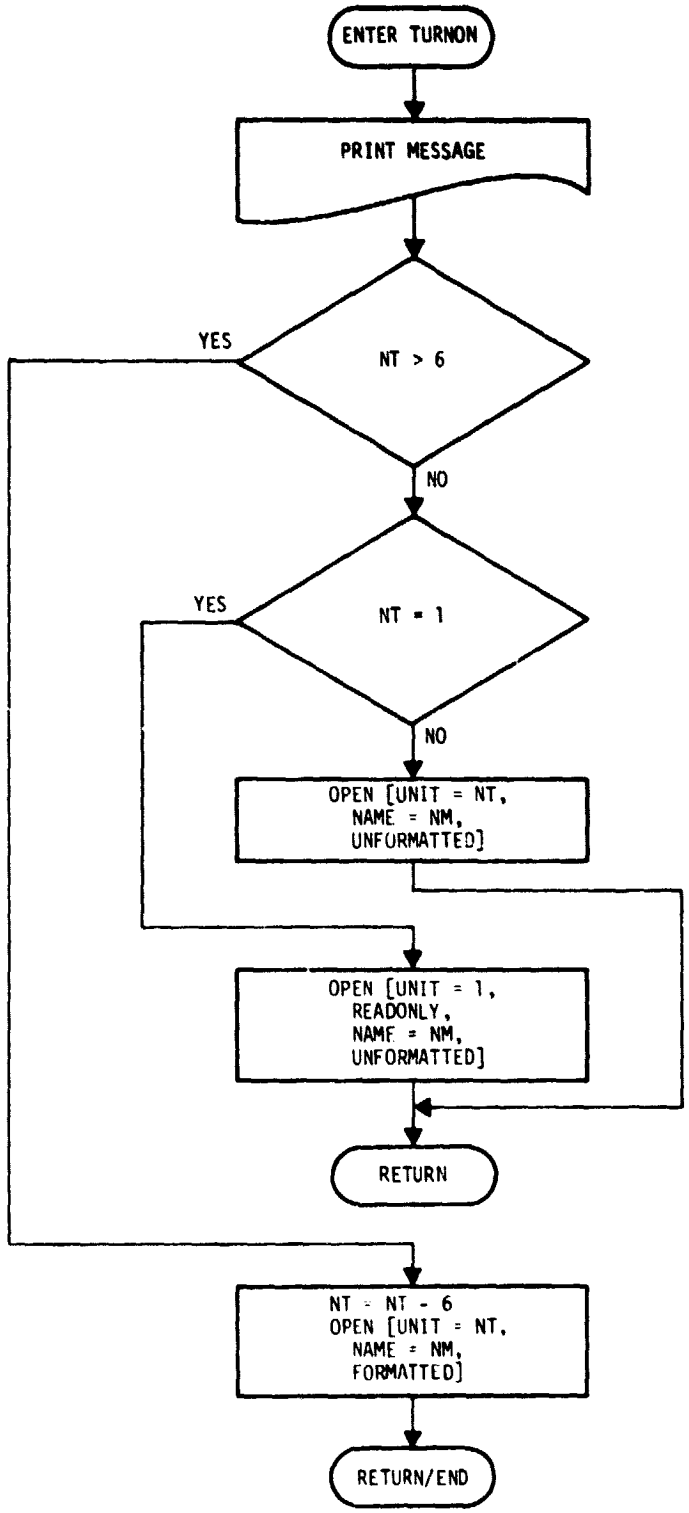

Figure 17.– Flow diagram for subroutine TURNON.

ORIGINAL PAGE IS
OF POOR QUALITY

```
FORTRAN IV-PLUS V02-51          07148130     05-MAY-78          PAGE 1
TURNON.FTN        /TR:BLOCKS/WR

              SUBROUTINE TURNON(NT)
              IMPLICIT INTEGER(A-Z)
        C**************************************************************
        C     ROUTINE OPENS DIRECT ACCESS FILES FOR PROGRAM CCITGA
        C     IF UNIT NUMBER PASSED (NT) IS >6, OPEN A FORMATTED FILE
        C     ELSE OPEN AN UNFORMATTED FILE
        C     USE UNIT 1 FOR A FILE WHICH MUST BE 'OLD'
        C**************************************************************
              BYTE NM(25)
              COMMON/NAME/NM
              PRINT 100,NT,(NM(N),N=1,24)
        100   FORMAT(1H0,'  OPENING UNIT ',I1,' ON DISC TO ACCESS FILE '
             *,24A1,//)
              IF(NT.GT.6) GO TO 2
              IF(NT.EQ.1) OPEN(UNIT=NT,NAME=NM,READONLY,
             * TYPE='OLD',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
              IF(NT.EQ.1) RETURN
              OPEN(UNIT=NT,NAME=NM,TYPE='UNKNOWN',ACCESS='SEQUENTIAL',
             1 FORM='UNFORMATTED')
              RETURN
        2     NT = NT - 6
              OPEN(UNIT=NT,NAME=NM,TYPE='UNKNOWN',ACCESS='SEQUENTIAL',
             1 FORM='FORMATTED',CARRIAGE CONTROL='FORTRAN')
              RETURN
              END
```

Figure 18.— Listing for subroutine TURNON.

