(12) LEVEL *III*
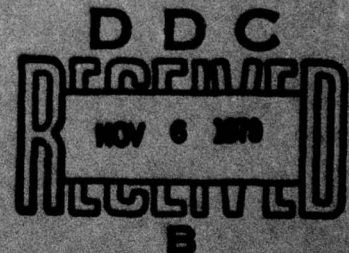
RADC-TR-79-179, Vol II (of five)
Final Technical Report
September 1979

AUTOMATED AIR INFORMATION PRODUCTION SYSTEM, PHASE I
Publishing Subsystem

Synectics Corporation

S. Edelblum
S. Borden

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

79 11·05 261

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-79-179, Vol II (of five) has been reviewed and is approved for publication.

APPROVED:

JOHN R. BAUMANN
Project Engineer

APPROVED:

HOWARD DAVIS
Technical Director
Intelligence & Reconnaissance Division

FOR THE COMMANDER:

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (IRRP), Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

(19) TR-79-179-VOL-2

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-79-179, Vol II (of five)	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AUTOMATED AIR INFORMATION PRODUCTION SYSTEM, PHASE I. Volume II. Publishing Subsystem.		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report. 26 Apr 77 — 26 Jul 78
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) S. Edelblum S. Bowden		8. CONTRACT OR GRANT NUMBER(s) F30602-77-C-0065
9. PERFORMING ORGANIZATION NAME AND ADDRESS Synectics Corporation 310 E. Chestnut Street Rome NY 13440		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 91370307
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (IRRP) Griffiss AFB NY 13441		12. REPORT DATE September 1979
		13. NUMBER OF PAGES 145
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: John R. Baumann (IRRP)

DDC
RECEIVED
NOV 6 1979
B

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Automated Cartography
Automated Flight Information Product Production
Text Composition and Editing
Electron Beam Recorder Film Output

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report details developmental efforts in providing the initial phase of a fully automated Air Information Production System (AAIPS) for the Defense Mapping Agency Aerospace Center. The system is used to produce DOD Flight Information Publications (FLIPS); Navigation/Planning and Special Purpose Charts; Special Products; and the Automated Air Facility Information File. The requirements, functional design and operational considerations of the AAIPS Charting, Air Facilities, and Publishing Subsystems are presented. The principal purpose of the three subsystems is the reduction of the labor (manual) required (Cont'd)

DD FORM 1473
1 JAN 73

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

407633   LB

Item 20 (Cont'd)

for the revision and publication of information critical to flight operations
and logistical planning. Improvement of response time between receipt of
changes to air navigation/air facilities data and the dissemination of new
data to all users, is also provided. The Publishing Subsystem permits publi-
cations to be produced on electronic equipment and extends the power and
flexibility of digital manipulation to the updating and reformatting of publi-
cations. The Air Facilities Subsystem provides maintenance of the AAFIF data
bases, selective data base retrieval, special report generation and generation
of formatted tape files for film negative output. The Charting Subsystem pro-
vides capture, revision and output of graphic data appearing throughout the
DMAAC Flight Information Publications, through preservation of data in
digital form and providing techniques to simplify alteration of the data.

TABLE OF CONTENTS

LIST OF FIGURES

SECTION I

INTRODUCTION

1.0 Background

DMAAC is responsible for producing long and complex books or publications that are part of the FLIP (Flight Information Package) that controls and supports safe navigation for military aircraft all over the world. The publications were prepared by typing them onto cards a line at a time on a varitype machine and storing the lines, in order, in a huge "tub" file (see Figure 1).

When it was time to print the book the cards were run through a special camera that photographed them (see Figure 2). Then the pictures or illustrations that went into the book were taped together with the photographed lines of text and sent to the printer.

A great deal of effort is involved in handling the cards and in keeping track of the cards as pages in the publication. For example, adding information to a page late in the update cycle could result in shifting whole paragraphs for many following pages, renumbering all of the pages and an unbalanced or unnecessary page or two.

1.1 Purpose

Use of the manual system has several disadvantages:

o   reliance upon a sequential camera which is increasingly difficult to maintain

o   lack of flexibility in reformatting publications to meet new requirements

o   a substantial manpower burden

The purpose of the DMAAC publishing subsystem is to:

o   permit the publications to be produced on modern electronic equipment

o   to extend the power and flexibility of digital manipulation to the updating and reformatting of the publications

o   to reduce the manpower required to produce the publications

Tub File

Cards

Card

Varitype

ABC
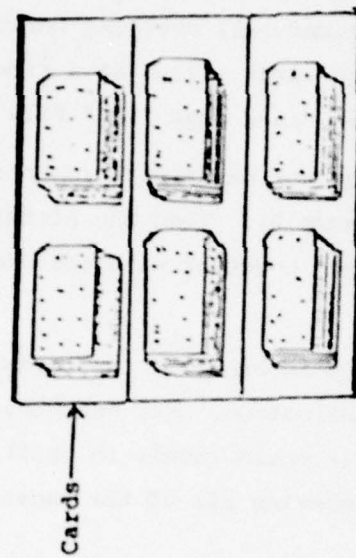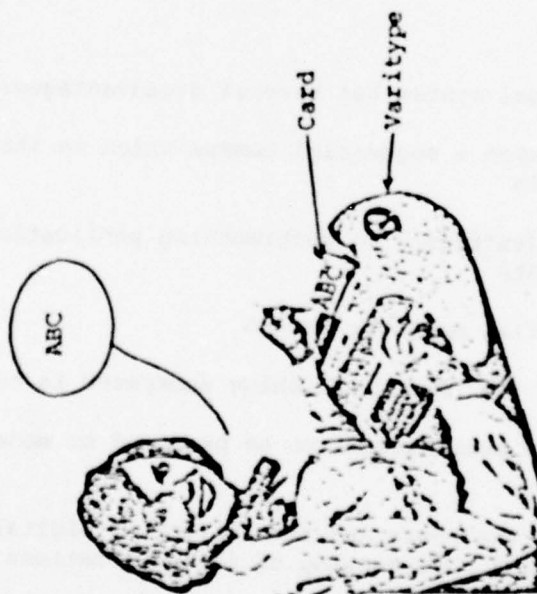
Maintaining a FLIP Publication on Cards

FIGURE 1

Cards

Camera

Illustrations

Negative

Paste Up
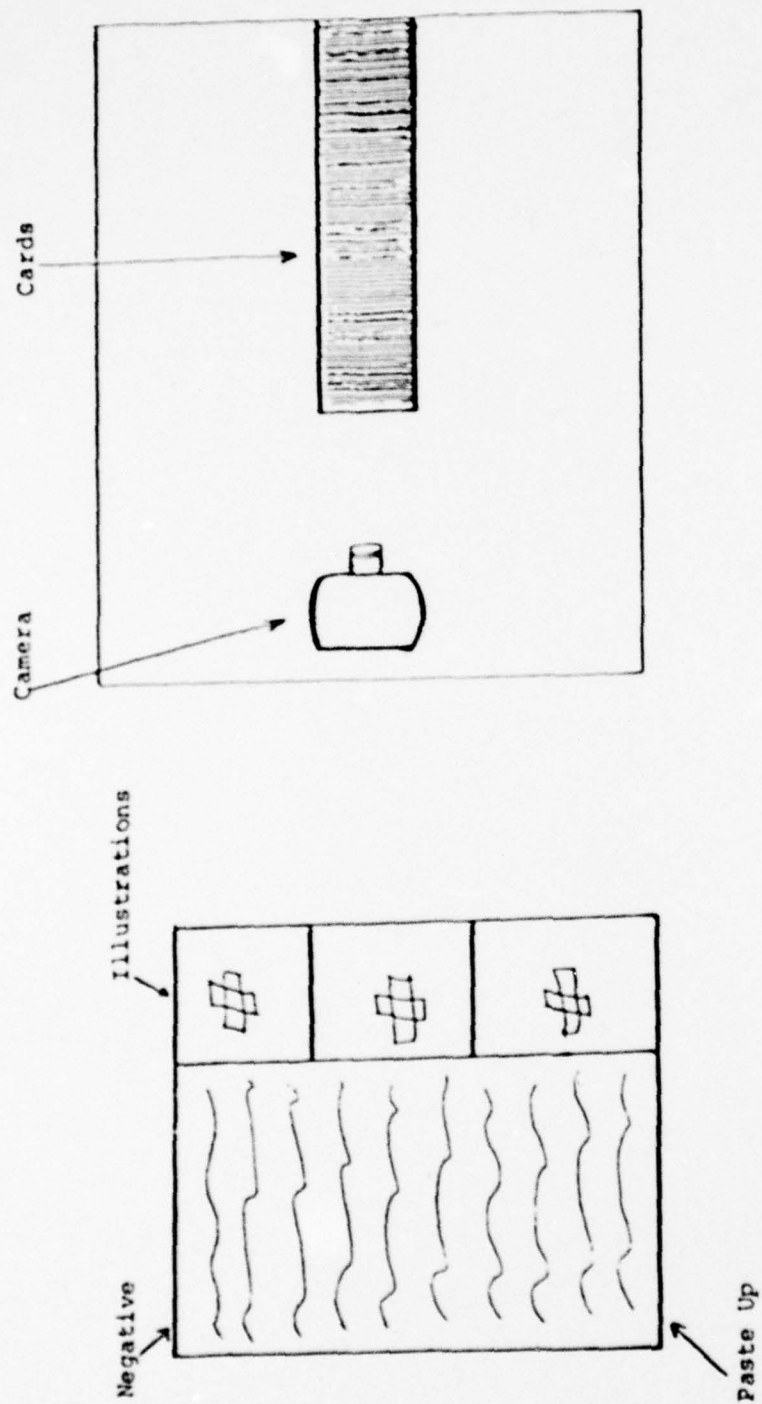
Preparing a Publication for Printing
FIGURE 2

SECTION II

PUBLISHING SUBSYSTEM FUNCTIONAL REQUIREMENTS

2.0 Publishing Subsystem Requirements and Capabilities

The Publishing Subsystem must provide a well human engineered, cost-effective, higher quality replacement of the manual document-to-tub file-to varitype-to-tub file-to-camera system that it replaces. The system must be:

- o   Easy to use
- o   Fast
- o   Highly automatic
- o   Capable of insuring high quality control

It must be easy to use to avoid extensive training, excessive errors, operator frustration and production delays. It must be fast to insure continuity of update sessions, shortened schedules, high productivity and low operator fatigue; it must be a flexible and responsive tool that justifies familiarity and confidence.

It must be highly automatic to relieve the operator of manual routine chores (e.g., justification and repagination); to enable the operator to concentrate almost exclusively on quality and content of the publication.

It must be capable of insuring high quality control through system proofing aids and reports, the ability to distinguish symbology in update mode, comprehensive error diagnostics, flexible error correction procedures and dependable automated assistance replacing complex and tedious manual procedures.

2.1 Workload

The Aeronautical Information Department (AD) of DMAAC publishes flight and air facilities information. These publications are used by DoD agencies, U&S Commands, military services and other authorized users for flight operations and logistical planning. These publications include:

- o   One Volume, FLIP General Planning (GP), published biannually
- o   Three Volumes, FLIP Area Planning (AP1, AP2, AP3), published quarterly

2-1

o   Three volumes, FLIP Area Planning Special Use Airspace (AP1A, AP2A, AP3A), published quarterly

o   One Volume, FLIP Military Planning Routes (AP1B), published quarterly

o   One Volume, Instrument Flight Rules USA, published biannually

o   One Volume, Visual Flight Rules USA, published biannually

o   Five Volumes, FLIP Area Supplements, published every four weeks

o   One Volume, Chart Updating Manual Supplement (CHUM), published monthly and biannually

o   One Volume, Bulletin Digest, published monthly and biannually

These publications result in about 140 issues, 22,000 pages and 1.5 million lines of text per year. The information requires approximately 18 million alphanumeric characters of which approximately 9 million characters are changed annually.

2.2 Data Base

The data base structure of the Publishing Subsystem has been designed to accommodate that data necessary for the production of the flight information publications prepared by the Defense Mapping Agency Aerospace Center and to provide for the ready access and maintenance of this data. Care has been exercised to avoid restrictions which would likely cause substantial redesign and conversion when enhancements to the subsystem result in additional maintenance functions and access requirements.

Where repeated occurrence of a data item would be a particular burden on storage, maintenance, or manual editing, data items have been incorporated to allow relief as in the case of alternate font and tabulation specifiers.

The data base is intended to be supported by a Data General Real Time Disk Operating System, although there are no inherent aspects which would encumber its support on any system which supports a mass storage device with equivalent characteristics to that of moving head disk.

SECTION III

PUBLISHING SUBSYSTEM DESIGN

3.0 Publishing Subsystem Design Criteria

The DMAAC Publishing Subsystem was designed to create and maintain complex flight information publications (FLIP) used by military pilots all over the world.

The system as developed possesses many unique advantages over existing systems, be they of word processing or typesetting origin:

a. Utilization of a CRT which displays 3960 characters of text at once (132 x 30).

b. On-line hyphenation and justification of typesetting quality.

c. Multi-font and font size capability including:

    (1) Individual character width definition.

    (2) Automatic leading control, permitting any font/size combinations.

    (3) Rapid identification of font/size on the CRT.

    (4) Accommodation of special symbols and foreign languages.

d. Complete automatic repagination capability including page breaking controls, column balancing, header generation by specification and from context, page numbering format control (e.g. 4-21 where chapter number prefaces page number).

e. Comprehensive, simple, edit capabilities which are easier to use than typical word processing systems (because they combine line context edit capabilities with cursor positioning capabilities) yet as powerful as typesetting systems because automatic on-line justification occurs simultaneously with update. These include:

    (1) Insert

    (2) Delete

    (3) Change (with repeat if desired)

    (4) Search (with repeat if desired)

    (5) Position cursor (forward, backward, up, down, by line number, etc.)

    (6) Scrolling

    (7) Hyphenation defeat or override

f. Multiple column definition.

g. Nesting of columnation (columns within columns).

h. Left, right and center justification, applying to the page width, column width or between any two specified tab positions (e.g., center justify between positions (1/2 pica) 10-50).

i. Diagram insertion and maintenance capability eliminates manual paste up.

j. Tabulation and automatic indent controls.

k. Directory access to multiple versions of books and chapters.

l. Flexible book and chapter format definition, or on-line redefinition.

m. Complete global edit capability permitting string substitution or deletion and simultaneous rejustification at high speed throughout the document.

n. Password protection by function.

o. Maintenance of a line index within a page enabling direct access to any part of the page either by cursor positioning, scrolling, or instantly, by specifying a line number.

p. Maintenance of update bars for technical publications.

q. Maintenance of CRT and proof list proofing flags indicating where changes have occurred.

r. Complete logging of all commands and production of a management information report based upon the log.

s. Auxiliary monitor which displays typesetting information allowing manual override of automatic decisions.

t. Proofing mode which logs approval or review status enabling document control.

u. Ability to handle a variety of special symbols, graphics or logos.

v. User definition and automatic system maintenance of and access to any desired index to the entire publication (e.g., access by paragraph number, embedded text, titles, headers, footnotes, etc.)

w. Maintenance of pages as distinct entities enabling document update and instant access by referring to the original page number; regardless of additions, changes, reformatting, or deletions in the interim. Text is not redistributed among pages until repagination is specified. After repagination the original is separately maintained and available as a "version."

x. Storage capacity is available to store from 50,000 to 1,200,000 pages of information depending upon configuration selected.

y. Programmed in a high level language (FORTRAN), allowing rapid additional programming and support to meet specific user requirements now and in the future. System improvements will be available to users without hardware upgrade.

z. Human engineered; training courses available to permit comprehensive operator training.

aa. System controls and complex features when not invoked remain invisible to the operator. The system can perform at the operator's level; in less than one hour a typist with no previous exposure to the system can compose typographic quality correspondence!

bb. Usable in a multi-user environment on a suitable, general purpose (currently operational on a Data General Eclipse minicomputer) computer, allowing users to run other applications of their choice simultaneously. Furthermore, both the terminals and the printer are usable for other user applications when not used for publishing or word processing applications.

cc. Automatic column balancing and page makeup.

dd. Vertical tabulation.

ee. On-line HELP function for interactive format and repagination specification assistance.

ff. Columns, page width, tabs, indents, font/size, update bars and hyphenation controls propagate throughout the entire document but can be easily redefined at any chosen place in the text; providing complete and flexible format control of any document regardless of size or makeup.

gg. User definable function keys and mnemonics permitting the automatic entry of any combination or commands and textual date of any length. There is no limit to the number of mnemonics (trillions!) which may be defined for this purpose.

3.1 Publishing Subsystem Structure Overview

The structural components of the Publishing Subsystem are depicted in Figure 3. The major functional areas are:

o   Log-on/log-off

o   Publication identification and creation

o   Display manipulation

o   Update pages

o   File management

o   Publication reports and statistics

o   Repagination and output to EBR

o   Publication proofing

3.1.1 Log-on/Log-off

This process permits access to the system only after the entering of a password. A hierarchy of password validity insures that only authorized
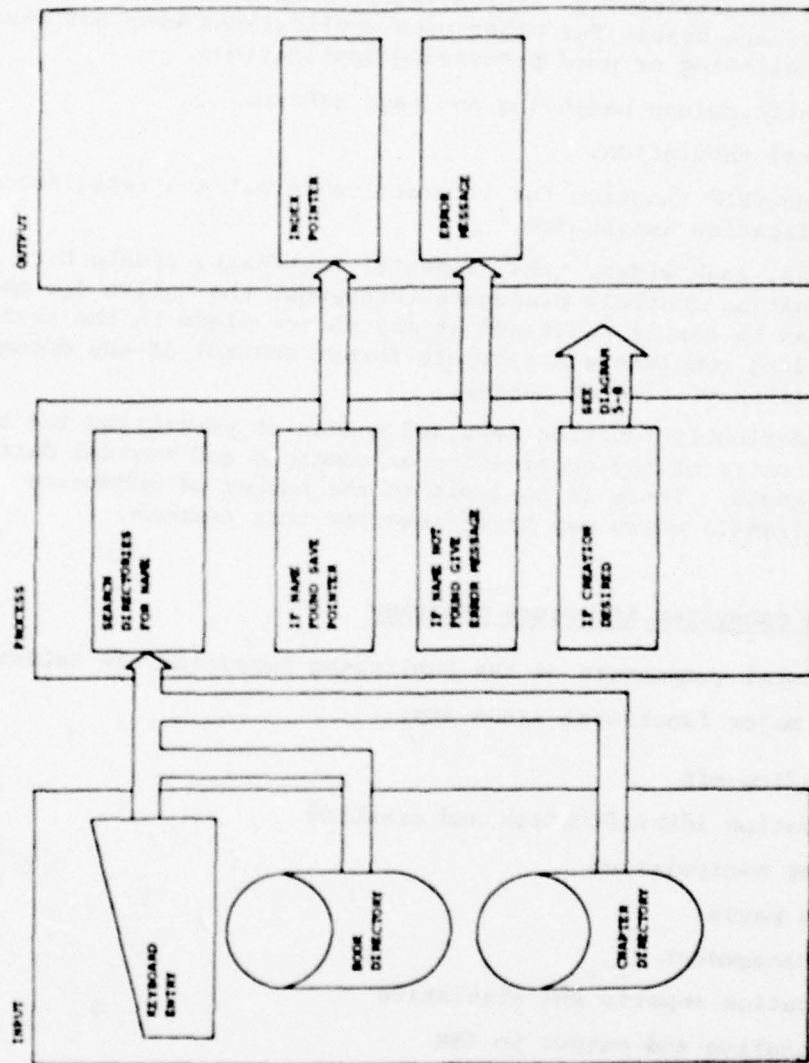
Figure 3 Diagram Identify Select Book or Chapter

personnel can utilize specific system functionality. For example, the password QRSTWVW may grant access to publications for revision and update but not for repagination, delete book, etc.

3.1.2 Publishing Identification and Creation

The functions to be performed under this category (see Figure 4) are:

o   Print book or chapter

o   Identify or select book or chapter

o   Update book or chapter

A book file (see Section 3.2) serves as an index to all chapters and chapter-versions (current and past publications having the same title and format). A chapter file contains format definition and heading information as well as the text and operation codes that form the substance of the publication. Chapters are divided into page divisions until the cut-off point in a publication cycle results in repagination and output. This is because the best reference an analyst has to a change is the page on which it appeared in the last published version. If page divisions were not maintained during the update cycle it would be difficult to reach desired places within the publication as text migrated from page to page reflecting update activity.

The publishing subsystem maintains indices to pages within chapters. This practice permits the system to automatically generate chapter page headers, ensure chapter integrity during repagination, and provide faster response times than associated with a simple book/page index.

Figure 5 illustrates the functionality of user selection and identification of book and chapter-version which occurs prior to any update session.

Figure 6 illustrates the function of actually changing the defining parameters of a book index or chapter header. For a more detailed description of how this function is implemented see Section 3.3, Personnel Functions, where an explicit example of the update procedure is depicted.

PUBLISHING ID/CREATION

UPDATE BOOK OR CHAPTER

IDENTIFY/SELECT BOOK OR CHAPTER

PRINT BOOK OR CHAPTER
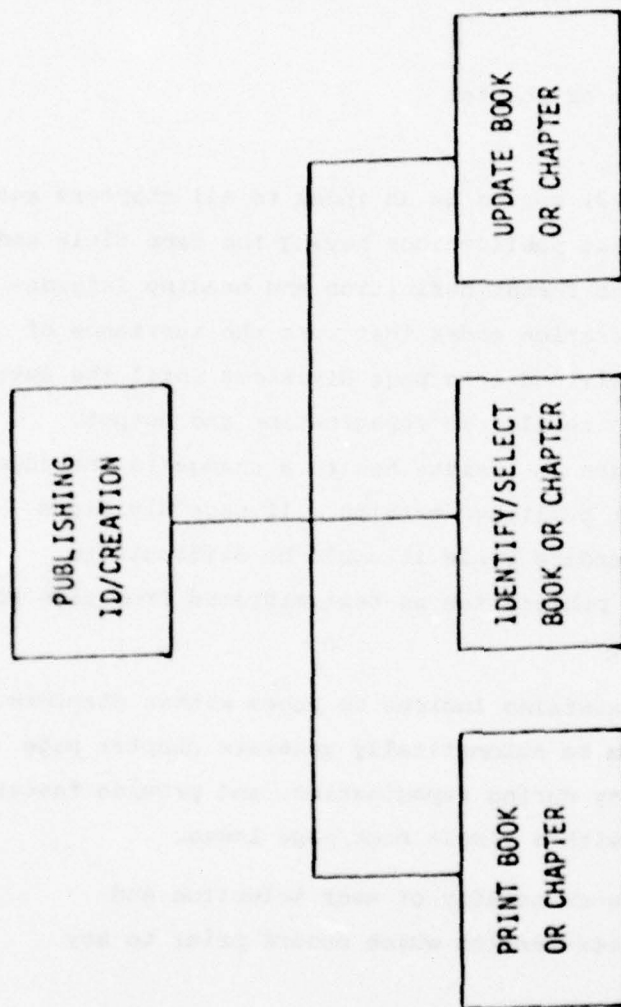
Figure 4   Publishing ID/Creation
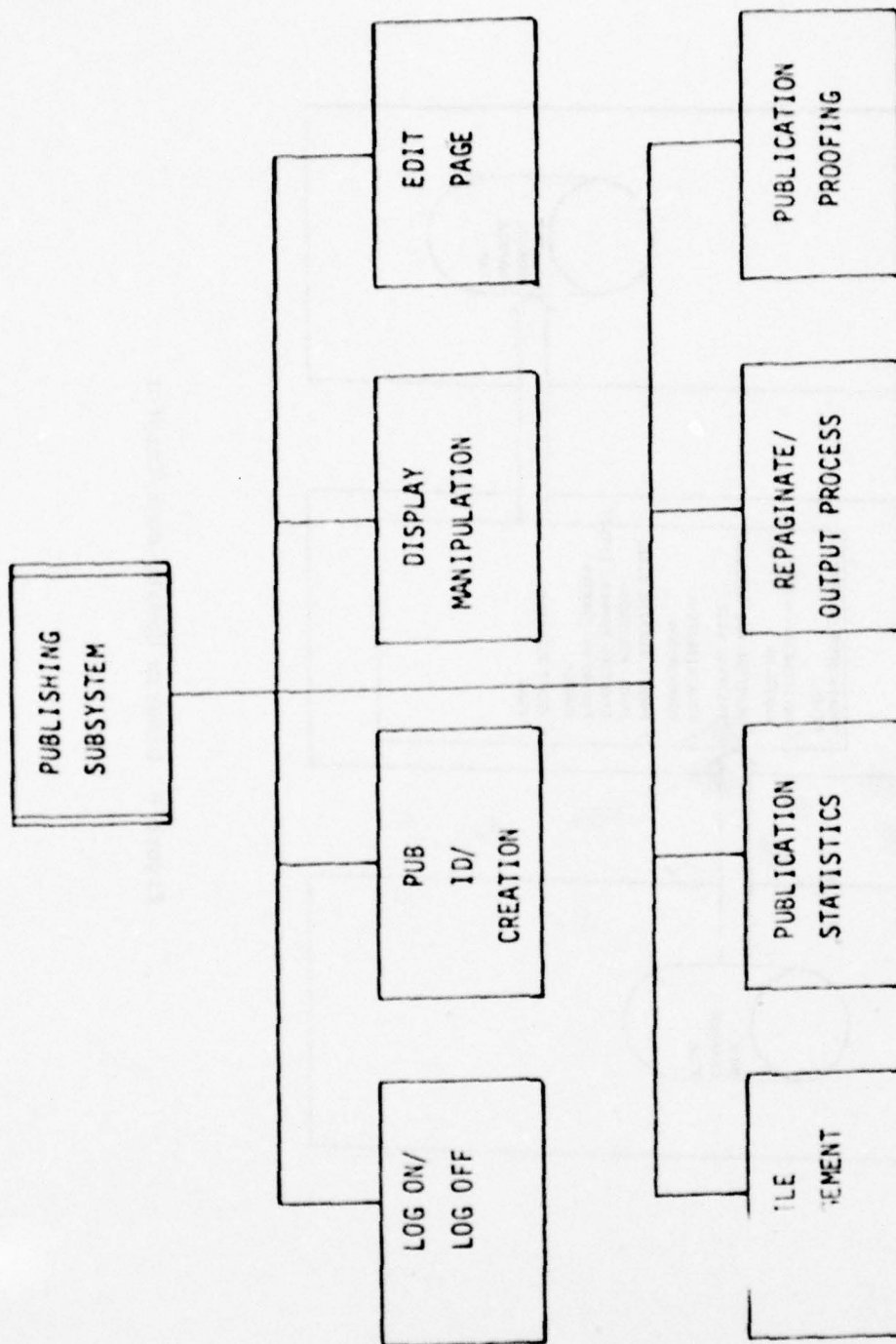
Figure 5   Publishing Subsystem

IDENTIFY/SELECT
BOOK OR CHAPTER

SEE
DIAGRAM
5.5

UPDATE BOOK OR CHAPTER
FILE:

/ ADD/DELETE/CHANGE
CHAPTER

/ VERTICAL AND HORIZON-
TAL PAGE SIZE

/ FOUR SIZE/FONT
COMBINATION

/ PAGE NUMBERING SIZE,
FONT, POSITION,
STARTING NUMBER, TITLE
FORMAT OR CHAPTER
NUMBER

/ INDEX REQUIRED
FLAG

KEYBOARD
ENTRY

BOOK
CHAPTER
FILE

UPDATED
BOOK OR
CHAPTER
FILE
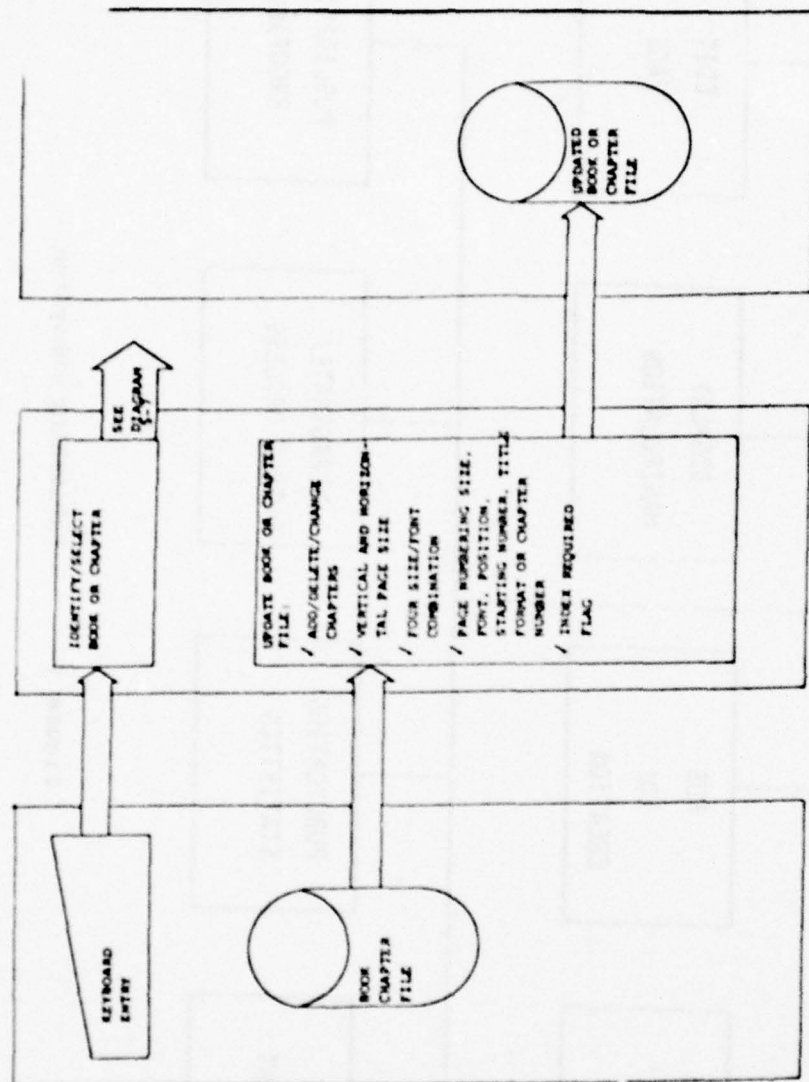
Figure 6   Diagram Update Book/Chapter

3-8

3.1.3 Display Manipulation

At the core of the Publishing Subsystem is the ability to revise the contents of an existing chapter. To actually perform this function, however, an interface to the system must exist which permits the rapid display of that body of text which must be updated. It is this vital function of getting to a page, moving the display cursor within the page and scrolling the display which is functionally incorporated into Display Manipulation (see Figure 7).

Figure 8 illustrates the two methods of reaching a page before updating a chapter. If a page number is specified, the system will retrieve the corresponding page of data.

If a page number is entered, the page is directly retrieved and the first 12 lines of the page are displayed; one may scroll forward to view the rest of the page.

Indexed documents permit use of the "+" and "-" keys to go from page to page displaying facility names as a guide to page content. Figure 9 shows the function of paging forward or backward in this manner.

Figure 10 shows the display functions available while actually revising a page. Cursor-control positions the cursor for add or delete functions, as well as causing the screen to scroll when the cursor is ordered off the screen.

The display commands offer a more rapid means of controlling which part of the page is to be shown on the screen. Using the display function also provides the advantage of maintaining the cursor in the same position on the screen.

