2.0 FILE DESIGN AND CREATION, by Katharine E.W. Mather

2.1 *DISK FILE STRUCTURE*

Each entry consisted of the Enga word, a number of constant elements (items 1.3.3 through 1.3.12), the English gloss (1.3.13), and up to nineteen variable length items (items 1.3.14 through 1.3.33). Not all of the items existed for each entry. The information pertaining to each dictionary entry was punched on cards for input to the computer. The cards were edited and the information transferred to disk files. The structure of the files was designed to minimise the storage required and the access time during retrieval of information.

2.1.1 Variable Length Data

For ease and speed of retrieval, direct access was used. Since direct access files on the 360/50 must contain fixed length records, nineteen files (each having a different record length) were used to store the variable length items and the English gloss. The record sizes ranged from 40 to 760 characters, in multiples of 40 characters. A file containing records 800 characters long was also established, but has not yet been required. Each item was stored in the shortest record length possible and assigned to the file which contained records of that length. The record length file format is shown in 2.1.7.

2.1.2 Fixed Length Data

The fixed length information relating to each Enga item was stored in a master file, which contained one record per Enga word. Each record in the master file contained an Enga item, the related fixed length information, and the addresses of the storage locations of the English gloss and any items associated with the Enga item. The address area for items which did not exist was set to zero. The master file record format is shown in 2.1.6.

Mather, K.E.W. "File Design and Creation". In Lang, A., Mather, K.E.W. and Rose, M.L. editors, *Information storage and retrieval: A dictionary project*. **D-8:19-64**. Pacific Linguistics, The Australian National University, 1972. DOI:*10.15144/PL-D8.19*

Since the dictionary file was to be sorted, space was left in the master file record format for the address of the next word in the sorted order. After sorting, these addresses were written into the file (see 4.1). Thus, the file could be accessed both sequentially and in alphabetical order.

2.1.3 Record Addresses

Each entry was given a unique sequential identification number which served as its address in the master file. Records in the other nineteen files were stored sequentially as they occurred. At the beginning of each run, control cards were read, which contained counts of the number of records stored in each of the files. These counts were updated during the run, the new record count serving as the address for the new record. The control cards for the succeeding run were punched at the end of processing.

2.1.4 Record Retrieval

This organisation of the file made it possible to access directly any part of the information associated with each dictionary entry. During nearly all retrieval operations, the master only needed to be read for all entries. Only after an entry had satisfied the retrieval criteria were any associated items required for the printout read from the relevant files. Thus unnecessary data transmission was avoided.

2.1.5 File Sizes

The lengths of the twenty files used were as follows. File number seven was the master file. The other files contained variable length information relating to the main Enga entries.

File No.	*No. of characters*	*No. of words*
7	265	
13	40	10
14	80	20
15	120	30
16	160	40
17	200	50
18	240	60
19	280	70
20	320	80
21	360	90

(continued)

File No.	*No. of characters*	*No. of words*
22	400	100
23	440	110
24	480	120
25	520	130
26	560	140
27	600	150
28	640	160
29	660	170
30	720	180
31	760	190

2.1.6 Master File Record Format

Character positions	*No. of characters*	*Contents*	
1-5	5	Word No.	
6	1	Card No. (=1)	
7-11	5	Address of next word	
12-36	25	Enga word	
37-38	2	Verb for Noun (exist. v.)	
39	1	?	
40-43	4	Codes (semantic domain)	
44-53	10	Tone code	
54-83	30	Source	
84-93	10	Thesaurus	
94-100	7	Code-Pt. of speech	
101-102	2	Dialect	
103-125	23	Loan word	
126-127	2	File No.	02 English
128-132	5	Address	
133-134	2	File No.	03 Synonyms
135-139	5	Address	
140-141	2	File No.	04 Class inclusion
142-146	5	Address	

(continued)

Character positions	*No. of characters*	*Contents*		
147-148	2	File No.	05	Attributive
149-153	5	Address		
154-155	2	File No.	06	Function
156-160	5	Address		
161-162	2	File No.	07	Contingency
163-167	5	Address		
168-169	2	File No.	08	Spatial
170-174	5	Address		
175-176	2	File No.	09	Operational
177-181	5	Address		
182-183	2	File No.	10	Comparison
184-188	5	Address		
189-190	2	File No.	11	Exemplification
191-195	5	Address		
196-197	2	File No.	12	Provenience
198-202	5	Address		
202-204	2	File No.	13	Grading
205-209	5	Address		
210-211	2	File No.	14	Time when
212-216	5	Address		
217-218	2	File No.	15	Explicative (why)
219-223	5	Address		
224-225	2	File No.	16	Constituent
226-230	5	Address		
231-232	2	File No.	17	Circularity
233-237	5	Address		
238-239	2	File No.	18	Antonyms, Ost. Def.
240-244	5	Address		
245-246	2	File No.	19	How, When do it
247-251	5	Address		
252-253	2	File No.	20	Comments of inform.
254-258	5	Address		
259-260	2	File No.	21	Quotation & source
261-265	5	Address		

2.1.7 Record Length File Format

Character position	*No. of characters*	*Contents*
1-6	6	Word number
7-8	2	Card number
9+		Record

This format was used for all records stored in the recorded length files, except those containing an English gloss. In these, positions 9-16 were left blank. It was originally intended to sort the file on the English gloss and store the address of the next record in sort order within the record (in characters 9-16), as was done with the Enga sort (4.1). However, this was not possible, since some items had more than one English gloss. A special program had to be written for the English sort and English-Enga print (see 4.3 below), and the space left in the English gloss records was not used.

2.2 *DISK FILE CREATION*

2.2.1 Card Identification

Twenty-one card types were used; the card format is shown in 2.2.5. The first six columns of each card contained the item identification number. Columns 7 and 8 contained the card numbers which defined the nature of the information on the card.

2.2.2 Contents of Card Types

Card types 1 and 2 contained the Enga item and all the fixed length information (items 1-12). The English gloss (item 13) started in column 41 of card 2, continuing onto one or more cards, if necessary. Card types 3 to 21 contained the variable length items (items 14-33).

2.2.3 Continuation

Card types 2 through 21 could have up to ten continuation cards, numbered 1 to 10 in columns 79-80. (The data when coded required at most ten continuation cards. The number could be increased to accommodate longer character strings.) An asterisk in column 80 of the first card of a card type indicated that a continuation card should follow. The last card of the card type was blank in columns 79-80.

2.2.4 Input Requirements

The only requirements during input were that the cards for each entry be in ascending numerical order and that card types 1 and 2 must exist. The entries themselves did not need to be ordered, since the word number served as the disk file address. Any number of entries could be entered in the master file during each run.

2.2.5 Card Input Format

Card No.: Columns	*Item No.*	*Content*
All cards: 1-6	1	Entry number
All cards: 7-8		Card number
1: 9-33	2	Enga entry
1: 34-35	3	Existential verb
1: 36	4	Incomplete items
1: 37-40	5	Semantic domain
1: 41-50	6	Source of entry
1: 51-60	8	Notebook and tape
1: 61-70	7	Source of tone
1: 71-80	9	Thesaurus
2: 9-15	10	Grammatical class
2: 16-17	11	Dialect
2: 18-40	12	Language & source (if loan word)
2: 41-76	13	English gloss
All other cards: 80		Continuation asterisk *, numerals
3:	14	Cross References
4:	15	Definitions types: class inclusive
5:	16	Attributive
6:	17	Function
7:	18	Contingency
8:	19	Spatial
9:	20	Operational
10:	21	Comparison
11:	22	Exemplification & Instrumental
12:	23	Provenience & Objective
13:	24	Grading
14:	25	Time & Duration
15:	26	Explicative

(continued)

Card No.: Columns	*Item No.*	*Contents*
16:	27	Subjective
17:	28	List
18: 9-40	29	Antonyms
18: 41-76	30	Ostensive
19:	31	'How do they do it'
20:	32	Miscellaneous Information
21:	33	Illustrative Quotation/ Citation

2.2.6 Description of Creation Program

At the beginning of each run, a number of initialisation procedures were carried out, which included reading and printing control cards containing counts of the number of records stored in each of the record length files and counts of the number of records for which space had been allowed in each file.

The data cards were read one at a time and edited for sequence within each dictionary item. The fixed length information on cards 1 and 2 was stored as it was read in to be written later to the master file. Numeric items were checked for numeric validity and the orthography of the main Enga item was changed, unless the change made the item more than twenty-five characters long (see 1.3.2).

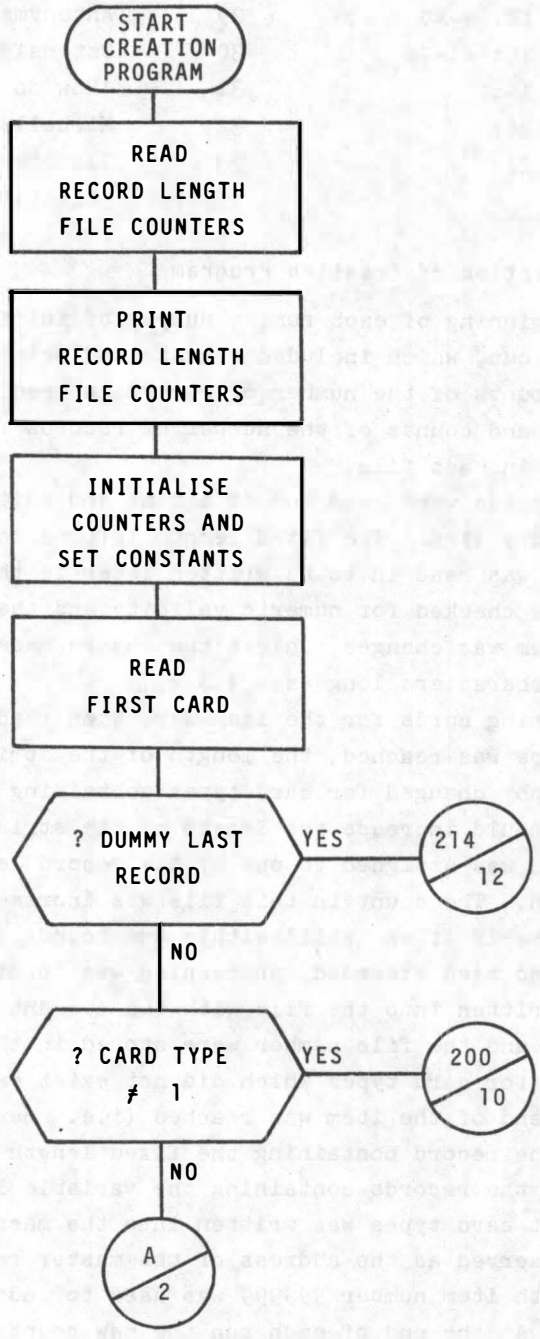
The remaining cards for the item were then read. When the end of each card type was reached, the length of the string was checked and the orthography changed for card types containing Enga (types 3-21); this change could increase the length of the string (record).

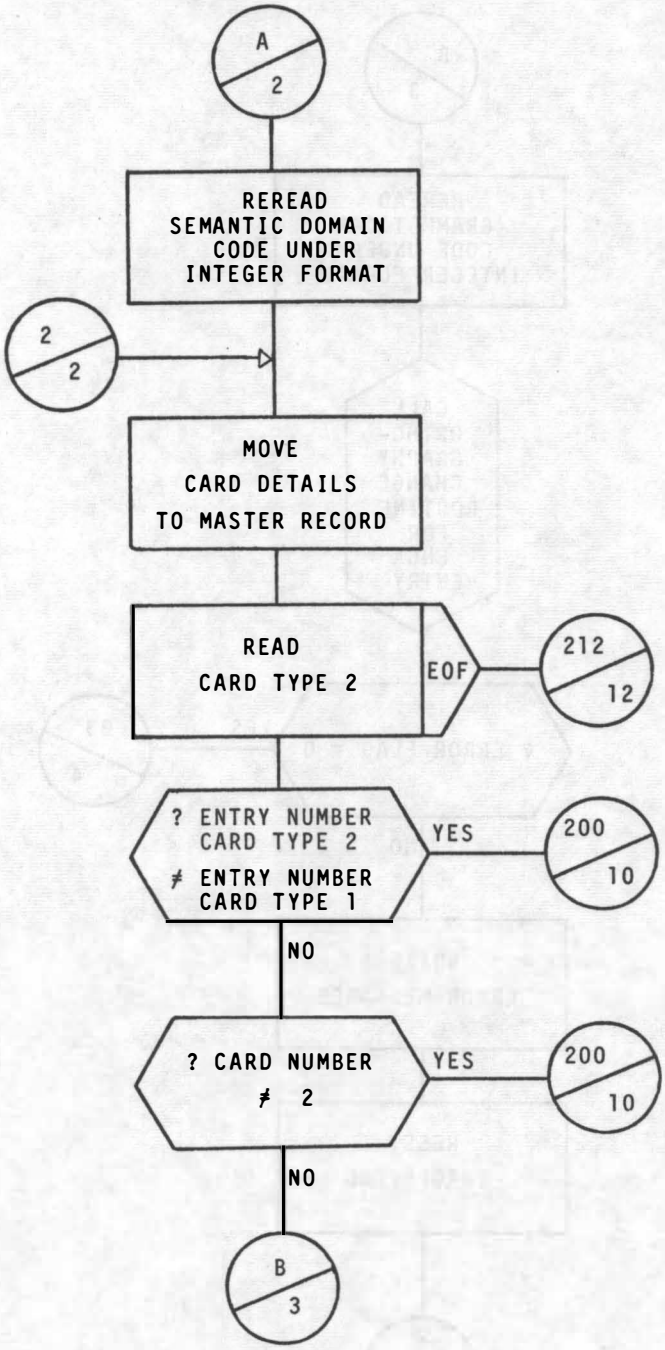
The record was assigned to one of the record length files according to its length. The count in this file was incremented by one and checked to see if it was still within the bounds of the file area. If the bounds had been exceeded, processing was terminated. Otherwise the record was written into the file with the current count as its address; this address and the file number were stored in the master record. The address area for card types which did not exist was set to zero.