3.3.9 SUBROUTINE DOTS8

3.3.9.1 Linkage

Subroutine DOTS8 is called by CCIT8 once via the main entry and 14 times via entry DOTS1. DOTS8 calls subroutine TURNON twice and subroutines PRINT8 (via entry PRINTD) and STCOD8 once.

3.3.9.2 Interface

Subroutine DOTS8 interfaces with TURNON via COMMON block NAME (see section 3.3.2.2.1) and with CCIT8 via COMMON blocks BUF (see section 3.3.1.2.1) and DOTS (see section 3.3.4.2.1).

3.3.9.3 Input

The input is dot label information from the CCIT 'D' records.

3.3.9.4 Output

Subroutine DOTS8 writes formatted, card-image records onto two disk-based output files opened on the initial call to the routine.

3.3.9.5 Storage

This subroutine requires 1052 words of storage.

3.3.9.6 Description

DOTS8 processes CCIT 'D' records into two formatted files of analyst-labeled dots. When called as DOTS8, the routine initializes the unit parameters, NT and MT, and the dot counters, KOUNT1 and KOUNT2. Then the elements of the array NM are set to name the file to receive the type 1 analyst-labeled dot data, and TURNON is called to open this file. NM(24) is redefined (1 → 2) to provide the name of the type 2 dot output file, and TURNON is called to open this file. Subroutine STCOD8 is called to obtain the two-byte parameter ST, the alphabetic state code for the segment. Control then returns to CCIT8.

When called as DOTS1, the routine processes one 720-byte 'D' record. For each analyst-labeled dot, one record is written. For type 1 dots, the data are written onto unit 2; for type 2 dots, the data are written onto unit 3. KOUNT1 is incremented for each type 1 dot, and KOUNT2 is incremented for each type 2 dot.

After processing all 209 dots (14 calls from CCIT8), a blank record is written into each output file. Then both output files are closed, and a message listing KOUNT1 and KOUNT2 is printed via a call to the PRINT8 subroutine entry PRINTD.

3.3.9.7 Flow Chart

The flow diagram for subroutine DOTS8 is given in figure 19.

3.3.9.8 Listing

The listing for this subroutine is given in figure 20.