3.1.4 Update Page

The Update page function provides absolute control of the content and format of the document. Figure 11 shows this function as composed of:

- o   Manipulate format
- o   Search for string
- o   Change/insert/delete string
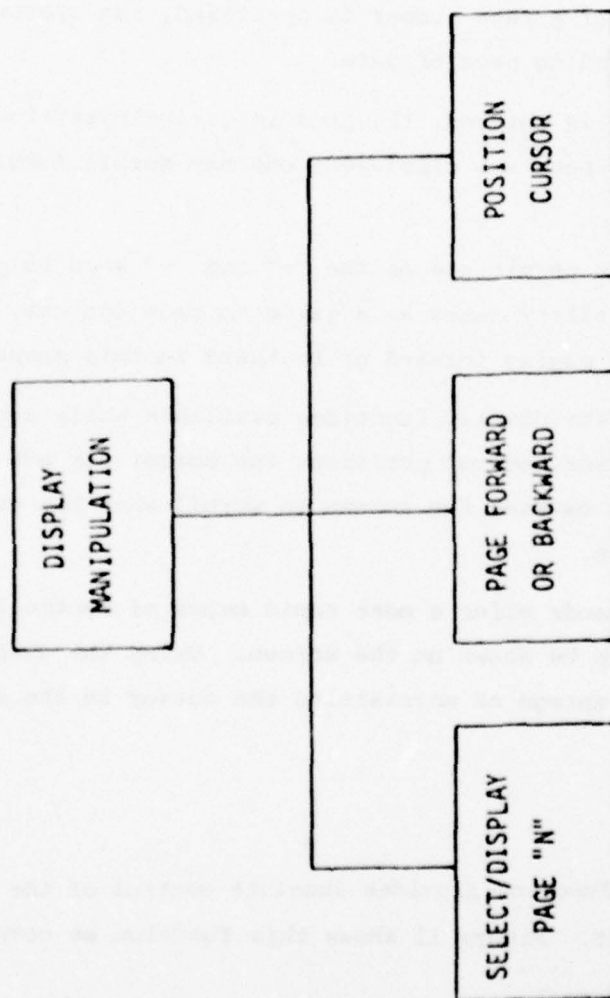
Figure 7   Display Manipulation
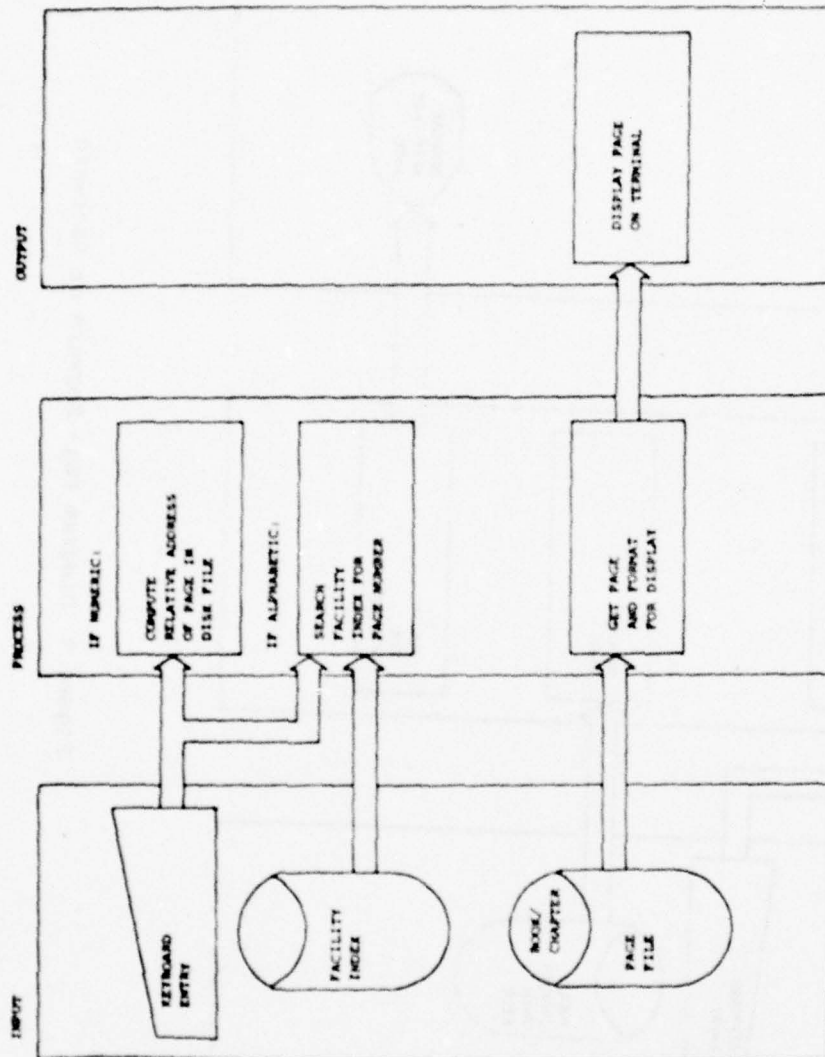
Figure 8 Diagram Select/Display Page "N"

3-11

INPUT

PROCESS

RELATIVE ADDRESS OF CURRENT PAGE

KEYBOARD ENTRY

BOOK/ CHAPTER PAGE FILE

COMPUTE RELATIVE ADDRESS OF NEXT/LAST PAGE

RETRIEVE PAGE

FORMAT FOR DISPLAY

DISPLAY NEXT/LAST PAGE
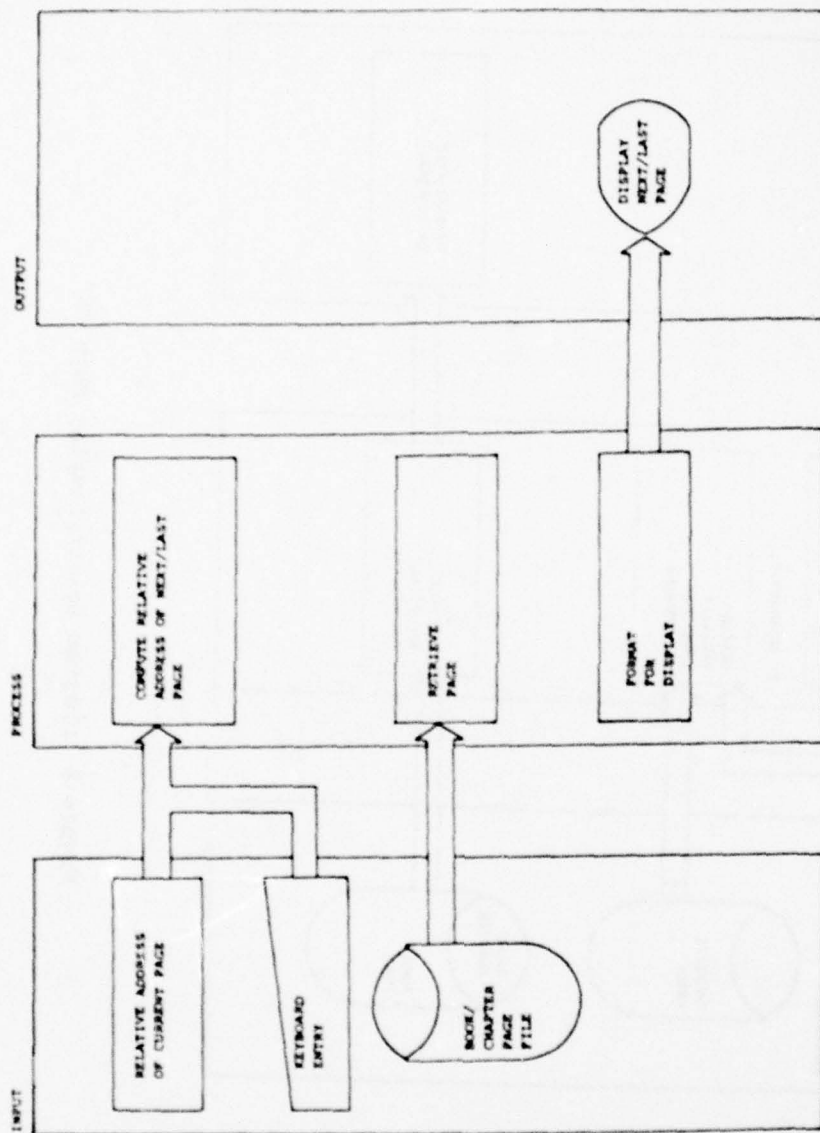
Figure 9 Diagram Page Forward or Backward

3-12
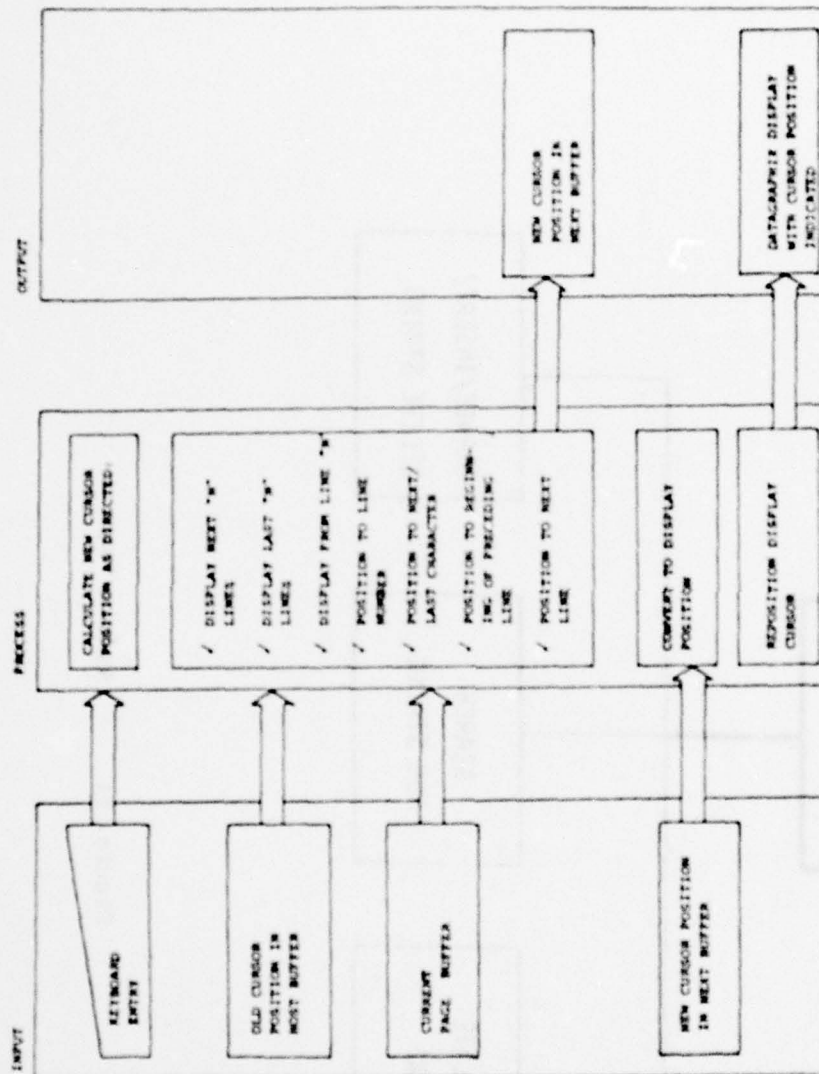
Figure 10 Diagram Position Cursor

3-13
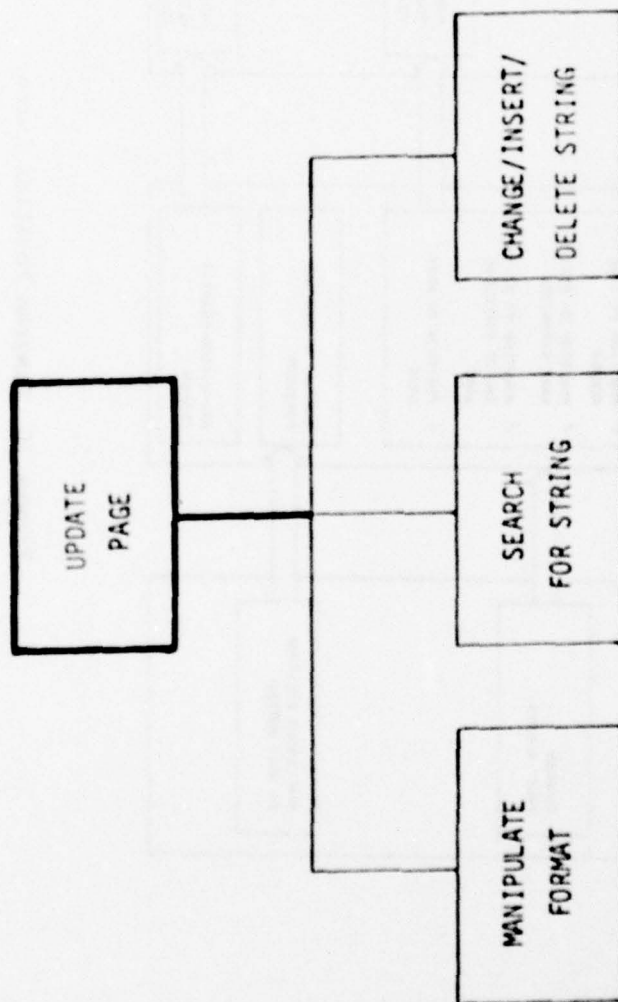
Figure 11   Update Page

3-14

Figure 12 illustrates the ability of the system to store and manipulate format operands such as tabulation, columnation, justification, hyphenation control, font/size selection and diagram insertion. Essentially, the operand supplied by the operator causes a command to be stored in the body of the text which results in correct formatting both on the screen and during pagination.

Figure 13 shows the search for a given string or combination of characters. It is used both as a cursor positioning function and as a vital subfunction in the changing of character groups.

Figure 14 illustrates the function of actually inserting characters and operands into the page of text. Positioning the cursor to a character and invoking the delete function causes the character to be removed from the page of text. The change command is used to replace a given character group or string with another of the same or different size. Automatic justification will expand or contract the line accordingly after this command is invoked.

Additionally this command can be invoked to change any number of consecutive occurrences of a string on a page to another string of same or different size.

3.1.5 File Management

Utilities having to do with file management (see Figure 15) include:

o   Update log file

o   Update font file

o   Store package files

o   Create book or chapter

A log (see Figure 16) of all system functions is kept including:

o   Function performed

o   Time of day performed

o   Date

o   Password identification

INPUT

KEYBOARD ENTRY

OLD PAGE

PROCESS

INSERT CONTROL COMMANDS INTO PAGE FOR FOLLOWING FUNCTIONS:
/HORIZONTAL AND VERTICAL TABS
/PAGE COLUMNATION
/FONT/SIZE SELECTION
/LEFT/CENTER/RIGHT JUSTIFICATION
/INSERT DIAGRAM
/AUTOMATIC HYPHEN-ATION CONTROL

DEFINE CHAPTER FORMAT:
/ PAGE SIZE
/ PAGE NUMBER
/ POSITION

OUTPUT

UPDATED PAGE

CHAPTER HEADING
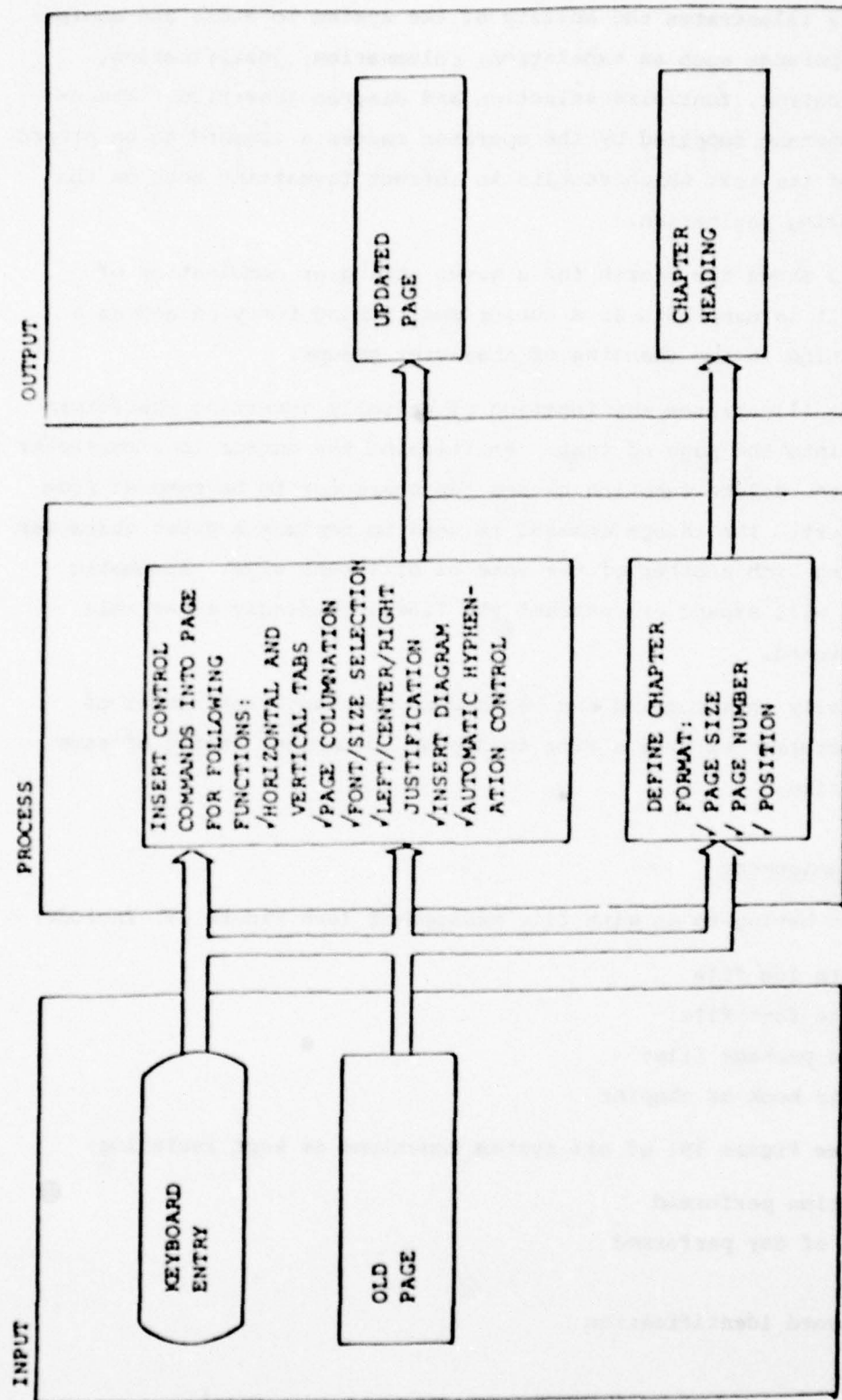
Figure 12   Diagram Manipulate Format

3-16

INPUT

KEYBOARD ENTRY
OR COMMAND

PAGE
CONTENTS

PROCESS

EXAMINE PAGE
CONTENTS FOR
STRING EQUALITY

POSITION CURSOR
WITHIN INCORE
BUFFER

OUTPUT

CURSOR
POSITIONED
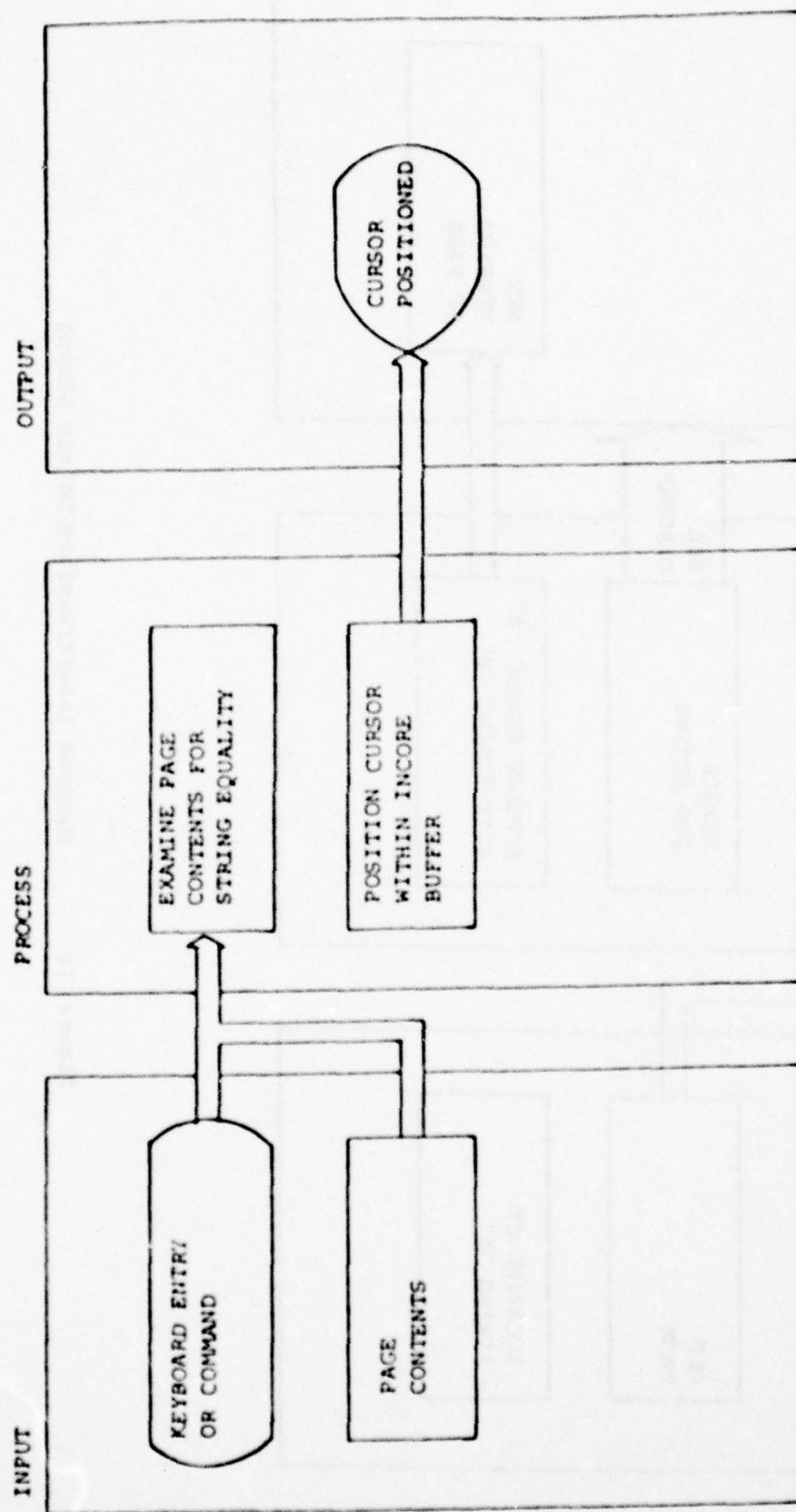
Figure 13   Diagram Search for String

3-17
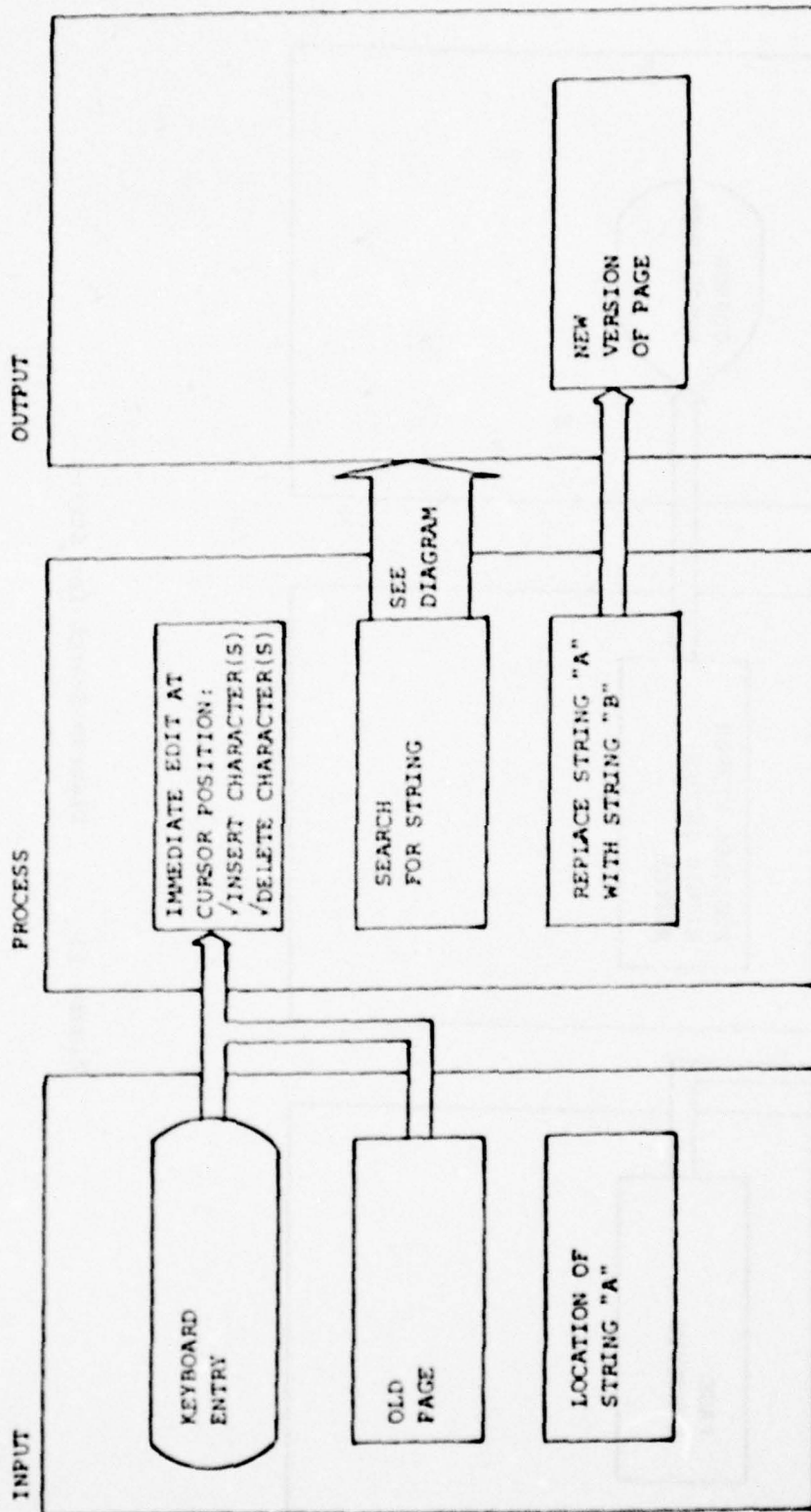
Figure 14   Diagram Insert/Replace/Delete String
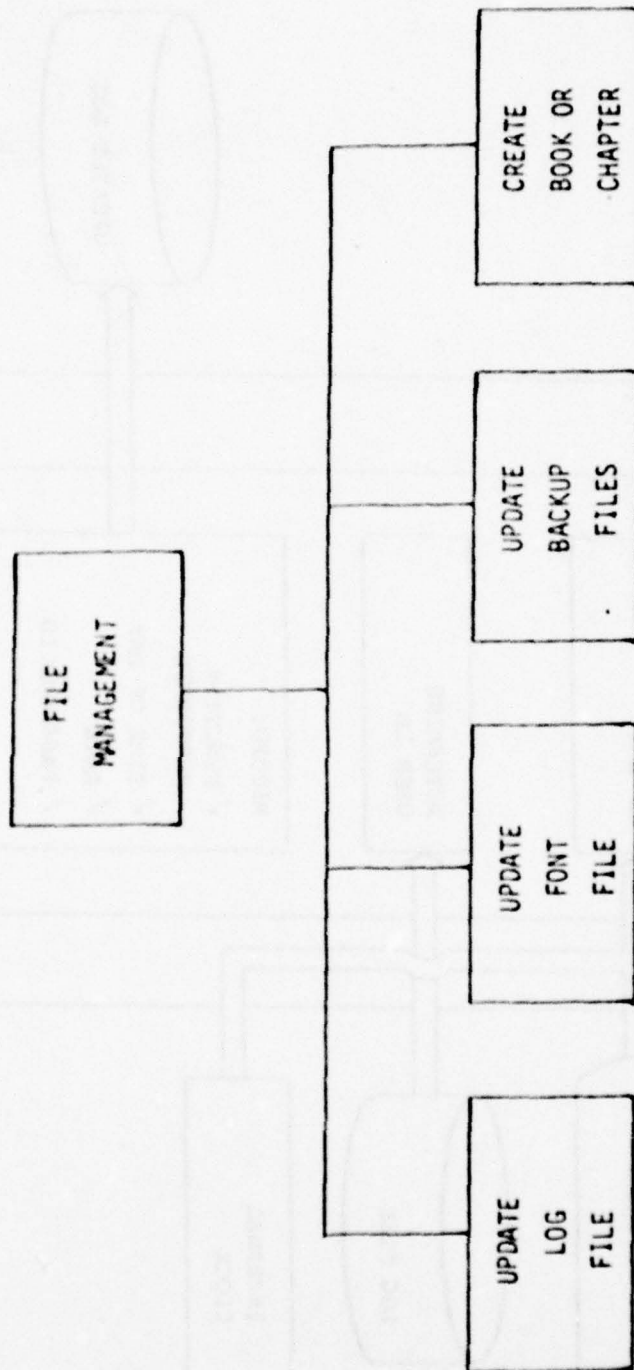
Figure 15   File Management
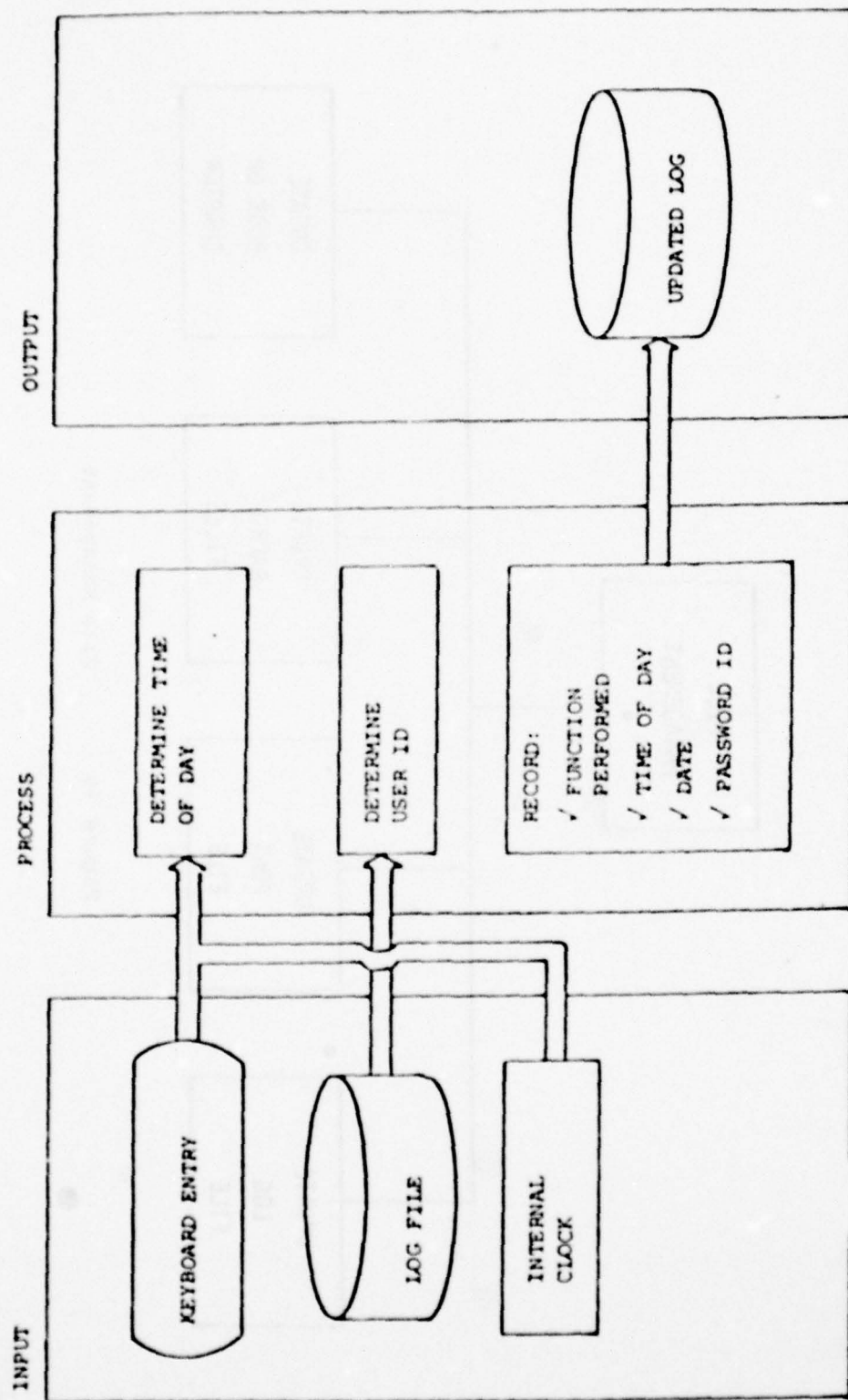
Figure 16   Diagram Update Log File

From this log management statistics may be generated. Additionally a proof management report reflecting the status of a publication through proofing is available to insure adequate proof management. Figure 17 shows a screen version of this report which may also be obtained in hard-copy form.

The font file forms the link between the representation of characters within the publishing subsystem and their depiction upon the EBR output device. Figure 18 illustrates this process. Essentially, the contents of this file are the codes by which the EBR will recognize the character set- and size-information regarding the size of the character to be produced on the film prior to blow-up to lift size.

Figure 19 shows the back-up procedure for Publishing Subsystem files. This procedure utilizes MRDOS system software.

Figure 20 illustrates the procedure for creating an empty book or chapter with default values. This is used in rare circumstances to merge books or chapters.

3.1.6 Publication Statistics

Figure 21 shows the functional areas of publication statistics to be comprised of collection and display of the statistics.

Figure 22 illustrates the method of statistics capture, which occurs during the creation of the system log file. Additionally since all changed characters are flagged it is possible to derive a number representing the number of characters entered to the publication (though not the number deleted). Net change can be calculated by comparing the character counts for successive publications.

Figure 23 illustrates how these statistics are normally retrieved and printed for management use.