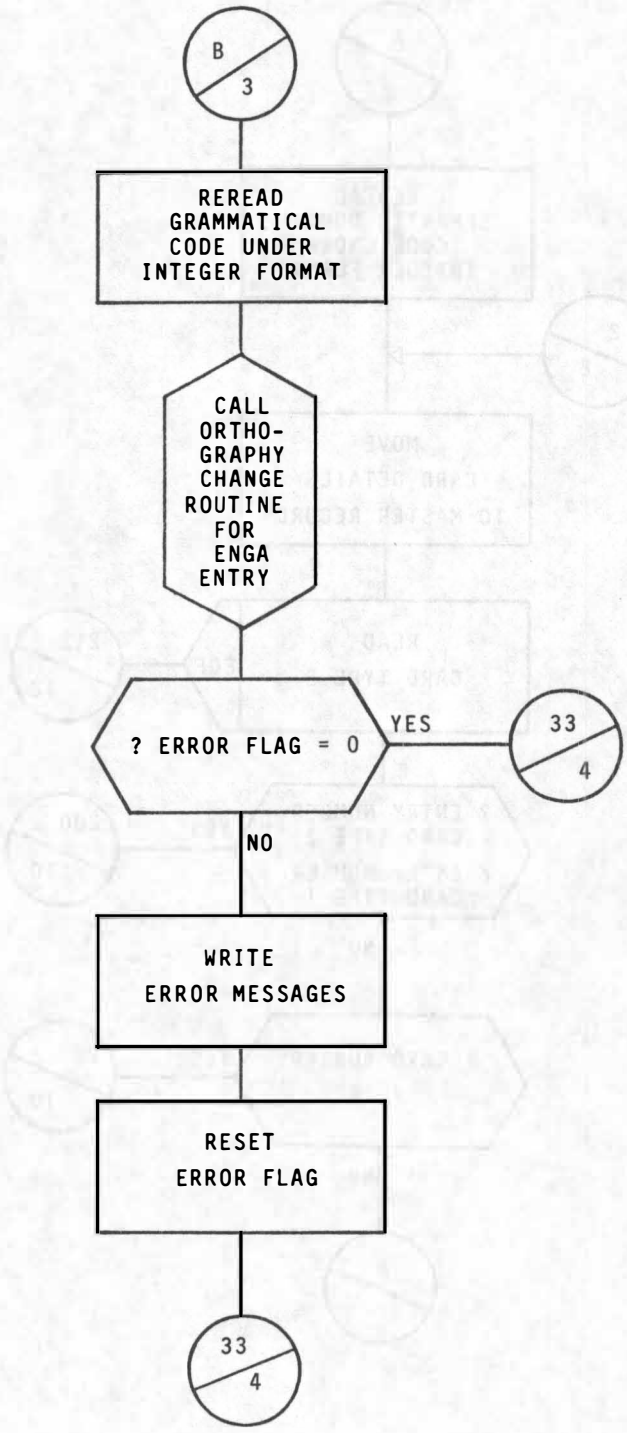
When the end of the item was reached (i.e. when another card type 1 was read), the record containing the fixed length information and the addresses of the records containing the variable length information of the different card types was written into the master file. The number of the item served as the address of the master record.

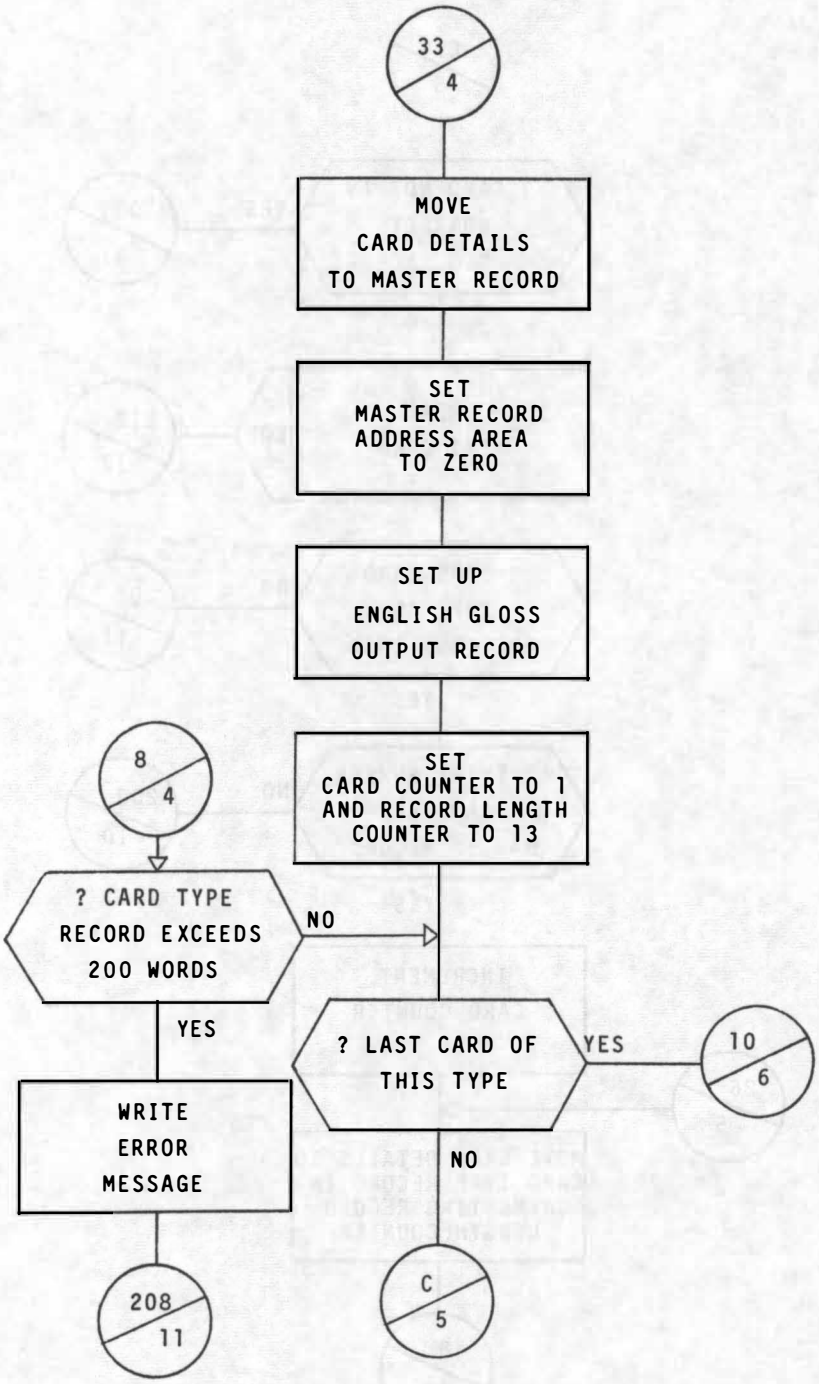
A card with item number 999999 was used to indicate the end of the card input. At the end of each run the new counts for the record length files were printed and also punched to provide the control cards for the following run.

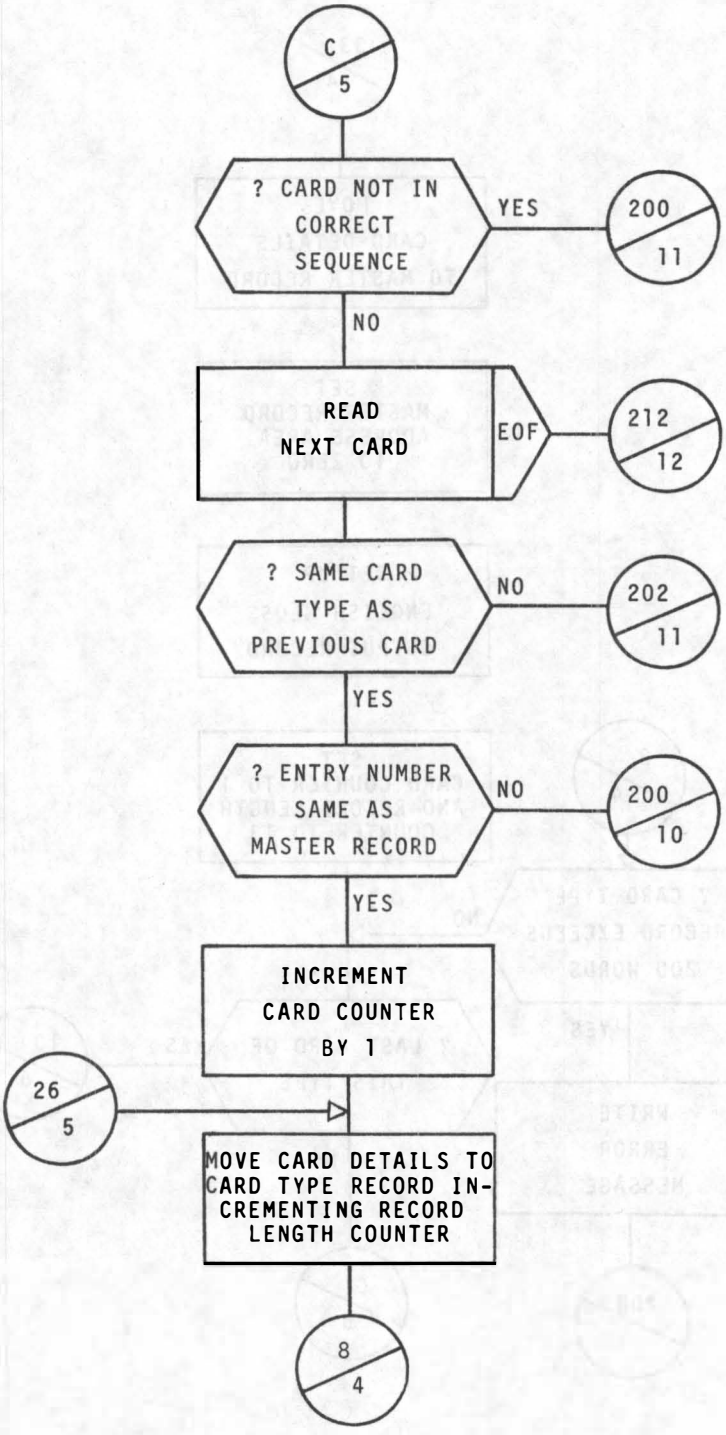
2.2.7 Flowchart

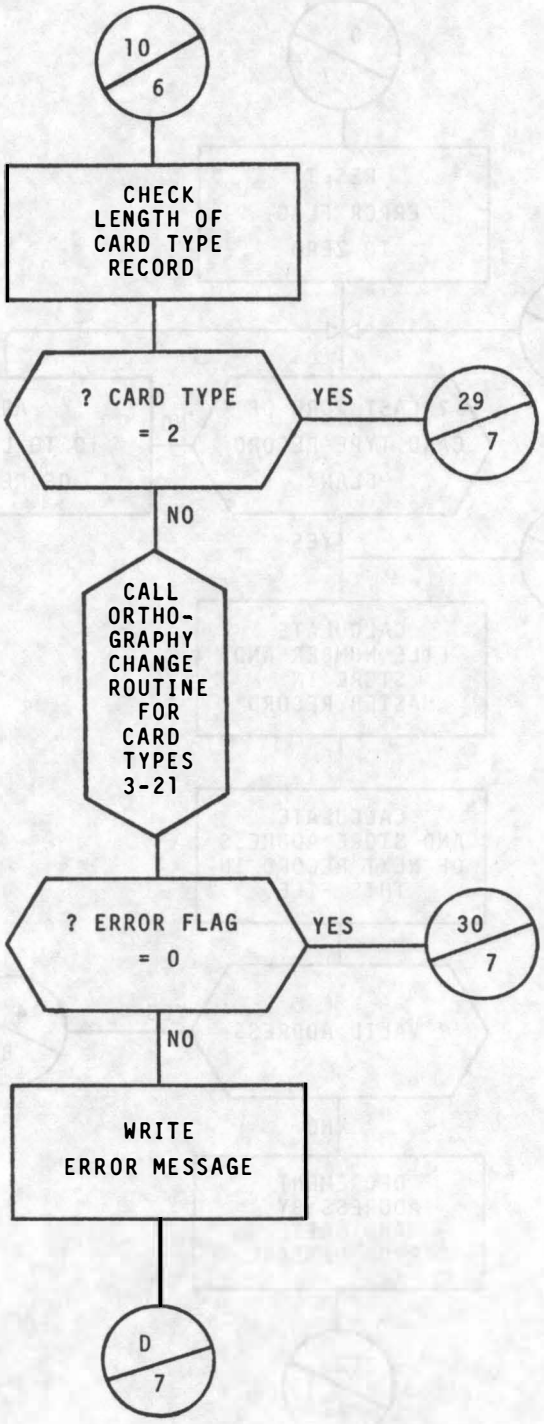


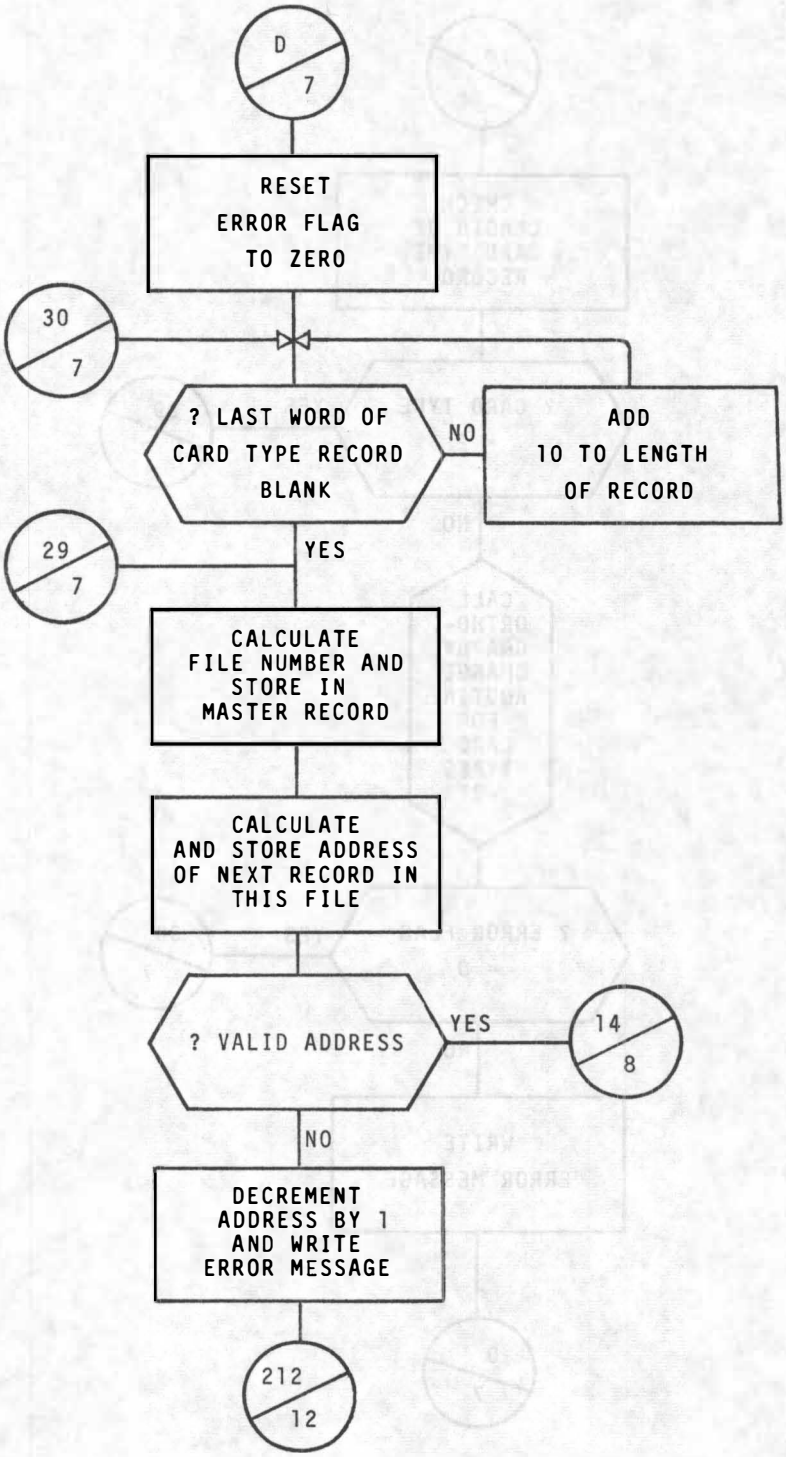

A
2
REREAD
SEMANTIC DOMAIN
CODE UNDER
INTEGER FORMAT
2
2
MOVE
CARD DETAILS
TO MASTER RECORD
READ
CARD TYPE 2
EOF
212
12
? ENTRY NUMBER
CARD TYPE 2
≠ ENTRY NUMBER
CARD TYPE 1
YES
200
10
NO
? CARD NUMBER
≠ 2
YES
200
10
NO
B
3