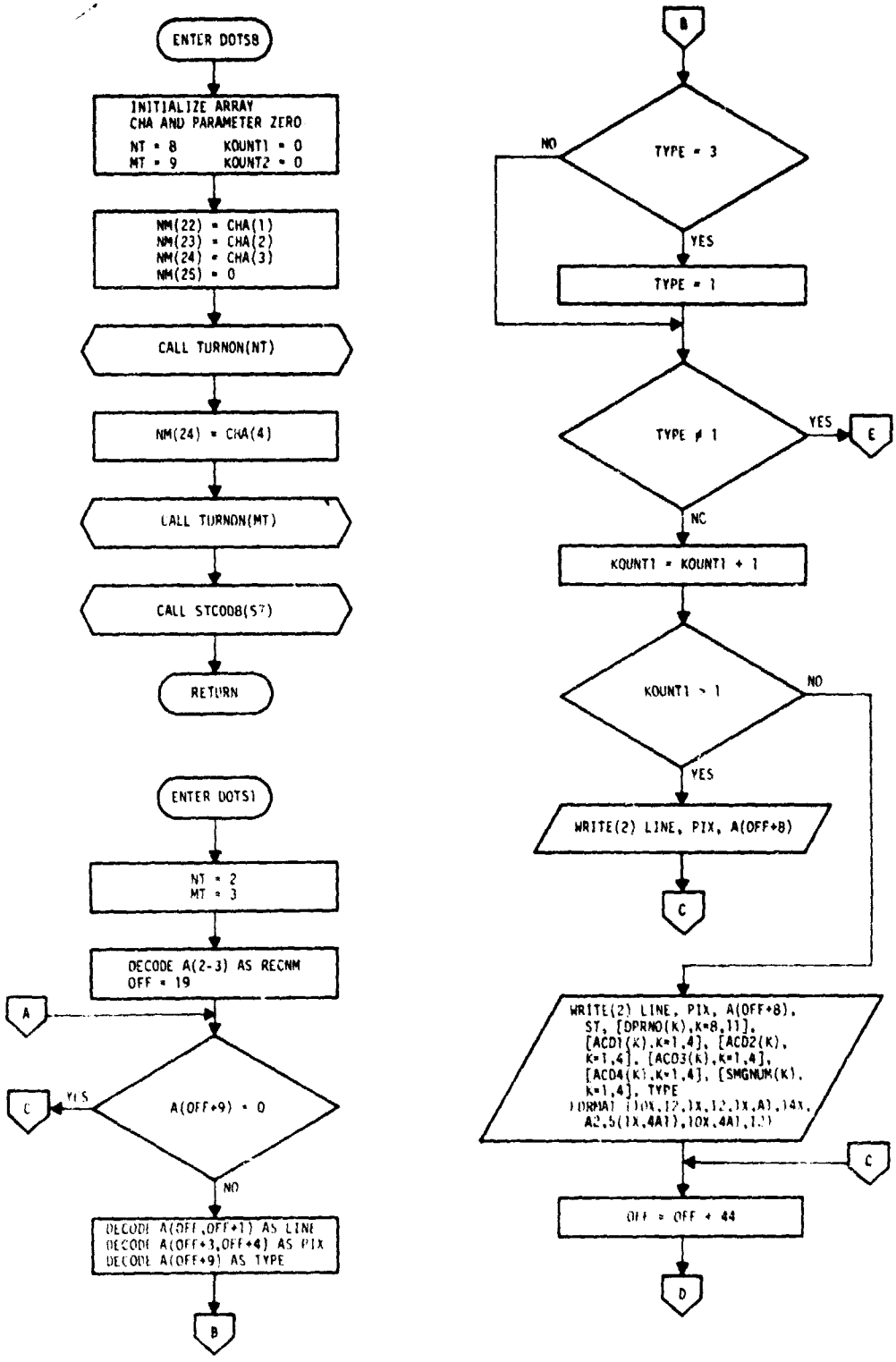

Figure 19.– Flow diagram for subroutine DOTS8.