3.1.7 Repaginate and output process

The repaginate and output process is the most complex function of the Publishing Subsystem (see Figure 24). All the embedded operands controlling

BOOK	CHAP/VERS	PAGE	ACCESS	DATE	TIME	STATUS	#INS	#DEL
AP2	1/ 1	1	UPDATE	5/ 8/78	728	FILED	9	53
AP2	1/ 1	1	UPDATE	5/ 8/78	728	FILED	1	5
AP2	1/ 1	1	UPDATE	5/ 8/78	728	FILED	5	22
AP2	1/ 1	1	UPDATE	5/17/76	858	FILED	0	1
AP2	1/ 1	1	UPDATE	5/17/78	1149	FILED	1	0
AP2	1/ 1	2	UPDATE	5/ 8/78	728	FORMATED	0	0
AP2	1/ 1	2	UPDATE	5/17/78	858	FORMATED	0	0
AP2	1/ 1	2	UPDATE	5/17/78	1149	FORMATED	0	0
AP2	1/ 1	3	UPDATE	5/ 8/76	728	FORMATED	0	0
AP2	1/ 1	3	UPDATE	5/17/78	858	FORMATED	0	0
AP2	1/ 1	3	UPDATE	5/17/78	1149	FORMATED	0	0
AP2	1/ 1	4	UPDATE	5/17/78	858	FORMATED	0	0
AP2	1/ 1	4	UPDATE	5/17/78	1149	FORMATED	0	0
AP2	1/ 1	5	UPDATE	5/ 8/76	838	FILED	3	42
AP2	1/ 1	5	UPDATE	5/ 8/78	911	FILED	1	3
AP2	1/ 1	5	UPDATE	5/17/78	858	FILED	0	2
AP2	1/ 1	5	UPDATE	5/17/78	858	FILED	0	0
AP2	1/ 1	5	UPDATE	5/17/78	1149	FILED	0	0

Figure 17   Screen Version

INPUT

KEYBOARD
ENTRY

FONT FILES

PROCESS

UPDATE:
✓ EBR CODE FOR
  EACH ASCII CHAR.
✓ HORIZONTAL
  INCREMENT (IN
  POINTS) FOR A 9
  POINT CHARACTER
✓ EBR VALUE FOR
  SPECIAL
  CHARACTERS
✓ HORIZONTAL
  INCREMENT (IN
  POINTS) FOR A 9
  POINT SPECIAL
  CHARACTER

OUTPUT
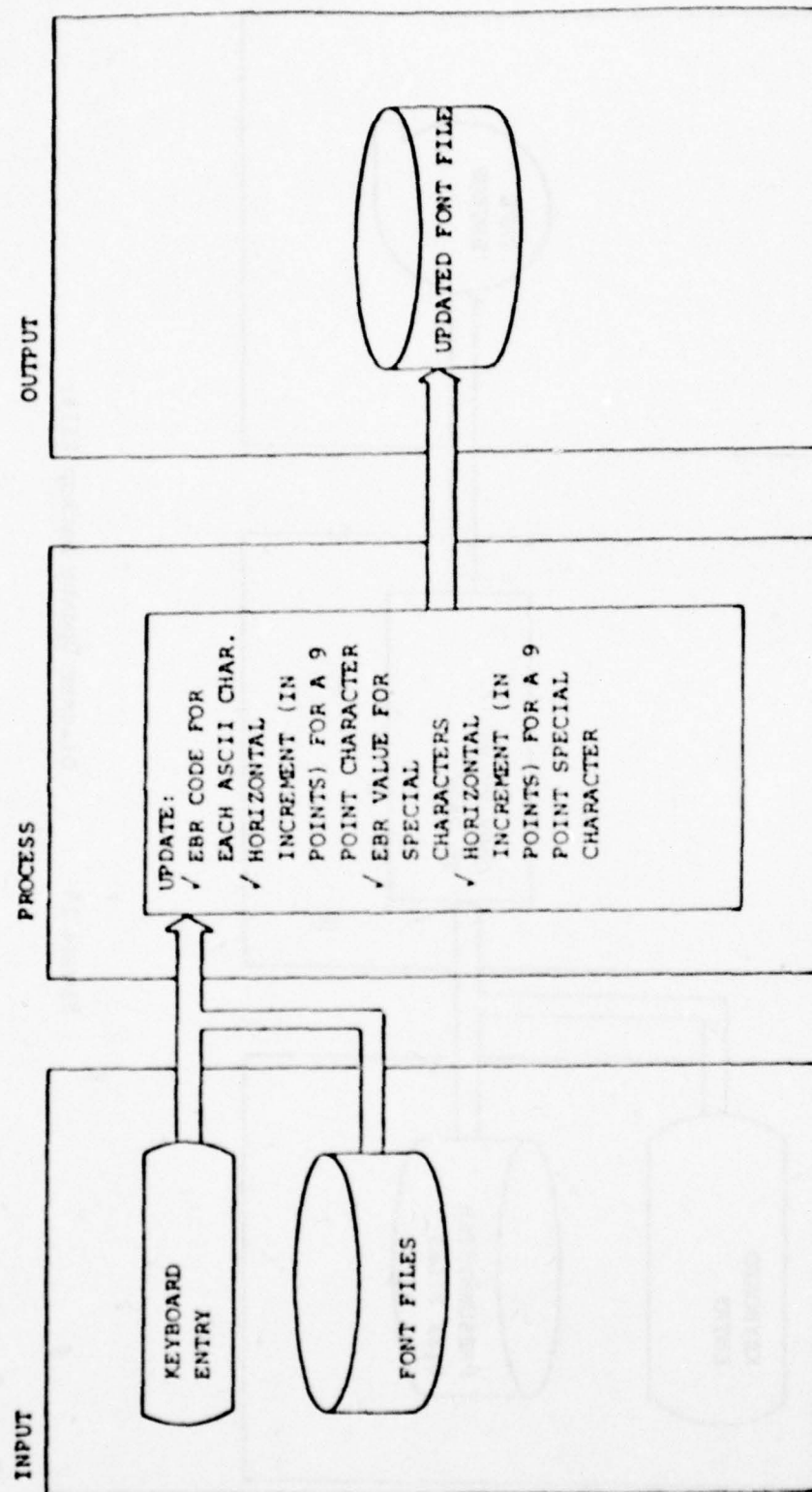
UPDATED FONT FILE

Figure 18   Diagram Update Font File

3-23

INPUT

PROCESS

OUTPUT

KEYBOARD ENTRY

PRESUBSYSTEM DISK FILES

COPY FILES TO TAPE

TAPE BACKUP
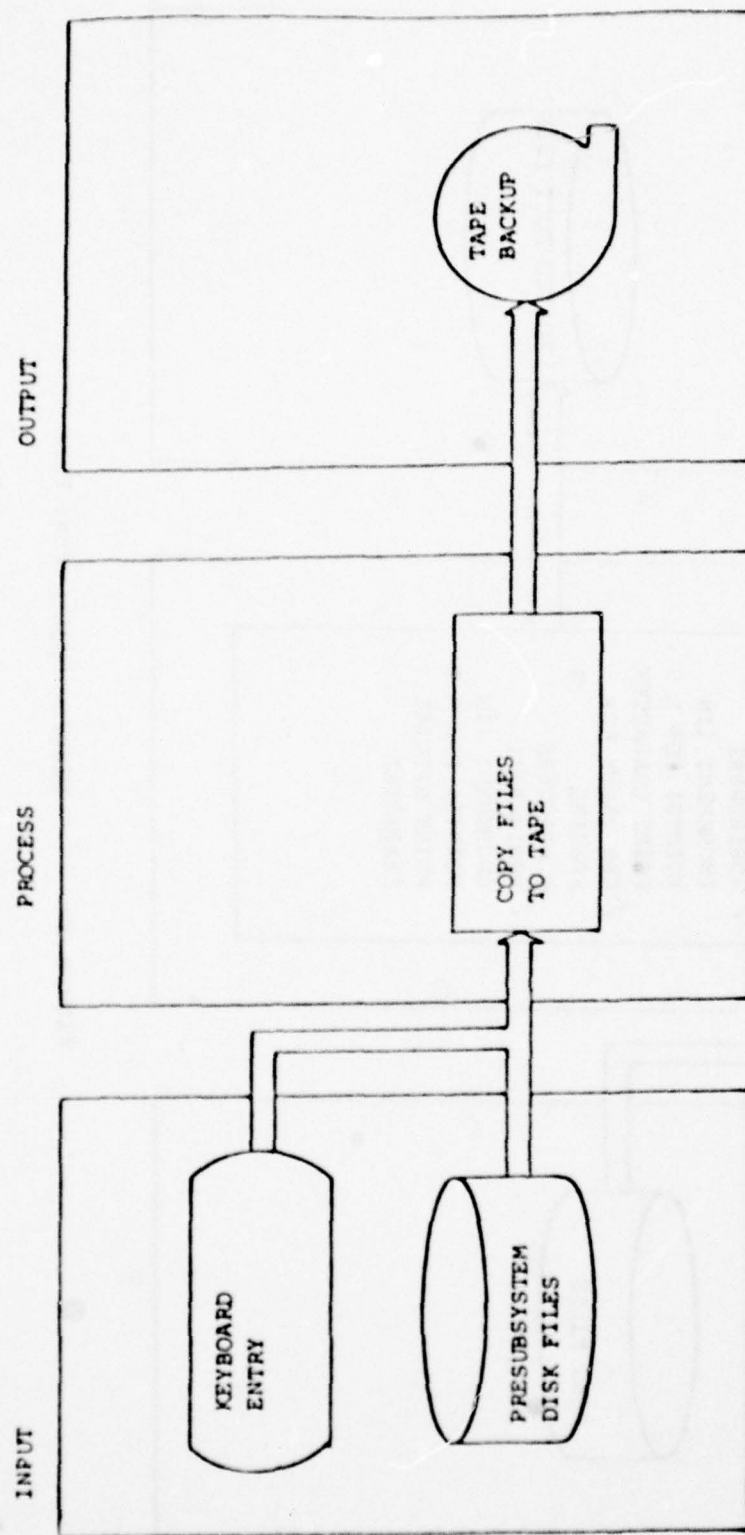
Figure 19   Diagram Update Backup Files

3-24
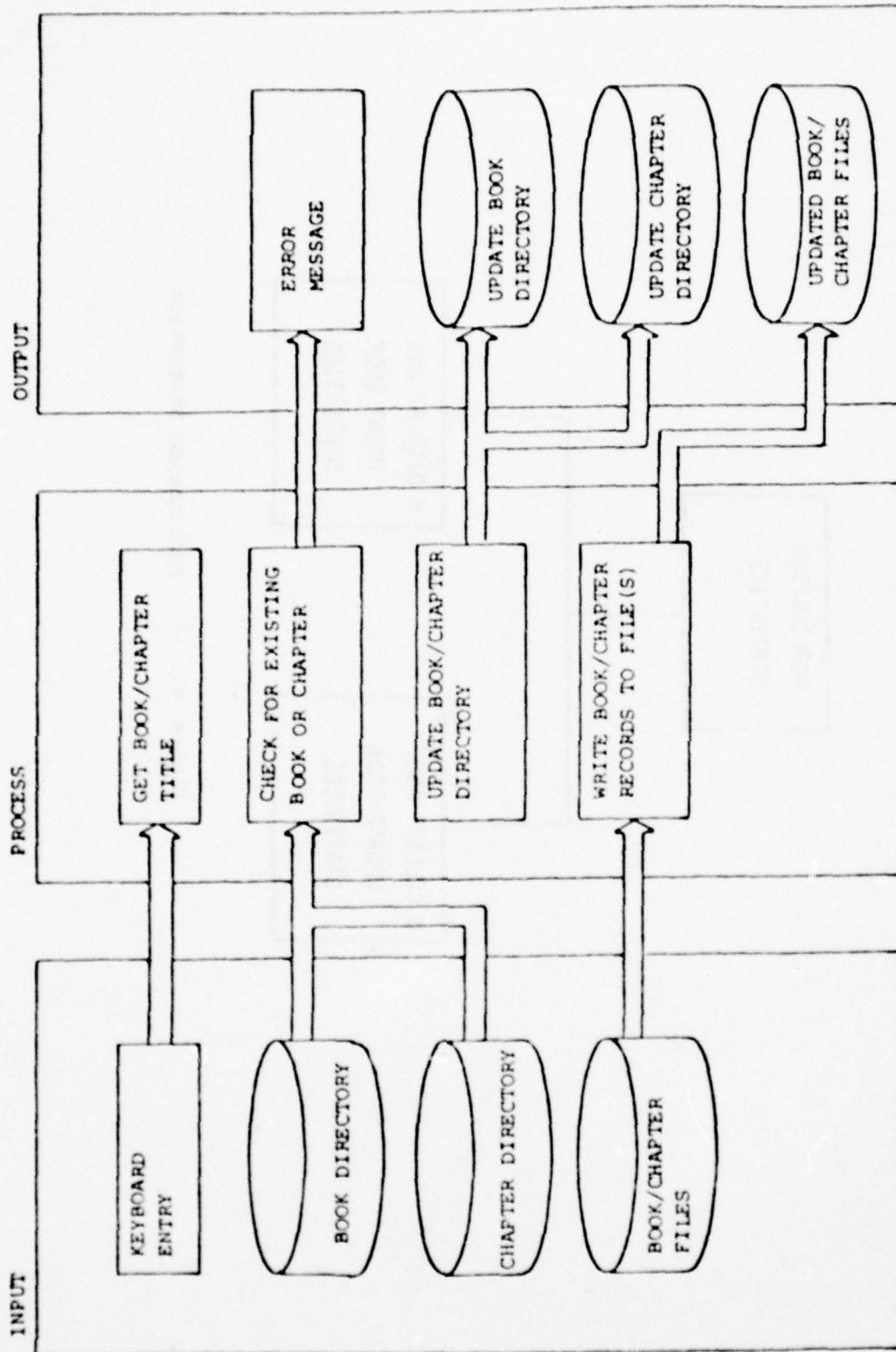
Figure 20   Diagram Create Book or Chapter

3-25
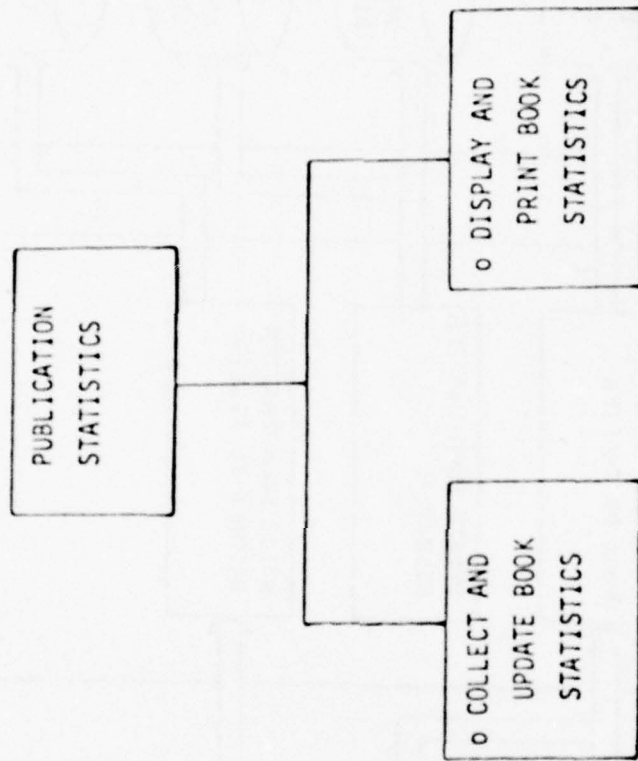
Figure 21   Publication Statistics

INPUT

BOOK FILE

LOG

PROCESS

ZERO CUMULATIVES

FOR EACH LOG ENTRY

ADD TO CUMULATIVE AND PRINT ENTRY

OUTPUT

LOG LISTING

CUMULATIVE SUMMARY DATA
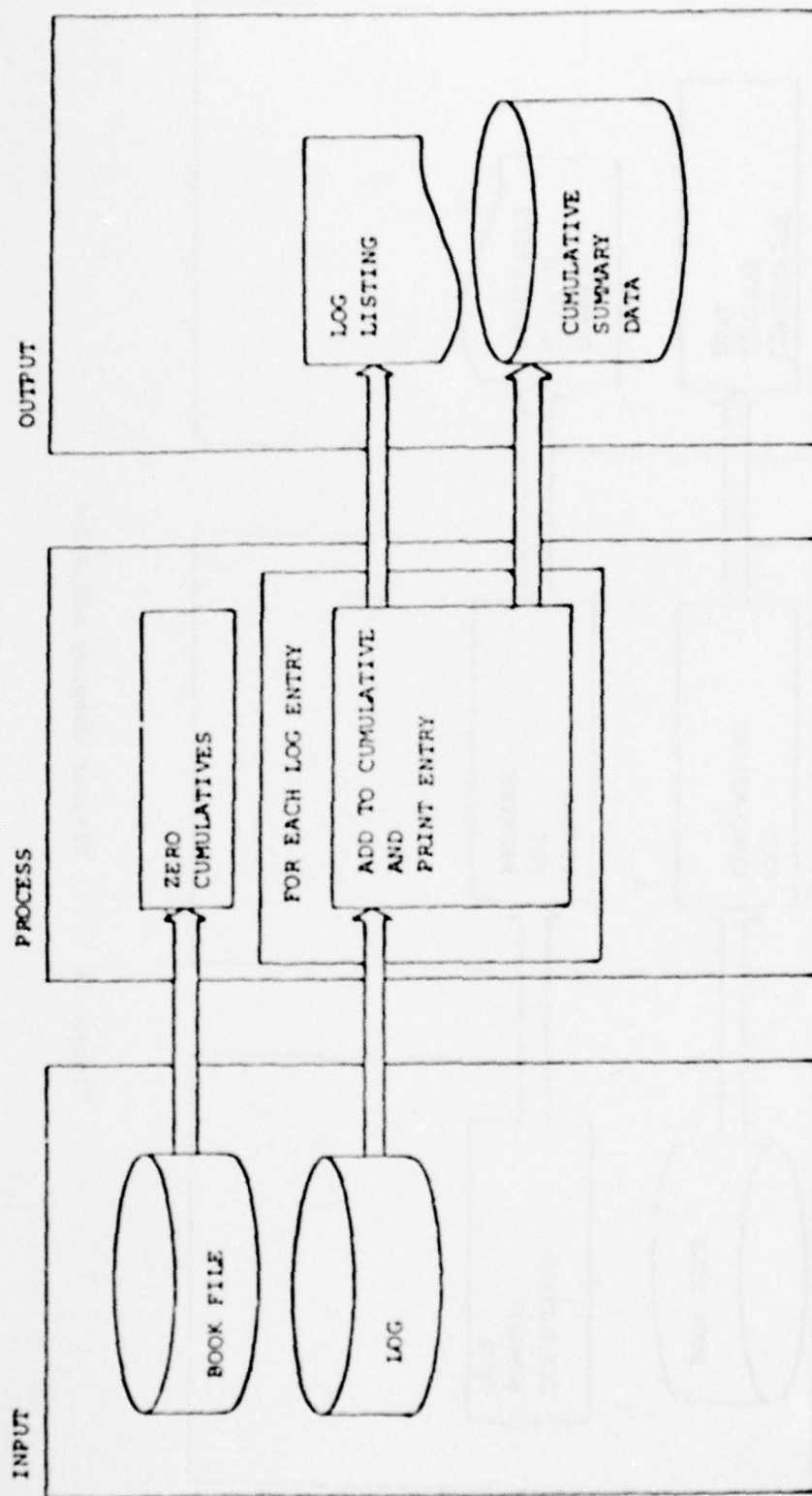
Figure 22   Diagram Collect and Update

3-27

INPUT

PROCESS

OUTPUT

BOOK FILE

CUMULATIVE SUMMARY DATA

GET CUMULATIVES

FORMAT FOR PRINTER

CUMULATIVE SUMMARY DATA

SUMMARY LIST OF STATISTICS
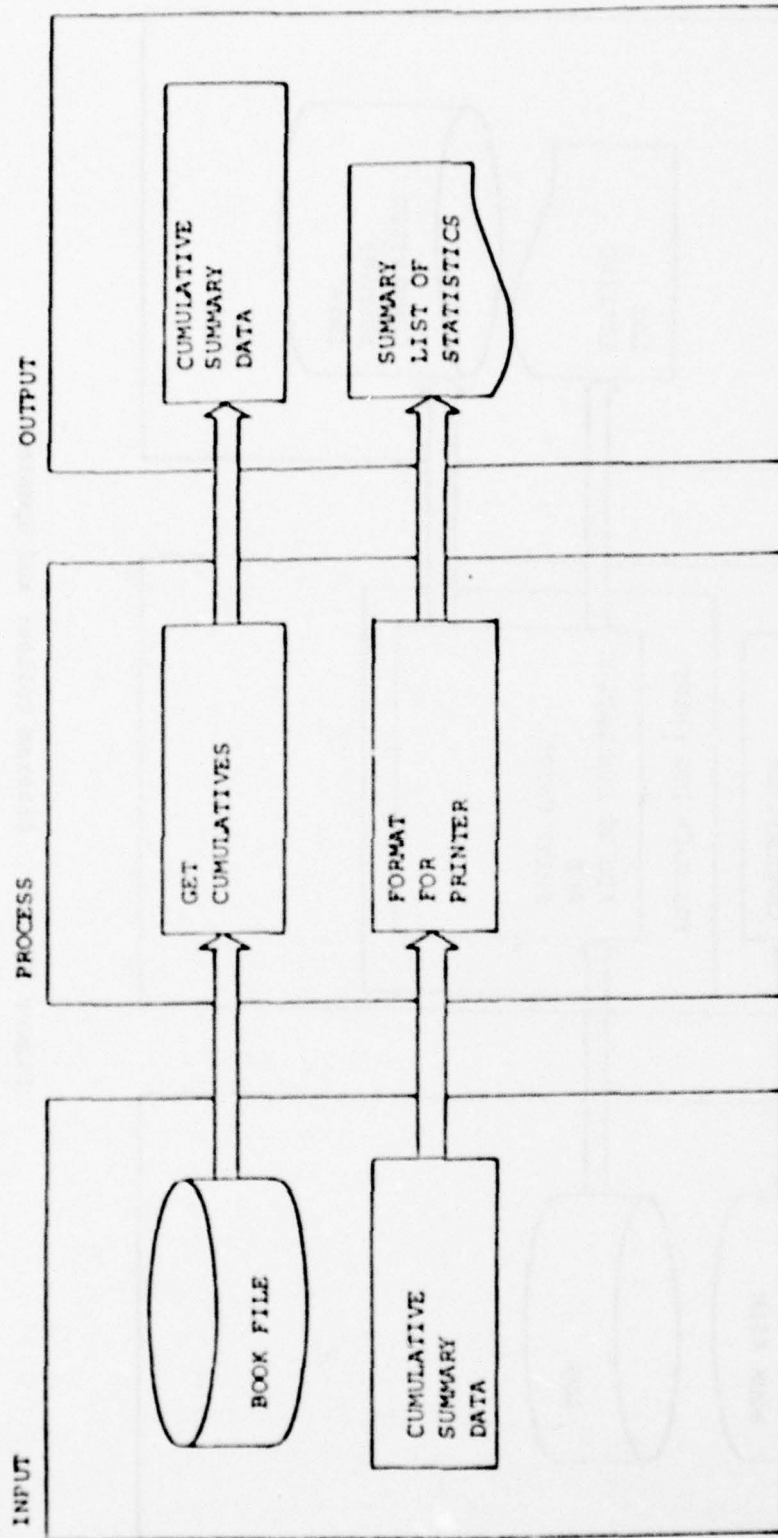
Figure 23   Diagram Display and Print

3-28
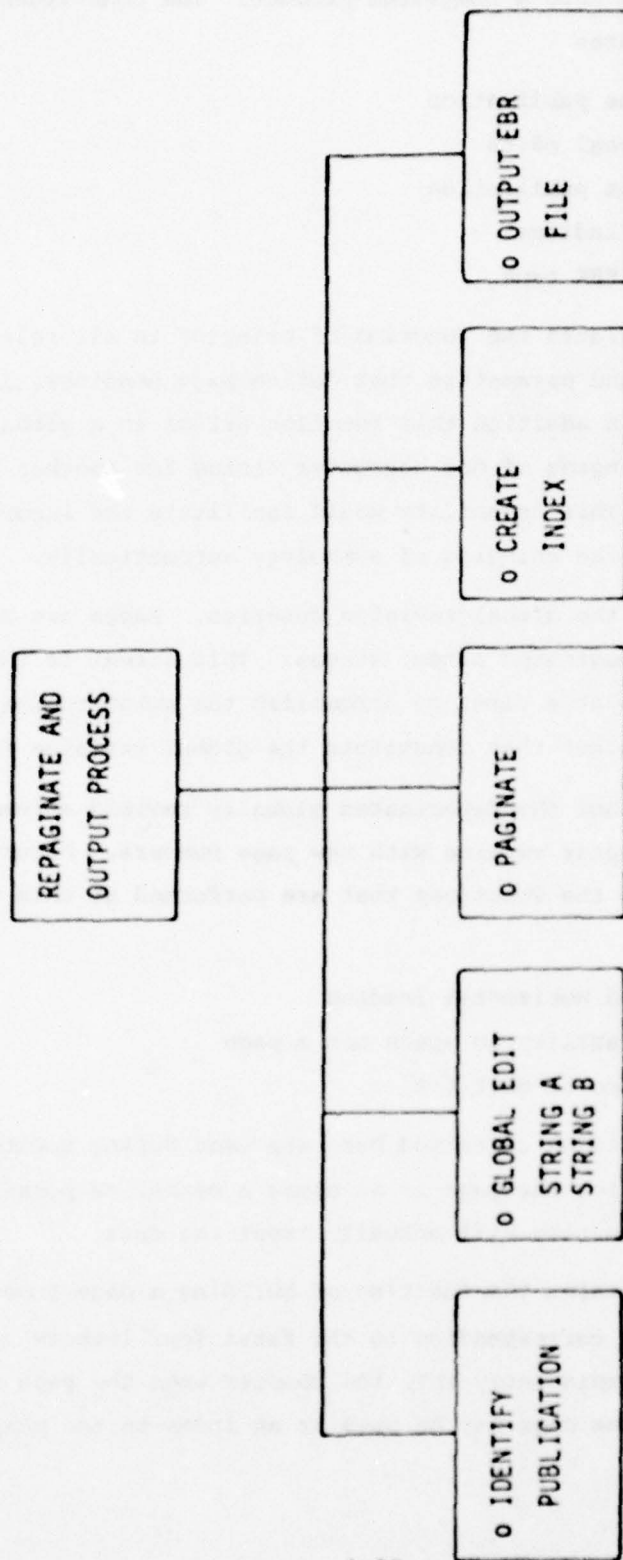
Figure 24   Repaginate and Output Process

3-29

format are translated into a completed product. The constituents of the repaginate function are:

- o   Identify the publication
- o   Perform global edits
- o   Paginate the publication
- o   Create new indices
- o   Output the EBR tape

Figure 25 illustrates the function of bringing in all relevant book and chapter descriptors and parameters that define page headings, leading requirements, etc. In addition this function brings in a global edit file which permits the changing of one character string for another throughout the entire chapter. This capability would facilitate the incorporation of new abbreviations on the changing of symbology automatically.

Figure 26 shows the global revision function. Pages are brought in and merged into a depaginated output stream. This stream is then checked, a group of characters at a time, to accomplish the substitution of one set of characters for another that constitute the global revision function.

Figure 27 shows how the depaginated globally revised stream is transformed into a new chapter version with new page numbers. Figure 27 (continued) lists some of the functions that are performed at this time. Other functions include:

- o   Vertical and horizontal leading
- o   Look ahead ability to space out a page
- o   Determination of best fit

Many of the functions contained here are used during update on a single page at a time to depict the page in as close a manner as possible to the way the repaginate function will actually treat the data.

Figure 28 illustrates the function of building a page index for selected publications corresponding to the first four letters of a facility name. This enables rapid entry into the chapter when the page number is not shown. Any text on the page may be used as an index to the chapter.
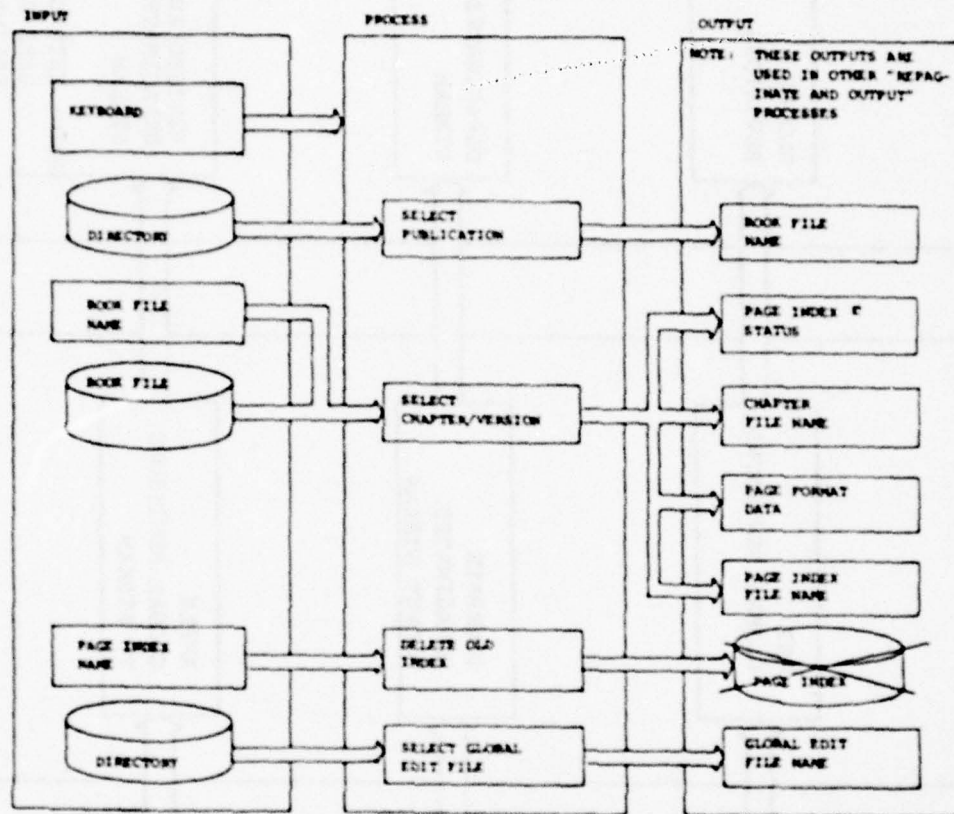
Figure 25 Diagram Identify Publication

INPUT

GLOBAL REVISIONS FILE NAME

GLOBAL REVISIONS FILE

CHAPTER FILE NAME

CHAPTER FILE

GLOBAL REVISIONS

DEPAGINATED STREAM

PROCESS

GET GLOBAL REVISIONS

GENERATE DEPAGINATED OUTPUT STREAM

APPLY GLOBAL REVISIONS TO STREAM

OUTPUT

GLOBAL REVISIONS

DEPAGINATED STREAM

REVISED AND DEPAGINATED STREAM

NOTE: THIS OUTPUT IS USED BY PAGINATE
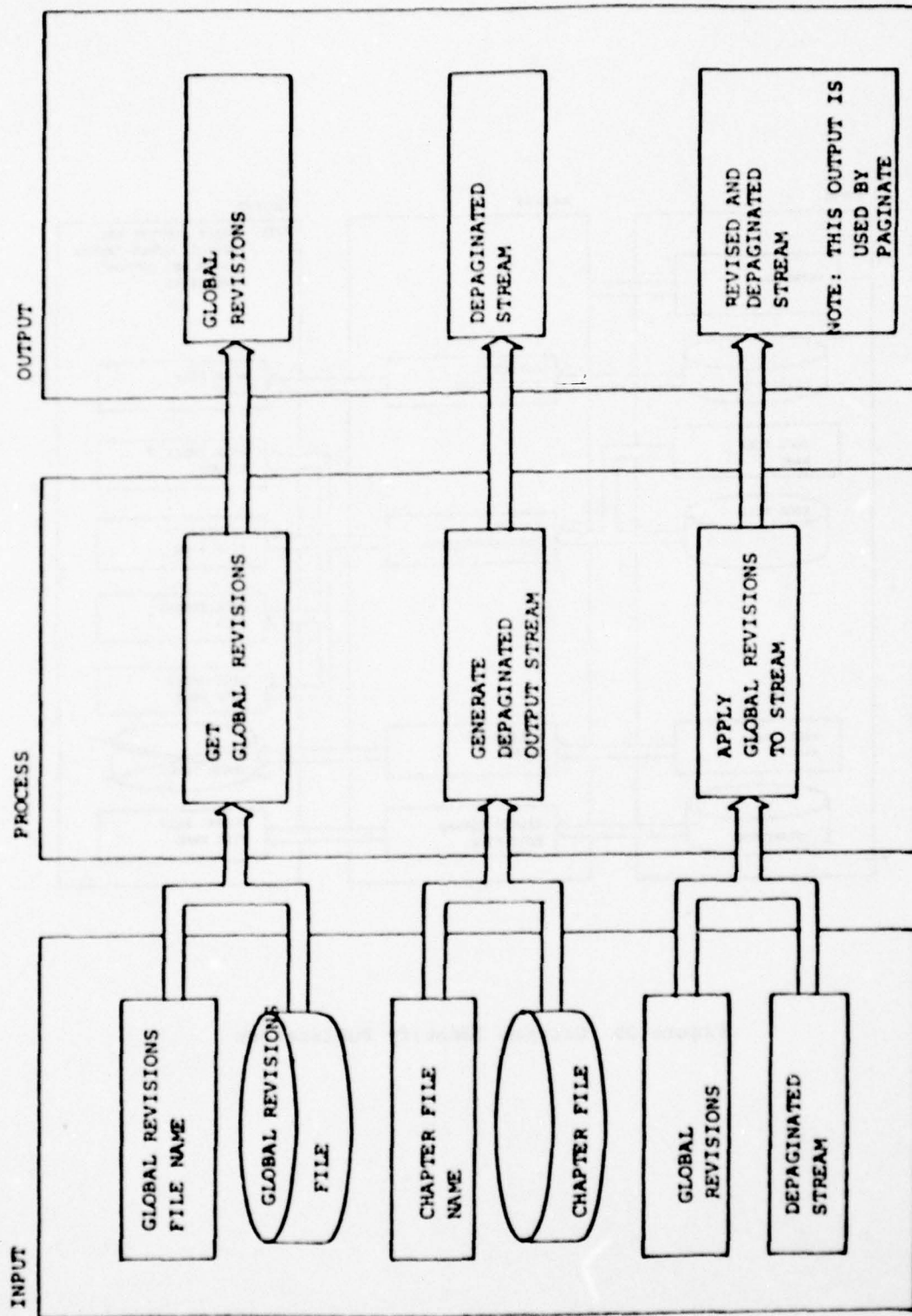
Figure 26   Diagram Global Revision

3-32
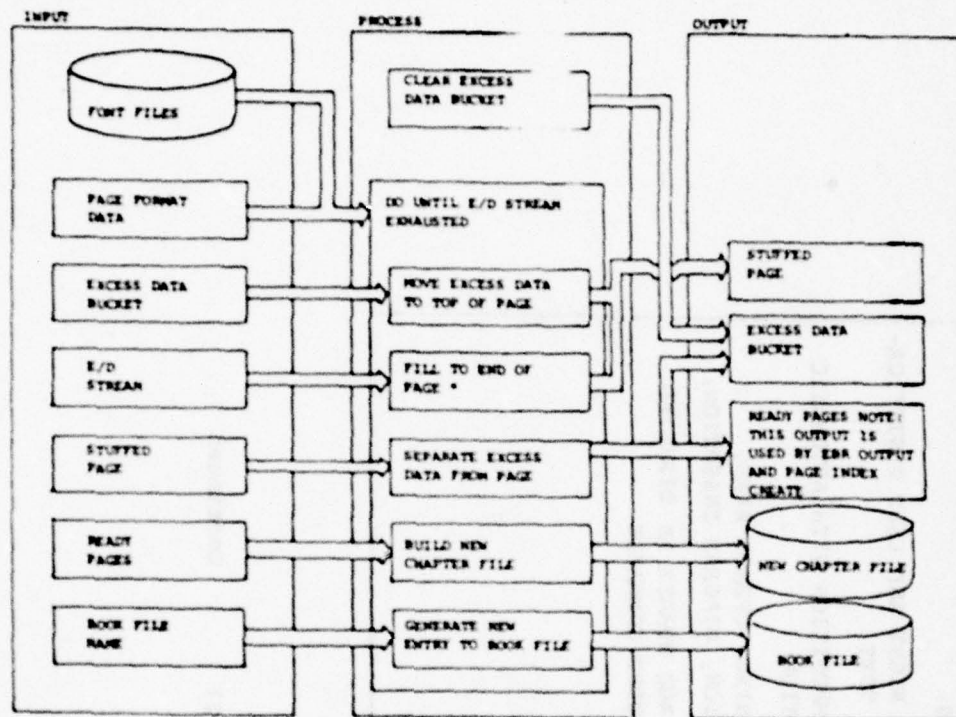
Figure 27 Diagram Paginate

FILL TO END OF PAGE WHILE
PERFORMING:

✓ CENTER, RIGHT AND LEFT JUSTIFICA-
TION OF TEXT

✓ LINE COMPOSITION WITH AUTOMATIC
HYPHENATION

✓ UNDERLINING, UPDATE BARS,
TABULATION, DIAGRAM INSERTION,

✓ ENTER PAGE NUMBER AS DIRECTED
BY EMBEDDED COMMANDS

Figure 27   Continued

INPUT   PROCESS   OUTPUT

DO IF PAGE INDEX STATUS
INDICATES PAGE INDEX
REQUIRED

PAGE INDEX
STATUS

DO UNTIL NO MORE READY
PAGES

EXTRACT FIRST FOUR
LETTERS IN FONT
FOUR AND WRITE TO
INDEX

READY
PAGES

PAGE INDEX FILE
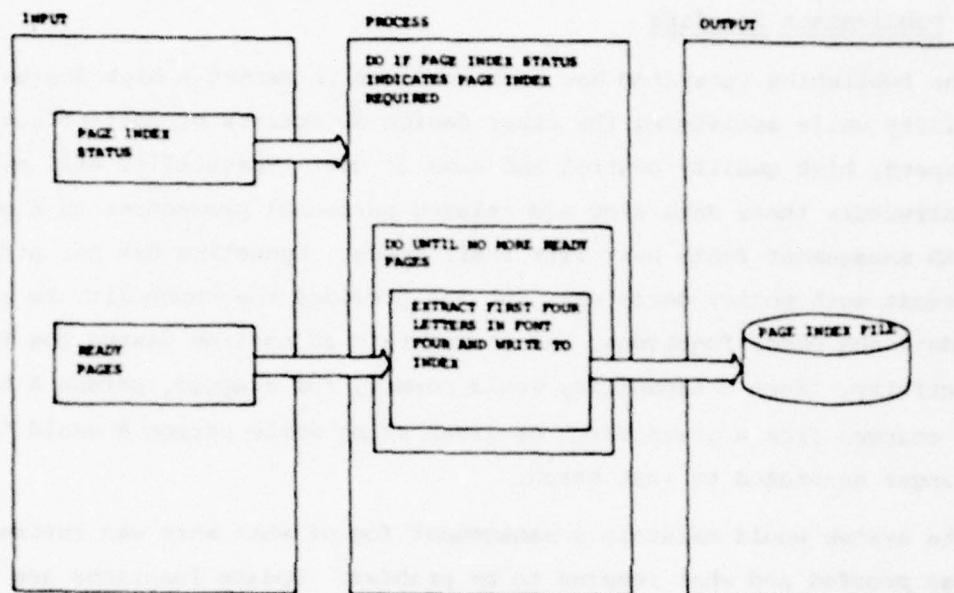
Figure 28 Diagram Create Index

3-35

Figure 29 shows the translation process whereby the revised and paginated chapter is converted to a format usable on the EBR output device. Once the translation has occurred the formatted publication becomes available to the Charting Subsystem as a magnetic tape for merging the necessary graphics with the formatted pages produced on the Publishing Subsystem.