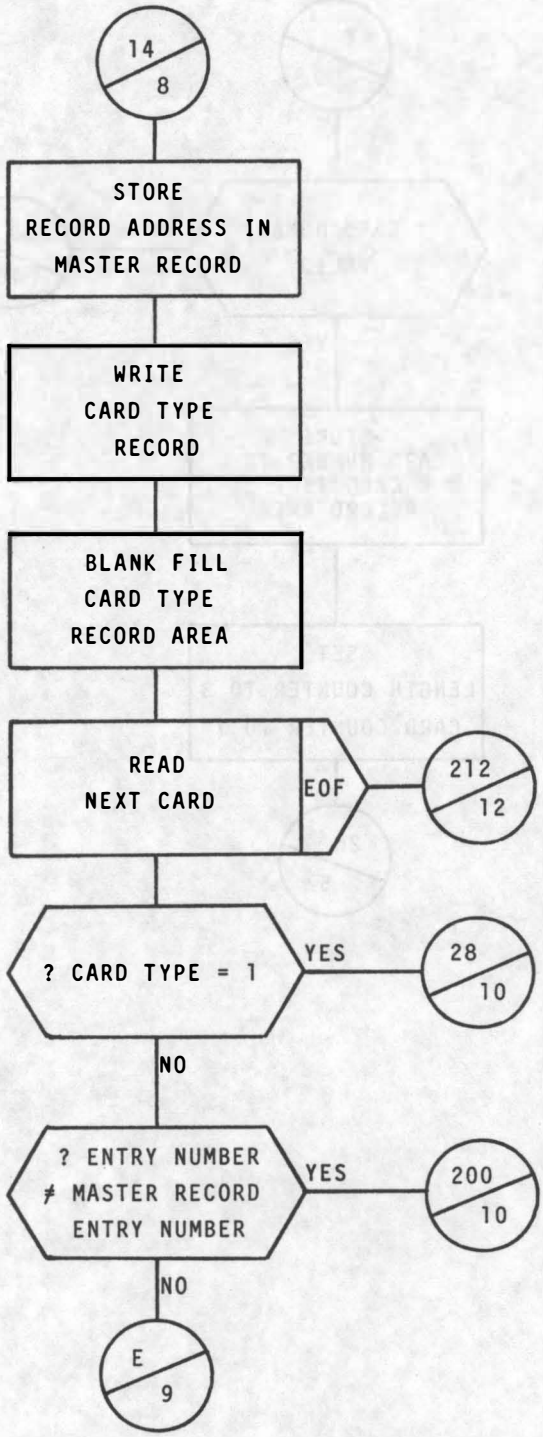

B
3
REREAD GRAMMATICAL CODE UNDER INTEGER FORMAT
CALL ORTHO-GRAPHY CHANGE ROUTINE FOR ENGA ENTRY
? ERROR FLAG = 0
YES
33
4
NO
WRITE ERROR MESSAGES
RESET ERROR FLAG
33
4

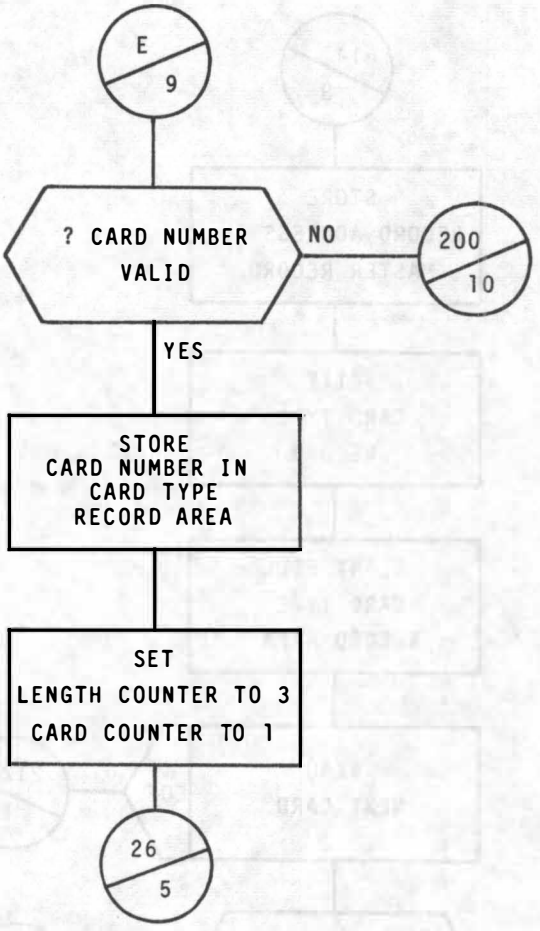

33
4
MOVE
CARD DETAILS
TO MASTER RECORD
SET
MASTER RECORD
ADDRESS AREA
TO ZERO
SET UP
ENGLISH GLOSS
OUTPUT RECORD
SET
CARD COUNTER TO 1
AND RECORD LENGTH
COUNTER TO 13
8
4
? CARD TYPE
RECORD EXCEEDS
200 WORDS
NO
YES
? LAST CARD OF
THIS TYPE
YES
10
6
NO
WRITE
ERROR
MESSAGE
208
11
C
5

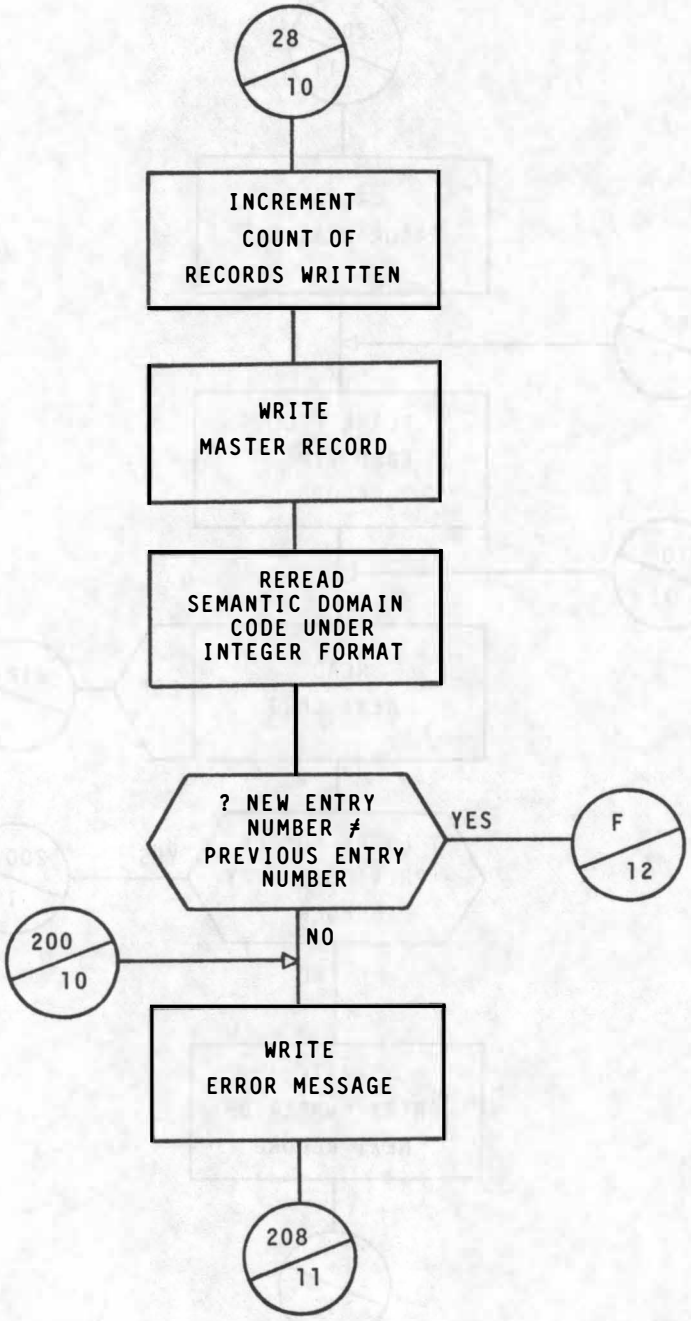

C
5
? CARD NOT IN CORRECT SEQUENCE
YES
200
11
NO
READ NEXT CARD
EOF
212
12
? SAME CARD TYPE AS PREVIOUS CARD
NO
202
11
YES
? ENTRY NUMBER SAME AS MASTER RECORD
NO
200
10
YES
INCREMENT CARD COUNTER BY 1
26
5
MOVE CARD DETAILS TO CARD TYPE RECORD INCREMENTING RECORD LENGTH COUNTER
8
4

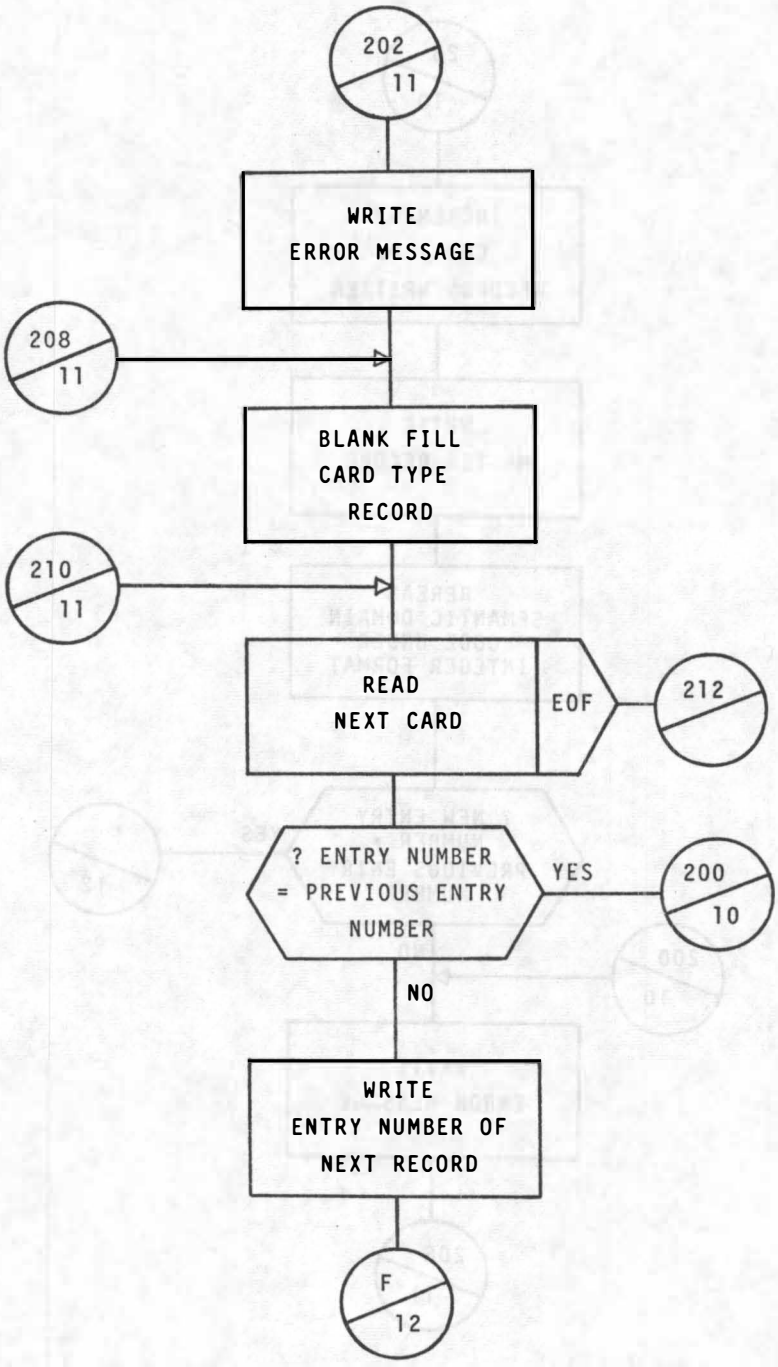

10
6
CHECK LENGTH OF CARD TYPE RECORD
? CARD TYPE = 2
YES
29
7
NO
CALL ORTHO-GRAPHY CHANGE ROUTINE FOR CARD TYPES 3-21
? ERROR FLAG = 0
YES
30
7
NO
WRITE ERROR MESSAGE
D
7