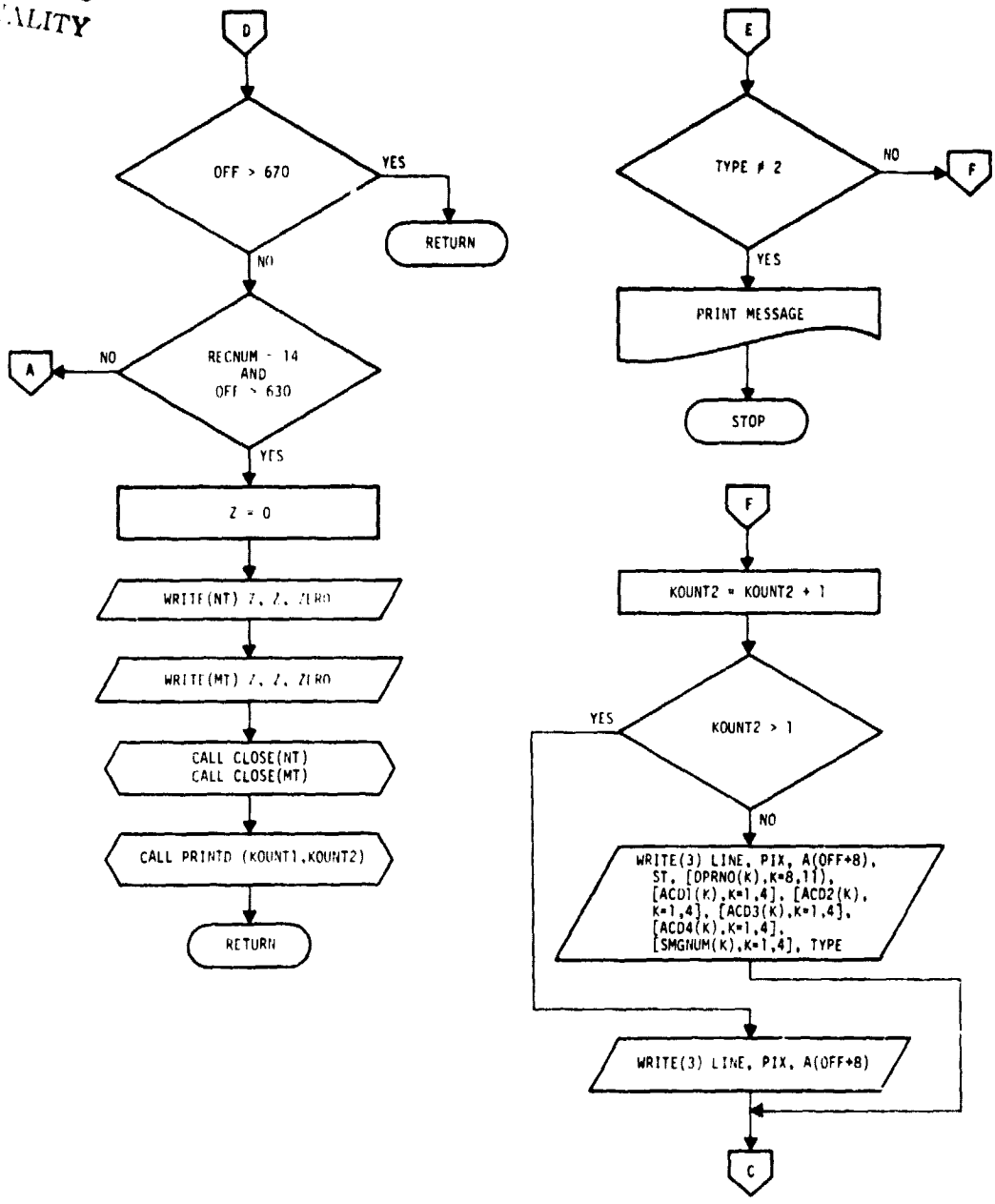

Figure 19.– Concluded.

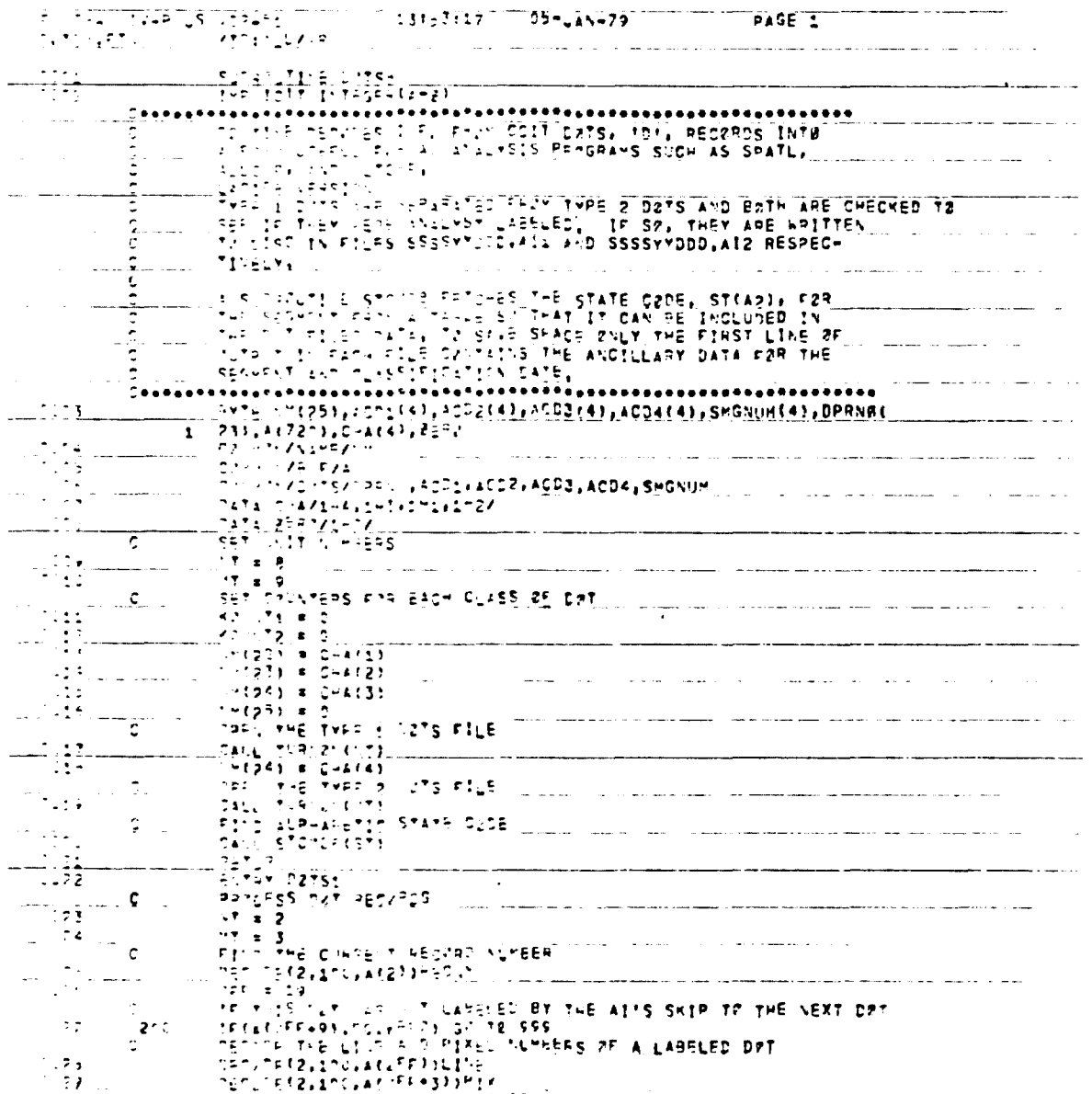

Figure 20.— Listing for Subroutine DOTS8.