3.1.8 Publication Proofing

The Publishing Subsystem has been designed to permit a high degree of versatility while satisfying the other design objectives of cost-effectiveness, speed, high quality control and ease of use. Versatility will permit AD to structure their data flow and related personnel procedures in a manner which AD management feels best fits their needs. Synectics has not presumed to forecast such policy decisions, but has provided the capability to separate the update and proof functions, and to maintain an on-line status log for each activity. Such a capability would permit, for example, person A to "edit" changes from a given batch of green slips while person B would "proof" the changes generated by that batch.

The system would maintain a management log of what work was revised, what was proofed and what remains to be proofed. Update functions are described below in Section 3.3 and the management log was shown in Figure 17.

Figure 30 graphically provides an overview of the proofing functions. An important facet of quality control is the ability to maintain accurate composition control parameters (e.g., font and type size parameters). Thus, composition control parameters are not displayed unless requested and are not, therefore, normally available for revision. During proofing they can be requested for display on the CRT or line printer as described in Figure 31.

Although the Publishing Subsystem is an interactive system and all updating, proofing, and revision functions can easily be performed through the system terminal, provisions have been made for generating hardcopy listings, chapters or books including composition. Figure 32 graphically portrays this function.

3-36
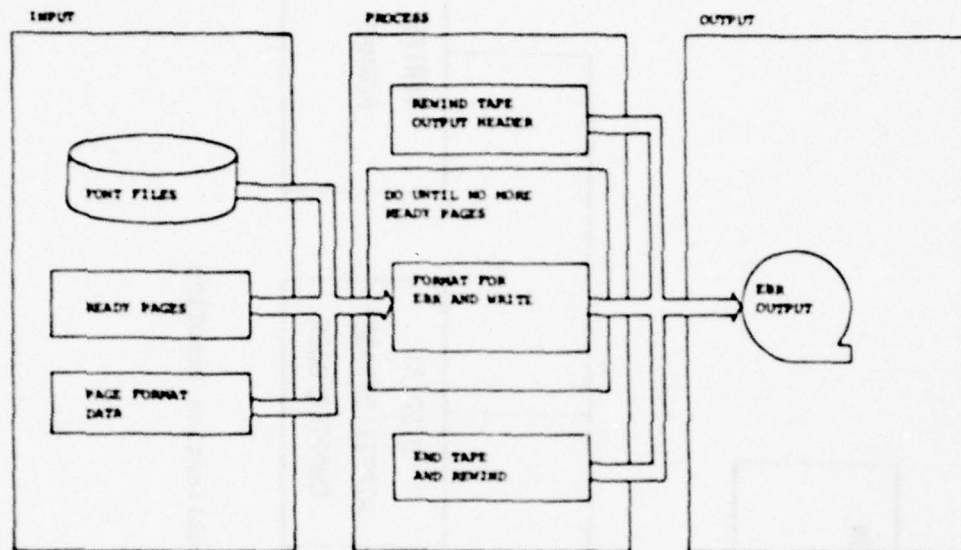
Figure 29   Diagram Output EBR File
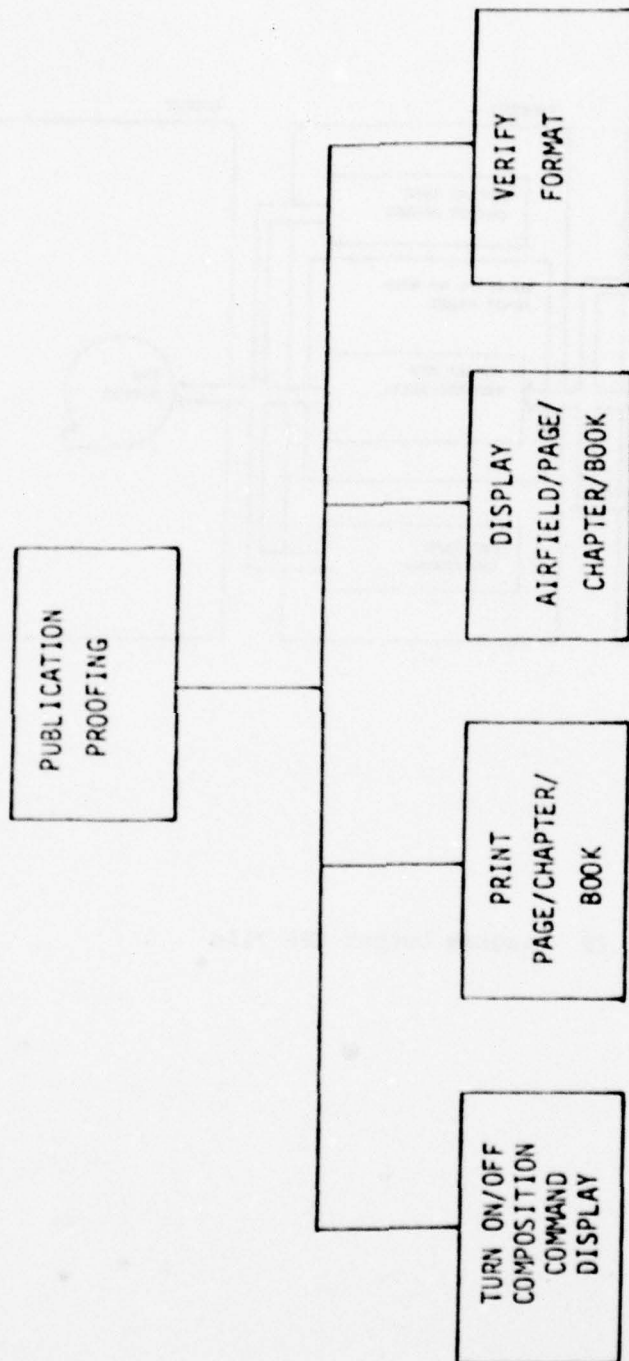
Figure 30   Publication Proofing
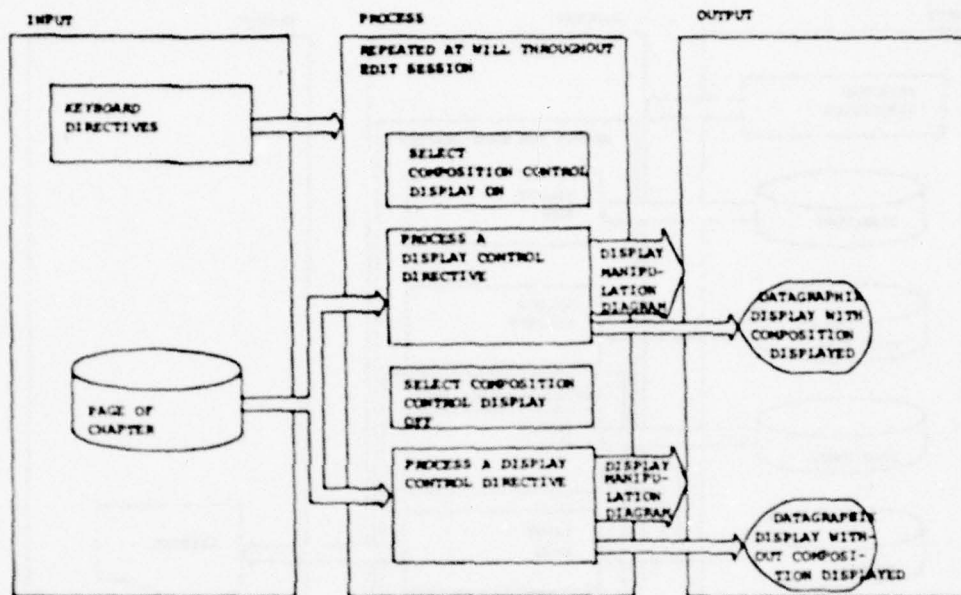
Figure 31   Diagram Turn On/Off Composition Display
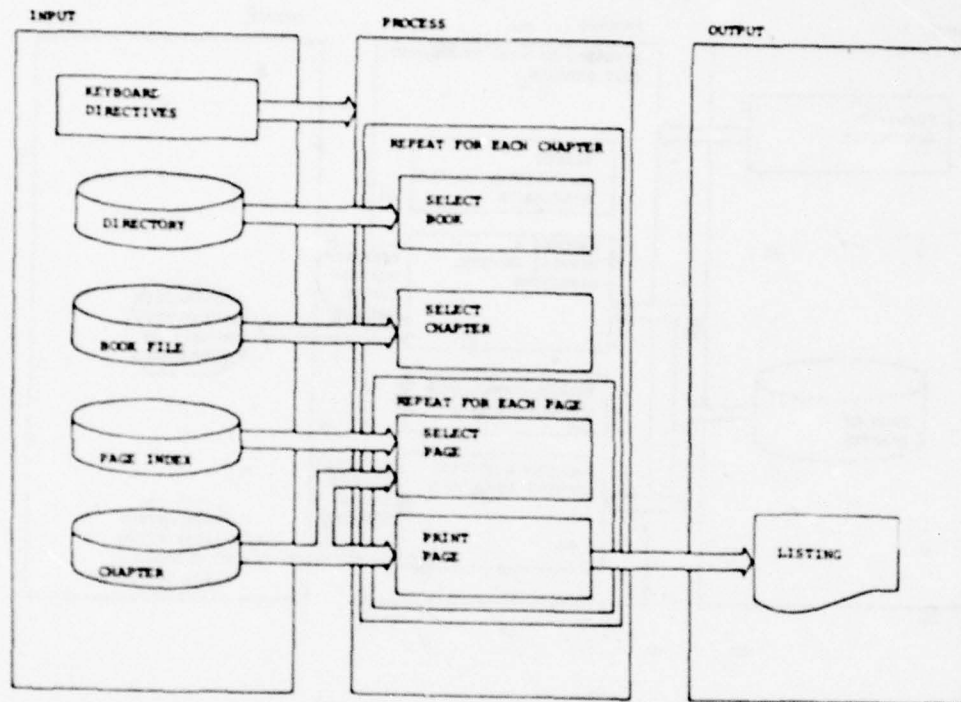
Figure 32 Diagram Print Page, Chapter, Book

During proof sessions no updates or revisions to the document can be effected. To accomplish an update the reviewer must, at minimum, exit proof mode and log-on in update mode. The operation sees the first 12 lines of text on the page just as in update mode (Figure 33). Format is verified on the terminal moving the cursor to the next change in composition control parameters and observing the small display of the font, type size and lead for the text that follows. Showing the proofer exactly what font and type size is being specified rather than trying to simulate the font and size on the CRT should help maintain a higher degree of quality control. Exact presentation of font and type size parameters does not leave any decision to chance interpretation of what is meant to be presented. Figure 34 is an overview of the verify process.

As detailed throughout the design discussion in Section 3, considerable emphasis has been given to human factors and quality control:

o   All currently used symbols will be available through the terminal

o   Composition control parameters are protected

o   "Update/Proof" management log protects system integrity

o   Items are retrieved by page or name

o   Chapter page headers are generated automatically

o   Publication parameters and user commands are segregated from publication text

o   The last command is displayed in addition to current or next command

3.2   Publishing Subsystem Data Base

Four types of files are used to maintain the publications. They are linked together as shown in Figure 35. The book file is a contiguous file of maximum length of 16,384 bytes. There is one book file for each publication. It serves as an index to each of the chapter versions (CV's) comprising the publication. The chapter file contains the contextual and formatting data for the publication. The font files contain that data needed to convert
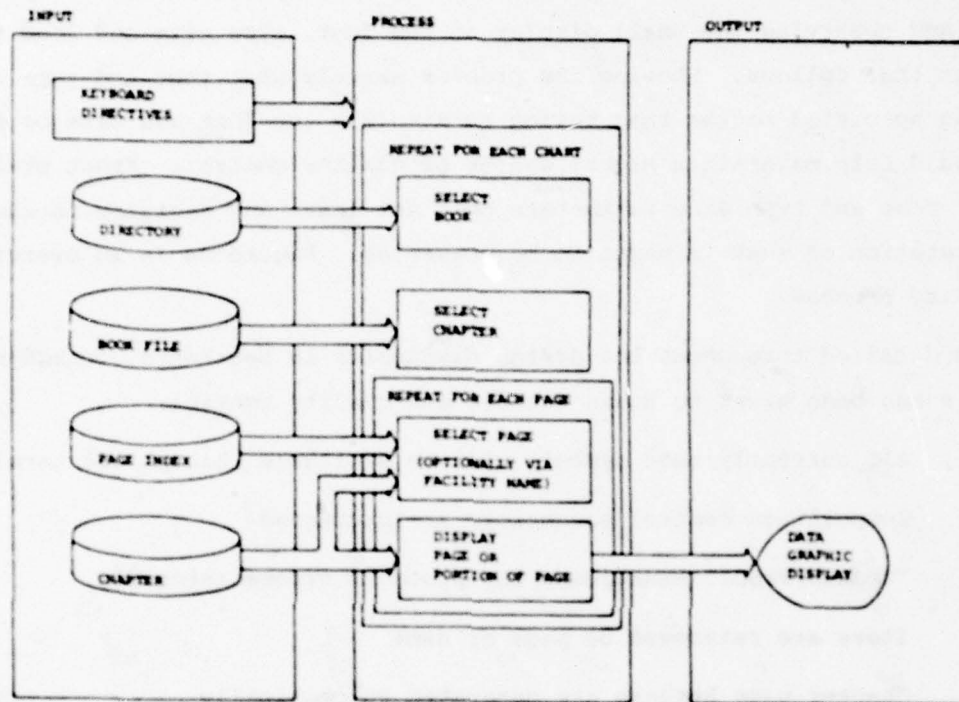
Figure 33 Diagram Display Airfields, Page, Chapter, Book

INPUT   PROCESS   OUTPUT

KEYBOARD
DIRECTIVES   →   SELECT 30
                                LINES OF
                                PAGE

PAGE OF CHAPTER   →   DO IN 10 SEC PERIODS

                                SPECIAL
                                DISPLAY   →   DATA
                                                        GRAPHIC
                                                        DISPLAY
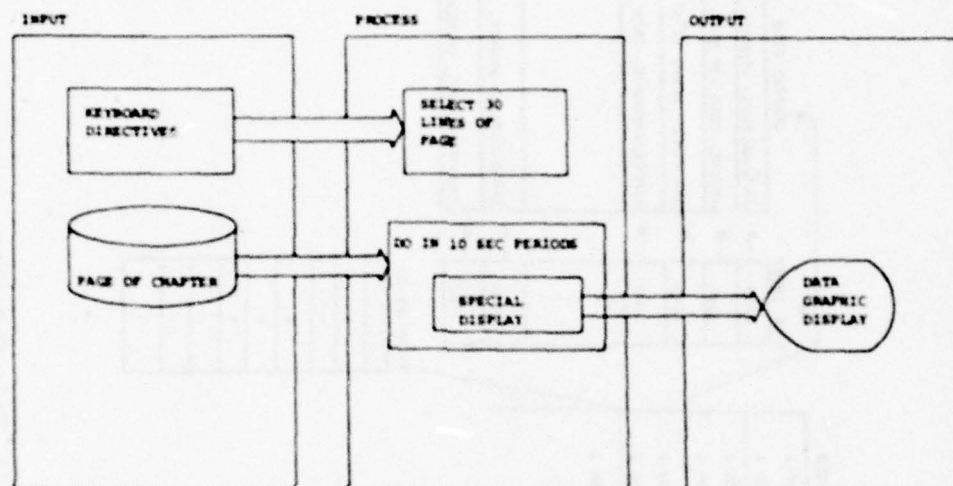
Figure 34   Diagram Program Verify Format
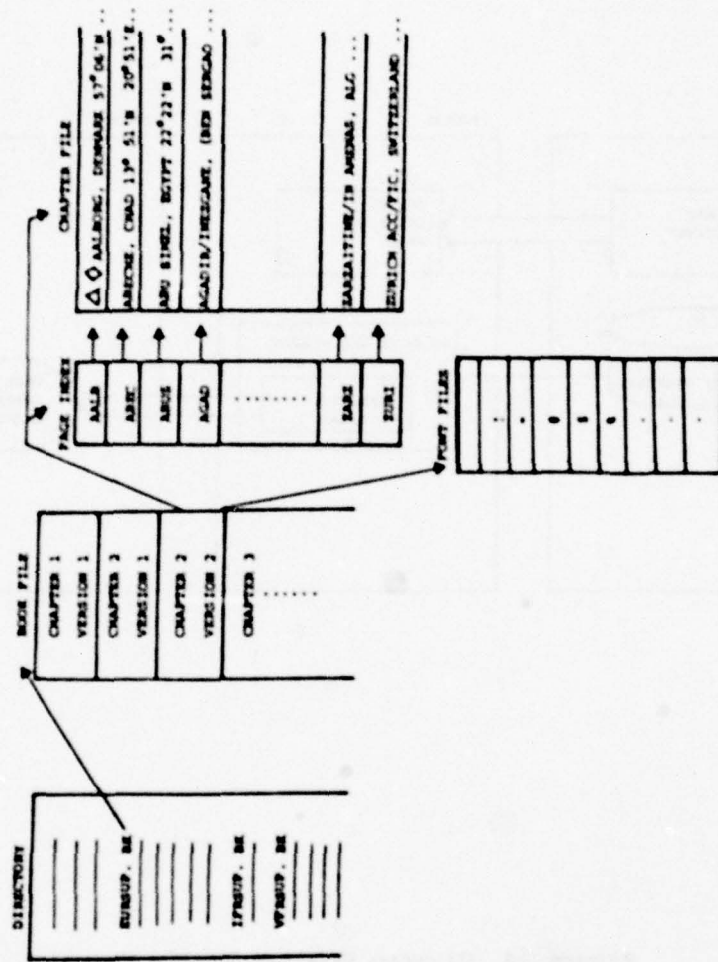
Figure 35 File Structure

the chapter file font designations into EBR designations. The page index
file is generated at publication time during repagination if the chapter
descriptor contains the appropriate indicator. It is used to address pages
of a supplement chapter by facility name. Additionally, there are system
utility files and transient files including:

o   RDOS command file (CoM-CM)

o   Publishing Subsystem Command File

o   System Log

3.2.1 Book File

The book file is a contiguous file of 16,384 bytes consisting of one
64 byte header followed by as many as 255 64-byte CV descriptors in a con-
tiguously organized file.

FILE HEADER

• Description	Bytes	Contents
1. File type identifier	2	"BK"
2. File type version	2	0
3. Number of bytes in header	2	64
4. Number of bytes per entry	2	64
5. Title of book	16	ASCII
6. Number of entries; chapter versions	1	0 to 255
(CV's) Filler	39	Nulls
	64	

CV DESCRIPTOR FORMAT

Field No.	Field Description	Bytes	Format	Contents
1	Chapter (CV) Title	16	RADIX 40	1-256
2	Chapter Number	1	Binary	1-256
3	Filler	1	Byte	Null
4	Version Number	2	Binary	1-6561
5	Vertical Page Size	2	Binary	Pica
6	Horizontal Page Size	2	Binary	Pica
7	Chapter File Record Size	2	Binary	Bytes
8	Chapter File Size	2	Binary	Records
9	Date of Last Update	2	Binary	Date
10	Date of Last Publication	2	Binary	0 or Date
11	Size of Font No. 1	1	Binary	Points
12	File Name of Font No. 1 Data	3	ASCII	FONTaaa.F
13	Size of Font No. 2	1	Binary	Points
14	File Name of Font No. 2 Data	3	ASCII	FONTaaa.F

Field No.	Field Description	Bytes	Format	Contents
15	Size of Font No. 3	1	Binary	Points
16	File Name of Font No. 3 Data	3	ASCII	FONTaaa.F
17	Size of Font No. 4	1	Binary	Points
18	File Name of Font No. 4 Data	3	ASCII	FONTaaa.F
19	Size of Page Number Font	1	Binary	Points
20	File Name of Page Number Font Data	3	ASCII	FONTaaa.F
21	Page Number of First Page	2	Binary	1-32767
22	Number of Pages	2	Binary	1-1024
23	Vert. Page Number Position (Top, Bottom)	1	ASCII	T, B
24	Horiz. Page Number Position (Inside, Center, Outside)	1	ASCII	I, C, O
25	Accompanying Page Number Data (Title, Nothing, CV Title, CV#)	1	ASCII	T,Blank,C
26	Page Index File Code (None, Will Generate, Yes, Will Stop)	1	ASCII	Blank, G,Y
	Filler	4	Bytes	Nulls
		64		

3.2.2 Chapter File Design

Each chapter file is a contiguous MRDOS file containing records of
16,384 bytes (32 sectors) each. The file name for this file is a function
of the book title and the chapter and version number as specified in the
CV descriptor. Each of these records represents one page of data as it
would appear in the publication or as subsequently updated. Repagination
occurs only during EBR output tape generation at publication time. Each
record contains a 32-byte header containing status information carried over
from the previous page followed by 16,352 bytes of contextual and formatting
data (see Figure 36). Although 16,352 bytes is triple the amount of room
required by a typical page, this much room has been allocated to accommodate
extensive modifications during the update process and although this contig-
uous structure will result in disk allocations of 6.5 million bytes for some
publications (versus 2 million for a more compact format) the fourfold (or
more) decrease in record access time makes it well worth this consumption of
a rather minor portion of the 192 million bytes of disk storage.
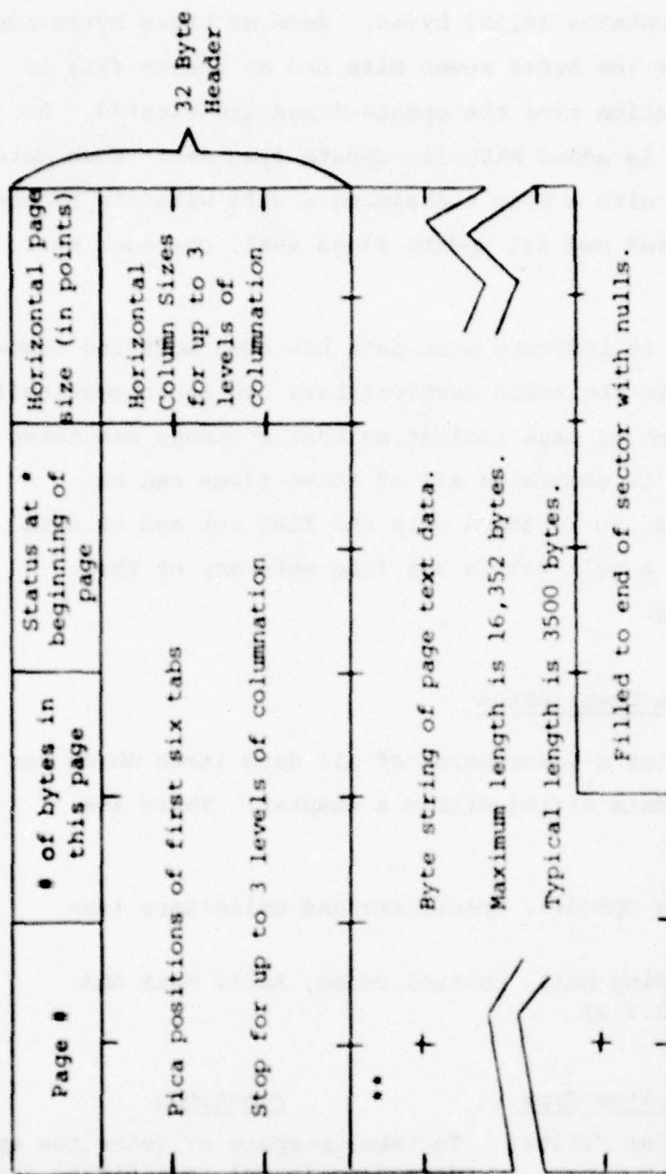
Figure 36. Chapter File Format

3.2.2.1 Update Flags

The body of each record contains 16,352 bytes. Each of these bytes contains an ASCII character in the low order seven bits and an update flag in the high order bit. At publication time the update flags are cleared. As data is subsequently added, it is added with its update flag set. When data are deleted, they are replaced with a byte containing a null with its update flag set (unless the data deleted had all update flags set), one such null for each string deleted.

The update flags are used to indicate what data has been modified since the last publication. With this the solid vertical bars can be automatically placed to the left of the column or page indicating that a change has taken place. Whenever a publication is generated all of these flags can be cleared. Then as data is added, it is added with the flag set and as data is deleted it is replaced with a null with a set flag when any of the deleted data has a cleared flag.

3.2.2.2 Chapter File Text Data Description

The following table contains a description of all data items which can be included in the ASCII text data string within a chapter. There are, however, two major divisions:

o Operations, including Opcodes, specifiers and delimiters (see 3.2.2.2.1)

o Nonoperations, including null, control codes, ASCII data and indicators (see 3.2.2.2.2)

Octal Code(s) of First Character	Data Item Type	Functions
000	Null or Filler	To take up space or (when the update flag is on) to indicate that items formerly in this position have been deleted since the last publication.
011 to 015 and 021 to 024	Control Code	To control horizontal (011) or vertical (013) tabulation, page/ column control (012 and 015) or termination (014), or font/size selection (021 to 024).

Octal Code(s) of First Character	Date Item Type	Functions
074 and 076	Operation	To specify column parameters, horizontal or vertical tab positions, font/size selections, diagram placement, horizontal justification, hyphenation or columnation restriction, titles, or horizontal column spanning lines.
040 to 177 except (074, 076, 133, 135, and 137)	ASCII Data	To specify contextual publication data.
133 and 135	Indicator	To specify enclosed text is to be underlined.
137	Indicator	To specify next byte contains a special or circled character.

3.2.2.2.1 Operations

Operations are delimited by an initial <(Octal 074) and a terminal> (Octal 076). Between the delimiting character is the one or two character opcode followed in some cases by addition specifiers. The opcodes are:

Opcode	Specifiers	Function	Level
H	No	Horizontal line spanning page or column (also terminates line)	4
CL	Yes	Specifies column parameters for data which follows (also terminates line)	3
TC	No	Terminate columnation	3
MS	No	Allow midstream column/page breaks	4
TM	No	Don't allow midstream column/page breaks	4
TB	Yes	Set horizontal tab stops	4
V	Yes	Set vertical tab stops	2
LJ	No	Select left justification (normal)	5
CJ	No	Select center justification (between tab stops)	5

Opcode	Specifiers	Function	Level
RJ	No	Select right justification (to next tab stops)	5
LD	No	Left justify in a field of dots	5
D	Yes	Specify position and identification of a diagram	5
TL	Yes	Specify a column title (implicit column break)	4
HY	No	Allow hyphenation (except across pages)	4
TH	No	Terminate hyphenation allowance	4

The level numbers indicate under what circumstances the operation can be used. Some of the operations (CL, TB, V, D and TL) can include one or more specifiers that have the following structure: a concatenation of data items delimited with a leading "<" and terminating ">." Within such a structure not all data items can be used. Specifically, only those data items with a level number greater than the level number of the item containing the specifier can be used. For example, suppose a diagram is to be specified. One of the specifiers that can be included in the data item for diagram specification designates the identification or label of the diagram. This specifier can only include data items of levels greater than 5. (5 is the level of diagram specification item.) Therefore, operation type data items cannot be used in the diagram specification.

3.2.2.2.2 Nonoperations

The other data items have the following levels:

Data Item	Level
Null or filler	1
Control codes	5
Operations	2-5 as described above
ASCII data	6

During data entry at an edit session the level of entry starts out at one and therefore nulls cannot be entered directly.

3-50

There are nine control codes:

Control Code	Octal	Function	Level
TAB (↑I)	011	Horizontal tabulation (like CR if no tabs)	5
LF (↑J)	012	Start on next line (next column if need be)	5
VT (↑K)	013	Vertical tabulation (like FF if no vertical tabs)	5
FF (↑L)	014	Start on next page/column	5
CR (↑M)	015	Start on next line (same column unless opcode MS)	5
DC1 (↑Q)	021	Select font/size 1	5
DC2 (↑R)	022	Select font/size 2	5
DC3 (↑S)	023	Select font/size 3	5
DC4 (↑T)	024	Select font/size 4	5

When font/size 3 or font/size 4 is selected, text in this font appearing on the Datagraphix terminal will be bright. At pagination time letters in font/size 4 will be used to build the page index file (if generation of this file is indicated) and text in this font will be displayed just prior to edit time as pages are turned in search of the correct page.

To indicate that text is to be underlined, the text should be enclosed in brackets. For example:

Subject to [LANDING FEES] at these

would appear in the publication as:

Subject to LANDING FEES at these

Spacing which is provided in response to a horizontal tab (indicated by an asterisk in the examples) will not be underlined. For example:

[FLIP Code*Grade*NATO Code]

would appear in the publication as:

FLIP Code   Grade   NATO Code

3.2.3 Font File Design

The font file is a fixed length contiguous MRDOS file 384 bytes in length. The name of the file is "FONT***.F" where "***" is any 3 alphanumeric

3-51

characters. The file is made up of 96 four byte entries each representing one of the 96 ASCII characters from Octal 40 to Octal 177. Each entry has the following format ("*" represents the ASCII character that the entry data describes):

Field No.	Field Description	Bytes
1	EBR output value for "*"	1
2	Horizontal increment (in points) for a 9 point "*"	1
3	EBR output value for a circle, dot or triangle "*"	1
4	Horizontal increment (in points) for a 9 point circle, dot, or triangle "*"	1
	TOTAL	4

This file is used at edit, pagination, and EBR output time.

3.2.4 Page-index file design

Generation of the page-index file occurs at repagination and EBR output generation time in response to an indicator in the CV descriptor of the book file. The page-index file is a sequential file of four byte ASCII entries, one such entry for each page in the CV file to which the index supports. The file name of the page-index file is identical to the file name of the corresponding CV file except for the extension which is ".CV" for the CV file and ".PI" for the page-index file.

Each four byte ASCII entry is developed from the first occurrence of a string of characters in font/size 4 in the page of the chapter corresponding to that entry. This file is then used at page select time (during an EDIT session) to generate a page number from a facility or paragraph name.

3.2.5 The Log File

The log is a sequential file of strictly ASCII data. Each entry contains the following data:

	# Char	Example
Year	2	77
Month	3	JUN

	# Char	Example
Day	2	30
Space	1	
Hour	2	18
Colon	1	:
Minute	2	16
Space	1	
Command	2	DC
Space	1	
Other Data Such as the Book Title		
Carriage Return	1	
Sometimes followed by Other Lines Which Begin with a Space and End with Carriage Returns		

The Other Lines May Include Error Messages:

Space	1	
Error Code	3	100
Space	1	
Error Message Possibly Containing Parametric Data Detailing the Error Condition		
Carriage Return	1	

For the background the name of this file is $PUBBLOG.TB. For the foreground, $PUBFLOG.TB. These files are maintained in chronologically ascending order.

3.2.6 Command file COM.CM

The action taken by the Command Line Interpreter (CLI) upon reading a command line is sufficiently flexible so that users can, if they wish, design programs to perform system command functions.

When either the background or foreground CLI reads a command line and does not recognize the first file name, the CLI always builds a command file before the save file of that name is loaded. The command file reflects an edited version of the command line. The name of the command file is COM.CM

if it is built by the background CLI. The foreground CLI builds a command file identical in structure to COM.CM, but the foreground command file is named FCOM.CM.