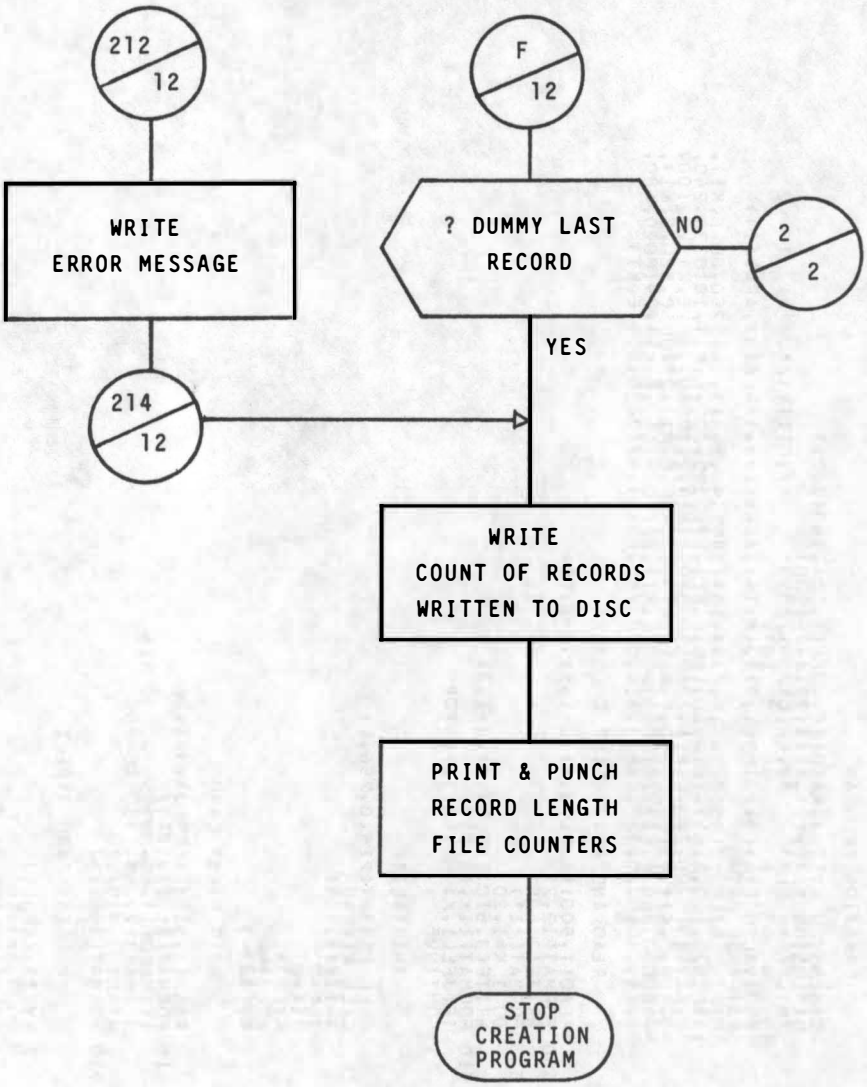

D
7
RESET
ERROR FLAG
TO ZERO
30
7
? LAST WORD OF
CARD TYPE RECORD
BLANK
NO
ADD
10 TO LENGTH
OF RECORD
29
7
YES
CALCULATE
FILE NUMBER AND
STORE IN
MASTER RECORD
CALCULATE
AND STORE ADDRESS
OF NEXT RECORD IN
THIS FILE
? VALID ADDRESS
YES
14
8
NO
DECREMENT
ADDRESS BY 1
AND WRITE
ERROR MESSAGE
212
12


14
8
STORE
RECORD ADDRESS IN
MASTER RECORD
WRITE
CARD TYPE
RECORD
BLANK FILL
CARD TYPE
RECORD AREA
READ
NEXT CARD
EOF
212
12
? CARD TYPE = 1
YES
28
10
NO
? ENTRY NUMBER
≠ MASTER RECORD
ENTRY NUMBER
YES
200
10
NO
E
9


E
9
? CARD NUMBER VALID
NO
200
10
YES
STORE CARD NUMBER IN CARD TYPE RECORD AREA
SET LENGTH COUNTER TO 3 CARD COUNTER TO 1
26
5


28
10
INCREMENT
COUNT OF
RECORDS WRITTEN
WRITE
MASTER RECORD
REREAD
SEMANTIC DOMAIN
CODE UNDER
INTEGER FORMAT
? NEW ENTRY
NUMBER ≠
PREVIOUS ENTRY
NUMBER
YES
F
12
NO
200
10
WRITE
ERROR MESSAGE
208
11


202
11
WRITE
ERROR MESSAGE
208
11
BLANK FILL
CARD TYPE
RECORD
210
11
READ
NEXT CARD
EOF
212
? ENTRY NUMBER
= PREVIOUS ENTRY
NUMBER
YES
200
10
NO
WRITE
ENTRY NUMBER OF
NEXT RECORD
F
12


212
12
WRITE
ERROR MESSAGE
F
12
? DUMMY LAST
RECORD
NO
2
2
YES
214
12
WRITE
COUNT OF RECORDS
WRITTEN TO DISC
PRINT & PUNCH
RECORD LENGTH
FILE COUNTERS
STOP
CREATION
PROGRAM