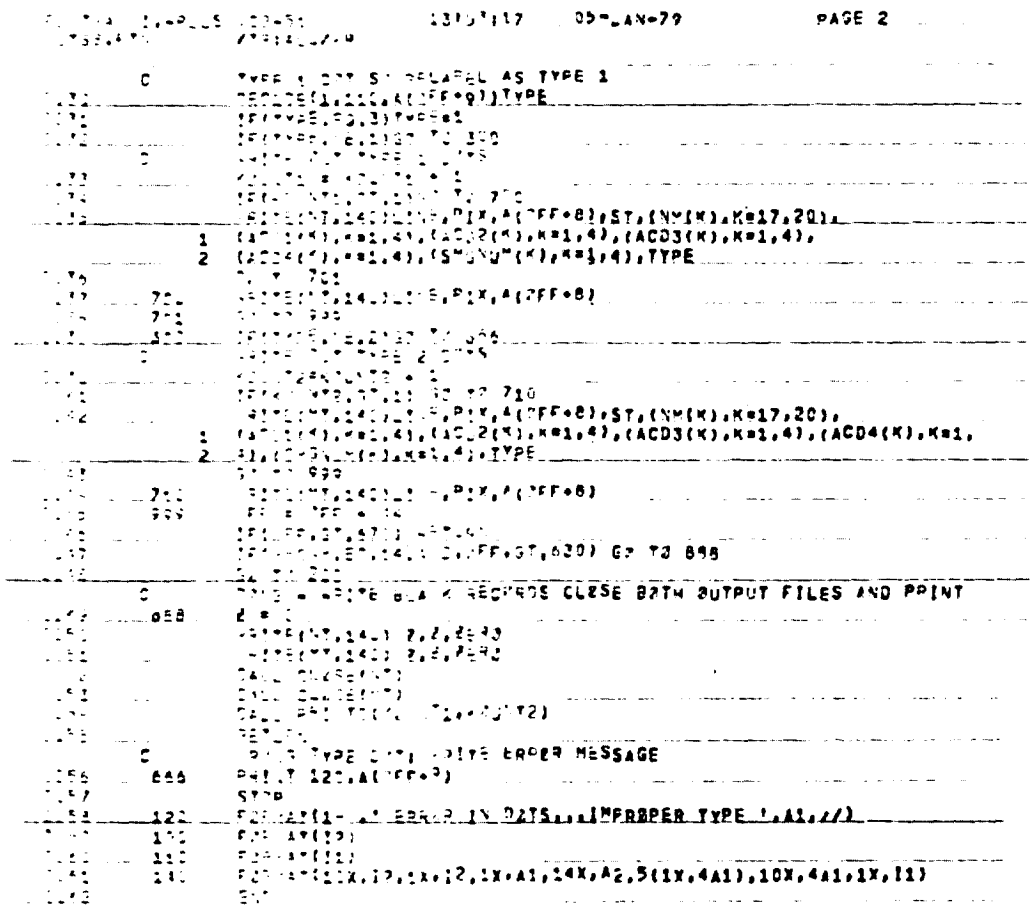

Figure 20.— Concluded.

3.3.10 SUBROUTINE STCOD8

3.3.10.1 Linkage

STCOD8 is called once by subroutine DOTS8.

3.3.10.2 Interface

STCOD8 interfaces with DOTS8 via COMMON block FNAME (see section 3.3.1.2.2) and passed parameter ST.

3.3.10.3 Input

The input to STCOD8 is the segment number.

3.3.10.4 Output

The output of STCOD8 is the two-letter state abbreviation.

3.3.10.5 Storage

This subroutine requires 1108 words of storage.

3.3.10.6 Description

STCOD8 locates the correct two-character alphabetic state code, ST, for a given segment number via table lookup. Note: The table given is only valid for AA LACIE U.S. Great Plains blind sites for the 1978 Transition Year.

3.3.10.7 Flow Chart

The flow diagram for subroutine STCOD8 is given in figure 21.

3.3.10.8 Listing

The listing for this subroutine is given in figure 22.