For example, suppose the user issues the command line:

FOO

The CLI does not recognize FOO as a known command word. It therefore builds a command file with the byte organization shown below:

Byte	Contents	
0	F	Each character of the file name occupies a byte. The file name is terminated by a null byte.
1	O	
2	O	
3	Null	
4		Four bytes (two words) are set aside for global switches of FOO. Each letter switch sets a bit. A sets bit 0 of the first word, etc., as shown in the switch/bit correspondence diagram below.
5		
6		
7		
	end-of-file	

Byte 1,3   Byte 2,4

bit: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	word 1
Q	R	S	T	U	V	W	X	Y	Z							word 2

Bit 15 of the second word in the global switches field is always set for files running under BATCH. This serves to indicate that it is running under BATCH instead of the CLI.

Note that the CLI does not attempt any interpretation of switches in building the command file. The CLI simply sets the appropriate bit.

Additional file name arguments and local switches are handled in the same way when the CLI builds the command file. Suppose the user types the command:

FOO/B AA ZZ/X MUMB

The CLI would then build the following command file:

Byte	Contents	
0	F	
1	O	Command file name FOO, terminated by null byte.
2	O	
3	null	
4	1	
5		Global switches of FOO. Bit 1 (switch B) set ON.
6		
7		
8	A	
9	A	Argument AA, terminated by null byte.
10	null	
11		
12		Four bytes set aside for local switches of AA. None set.
13		
14		
15	Z	
16	Z	Argument ZZ, terminated by null byte.
17	null	
18		
19		Local switches of ZZ. Bit 23 (switch X) set ON.
20	1	
21		
22	M	
23	U	
24	M	Argument MUMB, terminated by null byte.
25	B	
26	null	
27		
28		Local switches of MUMB. None set.
29		
30		

Since the CLI does not interpret switches, the user can set up program interpretation of such switches. This gives the user an added means of passing information to a program to be executed, since he can use switches as well as arguments.

A read line from a disk file will terminate on a null (as well as carriage return and form feed). This is quite useful in reading COM.CM and FCOM.CM arguments. The following example illustrates how a background user could read the first argument of the command file as well as its global switches.

```
        LDA            0,CFILE          ;COM.CM POINTER
        .SYSTM
        .OPEN          3                ;OPEN ON CHANNEL 3
        JSR            ERROR            ;??
        LDA            0,ARG1           ;FILE NAME AREA POINTER
        .SYSTM                          ;READ IT (THE NULL
        .RDL           3                ;TERMINATOR IS ALSO
        JSR            ERROR            ;TRANSFERRED)
        LDA            0,GLOB           ;POINTER FOR GLOBAL
        LDA            1,C4             ;SWITCHES
        .SYSTM                          ;READ FOUR BYTES
        .RDS           3
        JSR
        .
        .
        .
C4:     4
GLOB:   2*.GLOB
ARG1:   2*.ARG1
CFILE:  2*.CFIL

.GLOB   .BLK           2
.ARG1   .BLK           10
.CFIL   .TXT           *COM.CM*
```

When square brackets are detected in a command line, they are passed as a file name with bit 10 in the second word of switch information set (this bit follows the Z switch position). Commas within square brackets are treated in the same manner (bit 10 set).

3.2.7 Command File $PUBBCOM.TB

The command file generated by CLI (COM.CM) is deparsed before being processed by PUB. This deparsed data is approximately the same ASCII data as it was entered except that the "PUB," and the command code have been removed and the file is terminated by two consecutive carriage returns. The file is sequential and is named $PUBCOM.TB or $PUBFCOM.TB for background or foreground respectively.

3.3 Personnel Functions

Personnel functions for the Publishing Subsystem consist of exercising the command repertoire in performance of FLIP maintenance activities. This section contains a brief description of how the commands would be used to maintain a FLIP publication.

3.3.1 Password Entry

Even those that are allowed to use the publishing system may not be allowed to use all of the commands. A password permits the use of all PUB features. The terminal, in its normal state, will respond "R" after the key marked CARRIAGE RETURN is depressed (see Figure 37). This means that the system is Ready. PW is a mnemonic to tell the system that a password will be entered. The system will respond "ENTER PASSWORD:" (see Figure 38) and wait for password entry. The system hides the actual password intent by responding to any keys entered with the response or password (see Figure 39) whether it is typed correctly or NOT. The only way to detect an error in password entry is to attempt further commands--which will be rejected if the correct password was not entered (see Figure 40). If that happens, it is necessary to enter PW again and type in the correct password (see Figure 41).

3.3.2 Entering Update Mode

To get into a book to make changes, the command "UPDATE," tells the system to enter update or review mode (see Figure 42). Depressing the space bar delimits the command. The MNEMONIC of the book to be updated is then

09·12·18
R
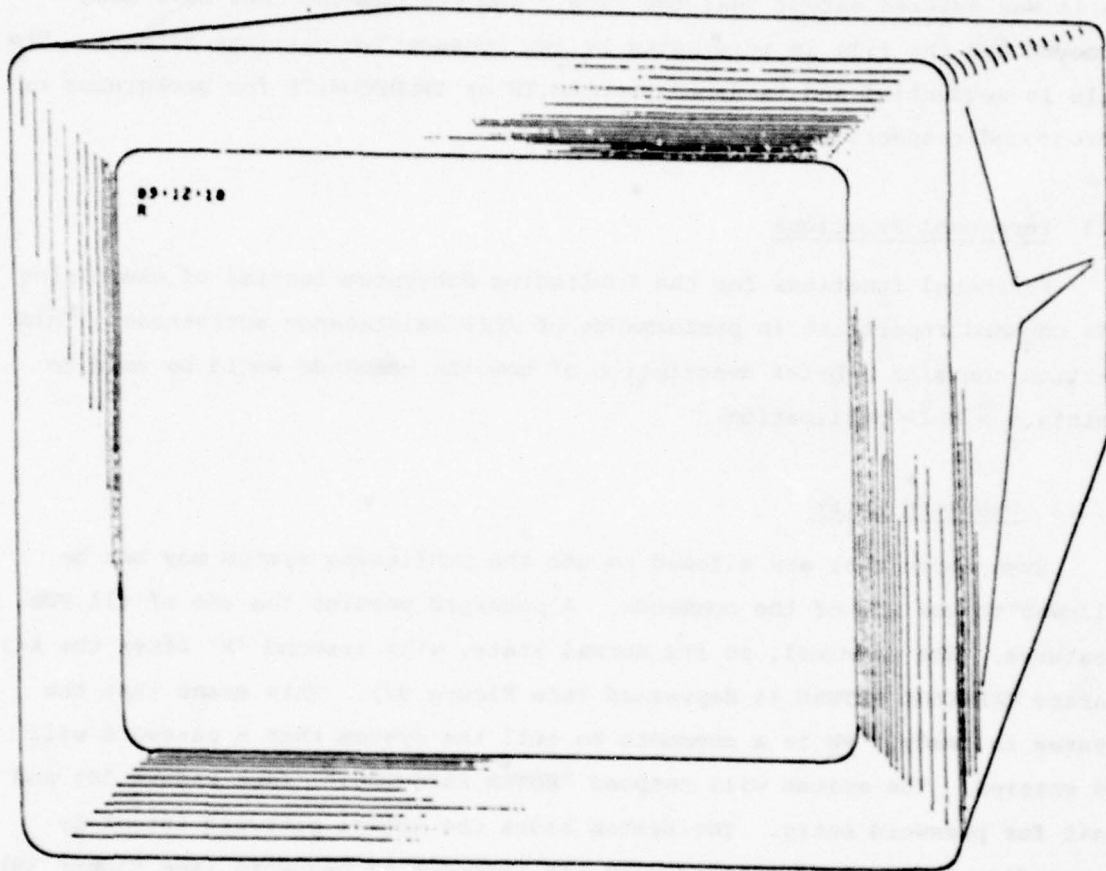
Figure 37   Key Entered: "Carriage Return"

03-12-10
R
PW

P.PW
  1978 MAR. 28   9:13 PW
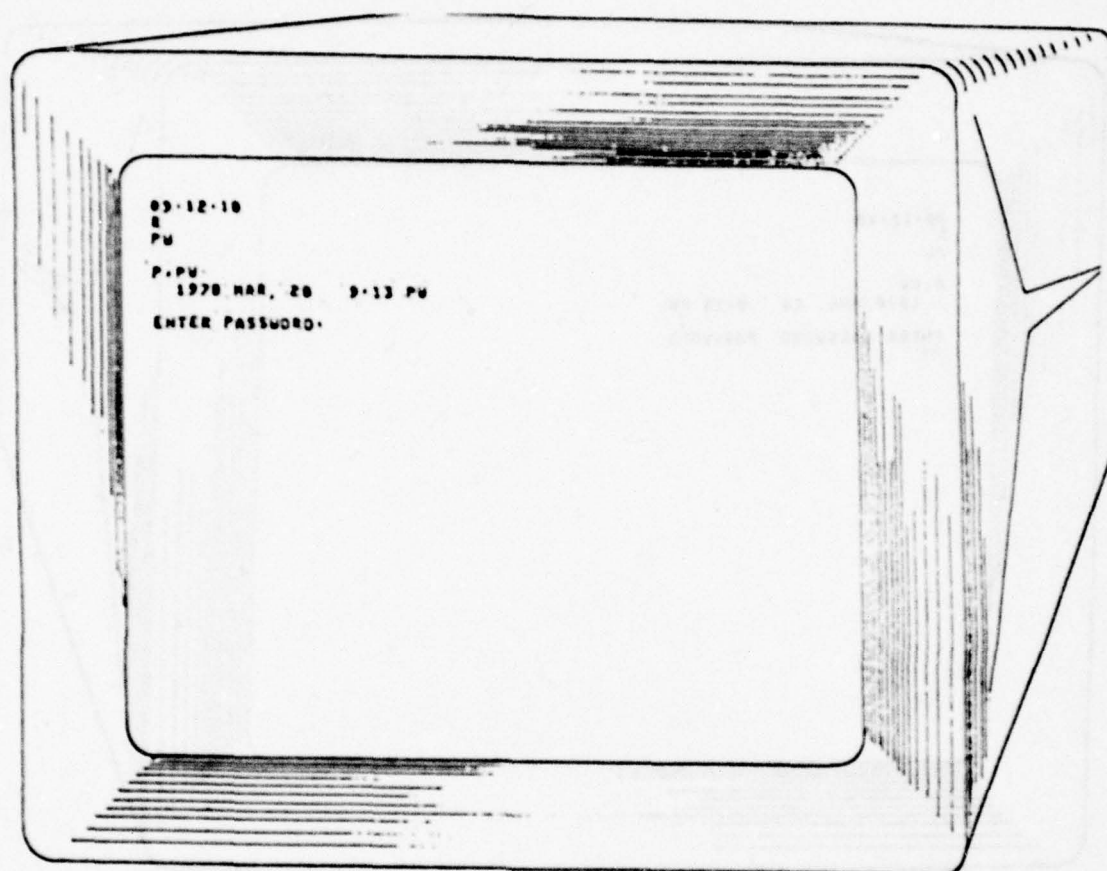
ENTER PASSWORD.

Figure 38   Keys Entered: "P""W"
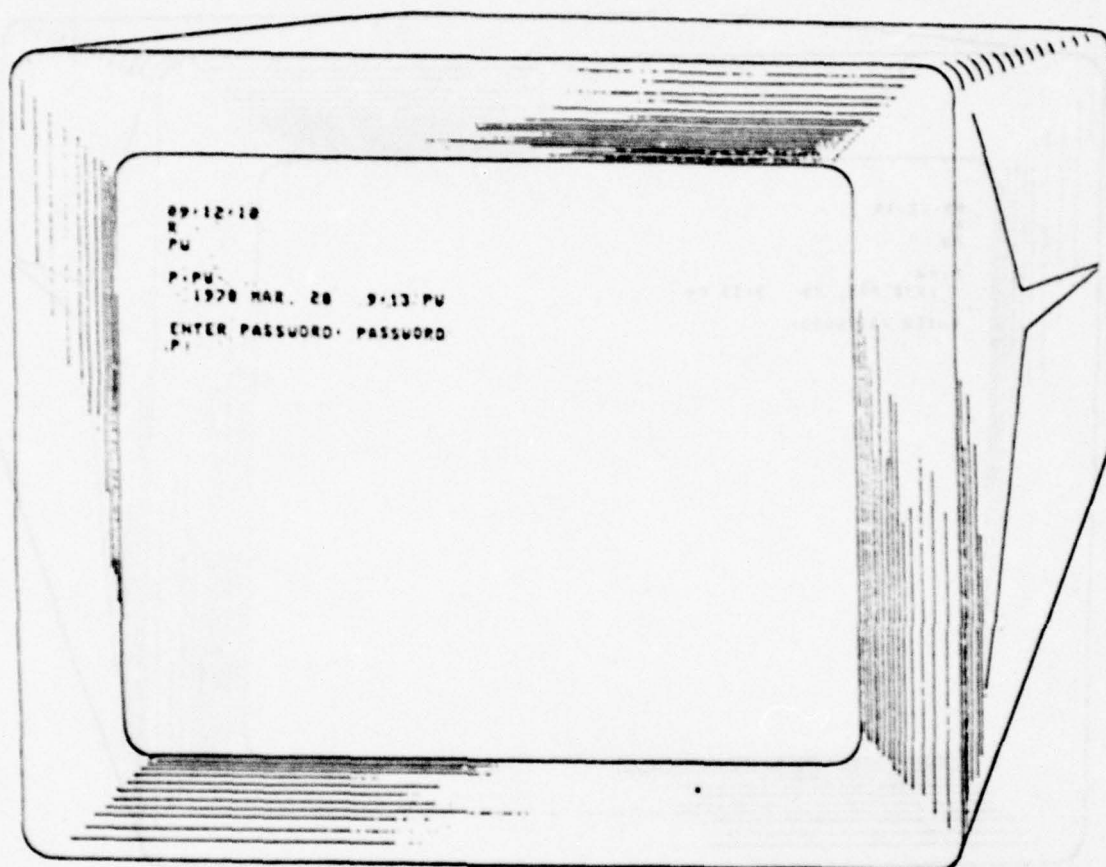
Figure 39 Keys entered: any eight keys or a valid password
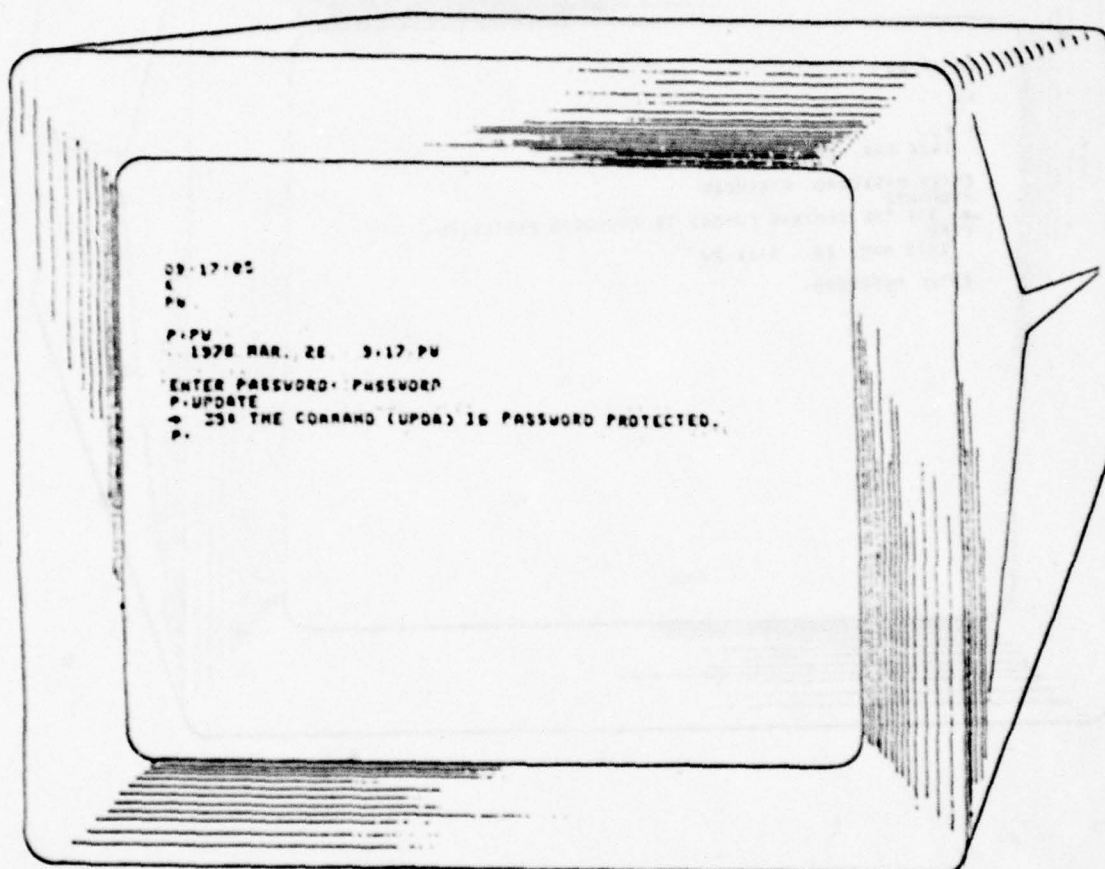
Figure 40

Keys entered: "U" "P" "D" "A" "T" "E"

(a valid PUB command)
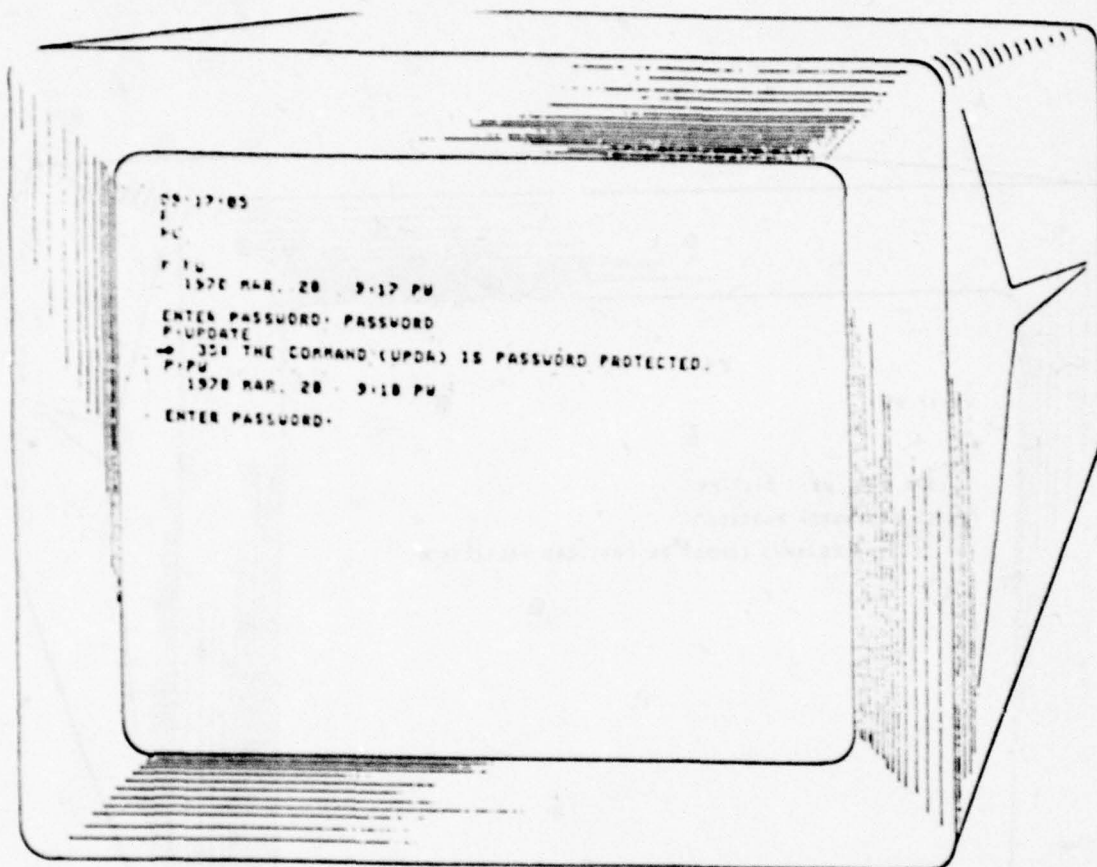
Figure 41 Keys entered: "P" "W"

```
09:17:05
R
PW

P-PW
  1978 MAR. 28   9:17 PV

ENTER PASSWORD: PASSWORD
P-UPDATE
  35$ THE COMMAND (UPDA) IS PASSWORD PROTECTED.
P-PW
  1978 MAR. 28   9:18 PV

ENTER PASSWORD: PASSWORD
P-UPDATE ES CS
```
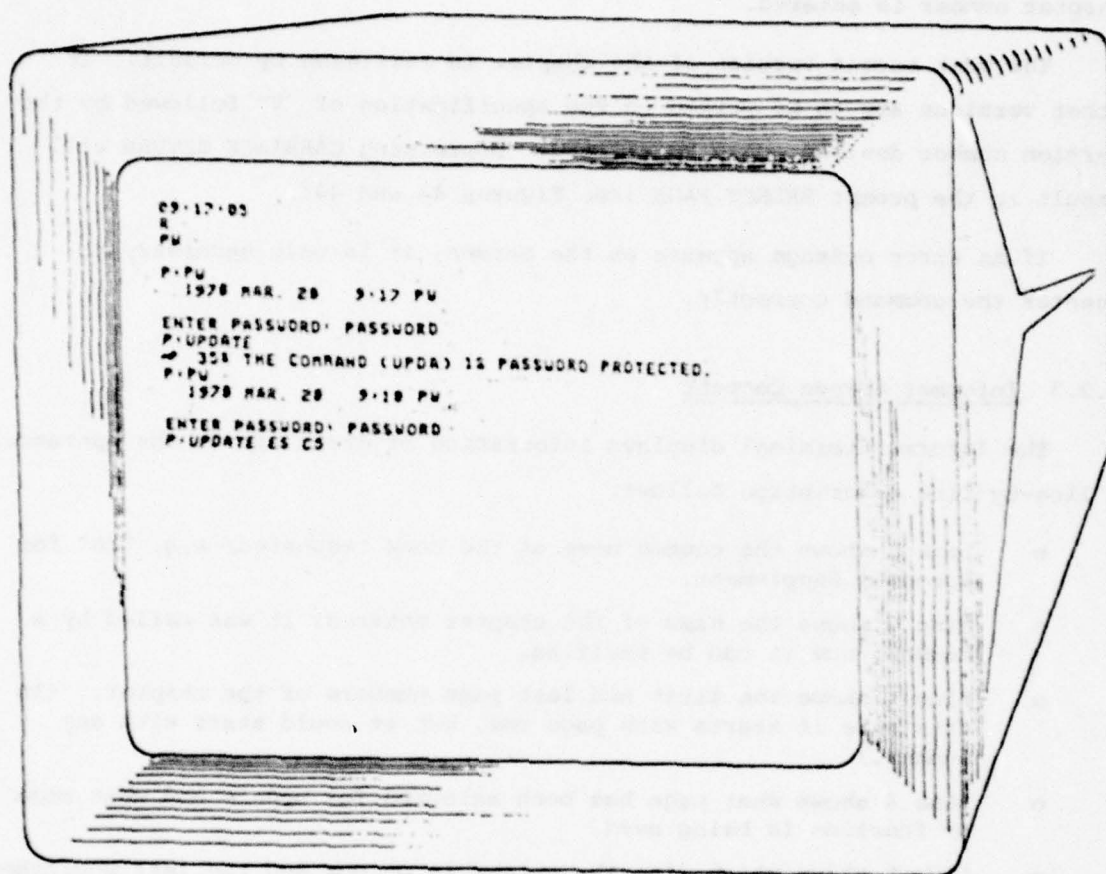
Figure 42   Using UPDATE

entered; e.g. "ES" for European Supplement. The space bar is again used to delimit the command. Finally, the nmemonic for chapter, "C"hapter, and the chapter number is entered.

The most recent version of the chapter is retrieved by default. If other versions are to be worked on the specification of "V" followed by the version number desired would be entered. Depressing CARRIAGE RETURN will result in the prompt SELECT PAGE (see Figures 43 and 44).

If an error message appears on the screen, it is only necessary to reenter the command correctly.

3.3.3 Informer Screen Content

The Informer terminal displays information of great use to the operator. A line-by-line description follows:

o   Line 1 shows the common name of the book requested; e.g. "ES" for European Supplement.

o   Line 2 shows the name of the chapter entered; it was called by a number, now it can be verified.

o   Line 3 shows the first and last page numbers of the chapter. (In this case it starts with page one, but it could start with any number.)

o   Line 4 shows what page has been selected for update and what mode or function is being used.

o   Line 5 shows which line the cursor is on now and the last published date.

o   Line 6 shows how far along the line the cursor is in printers measures (1/2 pica) and the date someone last updated this chapter.

o   Line 7 shows which of the four font/sizes the word at which the cursor is pointing uses in the publication.

o   Line 8 shows whether the character that the cursor is on is underlined and whether hyphenation can take place where the cursor is now or whether hyphenation has been disabled.

o   Line 9 shows whether Update Bars (lines on the side of the page) are to be produced for this part of the page if an update takes place and whether the system is allowed to make a new page starting from the current cursor position.

o   Line 10 shows, in printers measures (1/2 pica), how wide the page is.

o   Line 11 shows how much space will be left between lines.

SELECT PAGE
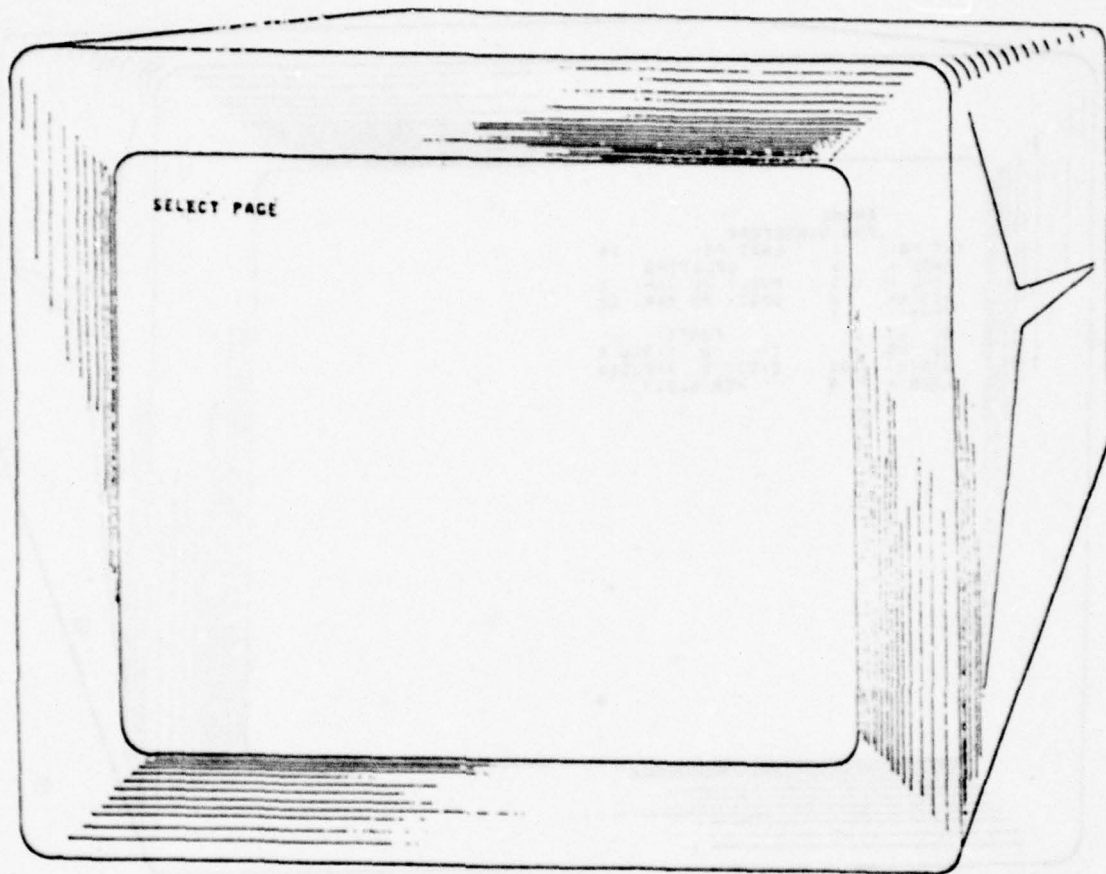
Figure 43

Now you are picking the page you
want to update.

ENAME
FAC DIRECTORY
1ST PG·   1   LAST PG·   94
PAGE ·   1   UPDATING
LINE ··   1   PUBL· 76 JAN. 1
POS'N·   0   UPDT· 78 MAR. 22
FONT ·   1
UL HY   FONTS·
U& RS   1·   0 3·BLD 0
WIDTH·   101   2·GOT 0 4·BLD10
LEAD ·   0   HDR·BLD10
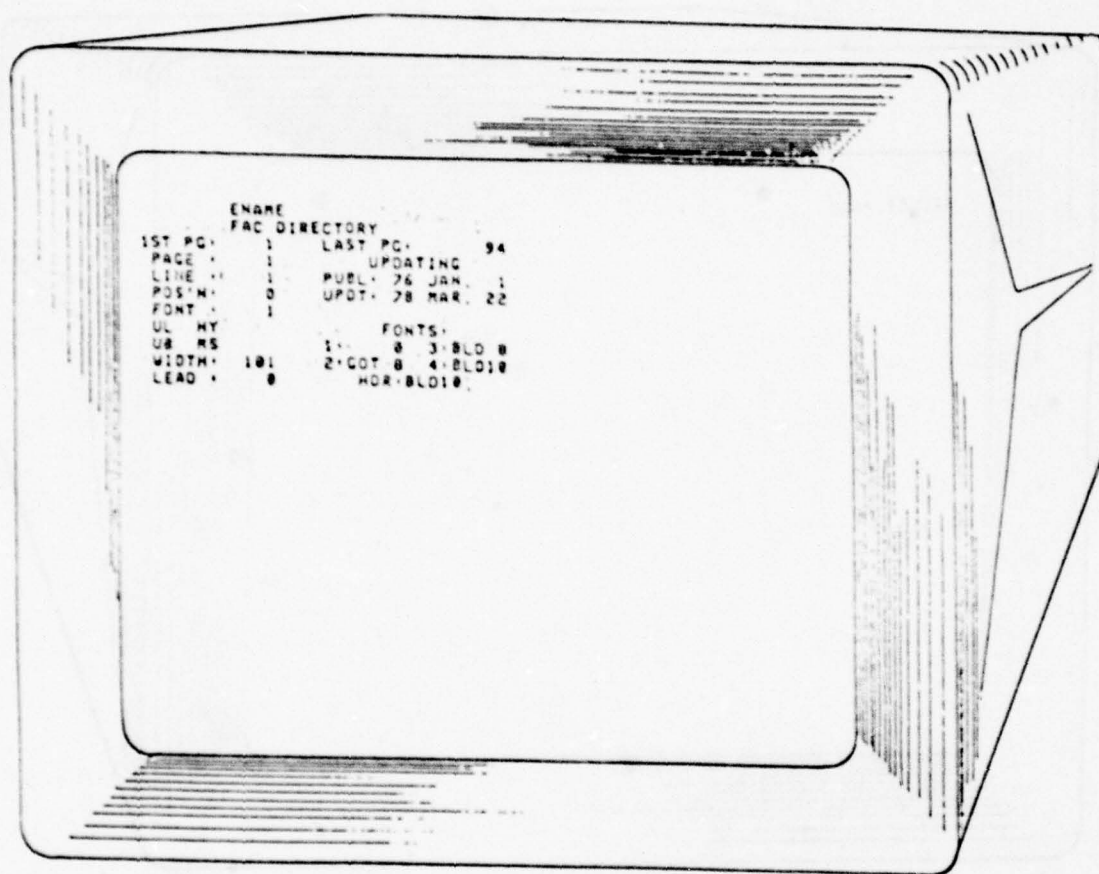
Figure 44   Small TV Screen

o   Under FONTS: are listed the four font sizes selected for this
     chapter; what they are and how big they are.

3.3.4 Page Selection

The page number may now be entered (see Figure 45) and a CARRIAGE RETURN
and the system will show the first few lines of the page (see Figure 46) and
request that confirmation of the page selection; by entering a ".". If "+"
or "-" is entered instead of "." you can go forward a page or backward a
page. The small screen will always show which page is being updated. After
entering the period the system will stay on the page until the mnemonic "X"
for e"X"it is entered.

3.3.5 Cursor Positioning Commands

Updates occur at the position of the cursor. Therefore, flexible cursor
positioning is very important. Positioning is effected as follows:

1. A line number entered, followed by a carriage return, such as 20),
   means position the cursor to line 20. If line 20 is on the screen
   the cursor goes directly to it, if it is not, the terminal will
   rapidly show you a new screen with six lines on it (17-23) and the
   cursor will be on line 20.

2. "A"dvance the cursor (also done by pressing the space bar) moves
   the cursor a space or a word to the right.

3. "BACKSPACE"--move the cursor back one word to the left.

4. "U"p--move the cursor up one line.

5. Line Feed--means move the cursor down one line. If the cursor is
   at the top of the screen or at the bottom, scrolling will take place.

6. "S"earch--allows you to look for a character string anywhere below
   you on the page.

7. "C"hange position the cursor to the next occurrence of a string of
   characters and replace it with another string of characters. For
   example, C ABC (control delete) 123 (control N) will change the next
   occurrence of "ABC" to "123." Depressing "R" will cause PUB to
   change the next occurrence of "ABC" to "123." "R" may be used until
   the page is exhausted. If the space bar is depressed at any time,
   the change command will stop.