2.2.8 Listing

```
C          CREATION PROGRAM
C
      DIMENSION KY(70),KARD(20),JFL(3,20),IND(2,21)
      DIMENSION ARRDV(2),ARRA(7),ARRB(200)
      INTEGER*4 BL/'    '/,KREC(209)/209*'    '/,CT(11)/'   *   1   2
     13   4   5   6   7   8   9  10'/
      EQUIVALENCE(KY(31),IND(1,2)),(KY(4),ARRA(1)),(KREC(3),ARRB(1))
      EXTERNAL ORTHO
      DEFINE FILE  7(5500,265,E,KAA),13(11800,40,E,K1),14(5250,80,E,K1),
     115(1520,120,E,K1),16(0960,160,E,K1),17(0390,200,E,K1),18(0330,240,
     2E,K1),19(0200,280,E,K1),20(0180,320,E,K1),21(0080,360,E,K1),22(007
     30,400,E,K1),23(0070,440,E,K1),24(0060,480,E,K1),25(0060,520,E,K1),
     426(0050,560,E,K1),27(0025,600,E,K1),28(0025,640,E,K1),29(0020,680,
     5E,K1),30(0020,720,E,K1),31(0020,760,E,K1),32(0020,800,E,K1)
C
C          READ AND WRITE FILE COUNTERS
C
      READ(1,500)((JFL(J,K),J=1,3),K=1,20)
  500 FORMAT(I2,2I8)
      WRITE(3,505)
  505 FORMAT('1')
      DO 1 K=1,20
      WRITE(3,510)(JFL(J,K),J=1,3)
  510 FORMAT(11X,I2,2I8)
      IF(JFL(1,K).NE.(K+12))STOP
    1 CONTINUE
C
C          INITIALIZE
C
      CALL ERRSET(215,0,256,1)
      CALL REREAD
      WRITE(3,505)
      IER=0
      KTA=0
      KY(2)=1
      KY(3)=0
      KY(12)=0
C
C          READ FIRST CARD
C
      READ(1,515)(KARD(J),J=1,20)
  515 FORMAT(I6,I2,18A4)
      IF(KARD(1).EQ.999999)GO TO 214
      IF(KARD(2).NE.1)GO TO 200
      READ(99,910)JAJ
  910 FORMAT(36X,I4)
C
C          PROCESS CARD TYPE 1
C
    2 KY(1)=KARD(1)
      DO 3 J=3,10
    3 KY(J+1)=KARD(J)
      DO 4 J=11,20
    4 KY(J+2)=KARD(J)
C
C          READ CARD TYPE 2
C
      READ(1,515,END=212)(KARD(J),J=1,20)
      IF(KARD(1).NE.KY(1))GO TO 200
      IF(KARD(2).NE.2)GO TO 200
```

```
      READ(99,915)JAJ,JBJ
  915 FORMAT(8X,I3,I4)
C
CC          CALL ORTHOGRAPHY CHANGE ROUTINE
C
      JVB=0
      IF(JAJ.EQ.2.OR.JAJ.EQ.8)JVB=1
      CALL CSDV(ARRA,ARRDV,25,25)
      CALL XTOPL1(ORTHO,ARRDV,JVB,IER,1)
C
CC          WRITE ERROR MESSAGES
C
      IF(IER.EQ.0)GO TO 33
      GO TO(31,32),IER
   31 WRITE(3,540)KY(1),(ARRA(J),J=1,7)
  540 FORMAT('0EXTENSION GT 25 - ',I4,4X,6A4,A1)
      IER=0
      GO TO 33
   32 WRITE(3,545)KY(1),KY(2),(ARRA(J),J=1,6)
  545 FORMAT('0SYNTAX ERROR - ',I4,I2,4X,25A4/(2X,31A4))
      IER=0
C
CC          MOVE DATA FROM CARD 2 TO MASTER RECORD
CC          SET MASTER RECORD ADDRESS AREA TO ZERO
C
   33 DO 5 J=3,10
    5 KY(J+20)=KARD(J)
      DO 6 J=33,70
    6 KY(J)=0
C
CC          SET UP ENGLISH GLOSS OUTPUT RECORD
C
      KREC(1)=KY(1)
      KREC(2)=2
      KREC(3)=BL
      KREC(4)=BL
      DO 7 J=11,20
    7 KREC(J-6)=KARD(J)
      KNT=14
      KR=1
C
CC          PROCESS CARD TYPES 2-21
C
    8 IF(KREC(201).NE.BL)GO TO 206
      IF(KARD(20).EQ.BL)GO TO 10
      IF(KARD(20).NE.CT(KR))GO TO 202
      READ(1,515,END=212)(KARD(J),J=1,20)
      IF(KARD(2).NE.KREC(2))GO TO 202
      IF(KARD(1).NE.KY(1))GO TO 200
      KR=KR+1
   26 DO 9 J=3,20
      KREC(KNT)=KARD(J)
    9 KNT=KNT+1
      KNT=KNT-1
      GO TO 8
C
CC          CHECK LENGTH OF RECORD FOR CARD TYPES 2-21
C
   10 DO 11 J=11,201,10
      IF(KREC(J).EQ.BL)GO TO 27
```

```
   11 CONTINUE
      WRITE(3,520)
  520 FORMAT('1',10X,'EXECUTION ERROR')
      GO TO 212
   27 J=J-1
C
CC         CALL ORTHOGRAPHY CHANGE ROUTINE FOR CARD TYPES 3-21
C
      IF(KREC(2).EQ.2)GO TO 29
      J=J*4-8
      CALL CSDV(ARRB,ARRDV,800,J)
      CALL XTOPL1(OPTHO,ARRDV,0,IER,0)
      J=J/4
      IF(IER.EQ.0)GO TO 30
      WRITE(3,545)KREC(1),KREC(2),(ARRB(K),K=1,J)
      IER=0
   30 IF(KREC(J).EQ.BL)GO TO 29
      J=J+10
      GO TO 30
C
CC         WRITE RECORD FOR CARD TYPES 2-21
C
   29 LF=J/10
      LU=LF+12
      IND(1,KREC(2))=LU
      JFL(3,LF)=JFL(3,LF)+1
      IF(JFL(3,LF).LE.JFL(2,LF))GO TO 14
      JFL(3,LF)=JFL(3,LF)-1
      WRITE(3,525)LU
  525 FORMAT('1',10X,'NO. OF RECORDS EXCEEDED IN FILE ',I2)
      GO TO 212
   14 IND(2,KREC(2))=JFL(3,LF)
      K1=JFL(3,LF)
      WRITE(LU'K1,530)(KREC(K),K=1,J)
  530 FORMAT(I6,I2,198A4)
   24 DO 25 K=11,J
   25 KREC(K)=BL
C
CC         READ CARD TYPES 3-21
C
      READ(1,515,END=212)(KARD(J),J=1,20)
      IF(KARD(2).EQ.1)GO TO 28
      IF(KARD(1).NE.KY(1))GO TO 200
      IF(KARD(2).GT.21.OR.KARD(2).LE.KREC(2))GO TO 200
      KREC(2)=KARD(2)
      KNT=3
      KR=1
      GO TO 26
C
CC         ADD TO COUNTER AND WRITE MASTER RECORD
C
   28 KTA=KTA+1
      KAA=KY(1)
      WRITE(3,905)KTA,KAA
  905 FORMAT(' ',I4,2X,I5)
      WRITE(7'KAA,535)(KY(J),J=1,70)
  535 FORMAT(I5,I1,I5,8A4,I10,18A4,20(I2,I5))
      IF(KARD(1).EQ.999999)GO TO 214
      READ(99,910)JAJ
      IF(KARD(1).EQ.KY(1))GO TO 200
```

```
      GO TO 2
C
C          ERROR MESSAGES
C
  200 WRITE(3,201)KARD(1),KARD(2)
  201 FORMAT('0',90X,'IDENT ERROR:',I6,I3)
      GO TO 208
  202 WRITE(3,203)KARD(1),KARD(2),KARD(20)
  203 FORMAT('0',90X,'CONTINUATION ERROR:',I6,I3,A4)
      GO TO 208
  206 WRITE(3,207)KARD(1),KARD(2)
  207 FORMAT('0',90X,'RECORD SIZE GT 200:',I6,I3)
  208 DO 209 J=11,201
  209 KREC(J)=BL
  210 READ(1,515,END=212)(KARD(J),J=1,20)
      IF(KARD(2).NE.1)GO TO 210
      IF(KARD(1).EQ.KY(1))GO TO 200
      WRITE(3,211)KARD(1)
  211 FORMAT('0',90X,'NEXT RECORD:',I6)
      IF(KARD(1).EQ.999999)GO TO 214
      GO TO 2
C
C          END OF RUN PROCEDURES
C
  212 WRITE(3,213)KY(1)
  213 FORMAT('0',10X,'RECORD NOT WRITTEN:',I6)
  214 WRITE(3,215)KTA
  215 FORMAT('1',10X,I6,' RECORDS FILED TO DISK'/1X)
C
C          WRITE AND PUNCH FILE COUNTERS
C
      DO 216 K=1,20
      WRITE(3,510)(JFL(J,K),J=1,3)
  216 WRITE(2,217)(JFL(J,K),J=1,3)
  217 FORMAT('V',I2,2I8)
      WRITE(3,505)
      STOP
      END
```

```
ORTHO: PROCEDURE(SB,JVB,IFLA,JM);
/* THIS SUBROUTINE OPERATES ON ENGA CARDS EXCEPT CARD TWO (ENGLISH).
   IT PERFORMS FOUR TYPES OF ORTHOGRAPHY CHANGES ON ALL WORDS PRESENTED
   TO IT INCLUDING ANY ENGLISH ON CARDS 3 - 21.
   TYPE 1:   LABIALIZATION.
          ANY CONSONANT + W GOES TO CONSONANT + U
          EG  BWAA -> BUAA
              KWAA -> KUAA   ETC.
   TYPE 2:   VOWEL CLUSTERS.
```

```
      LAST TWO VOWELS OF THREE VOWEL SET WHICH ARE NOT IDENTICAL
      ARE CHANGED AS FOLLOWS:
      V2=I -> IY
      V2=U -> UW
      V2=E -> Y
      V2=O -> W
      EG  KAIA -> KAIYA
          KOEA -> KOYA   ETC.
TYPE 3    VOWEL CLUSTERS IN VERB STEMS:
      IN VERBS IF  V1 V2 V3 PRECEDES -GI OR -GE, AND IF V2 + V3 ARE
      THE SAME V3 IS REMOVED.
      EG  MAIIGI -> MAIGI
TYPE 4    PRENASALIZATION.
      IN THE MIDDLE OF A WORD ONLY (NOT AT THE BEGINNING)
      B ->MB
      D -> ND
      G -> NG
      J -> NJ
      EG KADEGE -> KANDENGE
         AJA -> ANJA
THE SUBROUTINE IS IN THREE SECTIONS.
SECTION 1  REMOVES BLANKS FROM END OF STRING SUPPLIED.
SECTION 2 SPLITS OFF EACH ENGA WORD TO PASS ACROSS TO INTERNAL
          SUBROUTINE CHANGE.
SECTION 3 (SUBROUTINE CHANGE) PERFORMS THE FOUR TYPES OF ORTHOGRAPHY
          CHANGES.
PARAMETERS
SB - STRING CONTAINING ENGA WORDS
JVB = 1 IF MAIN ENTRY IF A VERB
    = 0 OTHERWISE
```

```
   IFLA = 0 NO ERRORS IN PROCESSING
        = 1 MAIN ENTRY WOULD EXPAND TO GREATER THAN 25 CHARACTERS
        = 2 TONE CODE HAS COME AFTER A SEMI-VOWEL (Y OR W)
   JM = 1 MAIN ENTRY
      = 0 CARDS 3-21
   OTHER IMPORTANT VARIABLES.
   JPOS - STARTING CHARACTER OF NEXT WORD.
   SA,SD,SR,S INTERMEDIATE STRINGS TO HOLD WORDS TEMPORARILY.
   SUB - USED TO EXAMINE ONE CHARACTER AT A TIME
   ICOUNT - COUNTS NO OF VOWELS FOUND IN SEQUENCE
   L - HOLDS POSITIONS OF V2 AND V3.
   FOUND - BRANCH TAKEN IF 3 VOWELS FOUND FOR TYPE 2.
   PROC - BRANCH TAKEN IF WORD ENDS IN GI OR GE FOR TYPE 3.
   R - BRANCH TAKEN IF 3 VOWELS FOUND FOR TYPE 3.                    */
DCL SB CHAR(800) VARYING,
IFLA BIN FIXED (31,0),
(JVB,NWD,JM) BIN FIXED (31,0);
/*  SECTION 1 - REMOVES BLANKS FROM END OF STRING                    */
JPOS=1;
DO I=LENGTH(SB) TO 1 BY -1;
TEST: IF SUBSTR(SB,I,1)¬=' ' THEN
/*  SECTION 2 - CHECKS FOR ERROR TYPE 1 AND SPLITS OFF NEXT WORD     */
(STRINGRANGE): ONE: BEGIN;
DCL SA CHAR(I),
SUB CHAR(1),
SD CHAR(800) VARYING;
IF JM=1 & I>20 THEN DO;
IFLA=1;
RETURN;
END;
KOUNT=1;
SA=SB;
IF JM=0 THEN SB=(800)' ';
SB='';
/*  START PROCESSING AT FIRST NON-BLANK CHARACTER.                   */
DO I=1 TO LENGTH(SA);
IF SUBSTR(SA,I,1)¬=' ' THEN DO;
JPOS=I;
SB=SB||SUBSTR(SA,1,I-1);
GO TO LOOP;
END;
END;
/*  SEARCH FOR NEXT WORD TERMINATOR AND CALL SUBROUTINE CHANGE       */
LOOP: DO I=JPOS TO LENGTH(SA);
SUB=SUBSTR(SA,I,1);
J=0;
IF SUB=';' | SUB=',' | SUB=' ' THEN DO;
```

```
DO J=1 TO LENGTH(SA)-I;
IF SUBSTR(SA,I+J,1)¬=' ' THEN
GO TO CALL;
END;
J=J-1;
END;
END;
CALL: N=I+J;
CALL CHANGE(SUBSTR(SA,JPOS,I-JPOS),IFLAG,SD);
/* TEST ERROR FLAG - RETURN ORIGINAL STRING IF ERROR. BLANK OUT
   REST OF STRING AND RETURN..                                    */
JPOS=N;
IFLA=IFLAG;
IF IFLAG¬=0 THEN DO;
SB=SA;
RETURN;
END;
SB=SB||SD||SUB;
KOUNT=KOUNT+1;
IF JPOS<=LENGTH(SA) THEN GO TO LOOP;
IF SUBSTR(SB,LENGTH(SB)-1,1)=SUBSTR(SB,LENGTH(SB),1)
THEN SB=SUBSTR(SB,1,LENGTH(SB)-1);
IF JM=1 THEN LEN=25;
ELSE LEN=800;
SD=(800)' ';
SB=SB||SUBSTR(SD,1,(LEN-LENGTH(SB)));
RETURN;
END ONE;
END;
IFLA=2;
RETURN;
CHANGE: PROCEDURE(S,IFL,SR);
DCL S CHAR(*),
SR CHAR(800) VARYING,
SUB CHAR(1),
L(2),
VOWEL(5) CHAR(1) INITIAL('A','E','I','O','U');
/* TYPE 1 CHANGE. SEARCH FOR CONSONANT + W AND CHANGE TO U.       */
IFL=0;
DO I=1 TO LENGTH(S)-1;
SUB=SUBSTR(S,I,1);
IF SUB<'A' | SUB>'Z' THEN GO TO END;
DO J=1 TO 5;
IF SUB=VOWEL(J) THEN GO TO END;
END;
IF SUBSTR(S,I+1,1)='W' THEN SUBSTR(S,I+1,1)='U';
END: END;
/* TYPE 4 CHANGE. SEARCH FOR B,D,G,J AND INSERT REQUIRED CHARACTER
   BEFORE COPYING STRING INTO INTERMEDIATE STRING SR.             */
SR=SUBSTR(S,1,1);
DO I=2 TO LENGTH(S);
SUB=SUBSTR(S,I,1);
IF SUB='B' THEN SR=SR||'M';
IF SUB='D'|SUB='G'|SUB='J' THEN SR=SR||'N';
SR=SR||SUB;
END;
/* TYPE 2 CHANGE. SEARCH BACKWARDS THROUGH WORD FOR THREE VOWELS IN A
   ROW OR SEPARATED ONLY BY TONES. STORE POSITIONS OF THESE THREE.*/
I=LENGTH(SR);
ICOUNT=0;
```

```
LOOP: IF I<3 THEN GO TO NXT;
DO J=I TO I-10 BY -1;
IF J<1 THEN GO TO NXT;
SUB=SUBSTR(SR,J,1);
IF SUB='''' THEN GO TO ROUND;
DO K=1 TO 5;
IF SUB=VOWEL(K) THEN DO;
ICOUNT=ICOUNT+1;
IF ICOUNT=3 THEN GO TO FOUND;
L(ICOUNT)=J;
GO TO ROUND;
END;
END;
FIX: ICOUNT=0;
I=I-1;
GO TO LOOP;
ROUND: END;
GO TO FIX;
/* CHECK FINAL TWO VOWELS ARE NON-IDENTICAL AND THEN PERFORM PRESCRIB-
   ED CHANGES PAYING SPECIAL ATTENTION TO TONE CODES.                 */
FOUND: IF SUBSTR(SR,L(2),1)=SUBSTR(SR,L(1),1) THEN GO TO FIX;
SUB=SUBSTR(SR,L(2),1);
IF SUB='A' THEN GO TO FIX;
IF SUB='E' THEN DO;
SUBSTR(SR,L(2),1)='Y';
IF SUBSTR(SR,L(2)+1,1)='''' THEN SR=SUBSTR(SR,1,J   )||''''||'Y'||
SUBSTR(SR,L(2)+2);
GO TO FIX;
END;
IF SUB='O' THEN DO;
SUBSTR(SR,L(2),1)='W';
IF SUBSTR(SR,L(2)+1,1)='''' THEN SR=SUBSTR(SR,1,J   )||''''||'W'||
SUBSTR(SR,L(2)+2);
GO TO FIX;
END;
IF SUBSTR(SR,L(2)+1,1)='''' THEN L(2)=L(2)+1;
IF SUB='I' THEN DO;
SR=SUBSTR(SR,1,L(2))||'Y'||SUBSTR(SR,L(2)+1);
I=L(2);
ICOUNT=0;
GO TO LOOP;
END;
IF SUB='U' THEN DO;
SR=SUBSTR(SR,1,L(2))||'W'||SUBSTR(SR,L(2)+1);
I=L(2);
ICOUNT=0;
GO TO LOOP;
END;
/* TYPE 3 CHANGE. CHECK FOR VERB AND LAST SYLLABLE BEING GI OR GE.   */
NXT: IF JVB=0 THEN GO TO RET;
LN=LENGTH(SR);
IF SUBSTR(SR,LN-3)='NGI'''|SUBSTR(SR,LN-3)='NGE''' THEN DO;
KK=LN-4;
GO TO PROC;
END;
IF SUBSTR(SR,LN-2)='NGI'|SUBSTR(SR,LN-2)='NGE' THEN DO;
KK=LN-3;
GO TO PROC;
END;
ELSE GO TO RET;
```

```
/* SEARCH BACKWARDS THROUGH WORD FOR THREE VOWELS                    */
PROC: ICOUNT=0;
DO I=KK TO KK-10 BY -1;
IF I<1 THEN GO TO RET;
SUB=SUBSTR(SR,I,1);
IF SUB='' THEN GO TO ND;
DO J=1 TO 5;
IF SUB=VOWEL(J) THEN DO;
ICOUNT=ICOUNT+1;
IF ICOUNT=3 THEN GO TO R;
L(ICOUNT)=I;
GO TO ND;
END;
END;
ICOUNT=0;
ND: END;
GO TO RET;
/* CHECK IF FINAL TWO EQUAL AND DELETE LAST ONE.                     */
R: IF SUBSTR(SR,L(2),1)¬=SUBSTR(SR,L(1),1)THEN GO TO RET;
SR=SUBSTR(SR,1,L(1)-1)||SUBSTR(SR,L(1)+1);
RET: IF JM=1 & LENGTH(SR)>25 THEN IFL=1;
DO I=1 TO LENGTH(SR)-1;
IF SUBSTR(SR,I,1)='Y'|SUBSTR(SR,I,1)='W' THEN
IF SUBSTR(SR,I+1,1)='' THEN DO;
IFL=2;
RETURN;
END;
END;
RETURN;
END CHANGE;
END ORTHO;
```

2.3 *UPDATING*

2.3.1 Description of Updating Program

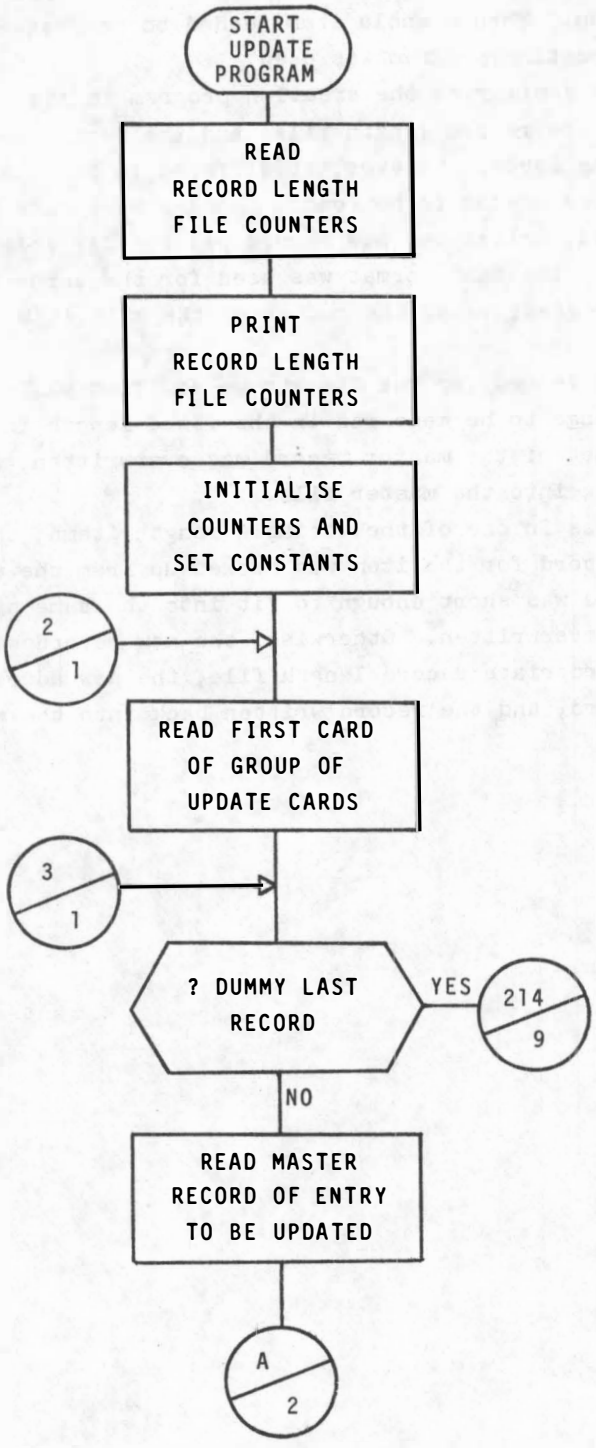
This program was used when information on one card type only required modification. When a whole item needed to be changed or a new item added, the creation program was used.