KG

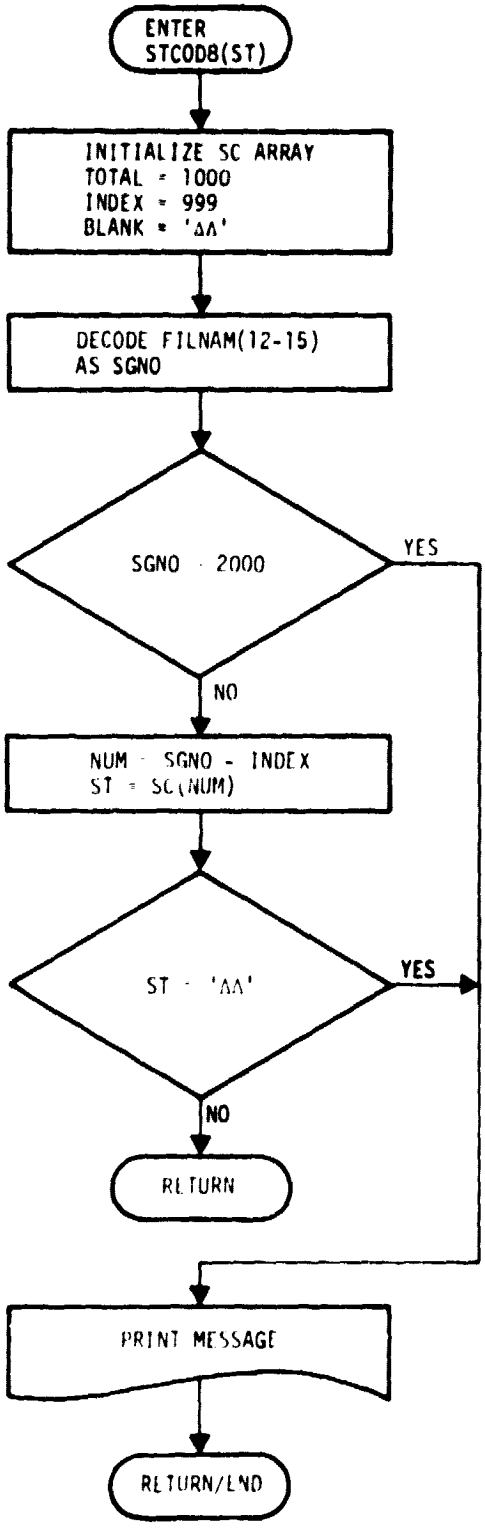

Figure 21.- Flow diagram for subroutine STCOD8.

Figure 22.— Listing for Subroutine STC008.

3.3.11 SUBROUTINE PRINT8

3.3.11.1 Linkage

PRINT8 is called by CCIT8 via entries PRINT8 and PRINTE, by HEADER via entry PRINTH, by DOTS via entry PRINTD, and by CLUST8 via entry PRINTC. All other called routines are Image Processor system routines.

3.3.11.2 Interface

PRINT8 interfaces with HEADER via COMMON block DOTS (see section 3.3.4.2.1), with CCIT8 via COMMON block FNAME (see section 3.3.1.2.2), with DOTS8 via passed parameters K1 and K2, and with CLUST8 via passed parameter RCNUM.

3.3.11.3 Input

The input to PRINT8 is the information to be printed and is passed to it via the various common blocks.

3.3.11.4 Output

PRINT8 prints reports on the line printer.

3.3.11.5 Storage

This subroutine requires 802 words of storage.

3.3.11.6 Description

PRINT8 provides most line printer reports for the CCIT8 processor. These reports provide processing records for the AA status and tracking activity. The routine uses system routines TIME and DATE to obtain data for header and trailer line printer messages for each run.

3.3.11.7 Flow Chart

The flow diagram for subroutine PRINT8 is given in figure 23.

3.3.11.8 Listing

The program listing for this subroutine is given in figure 24.