SELECT PAGE:1
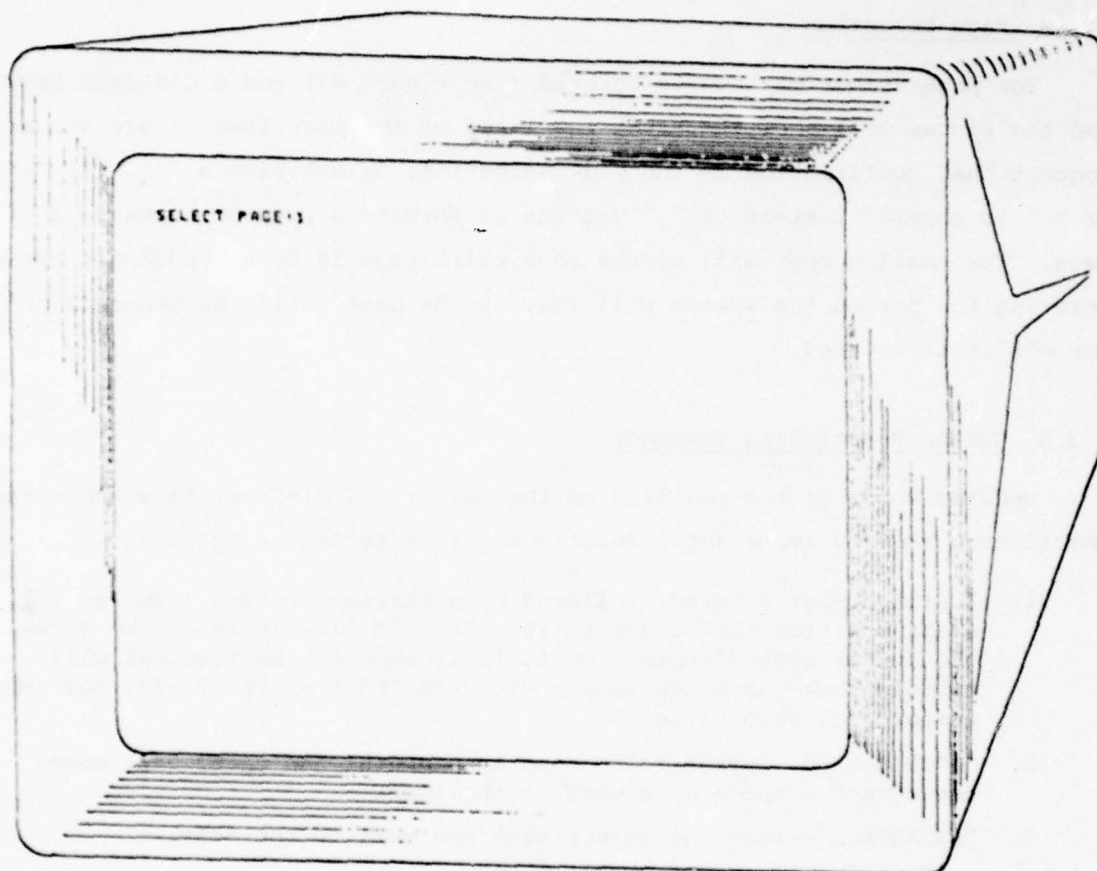
Figure 45 Request a page, key entered: "1"

Figure 46 You can see the first few lines of the page requested.

3.3.6   Insert Command

After entering the "." to confirm the page, positioning to line 5 and advancing the cursor to the end of the first word (the cursor always starts at the top left corner of the screen), "N" or "control key" "N" is entered (see Figure 47). Anything typed in will now be added to the page at the point where the cursor is. Depending upon what is entered the system will adjust the whole page as you enter. It would be very confusing to show all the lines moving around and changing every time a word or a command is entered, so whenever an insert, delete, or change is used, everything on the terminal following the cursor is erased. This does not mean that the system has destroyed the rest of the page. Everything is still there and can be easily called back onto the screen completes rejustified after inserting, deleting, or changing.

The word entered during insert does not go immediately onto the line where it is entered. The system shows it on the beginning of the next line. At the completion of each word entered, the system erases the word on the line below, hyphenates and makes line justification decisions and writes the word into its correct place (see Figures 48 and 49). A space, some special characters (like the underline bracket), and the end of the insert will always cause what is on the lower line to come up to where it belongs. Figure 50 shows a misspelling. To correct the error, hold the "control" key down and enter the "/DEL" command (see Figure 51). This command will erase one character at a time going backward while inserting in the middle of a word.

After completing the word and entering a space, the screen looks like Figure 52. Using the "control" "/DEL" command results in the whole last word being erased (see Figure 53). This is an important feature of the system. Whole words can be deleted as far back on the page as is required while in insert mode! For example, positioning the cursor beyond a word or group of words you want to change, and entering insert mode, the system permits control delete commands to erase the word or group of words to be changed and then allows entry of the replacement words; all without leaving insert mode.

Figure 47 Enter Insert Mode (Page disappears following the insert)

Figure 48 Inserting a Word

Figure 49 Entering a Space or Special Character or Terminating the Insert Sends Word to Proper Place

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

3-73

Figure 50 Enter Another Word

Figure 51 Use of "Control Del" Key to Erase Characters
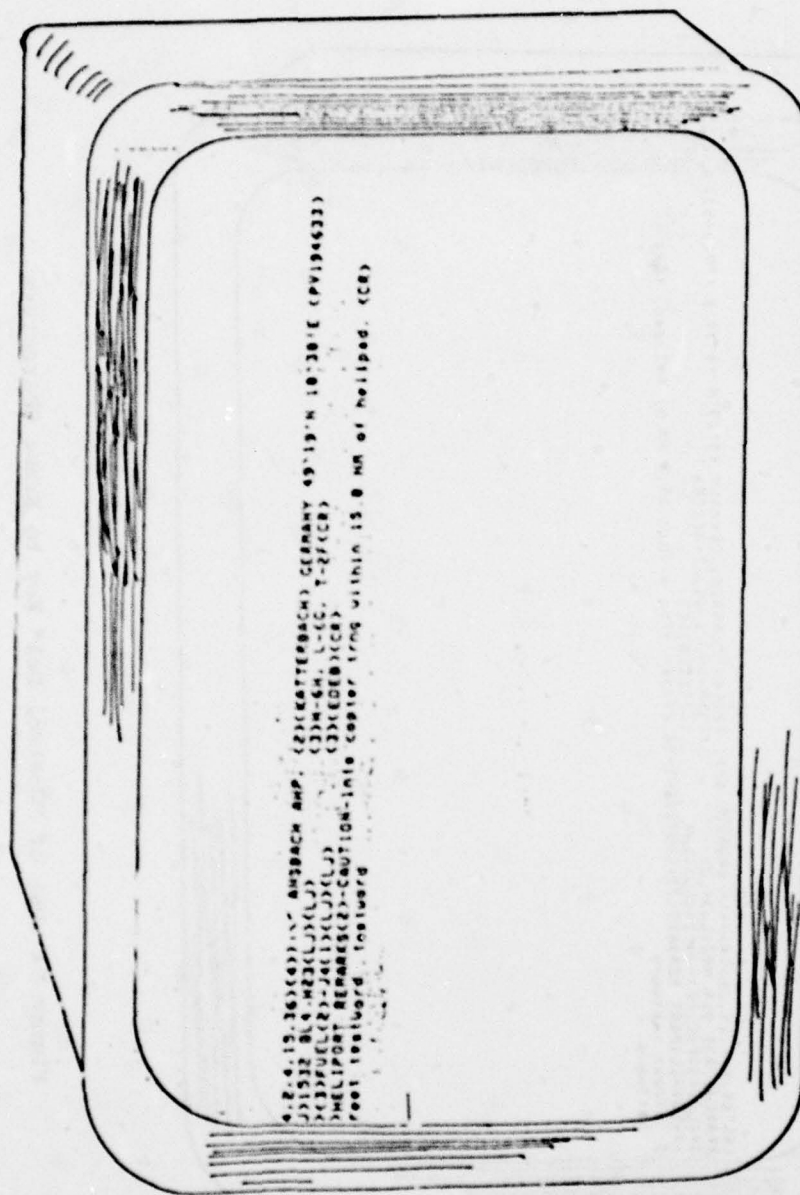
Figure 52 Send Word Up to Line

Figure 53 "Control Delete" Whole Word

Figure 54 shows what is on the page now (insert has been terminated and thereby a redisplay was requested). Nothing has been lost of the original page material.

3.3.7 <u>Exit and Miscellaneous Directory Requests</u>

Figure 55 shows the reply to "X" allowing exiting the page. Entering "QUIT" to the "P" prompt returns you to the "R" or "F" return the "<u>Select Page:</u>" prompt.

Two more commands that can be entered without a password while you have an "R" prompt are "LISTBOOKS" and "LISTFONTS" which return a directory of the books and fonts that are on the system (see Figure 56).

3.3.8 <u>Tabulations, Justification and Indents</u>

Tabulations, justification and indent are accomplished by means of one flexible command: <TB....>

The <TB....> command permits:

a. Setting any number of left justify tab stops at half pica positions in much the same way a typist would specify them at a typewriter:

   <TB, 5, 9, 12, 15, 21, 27> sets tabs at half pica locations 5, 9, 12, 15, 21, and 27. Depressing the TAB key produces the same effect as a typewriter; if the type has been set to position 12 (half pica) depressing the TAB key causes the next characters entered to begin at 15 (half pica) even if this was the first use of the TAB key in creating this line.

b. Setting an automatic indent for succeeding lines, for example:

   <TB 5, 9, 12>

   causes succeeding lines (not terminated by a carriage return to be automatically indented to position 5 (half pica).

c. Right justification is accomplished in two manners:

   (1) Right justification can occur between any two tab points using the specification format "X" R "Y;" e.g. <TB, 20 R 50>" used with the input <u>"TAB" ABC</u> will produce:

       0 (1/2 pica)   20 (1/2 pica)   50 (1/2 pica)   70 (1/2 pica)

Figure 54 Condition of Page After Insert
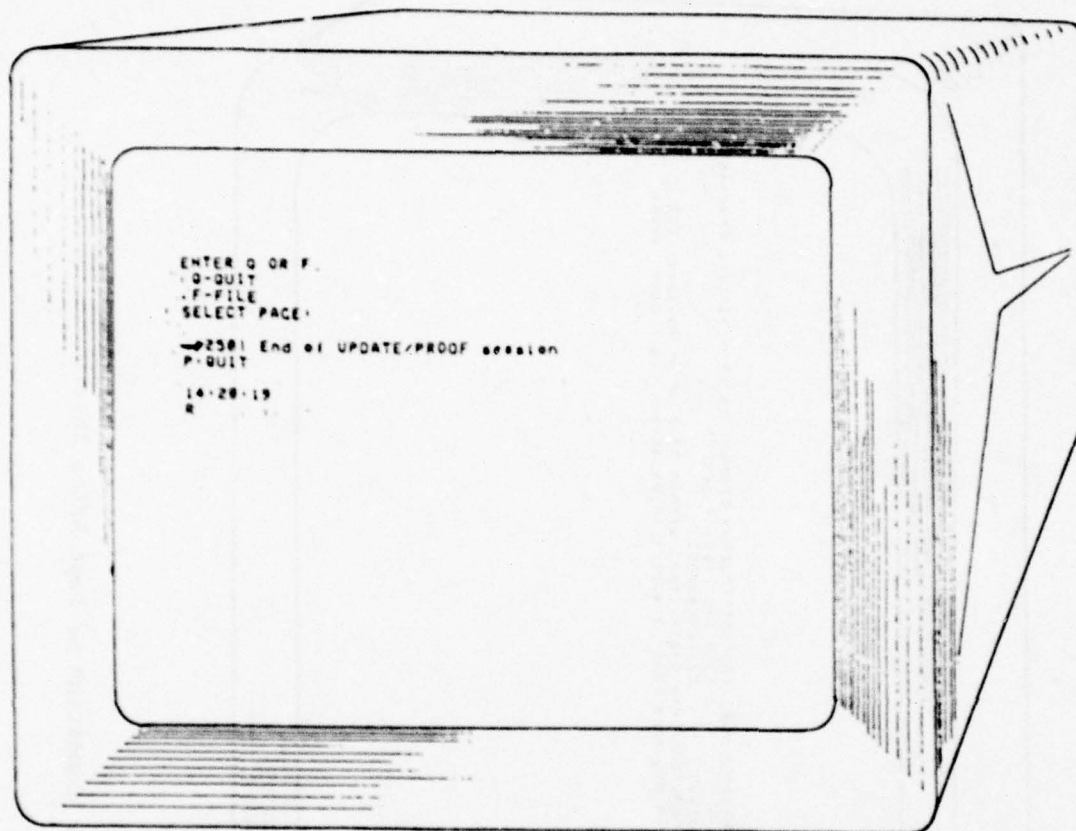
Figure 55 Ending an Update Session

```
09:16:20
R
LISTFONTS
FONTOLD.FT          384.
FONTFUT.FT          384
FONTGOT.FT          384
FONTST2.FT          384
FONTTST.FT          384
09:16:40
R
```
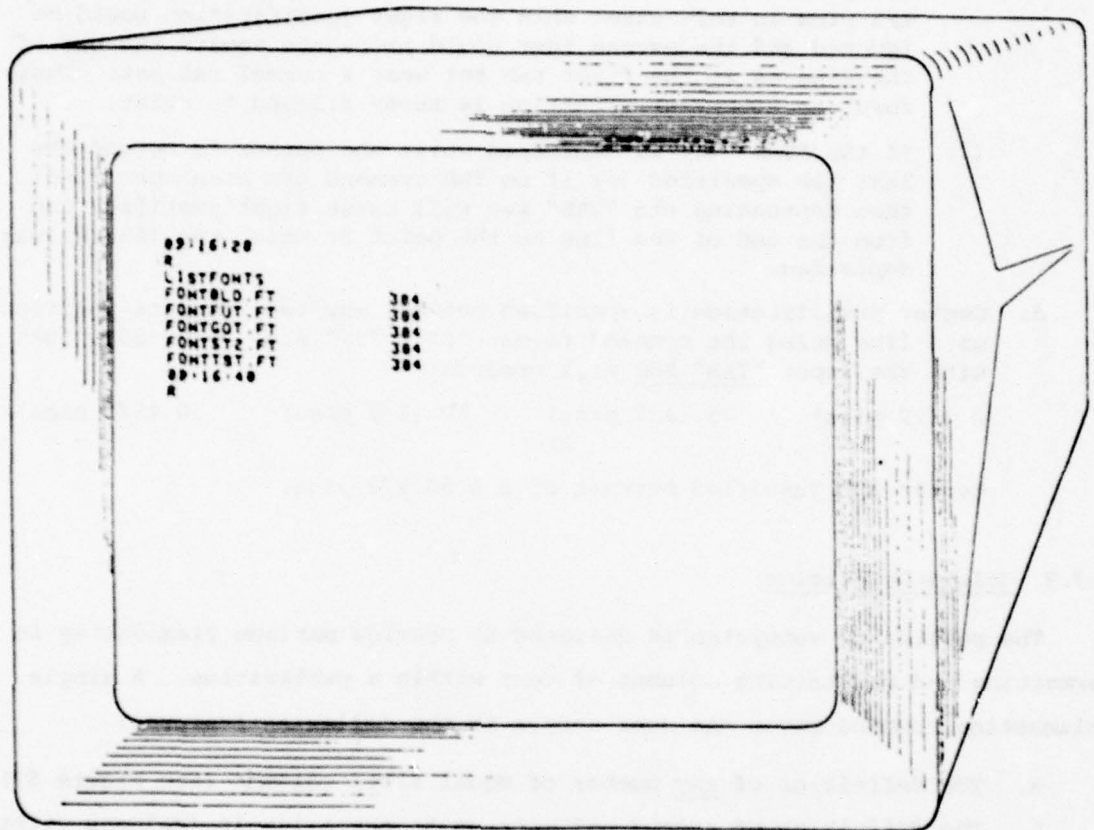
Figure 56 List Fonts

Where the text entered right justified ABC from 50 1/2 pica is right justified from the second positions specified. If enough text was entered to go beyond the first TAB set (20 1/2 pica in this case) then the right justification would be ignored and the excess text would propagate toward the end of the line as if the first tab set were a normal tab set. Therefore, an overprint situation is never allowed to exist.

(2) If the "TAB" key is depressed while the cursor is beyond the last tab specified (or if no TAB command has been specified) then depressing the "TAB" key will cause right justification from the end of the line to the point at which the TAB key was depressed.

d. Center justification is specified between any two 1/2 pica positions on a line using the command format "X" C "Y;" e.g. "<TB, 25 C 50>" with the input "TAB" ABC will produce:

0 (1/2 pica)   25 (1/2 pica)   50 (1/2 pica)   70 (1/2 pica)
                              ABC

Center ABC justified between 25 and 50 1/2 pica.

3.3.9 Column Definition

The publishing subsystem is designed to provide maximum flexibility in formatting and maintaining columns of text within a publication. A single columnation command gives the user access to the following features:

a. The definition of any number of equal sized columns (see Figure 57).

b. The definition of nested columns--up to three levels (columns within columns) (see Figure 58).

c. The ability to specify in half pica (see Figure 59) at all levels of nesting:

(1) The width of the column(s).

(2) The starting position of the (first) column.

(3) The gap between columns.

d. Automatic calculation of column parameters based on defaults or the specification of any of the three parameters.

The format of the column control command is:

<C I# of columns], Starting position of first column, width of the columns, gap between columns>
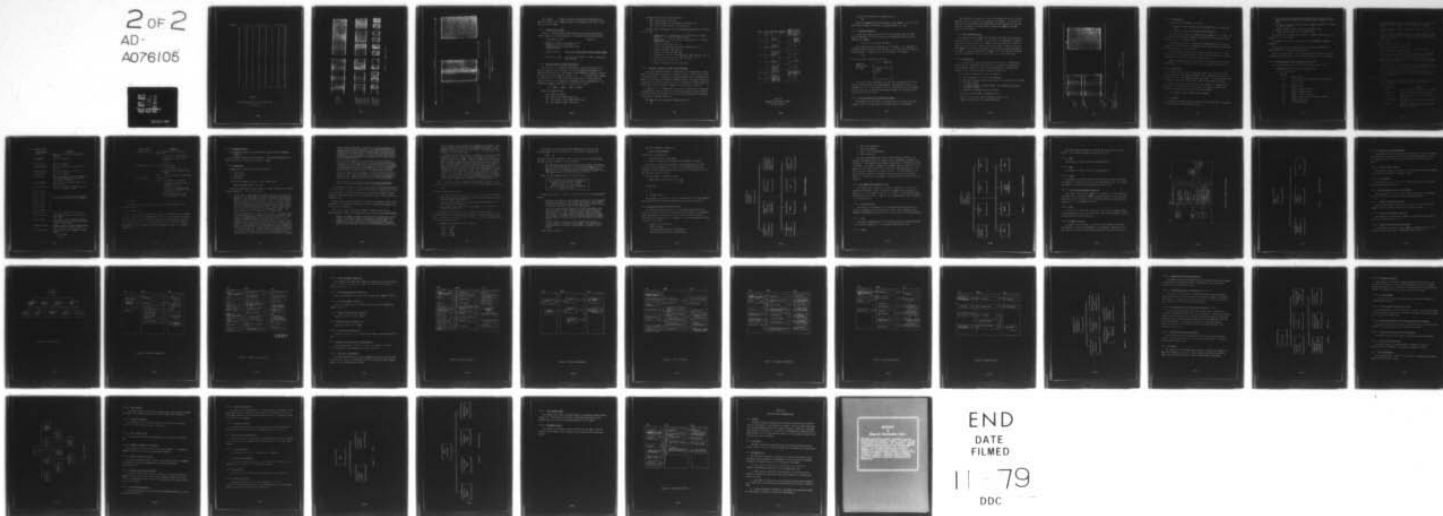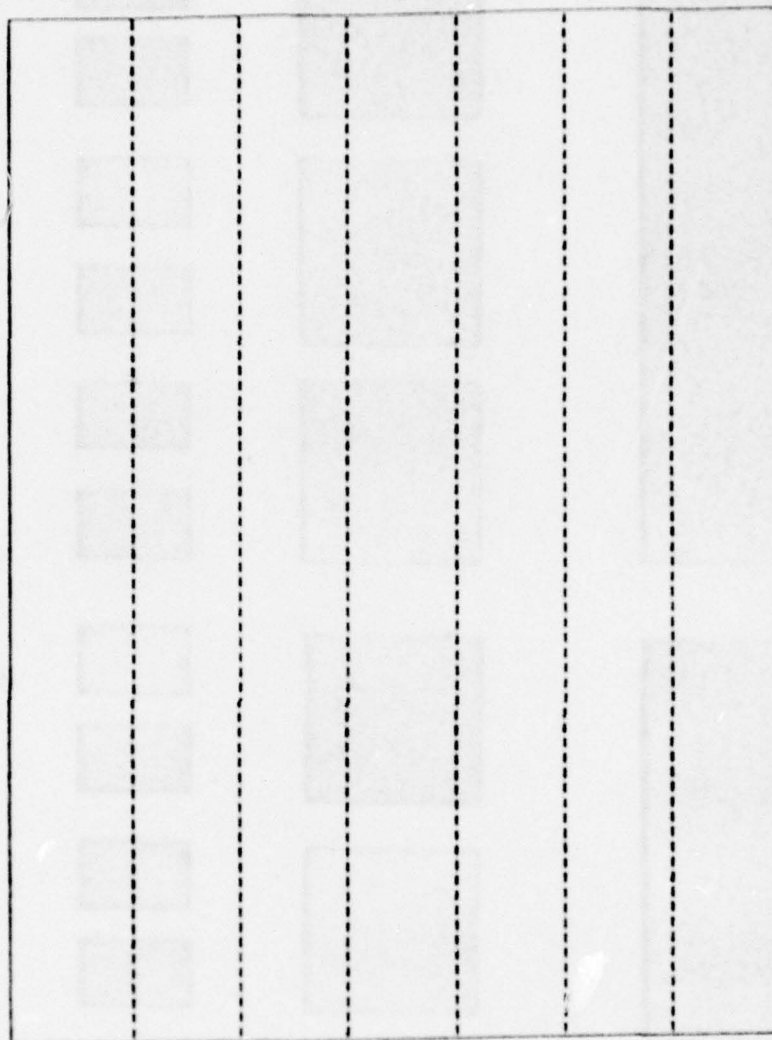
Figure 57

PUB Allows the Definition of Any Number
of Columns

2 Columns defined <C2>
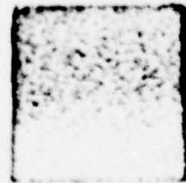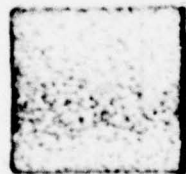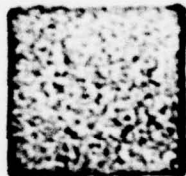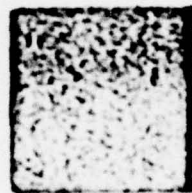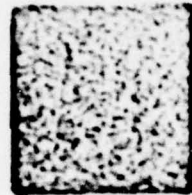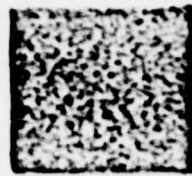
3 Columns defined within each of the two columns <C2><C3>
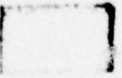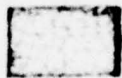
2 Columns defined within 3 columns

Figure 58 Nested Columns

Page Width

Column Width

Gap Between Columns

Column Width

Start Column Position
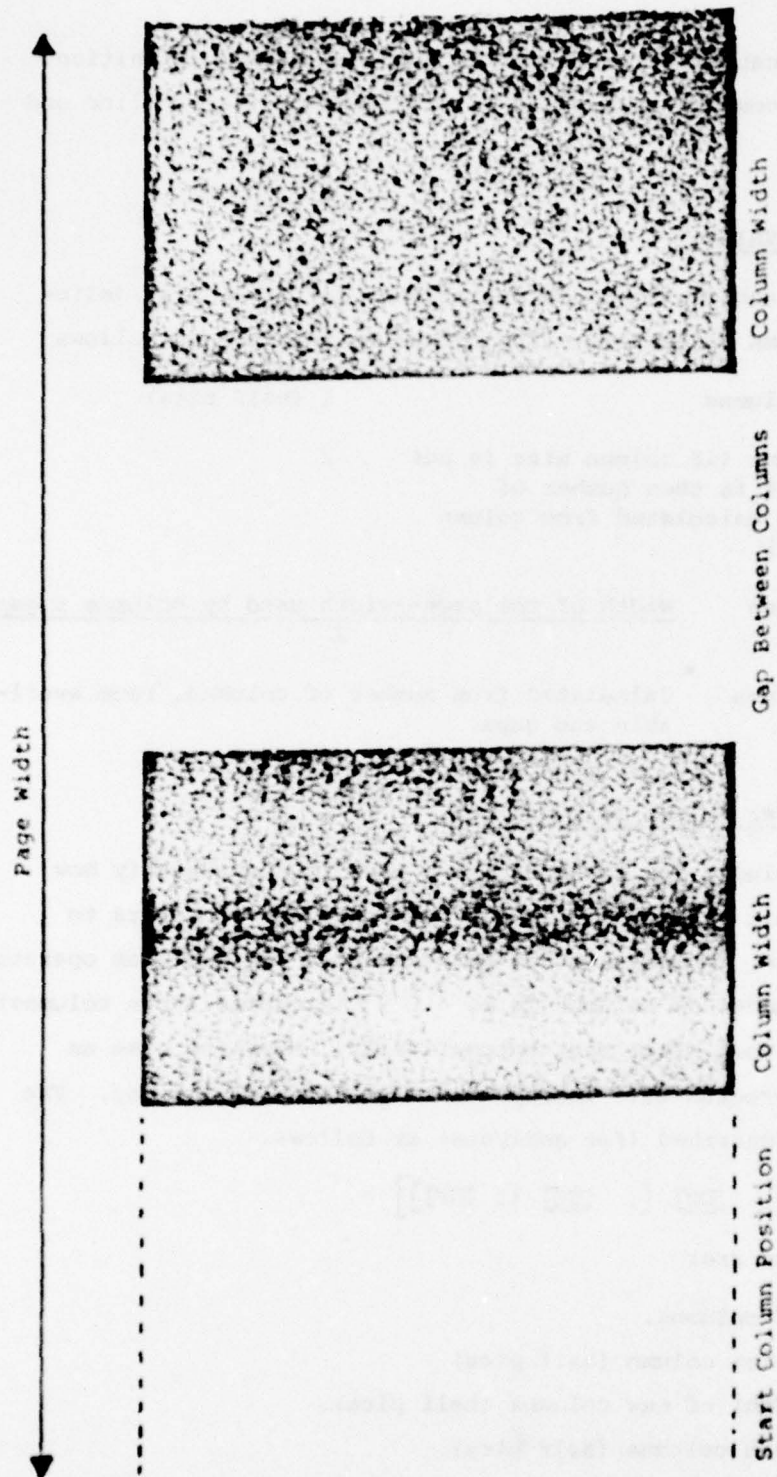
Figure 59

Specifying Width, Starting Position and Gap(s)
Between Column(s)

The command < . C> causes a return to the previous column definition. When the column setting command is used it is displayed on its own line and has its own line number.

3.3.9.1 Column Default Values

PUB will calculate and/or invoke default values for missing or deliberately unspecified column parameters. The default values are as follows:

	Gap between columns	1 (half pica)
a.	Number of columns (if column size is not specified; if it is then number of columns will be calculated from column size specifiers).	2
b.	Starting position	$\dfrac{\text{Width of the page--width used by columns \& gaps}}{2}$
c.	Size of the column	Calculated from number of columns, room available and gaps

3.3.9.2 Detailed Computer Column Calculations

This section is included for those having a need to know exactly how PUB will calculate default values. It is not necessary for operators to master these calculations. The system has been designed to permit an operator to simple specify the number of columns (e.g. < C 3 > produces three columns), the system does all the work after that automatically. Anything else an operator specifies the program will incorporate into its calculations. The columnation command is described (for analysts) as follows:

$$ < \; [.] \; c \; \boxed{NC} \; [, \; \boxed{SC} \; [, \; \boxed{CZ} \; [, \; \boxed{GP}]]] \; > $$

Where the parameters are:

NC: Number of columns.

SC: Start of 1st column (half pica).

CZ: Size (width) of new columns (half pica).

GP: Gap between columns (half pica).

Intermediate results are then defined as:

CW: Current width (half pica).

RM: Room available for new columns in CW (half pica).

TK: Portion of RM actually used (half pica).

Processing will then occur in the order presented based upon the following ten rules:

a. Command form .C causes return to previous level of column definition. No other processing occurs.

b. Command form C is identical to a line feed, LF . No other processing occurs.

c. If GP is not specified, $GP = 1$.

d. If NC is not specified and CZ is not specified, $NC = 2$.

e. If SC is specified, $RM = CW - SC$.

f. If SC is not specified, $RM = CW$.

g. If NC has not yet been determined, $NC = (RM + GP)/(CZ + GP + 1)$.

h. If CZ is not specified, $CZ = (RM + GP)/(NC) - GP - 1$.

i. $TK = (CZ + GP + 1) \cdot NC - GP$.

j. If SC is not specified, $SC = (CW - TK)/2$.

3.3.9.3 Operator Control of Column and Page Breaking

One of the publishing system's capabilities is to automatically repaginate a document after all update is complete. However, there are many cases in which an analyst or operator would not want PUB to divide a paragraph or table or other textual unit between columns or pages. To permit the operator to exercise this form of composition control over page and column format and content the system utilizes the command structure shown in Figure 60.

In addition to the commands which are shown in Figure 60 it is possible to permit PUB to make its own decisions concerning page and column breaking at the end of any line (whether the end of line is calculated by the system or forced by a carriage return (CR)).

The command which tells PUB to "break at will" is:

<MS>

What is Shown on TV	Which key(s) to Enter During Insert	What does the command do	See page(s) for command details	Does this command result in a page break to occur where specified	Does this command permit a new column where specified
<C> or <.C>	<C> or <.C>	Defines columnation		Yes	Yes and will balance columns to that point
<VT>	Control "VT" (k)	Invokes vertical tabulation		Yes	Yes
<(,"X","Y"..)>	<VT,"X","Y"..>	Defines vertical tabulation settings		Yes	Yes
<FF>	Control "L"	Forces new column		Yes	Yes (forced)
<LS>	<LS>	Draws a horizontal line across the column		Yes	Yes
<.LS>...<LS>	<.LS>...<LS>	Draws a box around text		Yes (before and after the box)	Yes (before and after the box)
<LF>	Line feed	Terminates a line and specifically permits column break to occur		Yes	Yes

Figure 60

Commands Which Affect Column

And Page Breaking

To shut off this feature the command used is:

<.MS>

These two commands extend across pages in the chapter. If MS is not
specified throughout the chapter, the system defaults to <.MS>.

3.3.10 <u>Vertical Tabulation</u>

You may space <u>down</u> a page as well as across a page; this is called
Vertical Tabulation. To set up the vertical tabulation positions on the
page use the command:

<VT, # of 1/2 pica down the page, #....>

Up to <u>six</u> vertical tables may be set on the page. It is important to
understand that the distance down the page is calculated from the bottom of
the page number or heading at the top of the page. For example:

<VT,4>

sets the vertical tabulation as follows:

```
                                    Page 45
    Number of      0
    1/2 pica down  1
    the page       2    Vertical tab set
                   3    here
                   4      ←┘
                   5
                   .
                   .
                   .
```
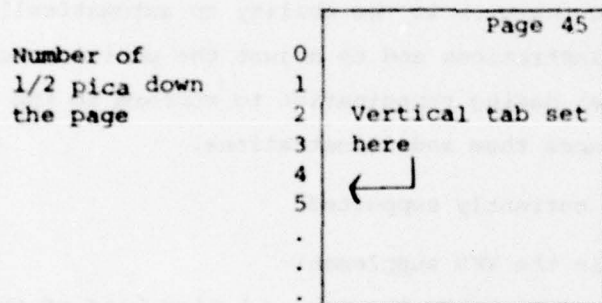
To use the Vertical Tabulation that has been set depress the "VT" key
(key 47) and the control key simultaneously. The system will echo "<VT>,"
informing you that it has advanced to the next available vertical tabulation
setting (just like the Horizontal Tabulation echoes "LJ" and moves to the
next horizontal tab setting).

3.3.11 <u>Draw a Horizontal Line Across the Column</u>

The command <LS> will cause a horizontal line to be drawn across the
column in which it is specified, at the point where it is specified.

The system does not show you a line on the screen but will put the command on a line all by itself with a separate line number. If <LS> falls as the first line of a page during repagination the line specified by the command will <u>not</u> appear in the publication (though the command will remain specified on the page).

3.3.12 Draw a Box Around Text

As we have seen the command <LS> produces a horizontal line across the column in which it is specified. In some cases it is desirable to create a box around a column or part of a column of text (see Figure 61). PUB will draw the box for you at a distance of 3 points from the text when you enter the command "<.L5>" (draw a horizontal line across the columns) followed by the text to be boxed followed by the command "<LS>" (draw another horizontal line and two vertical lines to the previous <.LS> line to form a box).

3.3.13 Diagram Entry

One of PUB's most important features is the ability to automatically leave rooms for diagrams and illustrations and to adjust the positions of those diagrams (compose the page) during repagination to conform to the changes in the text that references them and illustrations.