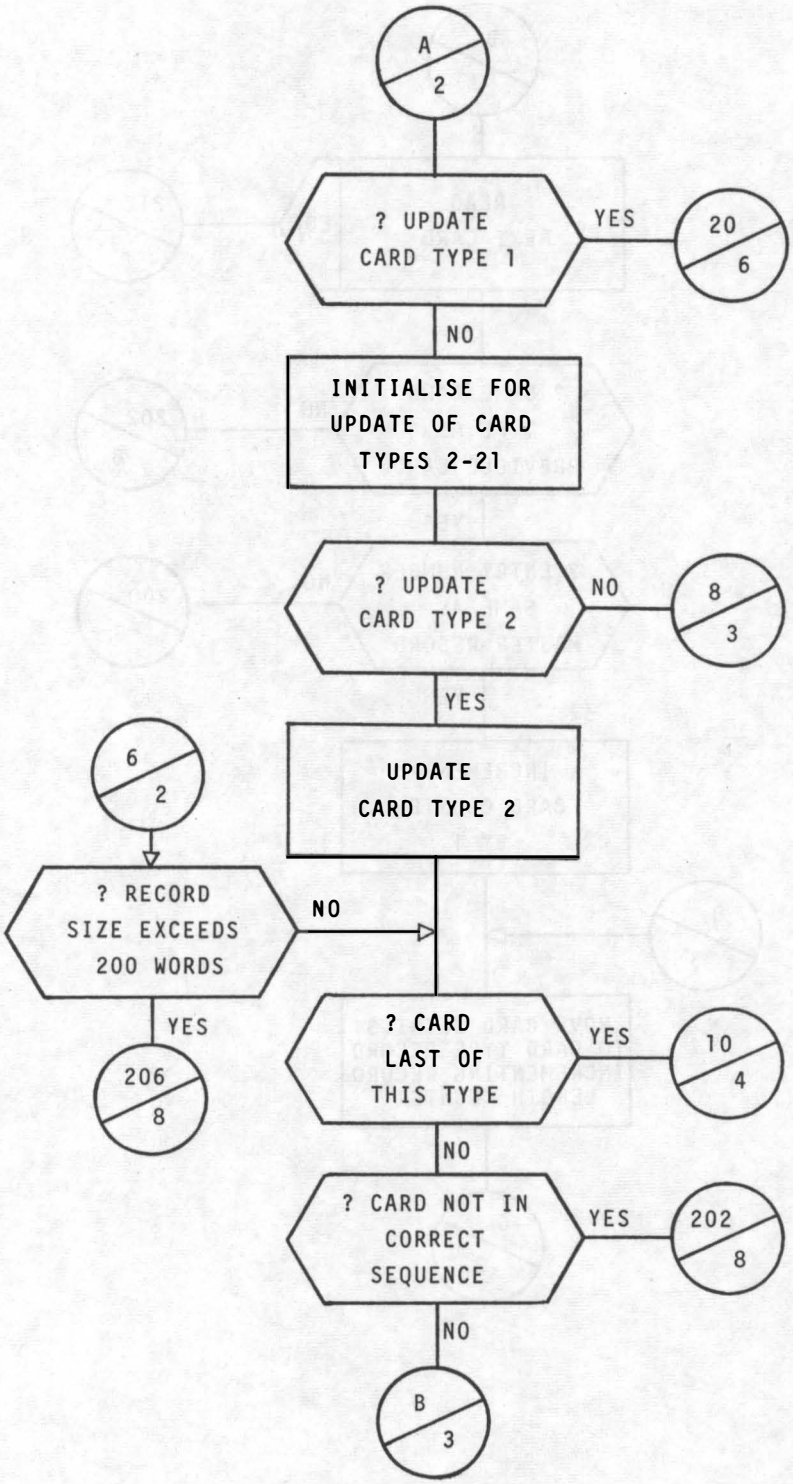
The program was similar to the creation program in its use of control cards for the record length files and the method used for reading and editing the cards. However, it differed in that only the card type to be corrected needed to be read. Changes were made by overwriting the existing record, unless the new record was too large for the area available. Exactly the same format was used for the cards as was used in the creation program; thus, the number of the main item appeared on every card.

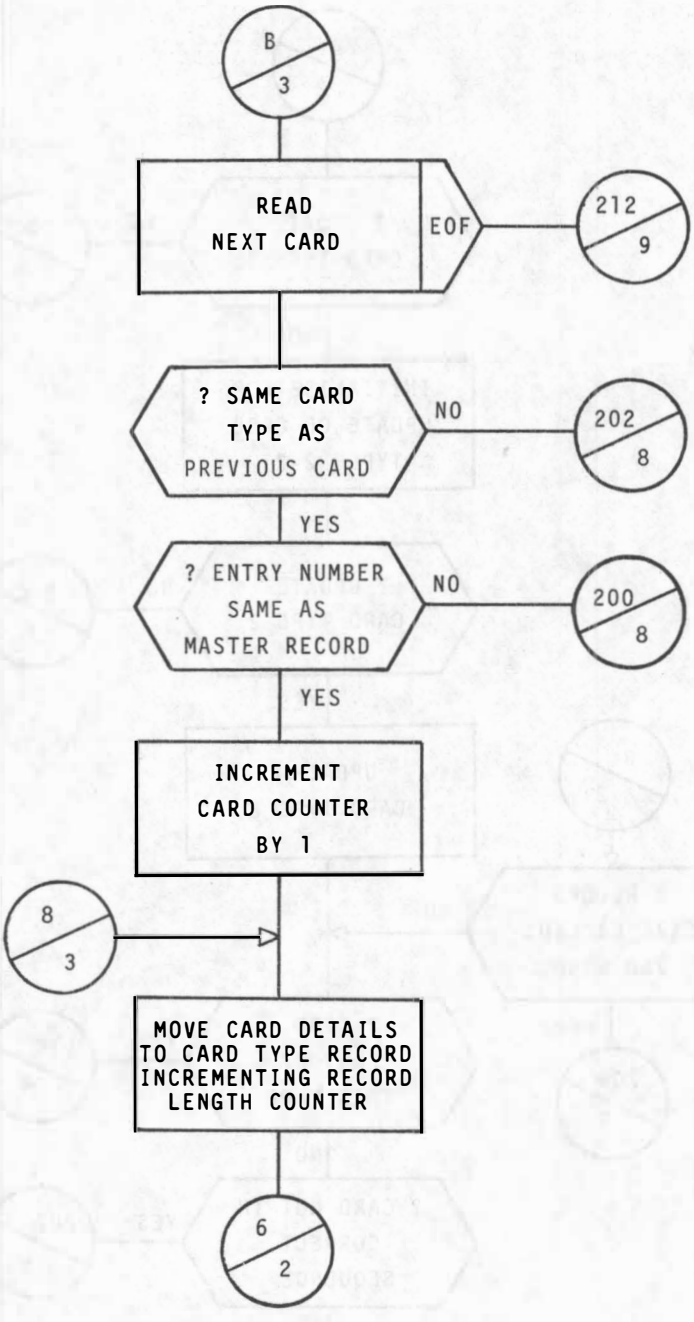
The master file record for the item to be modified was read into core store. If the change to be made was in the fixed length information, the appropriate part of the master record was overwritten, and the record written back into the master file.

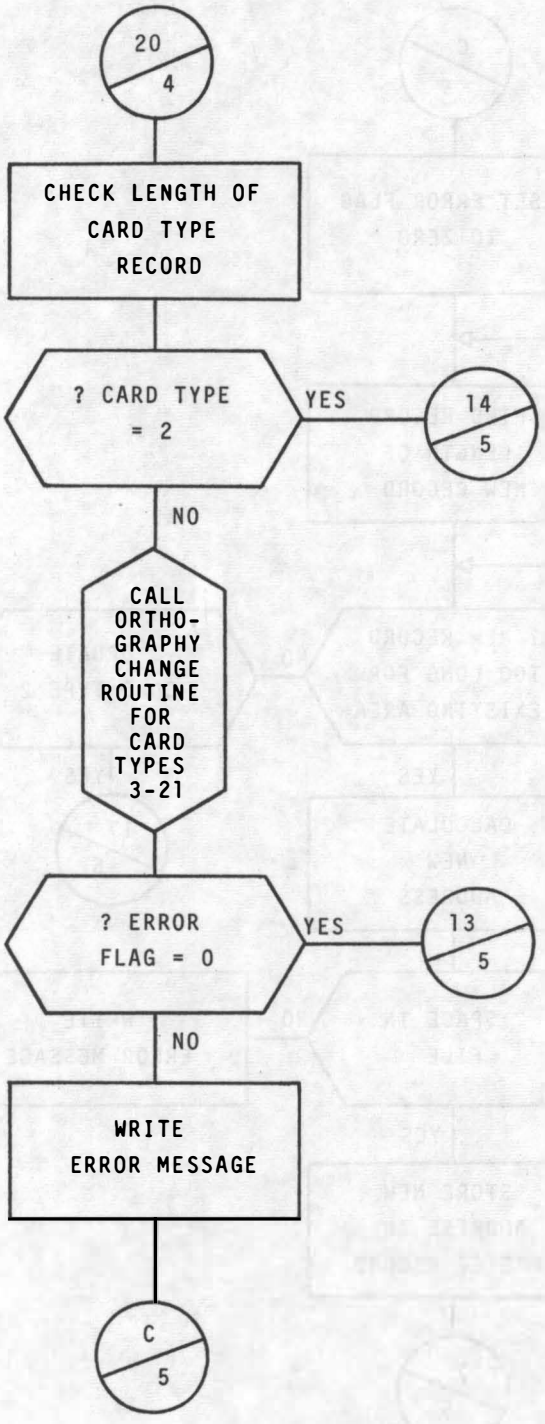
If the change was in one of the variable length items, the address of the existing record for the item was picked up from the master record. When the new record was short enough to fit into the same area, the existing record was overwritten. Otherwise, the new record was added to the end of the appropriate record length file, the new address stored in the master record, and the record written back into the master file.