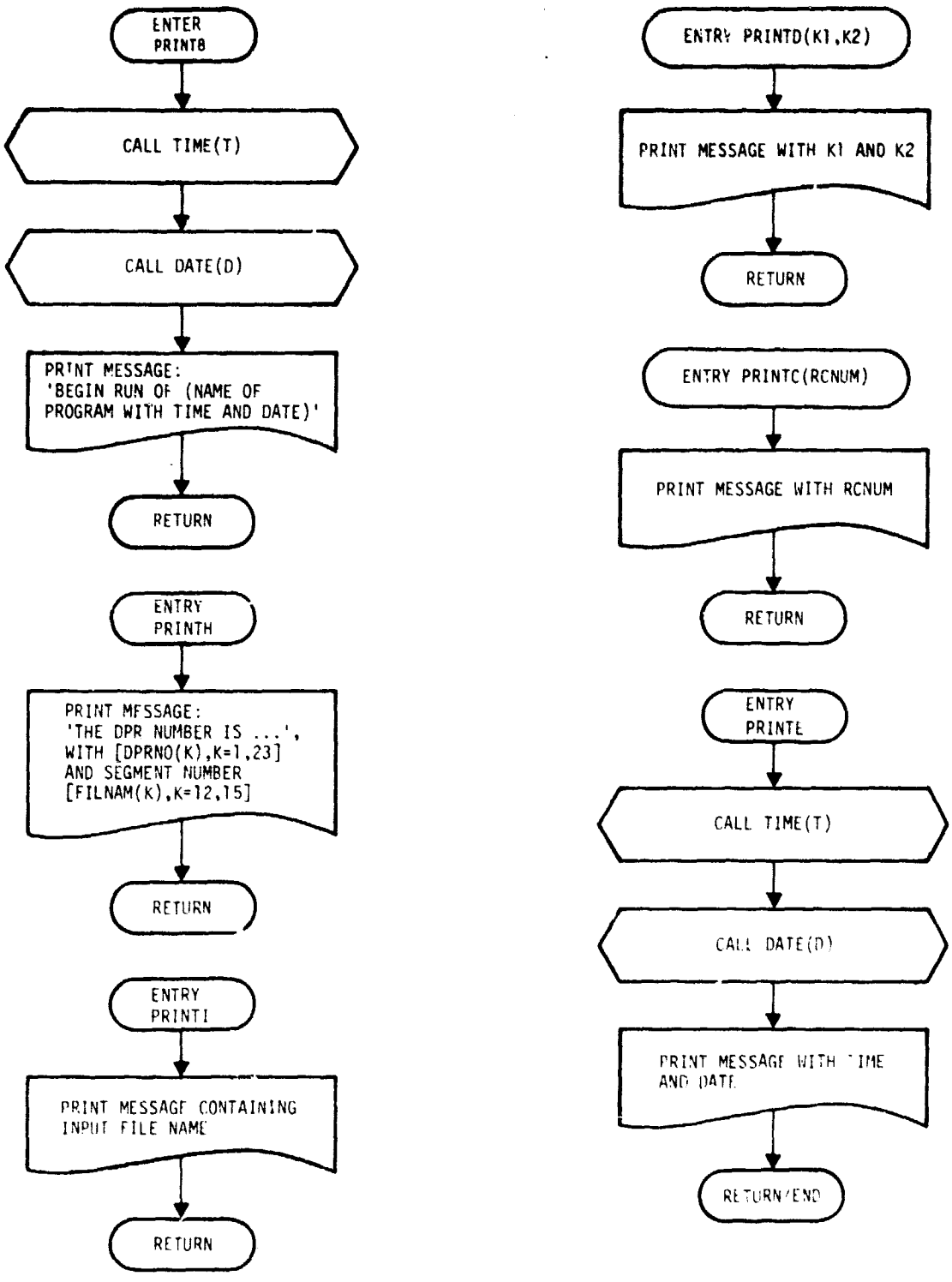

Figure 23.-- Flow diagram for subroutine PRINT8.