Five types of diagrams are currently supported:

a. <u>S</u>ide diagrams such as in the VFR supplement

b. <u>H</u>alf page diagrams occupying either the top or bottom half of the page

c. <u>Full</u> <u>P</u>age diagrams

d. In <u>C</u>olumn diagrams of variable height, but extending the width of the defined column

e. <u>R</u>egistration diagrams overlying the page

To specify these diagrams the command used is:

<"D" size in 1/2 pica, "Type of diagram" (S,H,P,C or R)

  "Name of diagram" (up to 10 characters long)>

2 columns are specified

<.LS> is inserted here

<.LS> is inserted here

Box is drawn around text included between <.LS> and <.LS>
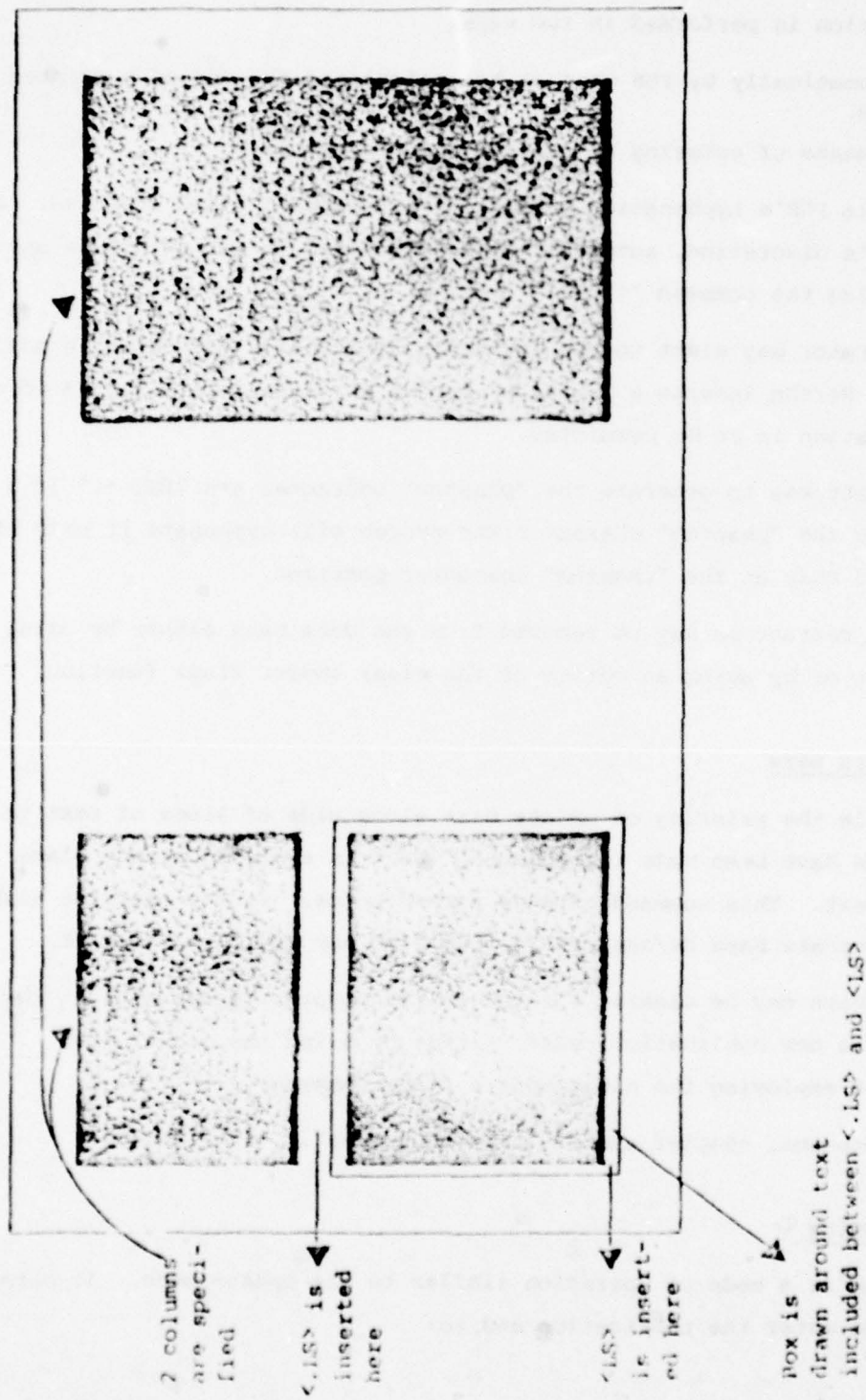
Figure 61

Specifying a Box Around Text

3-91

3.3.14 Hyphenation

Hyphenation is performed in two ways:

1. Automatically by PUB when it has calculated the end of a printed line.
2. By means of entering a "phantom" hyphen manually.

To enable PUB's hyphenation capability the command "<HY>" is used. At the operator's discretion, automatic hyphenation can be turned off at any time by issuing the command "<.HY>."

The operator may elect to specify where hyphenation may occur in any given word. He/she inserts a "phantom" hyphen at those points in the word where hyphenation is to be permitted.

The keystrokes to generate the "phantom" character are "ESC •." If a word contains the "phantom" character the system will hyphenate it only if necessary and only at the "phantom" character position.

Phantom characters may be removed from the data base either by using a global edit or by using an option of the clear update flags function.

3.3.15 Update Bars

To enable the printing of update bars along side of lines of text to which changes have been made the command "<UB>" is employed at any place within the text. This command extends across pages. If the operator wishes to suppress update bars he/she enters "<UB>" at any place in the text.

Update bars may be cleared for the entire chapter (presumably at the beginning of a new publication cycle) either by using the global edit feature or by employing the clear update flags command:

"CU, bookname, chapter number, version number"

3.3.16 Proofing

Proofing is a mode of operation similar to the update mode. It permits a reviewer to enter the publication and to:

a. Verify that all changes have been made correctly (he/she is aided by the asterisks or update bars indicating where changes have been made).

b. Utilize all the positioning and search commands available in UPDATE to inspect the page.

c. Accept, reject or leave the page for further review.

d. Obtain a proof listing on the system printer.

The system will not permit the reviewer to change a publication in any way while he is in Proof Mode.

The command used to enter Proof Mode is "PROOF BOOKNAME, CHAPTER #, VERSION #."

The system logs the reviewers activity. The log is available for report generation to monitor proofing activity.

When the reviewer enters a page in Proof Mode he is offered the option of obtaining a proof listing for that page by responding "L" to the prompt.

3.3.17 File Maintenance (Book, Chapter/Version, Font)

All file maintenance activities follow consistent update rules:

I. Command syntax for entering maintenance mode

    1. Specify the function:

        Function

CB	Create a book
LB	List a book and all chapter/versions associated with the book
DB	Delete a book
UB	Update a book
CC	Create a chapter/version
UC	Update a chapter/version
DC	Delete a chapter/version
CF	Create a font file with default EBR widths
UF	Update a font file
LF	List the font file

2. Specify the mnemonic name of the book or font (e.g., ES or NGC) (for book and font commands stop here and enter "carriage return").

3. Specify the chapter number desired in the form: C # e.g., "Cl" is chapter one.

4. Specify the version number desired in the form: V # e.g., "V1" is version one. (If you do not enter a version number but skip to step 5) Immediately then PUB will get you the last version number created.

5. Enter "carriage return."

II. Command syntax once in maintenance mode:

1. All fields displayed are numbered.

2. All fields having an asterisk next to them are protected: PUB will not allow you to change them.

3. After you enter the command (I above) you will get a numbered list of all fields and the current values in the field.

4. At any time the command "L carriage return" will show you the list again, as it now appears. If you just want one line enter "L#" and carriage return.

5. To change the value of a field enter the field number, "," the new value you wish the field to have, and a carriage return, e.g.:

   5,100   results in a value of 100 entered into field 5.

6. Entering a carriage return without a number will return you to the "P:" prompt (so you can use the other PUB functions) and store the changed file.

III. Field values

1. For book update and create you may only supply the book title which will subsequently appear on the Informer terminal.

2. For font updates:

Screen Fields	Meaning
OCTAL	Prompt for ASCII code you wish to update
1 EBR Code	Code that EBR uses to form character
2 Points	Width in points of this 36 point high character
3 ALT EBR Code	EBR code for this character when circled or dotted
4 Points	Width in points for this character when circled or dotted

3. For Chapter Update:

Screen Fields	Meaning
1 CU Title	Chapter title you wish to appear on Informer
2 * Chapter #	May not be changed
3 *	
4 * Version #	May not be changed
5 VERT PICA SIZE	Depth of the page in Pica (1/6") not 1/2 pica
6 HORZ PICA SIZE	Width of the page in Pica (1/6")
7 * Record Size	May not be changed
8 * File Size	Number of pages currently allocated for this chapter/version can be changed only during repagination
9 Last Updated	Date when last change was made to this chapter/version
10 Last Publicated	Date when repagination/EBR output was run on this chapter/version
11 Font 1 Size	
12 Font 1 Name	
13 Font 2 Size	Size in points and three letter mnemonic for font/sizes desired in this chapter
14 Font 2 Name	
15 Font 3 Size	
16 Font 3 Name	
17 Font 4 Size	
18 Font 4 Name	
19 Page # Font Size	Size of the page number in points
20 Page # Font NM	Three letter mnemonic of font used for page number
21 First Page #	First page of this chapter (used when creating the chapter; repagination will accept it as a default)
22 Number of Pages	Number of pages in chapter/verify (used when creating the chapter version)
23 Vert Page # Pos	Vertical positioning commands for page number

B = bottom

T = top

Screen Fields	Meaning
24 Horz Page # Pos	Horizontal positioning commands for page number
	I = inside (i.e., near binding)
	O = outside (i.e., away from binding)
	C = center
25 With Page #	Format of page number
	N = chapter number (e.g., 2-7)
	T = section title accompanies page number
	Blank = nothing accompanies page number
26 Page Index	Index maintenance
	Blank = no index for this chapter
	G = Generate an index next time repagination is run
	Y = Yes there is an index and it will be regenerated each time repagination is run
	J = don't create an index next time repagination is run

3.3.18 Titles

Titles are specified before the section to which they refer in the following manner:

<TL, name of title>

A "-" within the title name will be shown as a space within the document. If title is specified to accompany page number (see file maintenance, chapter version, field _) then the title will appear with the page number (whenever it is specified to be). If titles change within a page the first and last applicable title to the page are printed separated by a dash (e.g., ANGOLA-BERMUDA 3-17).

3.3.19 Management Report

The management report is called from the console using the command:

    MRREPORT

to an "R" prompt (before entering password). Individual chapter/versions
may be selected for reporting in response to prompt.

3.3.20 Repagination

Repagination consists of three major sections:

1. Global Edits

2. Repagination

3. EBR Output

The following command is used to invoke repagination:

    REPAG Bookname Chapter # Version #

PUB will respond with aids and prompts to enable the analyst to achieve
proper repagination. The first such aid is:

o You are about to be prompted for global edit sets. Each set you
   specify will be comprised of two parts. Each part is a string of
   characters, the type that are found in the pages of publications.
   The first string is called the search string. As the old (input)
   version is read it will be checked to see if it contains strings
   identical to the search string. Each time such a string is found,
   it is deleted and replaced with the second string in the global
   edit set, called the "replacement string." These changes to the
   publication will be reflected in all repagination output. The
   input version will not be affected by the global edits.

   You will be prompted for one string at a time. In response to any
   of these prompts, you can respond by entering a character string,
   by entering an immediate insertion character (control "N") or by
   entering an immediate deletion character (control "D"). If a
   character string is entered it can be as long as 30 bytes. Most
   characters take one byte, but others such as the circled or dotted
   characters (like the dotted triangle) take two. If you exceed 30
   bytes, an error message will be issued and you will be prompted to
   reenter the string. The character string is terminated by hitting
   the insertion character. If while entering the character string,
   you mis-type one or more characters, you can backout by hitting
   rubouts or start the string over by hitting the deletion character.

If the insertion character is entered with no string preceding it, then one of two things will result. If a search string was being prompted for, then the insertion character indicates that you are finished entering global edits. If a replacement string was being prompted for, then the insertion character indicates that there is no replacement string and that all instances of the search string are to be simply deleted.

If the deletion character is entered with no string preceding it, then one of two things will result. If a search string was being prompted for, then the deletion character will result in the restart of global edit entry; that is all global edit sets will be deleted. If a replacement string was being prompted for, then it indicates that the previous search string is to be deleted and reentered.

There is a limit to the total amount of global editions that can be specified. This limit is 511 bytes. If you would like to have a running tally of remaining bytes given just before each prompt for a search string, then respond accordingly to the next selection.

Read the above carefully since once you start specifying global edits, it will not be possible to have this message reprinted.

It is possible to elect to run just global edits by responding negatively to the repagination requests that follow specification of global edits. If it is only desired to run global edits then do not specify repagination pages because it takes much longer to effect repagination than it does to just do a global edit run.

After global edits are entered (if you wish to) repaginate without using global edits. A "control N" response to the first search string prompt will get you out of global edit.

Now you are ready to specify the number of pages you wish the new chapter/version to have. The system will show you the following HELP message:

Specify the number of pages to allocate for the new version of this chapter. Allow ample room for the expansion of pages in the old version, due to the insertion of new data into those pages. If insufficient room is allocated, this repagination will be aborted before all data from the old version is processed and recovery will

3-98

not be possible without redoing this repagination from start. Bear in mind, however, that the disk can accommodate only about 20,000 pages of data if it is used for nothing else. Therefore, if the maximum (2047) was always specified, only 10 chapter/versions could be stored on disk for all books.

Four numbers are given before you are prompted to specify the number of pages to allocate. These numbers should be used to judge a proper response to the prompt. The first number is labelled "minimum" and indicates the smallest number of pages which can be specified; any smaller number of pages would assure failure of repagination. The next number of labelled "marginal" and is the number of pages in the original version; if none of the data added to this chapter since the last time these pages were repaginated will cause new pages to form, then use this value. The next number is labelled "nominal" and is a number which allows some leeway for new page formation; you can select this value by responding with a carriage return. The last number, labelled "maximum" indicates the largest number of pages which can be allocated; system limitations prohibit the allocation of more than 2047 pages to a chapter/version.

After you have specified the number of pages (always estimate high all it costs you is a little space on the disk). You will be prompted as follows:

PAGES:

to which you may respond either:

1. With a page number (indicating that you wish this page repaginated).

2. With two page numbers separated by a dash (indicating that you wish a range of pages repaginated).

3. With a dash followed by a page number (indicating that you do not wish that page repaginated.

4. With a dash, page number, dash, page number (e.g., 120-130; not pages 120 through 130).

A carriage return ends the entry and returns the prompt "PAGES:." You may now continue to enter repagination requests. You may overlap page numbers is any fashion or order you please. The last entered page configuration is the one that will be used.

For example, you might enter the following:

PAGES: 16-44 ↓
PAGES: -18 ↓
PAGES: 23-79 ↓
PAGES: -65-68 ↓

This sequence by itself would cause repagination to occur on the following pages: 16, 17, 19-64, 69-79. If you were to continue with:

PAGES: 18

then page 18 would be reselected. When you are finished entering use two carriage returns to advance to the next prompt:

o   The repagination that you are setting up will result in the forma-
    tion of a new chapter/version. Specify the page number of the first
    page in this C/V. You may specify that this page number be the
    same as it is in the original version by entering a carriage return.
    Page numbers can be in the range from zero to 30000.

When you have responded, PUB will inform you that:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!   Page allocation has started.  Repagination   !!
!!      Parameter Specification will resume        !!
!!         when allocation has successfully        !!
!! completed.  Allocation takes approximately      !!
!!            0.3 seconds per page.                 !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

When page allocation is complete, PUB will give you another repagination prompt:

o   A three letter identifier was assigned to each font in this chapter.
    There may be as many as five of these fonts used in this chapter;
    four for the body of the text and one more for the lines at the top
    or bottom of the pages containing the page numbers.

    For each of these fonts an EBR selection code is required. The
    selection code is a number from zero to 1023 used by the EBR to
    identify the font to use. Defaults exist for many of the font
    identifiers. The default values are shown in parenthesis as part
    of the prompt. The default can be selected by typing a carriage
    return instead of typing a selection code followed by a carriage
    return.

    You can restart the selection code specification process by
    hitting a rubout in response to the prompt. This notice will be
    reissued whenever you enter a question mark in response to the
    prompt.

Rubout means "control ?"

The next repagination prompt is:

FALSE BOTTOM PARAMETER:

to which you can respond with:

1. A carriage return (echoes NONE)

2. Number of points (not to exceed half the depth of the page)

This feature is useful in spreading out data across pages to avoid intentionally left blank pages. It does this by artifically shrinking the pages. Caution should be employed in the use of this feature until you have experience with the results.

The last prompt in repagination says:

FACING PAGES; PAGE ON LEFT IS $\emptyset$ = EVEN

1 = ODD:

Enter either:

1. $\emptyset$

2. 1

3. Carriage return

This prompt is necessary to inform the system where inside page numbers will go and to permit overflowed paragraphs to occur on facing pages.

3.4 Publishing Subsystem Software Functions

The software required to support the Publishing Subsystem has been divided into modules according to subsystem process automation requirements. These modules were then placed into eight groups according to general functional or procedural topics or programs. These eight programs are shown in Figure 62 and include:

o Command line interpreter

o MRDOS utilities

o File maintenance facilities development

o Pagination, justification, global edition

Publishing Subsystem

Command Line Interpreter

MRDOS Utilities (I/O, task, etc.)

File Maintenance

Pagination Justification and Global Edits

EBR Output

Editor

HELP
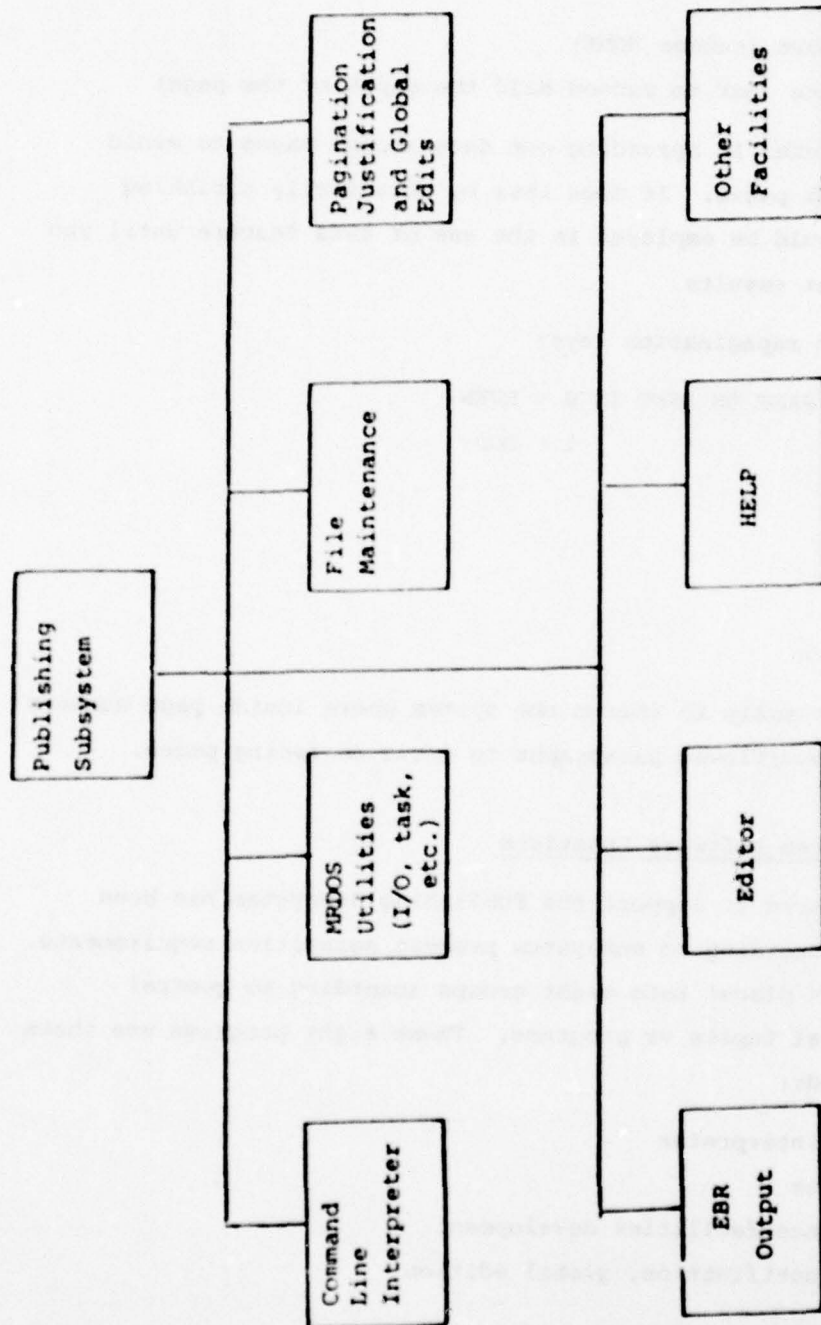
Other Facilities

Figure 62   Publishing Subsystem

o   EBR output generation

o   Editor development

o   Help (on-line documentation)

o   Other facilities

The first two programs are actually vendor supplied software. The
software development effort was divided into six development programs cor-
responding to the remaining six programs listed above. All modules are coded
in FORTRAN. Each module is comprised of FORTRAN routines, operations docu-
mentation, test data, and link edit instructions, when appropriate. To the
extent possible, modules do not share routines. In-line documentation is
of the categoric variety to introduce each routine and of the narrative
variety to introduce each "paragraph" (substantial block of code) within the
routines.

3.4.1 The Command Line Interpreter (CLI)

The command line interpreter, as shown in Figure 63, is used in the
Publishing Subsystem to accept and parse the command strings entered in
response to the CLI prompt "R" and then to dispatch to the appropriate main
level module such as "PUB" or "HELP." This program supports and is essential
to all user functions.

3.4.1.1 The "@" Function

This module is instrumental in the interpretation and processing of
those CLI commands which are bracketed by signs. These commands include
@BACKUP@ and @RESTORE@ for data base back up and archival storage maintenance.

3.4.1.2 LIST

This module reformats and lists selected entries of the MRDOS directory.
It is therefore needed for listing available fonts and book ID's.

3.4.1.3 PRINT
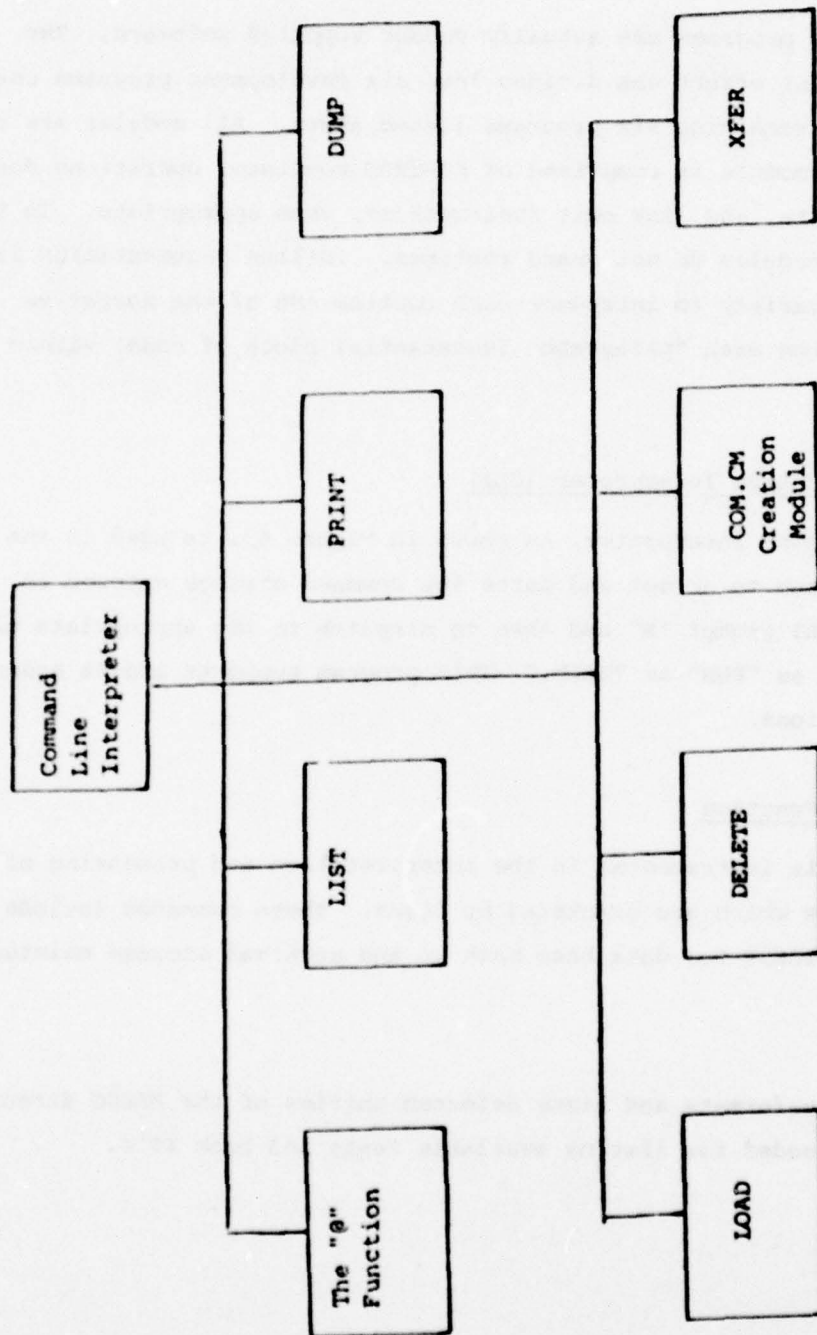
Figure 63 Command Line Interpreter

This module takes an ASCII file and directs this data to the line printer. It is therefore needed in printing the log.

3.4.1.4 DUMP

This module is used in effecting a system back up.

3.4.1.5 LOAD

This module is used in restoring a system back up.

3.4.1.6 DELETE

This module will delete an MRDOS file from the disk and thus release the storage it occupied on disk for use by other MRDOS files. It is therefore needed in deleting the log and deleting a font file.

3.4.1.7 COM.CM Creation Module (Figure 64)

This module generates the command file "COM.CM" (or in the foreground "FCOM.CM") which contains the command line data written into it by CLI and read back by "PUB" or "HELP." It is therefore essential to all functions except for CLI file management functions.

3.4.1.8 XFER

In the event of catastrophic data loss this module can support extraordinary maintenance and recovery procedures when adequate data back up has not been maintained or has been lost.

3.4.2 The MRDOS Utilities

The MRDOS utilities shown in Figure 65 are the modules provided by Data General to allow the programmer to take advantage of MRDOS abilities and conventions. Each of these modules are used by all user functions.
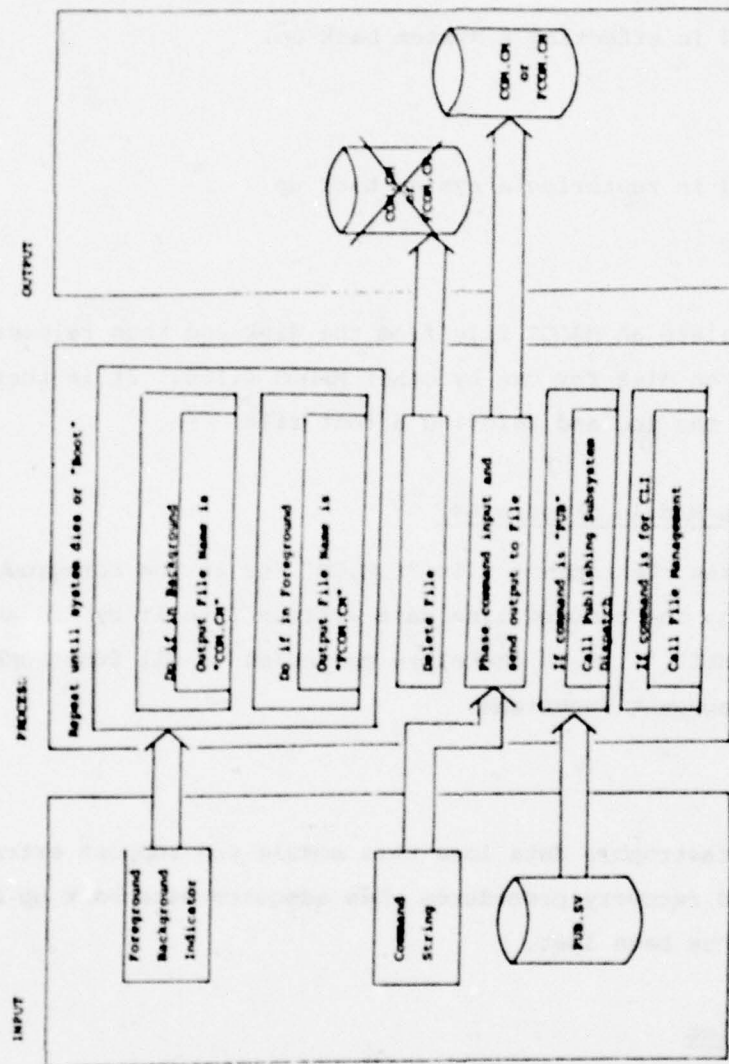
Figure 64   COM.CM Creation Module
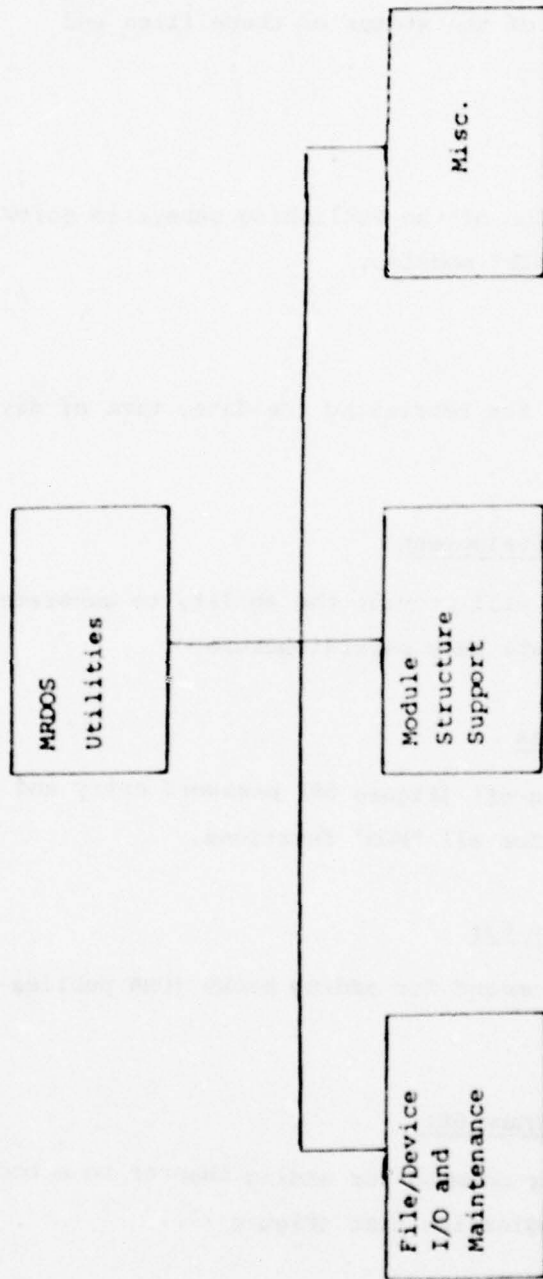
Figure 65   MRDOS Utilities

3.4.2.1 File/Device I/O and Maintenance

These modules allow for the creation and deletion of files, the reading and writing of file and device data, the control and maintenance of these files and devices and the monitoring of the status of these files and devices.

3.4.2.2 Module Structure Support

These modules support the division of the Publishing Subsystem software into overlays and "SWAPable"/"CHAINable" modules.

3.4.2.3 Miscellaneous

These modules include utilities for retrieving the date, time of day, and background/foreground status.

3.4.3 File Maintenance Facilities Development

This program shown in Figure 66 will provide the ability to generate and alter the publishing subsystem data base superstructure.

3.4.3.1 Password and Dispatch Modules

These modules support log-on/log-off (Figure 66) password entry and command line retrieval and dispatch for all "PUB" functions.

3.4.3.2 Book File Generation (Figure 67)

This supports the create book command for adding books (DMA publications) to the data base (Figure 20).

3.4.3.3 Chapter File Generation (Figure 68)

This supports the create chapter command for adding chapter to a book (Figure 68) and is also used at repagination time (Figure 27).
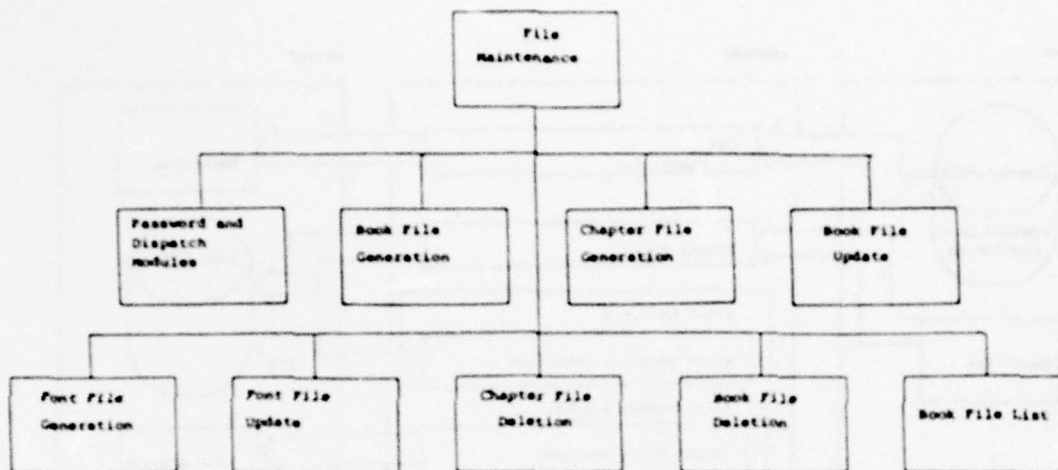
Figure 66 File Maintenance

INPUT

PROCESS

OUTPUT

Command File

PUBACOM.TD or PUBFCOM.TD

Booktitle

GET
   Booktitle

Create File

Error Code = 0

Write header to bookfile

File already exists.

Output error message.