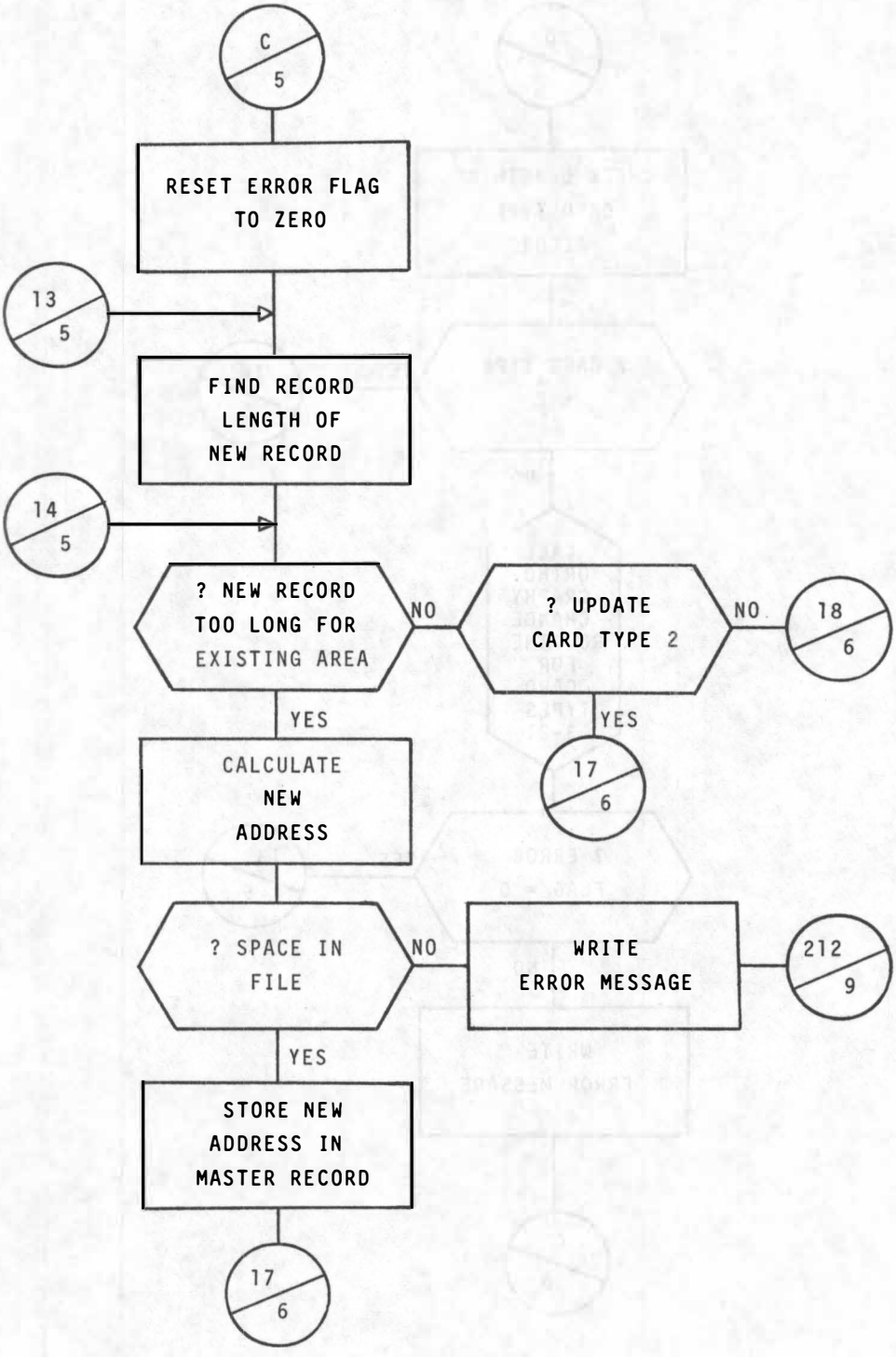
2.3.2 Flowchart

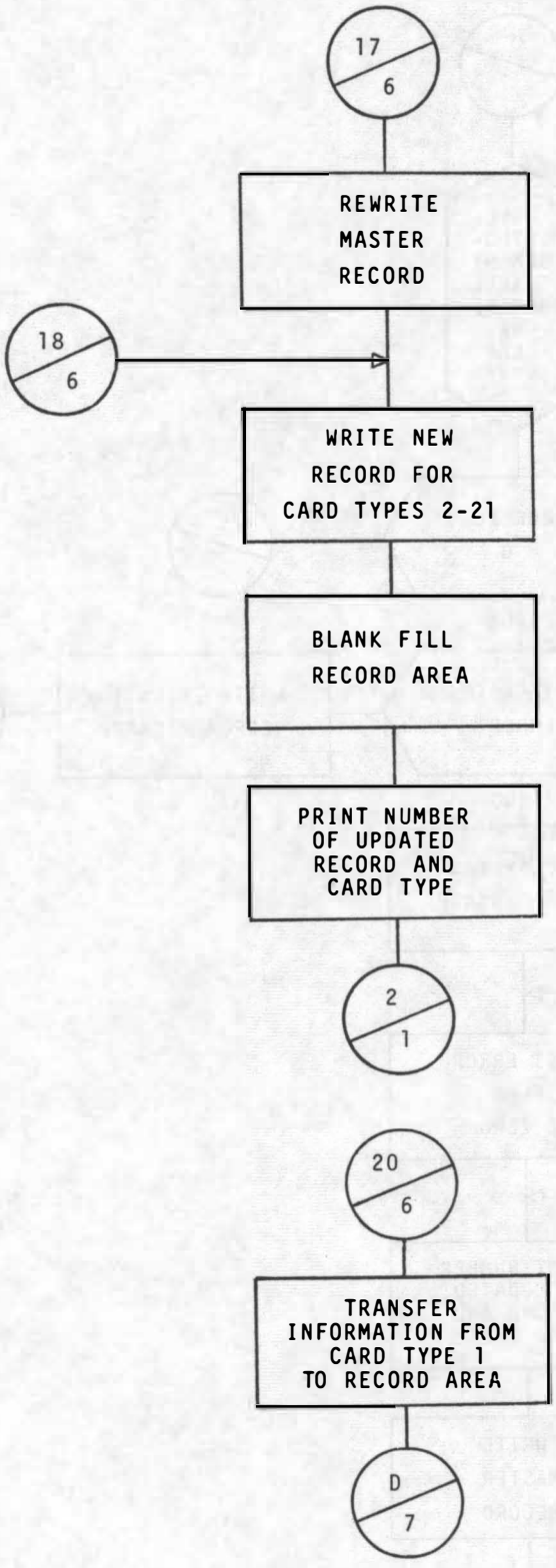


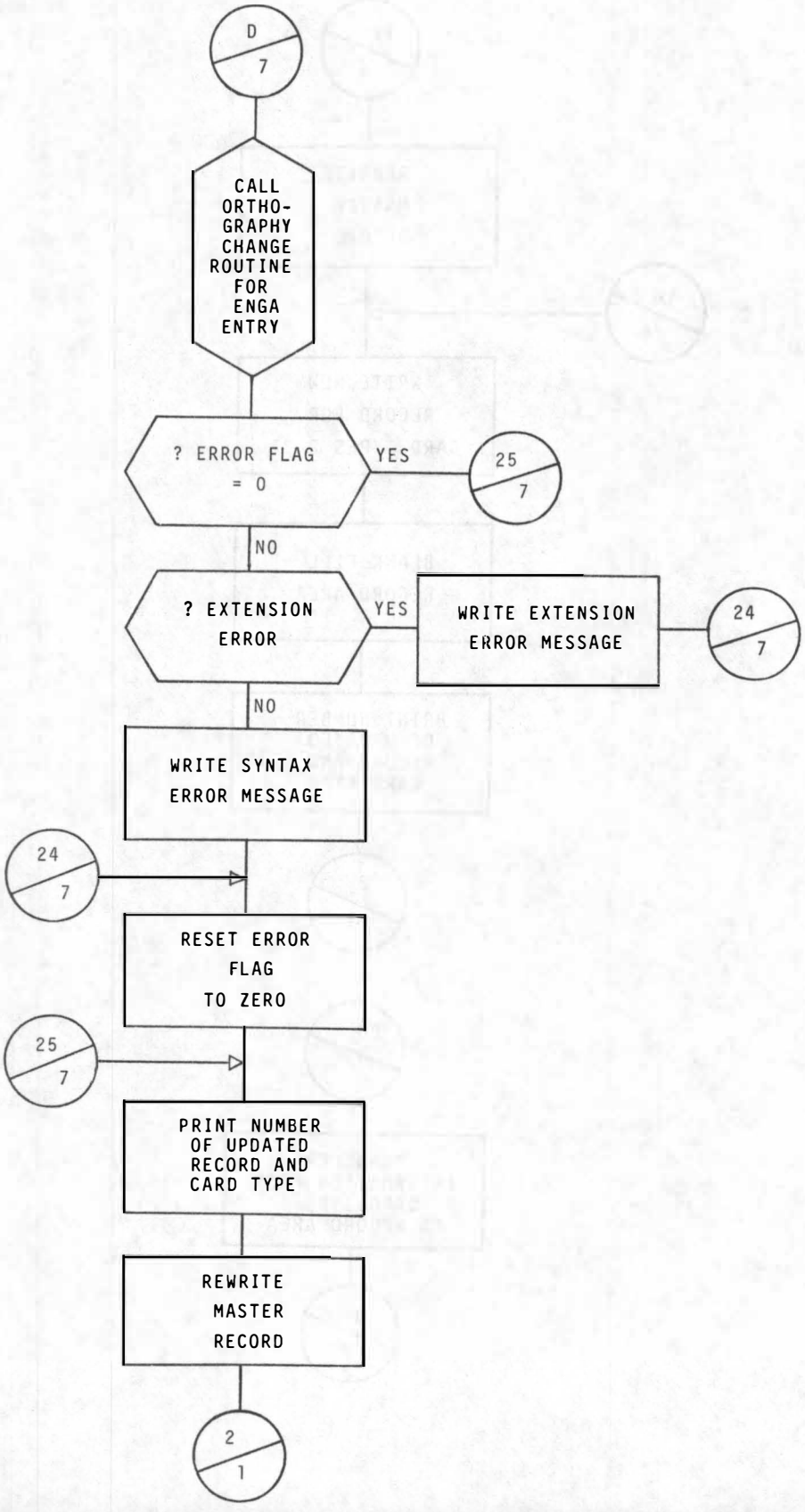

A
2
? UPDATE CARD TYPE 1
YES
20
6
NO
INITIALISE FOR UPDATE OF CARD TYPES 2-21
? UPDATE CARD TYPE 2
NO
8
3
YES
UPDATE CARD TYPE 2
6
2
? RECORD SIZE EXCEEDS 200 WORDS
NO
YES
206
8
? CARD LAST OF THIS TYPE
YES
10
4
NO
? CARD NOT IN CORRECT SEQUENCE
YES
202
8
NO
B
3

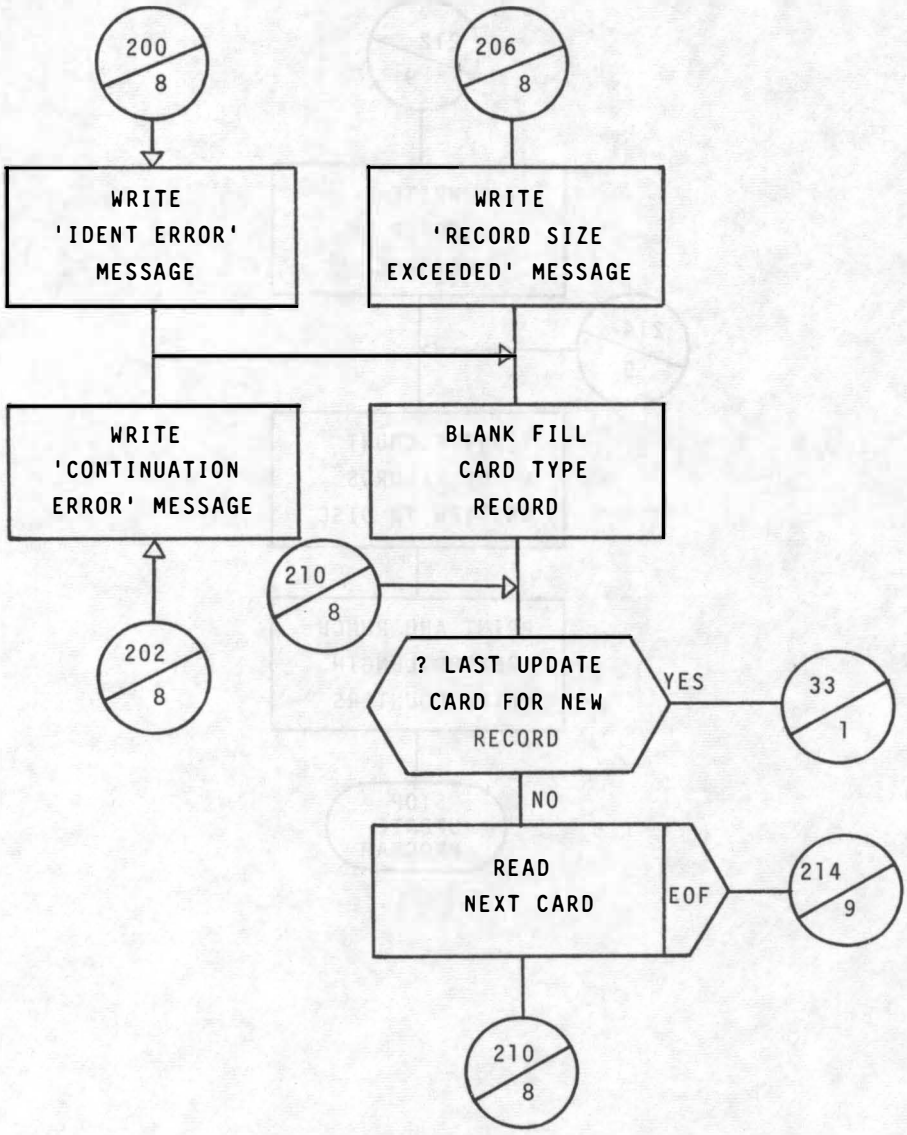

B
3
READ
NEXT CARD
EOF
212
9
? SAME CARD
TYPE AS
PREVIOUS CARD
NO
202
8
YES
? ENTRY NUMBER
SAME AS
MASTER RECORD
NO
200
8
YES
INCREMENT
CARD COUNTER
BY 1
8
3
MOVE CARD DETAILS
TO CARD TYPE RECORD
INCREMENTING RECORD
LENGTH COUNTER
6
2

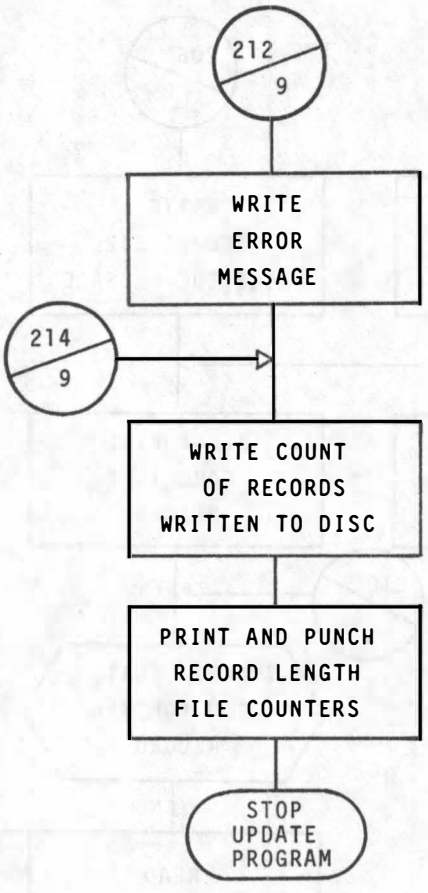

20
4
CHECK LENGTH OF CARD TYPE RECORD
? CARD TYPE = 2
YES
14
5
NO
CALL ORTHO-GRAPHY CHANGE ROUTINE FOR CARD TYPES 3-21
? ERROR FLAG = 0
YES
13
5
NO
WRITE ERROR MESSAGE
C
5


C
5
RESET ERROR FLAG TO ZERO
13
5
FIND RECORD LENGTH OF NEW RECORD
14
5
? NEW RECORD TOO LONG FOR EXISTING AREA
NO
? UPDATE CARD TYPE 2
NO
18
6
YES
YES
CALCULATE NEW ADDRESS
17
6
? SPACE IN FILE
NO
WRITE ERROR MESSAGE
212
9
YES
STORE NEW ADDRESS IN MASTER RECORD
17
6


17
6
REWRITE
MASTER
RECORD
18
6
WRITE NEW
RECORD FOR
CARD TYPES 2-21
BLANK FILL
RECORD AREA
PRINT NUMBER
OF UPDATED
RECORD AND
CARD TYPE
2
1
20
6
TRANSFER
INFORMATION FROM
CARD TYPE 1
TO RECORD AREA
D
7


D
7
CALL ORTHO-GRAPHY CHANGE ROUTINE FOR ENGA ENTRY
? ERROR FLAG = 0
YES
25
7
NO
? EXTENSION ERROR
YES
WRITE EXTENSION ERROR MESSAGE
24
7
NO
WRITE SYNTAX ERROR MESSAGE
24
7
RESET ERROR FLAG TO ZERO
25
7
PRINT NUMBER OF UPDATED RECORD AND CARD TYPE
REWRITE MASTER RECORD
2
1


200
8
206
8
WRITE
'IDENT ERROR'
MESSAGE
WRITE
'RECORD SIZE
EXCEEDED' MESSAGE
WRITE
'CONTINUATION
ERROR' MESSAGE
BLANK FILL
CARD TYPE
RECORD
210
8
202
8
? LAST UPDATE
CARD FOR NEW
RECORD
YES
33
1
NO
READ
NEXT CARD
EOF
214
9
210
8