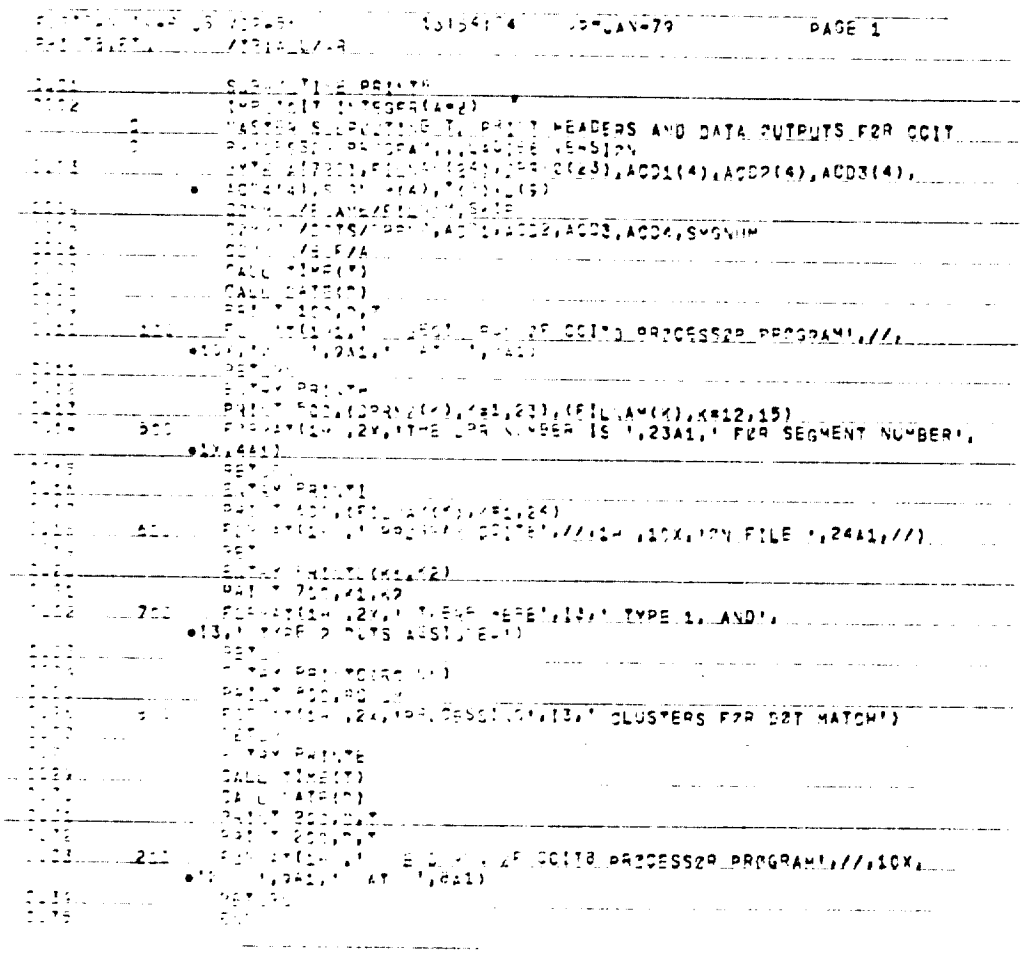

Figure 24.— Listing for Subroutine PRINT8.

4. OPERATIONS

This section presents all the information necessary to obtain proper execution of the CCIT8 processor program.

4.1 OPERATORS GUIDE

This section explains the system hardware configuration and execution (run) setup for the CCIT8.

4.1.1 HARDWARE CONFIGURATION

The nominal configuration is the Earth Observations Division/Data Techniques Laboratory (EOD/DTL) PDP 11/45 processor with the RSX 11-D operating system. The system must have the input CCIT files resident on either the system disk or a user disk. The output files are written onto the same disk and under the same user identification code (UIC) as the resident input data. The input files are created using program AACCIT, described in JSC-13893. (See section 2 of this specification.)

4.1.2 PROGRAM EXECUTION

4.1.2.1 INTERACTIVE SETUP

a. Edit file CCIT8.DAT for the proper file name and the value of parameter SKIP (24A1,I2). The file name takes the form:

DBX:[abc,d]SSSSYYDDD.wxy

where

X = Disk unit number

SSSSYYDDD = Input file name

wxy = Input file type; i.e., .CCO

[abc,d] = UIC for the input file

b. Mount the proper disk pack on the drive.

c. Type 'RUN CCIT8'.

d. When message CCIT8-STOP appears on the monitor, collect a single-page report at the line printer, and check the listing to ensure that the ending message was printed and that the various steps were properly executed.

4.1.2.2 BATCH SETUP

a. Prepare a batch run request detailing the disk configuration required.

b. Set up a batch run deck as in table 2. The required steps follow:

- Delete CCIT8.DAT.
- Create CCIT8.DAT with card images, as given in section 4.1.2.1
- Run CCIT8.TSK.

4.2 USERS GUIDE

The CCIT8 program is designed to obtain a small fraction of the data from a CCIT disk file and to reformat these data into a form directly used by several AA software modules. This program will not execute correctly for CCIT's other than those created under LACIE version 8.

4.3 MAINTENANCE DOCUMENTATION

Not applicable.

TABLE 2.— BATCH RUN DECK SETUP

```
$JOB/NAME=AA/MCR/LIMIT=99/ACCOUNT=110 6
$MCR PIP
CCIT8.DAT;*/DE
$CREATE CCIT8.DAT
   :
Card images for file name and SKIP parameter (24A1,I2)
   :
Blank card
$EOD
$MCR REM RSXBAT
$RUN CCIT8.TSK
$EOJ
```

APPENDIX

FORMAT OF .CLO FILE

APPENDIX

FORMAT OF .CLO FILE

Record 1: The first record contains a single integer representing the number of classes detailed in the next record. Sixteen bytes of data follow for each class (minimum of 2 classes, maximum of 26 classes).

Record 2:

Byte number	Data description (ASCII)
1	Class 1 label (W, S, G, N, etc.)
2-6	Pixel population; PPPPP
7-9	Uncorrected proportion; M.MM (implied decimal point)
10-12	Corrected proportion; N.NN (implied decimal point)
13-16	Variance; .VVVV (implied decimal point)
17	Class 2 label (W, S, G, N, etc.)
18-22	Pixel population; PPPPP
23-25	Uncorrected proportion; M.MM (implied decimal point)
26-28	Corrected proportion; N.NN (implied decimal point)
29-32	Variance; .VVVV (implied decimal point)
⋮	⋮

Record 3: This record cortains a single integer giving the number of clusters in the classification, CNUM.

Record 4: The fourth record contains 12 bytes of ASCII character data for each cluster; e.g., 12*CNUM bytes of data. The first 6 bytes of each group of 12 are the cluster label; e.g., NOCL17. The last six bytes of each group are the identity of the dot used to label the clusters; e.g., DOT103. Only type 1 dots are used to label clusters.

58