212
9
WRITE
ERROR
MESSAGE
214
9
WRITE COUNT
OF RECORDS
WRITTEN TO DISC
PRINT AND PUNCH
RECORD LENGTH
FILE COUNTERS
STOP
UPDATE
PROGRAM

```
C
C          CALCULATE NEW ADDRESS AND STORE IN MASTER RECORD
C
   15 JFL(3,LF)=JFL(3,LF)+1
      IF(JFL(3,LF).LE.JFL(2,LF))GO TO 16
      JFL(3,LF)=JFL(3,LF)-1
      WRITE(3,530)LU
  530 FORMAT('1',10X,'NO. OF RECORDS EXCEEDED IN FILE ',I2)
      GO TO 212
   16 IND(1,KREC(2))=LU
      IND(2,KREC(2))=JFL(3,LF)
C
C          REWRITE MASTER RECORD
C
   17 KAA=KY(1)
      WRITE(7'KAA,520)(KY(M),M=1,70)
C
C          WRITE NEW RECORD FOR CARD TYPES 2-21
C
   18 K1=IND(2,KREC(2))
      WRITE(LU'K1,535)(KREC(K),K=1,J)
  535 FORMAT(I6,I2,198A4)
      CO 19 K=)1,J
   19 KREC(K)=BL
      KTA=KTA+1
      WRITE(3,540)KTA,KREC(1),KREC(2)
  540 FORMAT(' ',I4,2X,I5,2X,I2)
      GO TO 2
C
C          UPDATE OF CARD TYPE 1
C
   20 READ(99,910)JAJ
  910 FORMAT(36X,I4)
      CO 21 J=3,10
   21 KY(J+1)=KARD(J)
      CO 22 J=11,20
   22 KY(J+2)=KARD(J)
      CALL CSOV(ARRA,ARRDV,25,25)
      CALL XTOPL1(CRTHO,ARRDV,JVB,IER,1)
      IF(IER.EQ.0)GO TO 25
      IF(IER.EQ.2)GO TO 23
      WRITE(3,545)KY(1),(ARRA(J),J=1,7)
  545 FORMAT('0EXTENSION GT 25 - ',I4,4X,6A4,A1)
      GO TO 24
   23 WRITE(3,525)KY(1),KY(2),(ARRA(J),J=1,6)
   24 IER=0
   25 KTA=KTA+1
      KAA=KY(1)
      KAB=1
      WRITE(3,540)KTA,KAA,KAB
      WRITE(7'KAA,520)(KY(J),J=1,70)
      GO TO 2
```

```
C
C          ERROR MESSAGES
C
  200 WRITE(3,201)KARD(1),KARD(2)
  201 FORMAT('0',20X,'IDENT ERROR:',I6,I3)
      GO TO 208
  202 WRITE(3,203)KARD(1),KARD(2),KARD(20)
  203 FORMAT('0',20X,'CONTINUATION ERROR:',I6,I3,A4)
      GO TO 208
  206 WRITE(3,207)KARD(1),KARD(2)
  207 FORMAT('0',20X,'RECORD SIZE GT 200:',I6,I3)
  208 DO 209 J=11,KNT
  209 KREC(J)=BL
  210 IF(KARD(1).NE.KREC(1))GO TO 3
      READ(1,515,END=214)(KARD(J),J=1,20)
      GO TO 210
C
C          END OF RUN PROCEDURES
C
  212 WRITE(3,213)KREC(1),KREC(2)
  213 FORMAT('0',20X,'RECORD NOT WRITTEN:',I6,I3)
  214 WRITE(3,215)KTA
  215 FORMAT('1',10X,I6,' RECORDS FILED TO DISK'/1X)
C
C          WRITE AND PUNCH FILE COUNTERS
C
      DO 216 K=1,20
      WRITE(3,510)(JFL(J,K),J=1,3)
  216 WRITE(2,217)(JFL(J,K),J=1,3)
  217 FORMAT('V',I2,2I8)
      WRITE(3,505)
      STOP
      END
```

```
ORTHO: PROCEDURE(SB,JVB,IFLA,JM);
/* THIS SUBROUTINE OPERATES ON ENGA CARDS EXCEPT CARD TWO (ENGLISH).
   IT PERFORMS FOUR TYPES OF ORTHOGRAPHY CHANGES ON ALL WORDS PRESENTED
   TO IT INCLUDING ANY ENGLISH ON CARDS 3 - 21.
   TYPE 1:   LABIALIZATION.
             ANY CONSONANT + W GOES TO CONSONANT + U
             EG  BWAA -> BUAA
                 KWAA -> KUAA   ETC.
   TYPE 2:   VOWEL CLUSTERS.
             LAST TWO VOWELS OF THREE VOWEL SET WHICH ARE NOT IDENTICAL
             ARE CHANGED AS FOLLOWS:
             V2=I -> IY
             V2=U -> UW
             V2=E -> Y
            V2=O -> W
             EG  KAIA -> KAIYA
                 KOEA -> KOYA   ETC.
   TYPE 3    VOWEL CLUSTERS IN VERB STEMS:
             IN VERBS IF  V1 V2 V3 PRECEDES -GI OR -GE, AND IF V2 + V3 ARE
```

```
          THE SAME V3 IS REMOVED.
          EG   MAIIGI -> MAIGI
   TYPE 4    PRENASALIZATION.
          IN THE MIDDLE OF A WORD ONLY (NOT AT THE BEGINNING)
          B ->MB
          D -> ND
          G -> NG
          J -> NJ
          EG KADEGE -> KANDENGE
             AJA -> ANJA
   THE SUBROUTINE IS IN THREE SECTIONS.
   SECTION 1  REMOVES BLANKS FROM END OF STRING SUPPLIED.
   SECTION 2 SPLITS OFF EACH ENGA WORD TO PASS ACROSS TO INTERNAL
             SUBROUTINE CHANGE.
   SECTION 3 (SUBROUTINE CHANGE) PERFORMS THE FOUR TYPES OF ORTHOGRAPHY
             CHANGES.
   PARAMETERS
   SB - STRING CONTAINING ENGA WORDS
   JVB = 1 IF MAIN ENTRY IF A VERB
       = 0 OTHERWISE
   IFLA = 0 NO ERRORS IN PROCESSING
        = 1 MAIN ENTRY WOULD EXPAND TO GREATER THAN 25 CHARACTERS
        = 2 TONE CODE HAS COME AFTER A SEMI-VOWEL (Y OR W)
   JM = 1 MAIN ENTRY
      = 0 CARDS 3-21
   OTHER IMPORTANT VARIABLES.
   JPOS - STARTING CHARACTER OF NEXT WORD.
   SA,SD,SR,S INTERMEDIATE STRINGS TO HOLD WORDS TEMPORARILY.
   SUB - USED TO EXAMINE ONE CHARACTER AT A TIME
   ICOUNT - COUNTS NO OF VOWELS FOUND IN SEQUENCE
   L - HOLDS POSITIONS OF V2 AND V3
   FOUND - BRANCH TAKEN IF 3 VOWELS FOUND FOR TYPE 2.
   PROC - BRANCH TAKEN IF WORD ENDS IN GI OR GE FOR TYPE 3.
   R - BRANCH TAKEN IF 3 VOWELS FOUND FOR TYPE 3.                    */
DCL SB CHAR(80C) VARYING,
IFLA BIN FIXED (31,0),
(JVB,NWD,JM) BIN FIXED (31,0);
/*  SECTION 1 - REMOVES BLANKS FROM END OF STRING                    */
JPOS=1;
DO I=LENGTH(SB) TO 1 BY -1;
TEST: IF SUBSTR(SB,I,1)=' ' THEN
/*  SECTION 2 - CHECKS FOR ERROR TYPE 1 AND SPLITS OFF NEXT WORD     */
(STRINGRANGE): ONE: BEGIN;
DCL SA CHAR(I),
SUB CHAR(1),
SD CHAR(800) VARYING;
IF JM=1 & I>20 THEN DO;
IFLA=1;
RETURN;
END;
KOUNT=1;
SA=SB;
IF JM=0 THEN SB=(80C)' ';
SB='';
/*  START PROCESSING AT FIRST NON-BLANK CHARACTER.                   */
DO I=1 TO LENGTH(SA);
```

```
IF SUBSTR(SA,I,1)¬=' ' THEN DO;
JPOS=I;
SB=SB||SUBSTR(SA,1,I-1);
GO TO LOOP;
END;
END;
/*  SEARCH FOR NEXT WORD TERMINATOR AND CALL SUBROUTINE CHANGE        */
LOOP: DO I=JPOS TO LENGTH(SA);
SUB=SUBSTR(SA,I,1);
J=0;
IF SUB=';' | SUB=',' | SUB=' ' THEN DO;
DO J=1 TO LENGTH(SA)-I;
IF SUBSTR(SA,I+J,1)¬=' ' THEN
GO TO CALL;
END;
J=J-1;
END;
END;
CALL: N=I+J;
CALL CHANGE(SUBSTR(SA,JPOS,I-JPOS),IFLAG,SD);
/* TEST ERROR FLAG - RETURN ORIGINAL STRING IF ERROR. BLANK OUT
    REST OF STRING AND RETURN..                                       */
JPOS=N;
IFLA=IFLAG;
IF IFLAG¬=0 THEN DO;
SB=SA;
RETURN;
END;
SB=SB||SD||SUB;
KOUNT=KOUNT+1;
IF JPOS<=LENGTH(SA) THEN GO TO LOOP;
IF SUBSTR(SB,LENGTH(SB)-1,1)=SUBSTR(SB,LENGTH(SB),1)
THEN SB=SUBSTR(SB,1,LENGTH(SB)-1);
IF JM=1 THEN LEN=25;
ELSE LEN=800;
SD=(800)' ';
SB=SB||SUBSTR(SD,1,(LEN-LENGTH(SB)));
RETURN;
END ONE;
END;
IFLA=2;
RETURN;
CHANGE: PROCEDURE(S,IFL,SR);
DCL S CHAR(*),
SR CHAR(800) VARYING,
SUB CHAR(1),
L(2),
VOWEL(5) CHAR(1) INITIAL('A','E','I','O','U');
/* TYPE 1 CHANGE. SEARCH FOR CONSONANT + W AND CHANGE TO U.          */
IFL=0;
DO I=1 TO LENGTH(S)-1;
SUB=SUBSTR(S,I,1);
IF SUB<'A' | SUB>'Z' THEN GO TO END;
DO J=1 TO 5;
IF SUB=VOWEL(J) THEN GO TO END;
END;
```

```
IF SUBSTR(S,I+1,1)='W' THEN SUBSTR(S,I+1,1)='O';
END:  END;
/* TYPE 4 CHANGE. SEARCH FOR B,D,G,J AND INSERT REQUIRED CHARACTER
   BEFORE COPYING STRING INTO INTERMEDIATE STRING SR.                */
SR=SUBSTR(S,1,1);
DO I=2 TO LENGTH(S);
SUB=SUBSTR(S,I,1);
IF SUB='B' THEN SR=SR||'M';
IF SUB='D'|SUB='G'|SUB='J' THEN SR=SR||'N';
SR=SR||SUB;
END;
/* TYPE 2 CHANGE. SEARCH BACKWARDS THROUGH WORD FOR THREE VOWELS IN A
   ROW OR SEPARATED ONLY BY TONES. STORE POSITIONS OF THESE THREE.   */
I=LENGTH(SR);
ICOUNT=0;
LOOP: IF I<3 THEN GO TO NXT;
DO J=I TO I-10 BY -1;
IF J<1 THEN GO TO NXT;
SUB=SUBSTR(SR,J,1);
IF SUB='''' THEN GO TO ROUND;
DO K=1 TO 5;
IF SUB=VOWEL(K) THEN DO;
ICOUNT=ICOUNT+1;
IF ICOUNT=3 THEN GO TO FOUND;
L(ICOUNT)=J;
GO TO ROUND;
END;
END;
GO TO FIX;
ROUND: END;
FIX: ICOUNT=0;
I=I-1;
GO TO LOOP;
/* CHECK FINAL TWO VOWELS ARE NON-IDENTICAL AND THEN PERFORM PRESCRIB-
   ED CHANGES PAYING SPECIAL ATTENTION TO TONE CODES.                */
FOUND: IF SUBSTR(SR,L(2),1)=SUBSTR(SR,L(1),1) THEN GO TO FIX;
SUB=SUBSTR(SR,L(2),1);
IF SUB='A' THEN GO TO FIX;
IF SUB='E' THEN DO;
SUBSTR(SR,L(2),1)='Y';
IF SUBSTR(SR,L(2)+1,1)='''' THEN SR=SUBSTR(SR,1,J   )||''''||'Y'||
SUBSTR(SR,L(2)+2);
GO TO FIX;
END;
IF SUB='O' THEN DO;
SUBSTR(SR,L(2),1)='W';
IF SUBSTR(SR,L(2)+1,1)='''' THEN SR=SUBSTR(SR,1,J   )||''''||'W'||
SUBSTR(SR,L(2)+2);
GO TO FIX;
END;
IF SUBSTR(SR,L(2)+1,1)='''' THEN L(2)=L(2)+1;
IF SUB='I' THEN DO;
SR=SUBSTR(SR,1,L(2))||'Y'||SUBSTR(SR,L(2)+1);
I=L(2);
ICOUNT=0;
GO TO LOOP;
END;
```

```
IF SUB='U' THEN DO;
SR=SUBSTR(SR,1,L(2))||'W'||SUBSTR(SR,L(2)+1);
I=L(2);
ICOUNT=0;
GO TO LOOP;
END;
/* TYPE 3 CHANGE, CHECK FOR VERB AND LAST SYLLABLE BEING GI OR GE.    */
NXT: IF JVB=0 THEN GO TO RET;
LN=LENGTH(SR);
IF SUBSTR(SR,LN-3)='NGI'''|SUBSTR(SR,LN-3)='NGE''' THEN DO;
KK=LN-4;
GO TO PROC;
END;
IF SUBSTR(SR,LN-2)='NGI'|SUBSTR(SR,LN-2)='NGE' THEN DO;
KK=LN-3;
GO TO PROC;
END;
ELSE GO TO RET;
/* SEARCH BACKWARDS THROUGH WORD FOR THREE VOWELS                     */
PROC: ICOUNT=0;
DO I=KK TO KK-10 BY -1;
IF I<1 THEN GO TO RET;
SUB=SUBSTR(SR,I,1);
IF SUB='''' THEN GO TO ND;
DO J=1 TO 5;
IF SUB=VOWEL(J) THEN DO;
ICOUNT=ICOUNT+1;
IF ICOUNT=3 THEN GO TO R;
L(ICOUNT)=I;
GO TO NO;
END;
END;
ICOUNT=0;
ND: END;
GO TO RET;
/* CHECK IF FINAL TWO EQUAL AND DELETE LAST ONE.                      */
R: IF SUBSTR(SR,L(2),1)¬=SUBSTR(SR,L(1),1)THEN GO TO RET;
SR=SUBSTR(SR,1,L(1)-1)||SUBSTR(SR,L(1)+1);
RET: IF JM=1 & LENGTH(SR)>25 THEN IFL=1;
DO I=1 TO LENGTH(SR)-1;
IF SUBSTR(SR,I,1)='Y'|SUBSTR(SR,I,1)='W' THEN
IF SUBSTR(SR,I+1,1)='''' THEN DO;
IFL=2;
RETURN;
END;
END;
RETURN;
END CHANGE;
END ORTHO;
```

3.0 FORTRAN RETRIEVAL TECHNIQUES, by Katharine E.W. Mather

The retrieval technique was essentially the same for all the required printouts. The whole of the master file was read, sequentially if a count was being performed, or in sort order for all listings. If a record satisfied the retrieval criterion being applied (e.g. no tones in Enga item or no English gloss), the Enga main item and any other necessary information from the master record were printed. If cross references or other card types were required to be printed, the master file record was checked to see if these records existed. If so, their addresses were picked up from the master record, and they were read and printed.

3.1 *SINGLE-PHASE RETRIEVALS*

Two basic types of retrieval problems existed. In some cases one sub-group only had to be selected and printed according to the values in a particular field or fields. Examples of this single phase retrieval are the lists of words with no tones, no English gloss, no existential verb or no grammatical class. For these the master file was read and the required information printed.

3.2 *MULTI-PHASE RETRIEVALS*

The second type of retrieval problem involved those cases in which a large number of sub-groups existed within the file. There were, for instance, thirty complex verb sub-groups and over eighty semantic domain sub-groups.

To produce listings of the groups without reading the master file again and again checking the same field each time, the master file was read only once and lists of addresses of words of each sub-group were produced. These lists were then stored in direct access disk files.

A second program was used to read and print the information for any

or all sub-groups. The list of addresses for the required group was first read from the disk file. Then the master record for each entry and any associated items required were read and printed as described above (3.0).

Mather, K.E.W. "File Design and Creation". In Lang, A., Mather, K.E.W. and Rose, M.L. editors, *Information storage and retrieval: A dictionary project.* **D-8:19-64**. Pacific Linguistics, The Australian National University, 1972. DOI:*10.15144/PL-D8.19*