Other error

Output error message

Booktitle

Book File

File already exists.

operating system error "n"
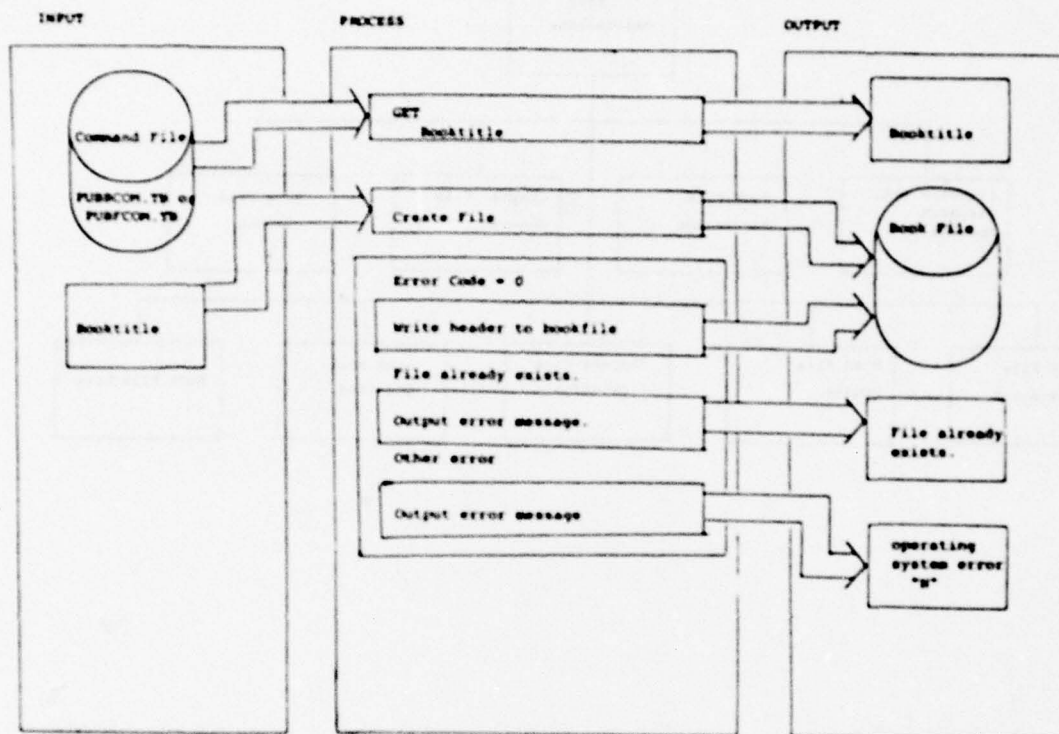
Figure 67 Book File Generation

3-110
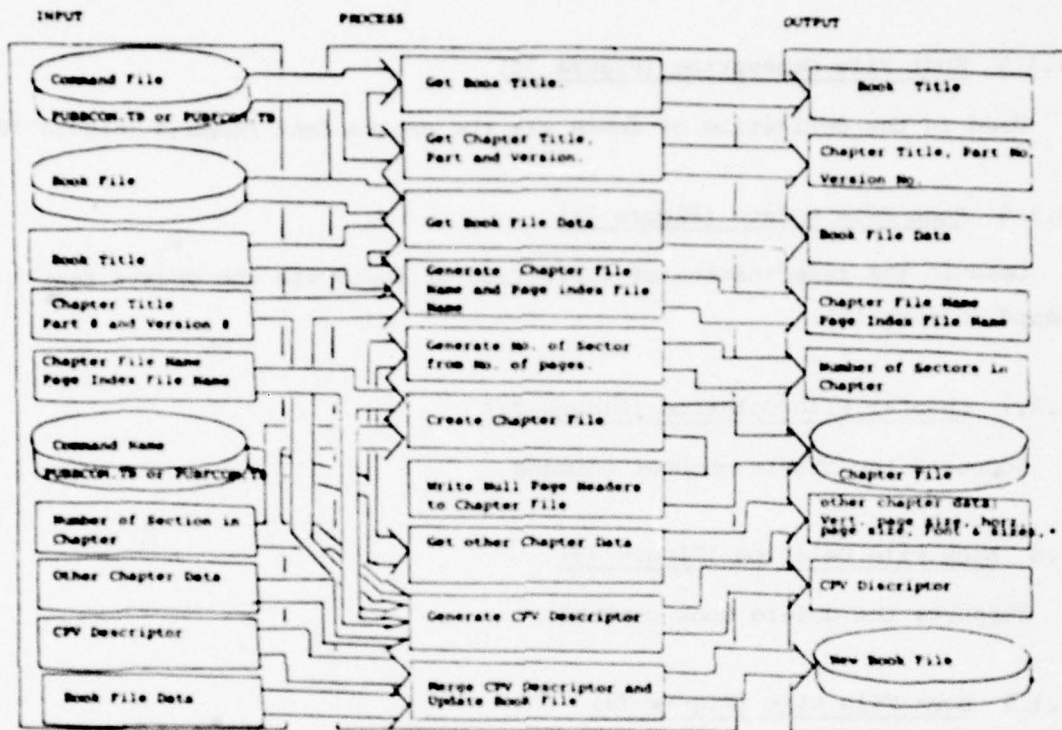
* First Page Number, Last page number, page number format, Page index.

Figure 68   Chapter File Generation

3-111

3.4.3.4 Book File Update (Figure 69)

This supports the update book command for specifying the font selection (Figure 6), page index generation (Figure 28), and general book update functions (Figure 6).

3.4.3.5 Font File Generation (Figure 70)

Used in the definition of fonts via the create font command (Figure 18).

3.4.3.6 Font File Update (Figure 71)

Used in the redefinition of correction of fonts via the update font command (Figure 18).

3.4.3.7 Chapter File Deletion (Figure 72)

Supports the delete chapter command.

3.4.8 Book File Deletion (Figure 73)

Supports the delete book command.

3.4.3.9 Book File List (Figure 74)

Provides the ability to list chapters and chapter descriptive data of a book.

3.4.4 Pagination, Justification, Global Edition

This program shown in Figure 75 provides those modules necessary for the global edit and repagination process (Figure 24).

3.4.4.1 New Version Installment

This module determines the correct parameters for creation of the new version of the chapter to be paginated and then invokes the create chapter module of the file maintenance program.
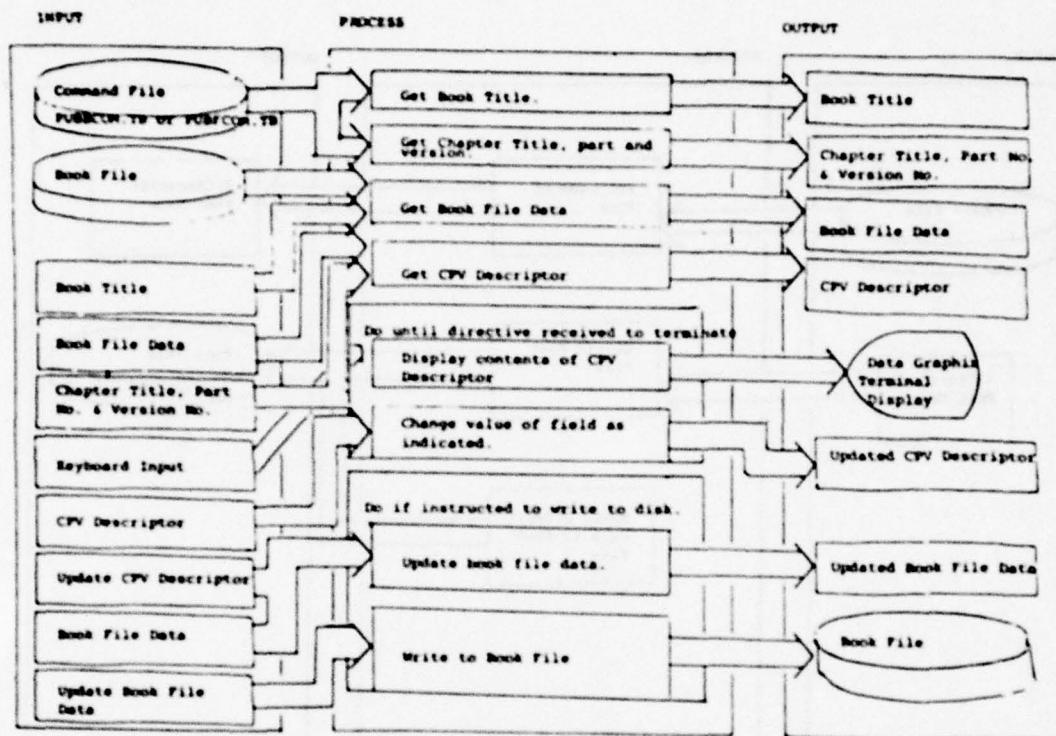
Figure 69 Book File Update

3-113

INPUT   PROCESS   OUTPUT

Command File   Get name of   3-Character
                               Font   Font Name

3-Character   Create Font   Font File
Font Name   File   Font***.F

                               Write Default
                               data to Font
                               File
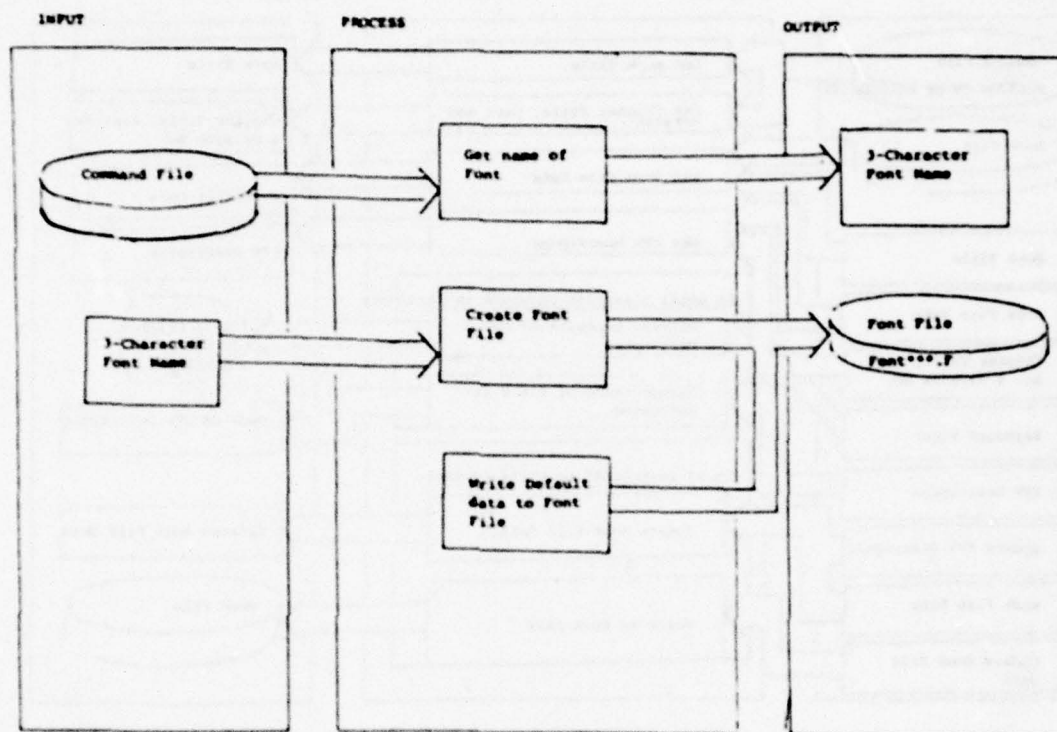
Figure 70 Font File Generation

3-114

INPUT   PROCESS   OUTPUT

Command File
PUBACOM.TB or PUBFCOM.TB

)-Character Font Name   Get Name of Font.   )-charater Font Name

Font File
Font one.p   Get Font Data.   Font Data

                               Do until Directive received to
                               terminate

Key Board Input   Get Octal Character pos. from   Octal Char. Pos.
                               Keyboard

                               Do until directive recieved to Data Graphis terminal
Octal Char. pos.   terminate.   display

                                   Display the entry

Font Data   Update data as directed   Updated Font Data

                               do if directed to save   Updated
                                                                Font File Font Assp.
)-charater Font name   write update data to Font File

Update Font Data   Enter update in log.   Log
                                                                PUBLILOG.TB or PUBFLOG.TB

Figure 71   Font File Update

3-115
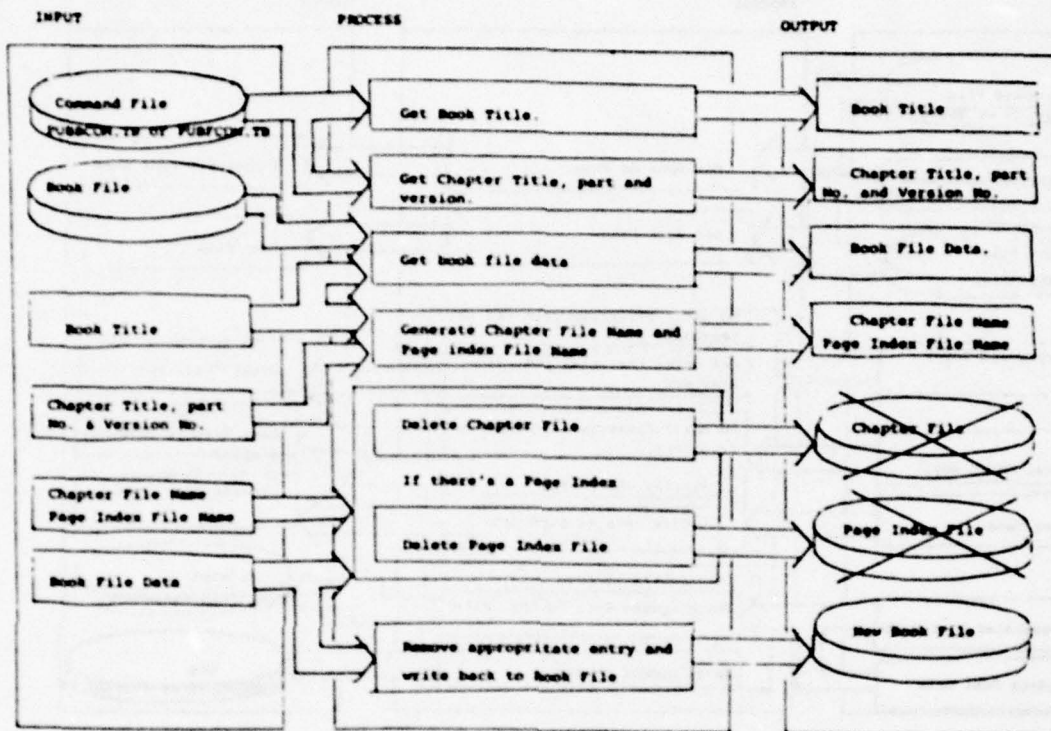
Figure 72 Chapter File Deletion

3-116

INPUT   PROCESS   OUTPUT

```
┌─────────────────────┐   ┌────────────────────────────┐   ┌─────────────────────────┐
│  ╭───────────────╮  │   │  ┌──────────────────────┐   │   │                         │
│ ( Command File    ) │   │  │ Get Book Title       │──┼───┼──▶┌──────────────────┐  │
│ ( ELBKIN.TB or    ) │──┼──▶  └──────────────────────┘   │   │  │ Book Title      │  │
│ ( PUBSPGM.TB       ) │   │                              │   │  └──────────────────┘  │
│  ╰───────────────╯  │   │  ┌──────────────────────┐   │   │                         │
│ ┌─────────────────┐ │   │  │ Get Book File Data   │──┼───┼──▶┌──────────────────┐  │
│ │ Book Title      │─┼───┼──▶  └──────────────────────┘   │   │  │ Book File Data  │  │
│ └─────────────────┘ │   │                              │   │  └──────────────────┘  │
│  ╭───────────────╮  │   │  ┌──────────────────────┐   │   │                         │
│ ( Book File       )─┼───┤  │ For each entry       │   │   │    ╭───────────────╮   │
│  ╰───────────────╯  │   │  │ ┌──────────────────┐ │   │   │   ( Chapter File   )   │
│                     │   │  │ │ Delete Chapter   │─┼─┼───┼──▶╰───────────────╯   │
│                     │   │  │ │ File             │ │   │   │                         │
│                     │   │  │ └──────────────────┘ │   │   │    ╭───────────────╮   │
│ ┌─────────────────┐ │   │  │ ┌──────────────────┐ │   │   │   ( Page Index File )   │
│ │ Book File Data  │─┼───┼──▶  │ If page index    │─┼─┼───┼──▶╰───────────────╯   │
│ └─────────────────┘ │   │  │ │ Delete Page Index│ │   │   │                         │
│                     │   │  │ └──────────────────┘ │   │   │    ╭───────────────╮   │
│                     │   │  │ ┌──────────────────┐ │   │   │   ( Log             )   │
│                     │   │  │ │ Make Log entry   │─┼─┼───┼──▶( PUBBLOG.TB or     )   │
│                     │   │  │ │ indicating       │ │   │   │   ( PUBFLOG.TB      )   │
│                     │   │  │ │ chapter deleted  │ │   │   │    ╰───────────────╯   │
│                     │   │  │ └──────────────────┘ │   │   │    ╭───────────────╮   │
│                     │   │  │ ┌──────────────────┐ │   │   │   ( Book File       )   │
│                     │   │  │ │ Delete Book File │─┼─┼───┼──▶╰───────────────╯   │
│                     │   │  │ └──────────────────┘ │   │   │                         │
│                     │   │  └──────────────────────┘   │   │                         │
└─────────────────────┘   └────────────────────────────┘   └─────────────────────────┘
```
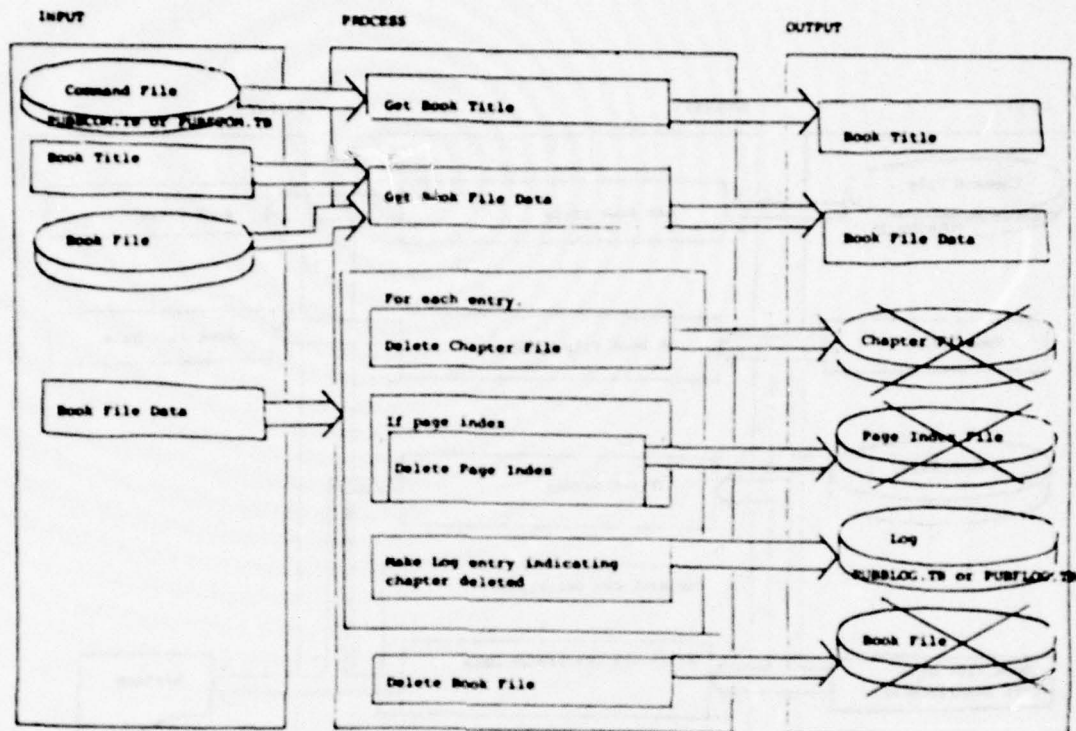
Figure 73   Book File Deletion

3-117

Figure 74 Book File List

3-118

Pagination, Justification & Global Edits

Stream Buffering and Global Editing

Depagination & Stream Generation

Diagram Allocation & Pagination

New Version Installment

Stream Validation Checking & ification
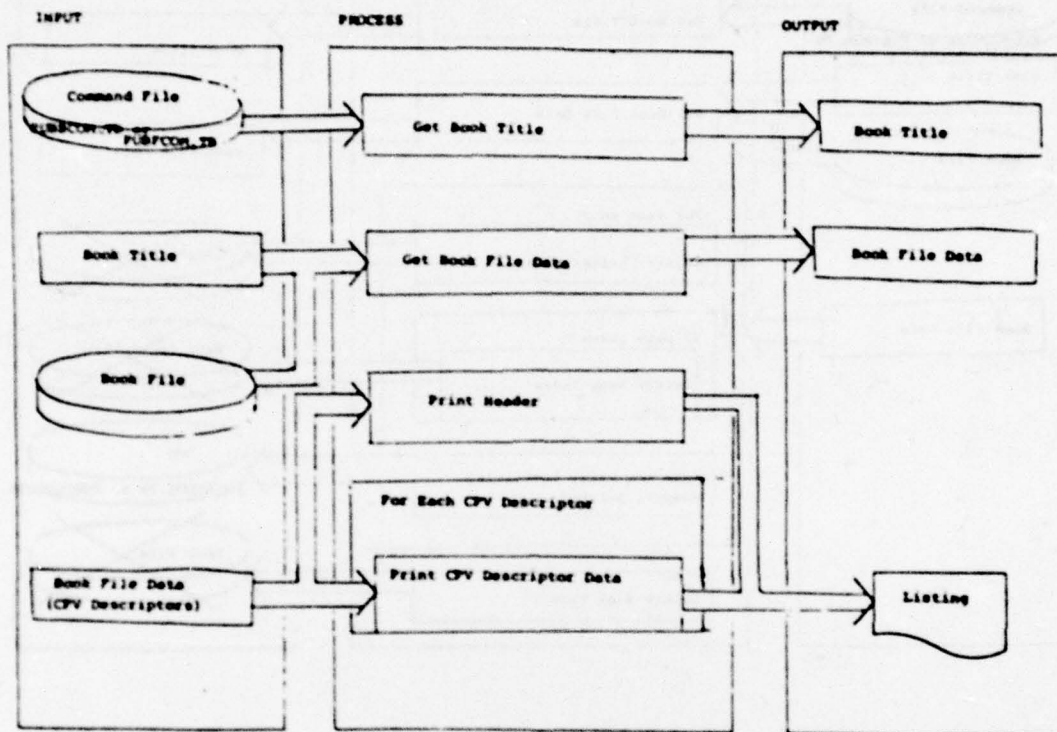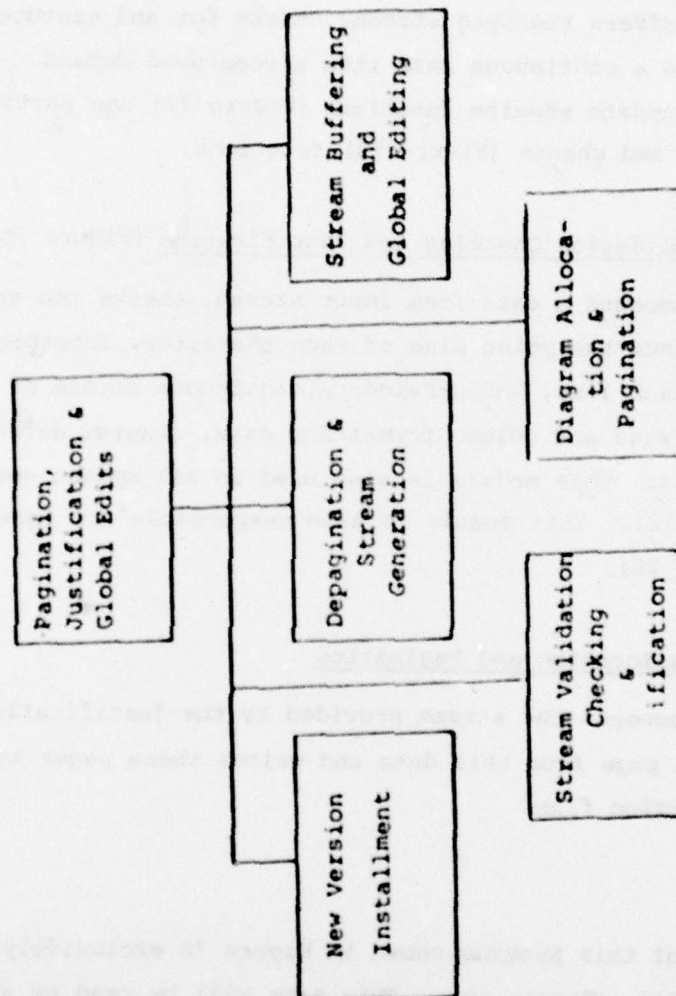
Figure 75   Pagination, Justification and Global Edits

3.4.4.2 Depagination and Stream Generation

This module reads data from the destination chapter version and provides a continuous byte stream upon demand. This module is also used in all functions of the edit session (Figure 75).

3.4.4.3 Stream Buffering and Global Revision (Figure 75)

This module buffers the byte stream, checks for and executes global edits, and provides a continuous data item stream upon demand. This module also supports all update session functions (Figure 12) and particularly the search (Figure 13) and change (Figure 14) functions.

3.4.4.4 Stream Validation Checking and Justification (Figure 75)

This module accepts a data item input stream, checks the validity of this data, determines the point size of each character, interprets other data items, justifies each line, and provides a continuous stream of uninterruptable text blocks, page and column formatting data, diagram definitions, and vector requirements. This module is also used in all update session functions (Figure 12). This module is also responsible for page index generation (Figure 28).

3.4.4.5 Diagram Allocation and Pagination

This module accepts the stream provided by the justification module (Figure 27) builds page from this data and writes these pages to the destination chapter version file.

3.4.5 EBR Output

The modules of this program shown in Figure 76 exclusively support the EBR output formation (Figure 29). This tape will be read by an electron beam recording system which will output a scaled image of each page.
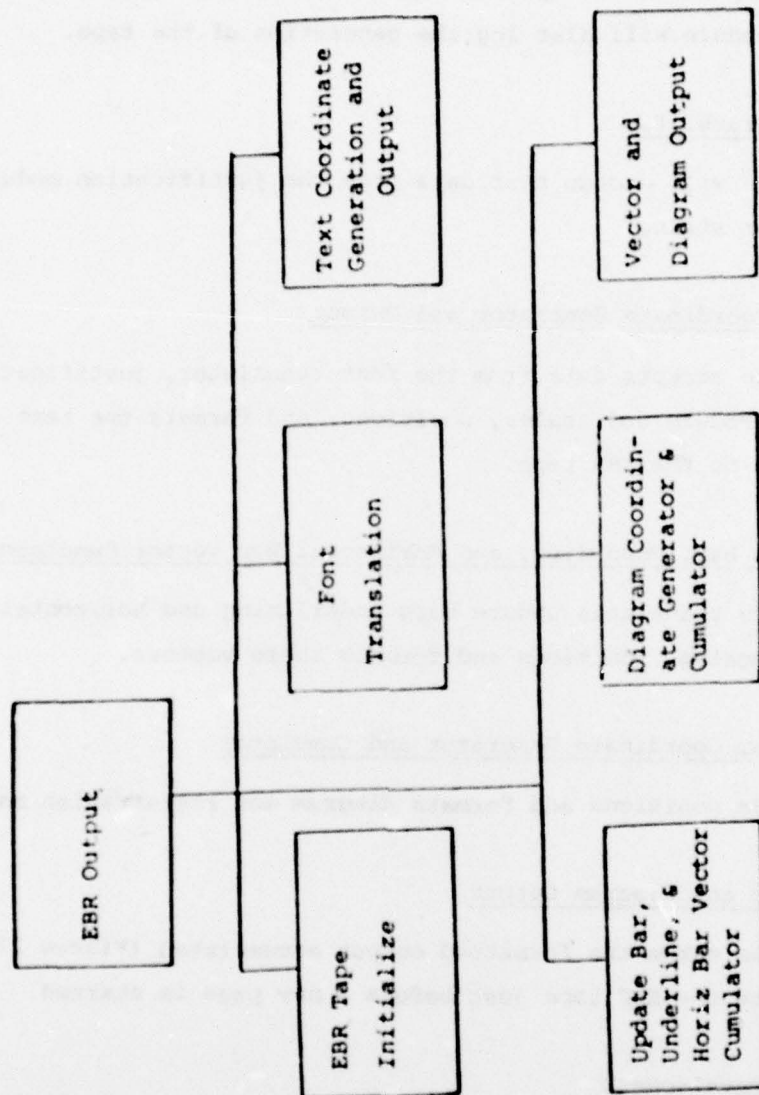
Figure 76   EBR Output

3.4.5.1 EBR Tape Initialize

This module will perform all tape maintenance required prior to the writing of data records. This includes assuring that a tape is properly mounted and ready for write operations, rewinding the tape, and writing the header. This module will also log the generation of the tape.

3.4.5.2 Font Translator

This module will encode text data from the justification module into an EBR character string.

3.4.5.3 Text Coordinate Generator and Output

This module accepts data from the font translator, justification module, and pagination module and scales, positions, and formats the text and writes it in this form to the EBR tape.

3.4.5.4 Update Bar, Underline, and Horizontal Bar Vector Cumulator

This module translates update bars underlining and horizontal bars into bectors, then scales, positions and formats these vectors.

3.4.5.5 Diagram Coordinate Generator and Cumulator

This module positions and formats diagram and registration markings.

3.4.5.6 Vector and Diagram Output

This module takes the formatted output accumulated (Figure 27) and writes it to the EBR tape just before a new page is started.

3.4.6 Editor Development

This program shown in Figure 77 will provide the capability to create, update, and proof pages of data.

Editor

Command
Interpret &
Dispatch

Small Display
Driver

Proof Listing
Generator

Log Entry
Generator

Display
Translators

Deletion
Insertion
Control

Large Display
Driver

Page Selection
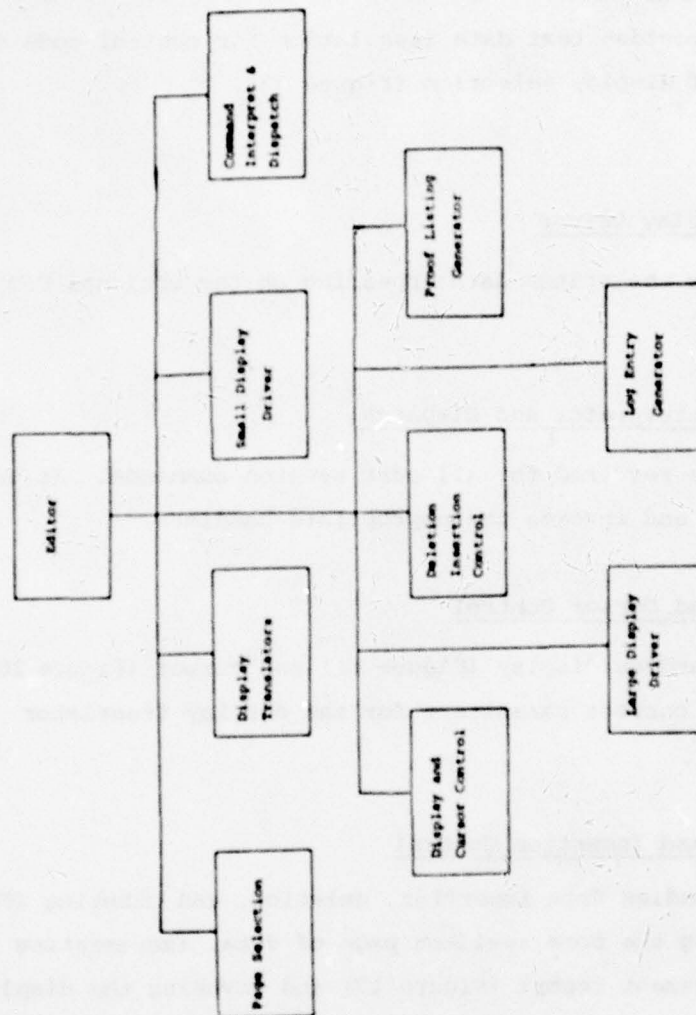
Display and
Cursor Control

Figure 77 Editor

3.4.6.1 Page Selector

This module supports selection of a page of data to be edited or proofed. This module also supports page selection via the page index (Figure 8).

3.4.6.2 Display Translators

This module provides text data translation for control code display selection and proof display selection (Figure 33).

3.4.6.3 Small Display Driver

This maintains the status data appearing on the optional CRT (Figure 44).

3.4.6.4 Command Interpreter and Dispatch

This module is required for all edit session commands. It interprets command input data and invokes the appropriate handlers.

3.4.6.5 Display and Cursor Control

This module handles display (Figure 33) and cursor (Figure 10) commands by determining the correct parameters for the display translator and display driver .

3.4.6.6 Deletion and Insertion Control

This module handles data insertion, deletion, and changing (Figure 14) commands by updating the core resident page of data, incrementing the edit count for the management report (Figure 17) and invoking the display control to reflect the change.

3.4.6.7 Proof Listing Generator

This module generates a proof listing of the page and directs it to the line printer (Figure 32).

3.4.6.8 Large Display Driver

All output to the data graphics is channeled through this module. This module maintains line numbers being displayed, the cursor position, and a line number index into the core resident page of data. This module is used by all edit session commands.

3.4.6.9 Log Entry Generator

This module writes the core resident page back to disk and makes the appropriate log entry for subsequent management listing (Figure 16).

3.4.7 HELP (On-line Documentation)

This program shown in Figure 78 will provide full operational documentation in response to the command "HELP" or the entry of a "?" to any "PUB" command.

3.4.7.1 The HELP Command

This documents all the "PUB" commands and CLI commands.

3.4.7.2 The On-line Documenter

This provides detailed description and instruction at any point in a command string in response to typing a question mark.

3.4.8 Miscellaneous

There are four miscellaneous software modules shown in Figure 79.

3.4.8.1 Book File Listing

This module supports the list book command which lists the chapter title and descriptor data for each chapter/version in a designated book and directs this listing to the line printer.
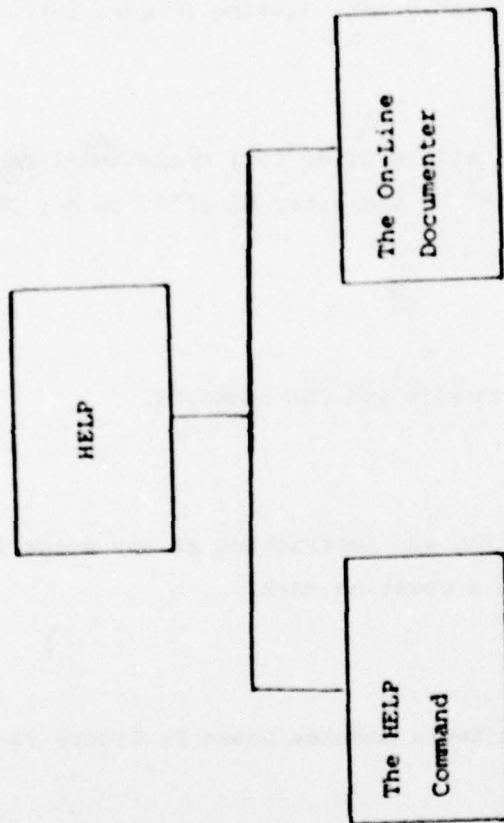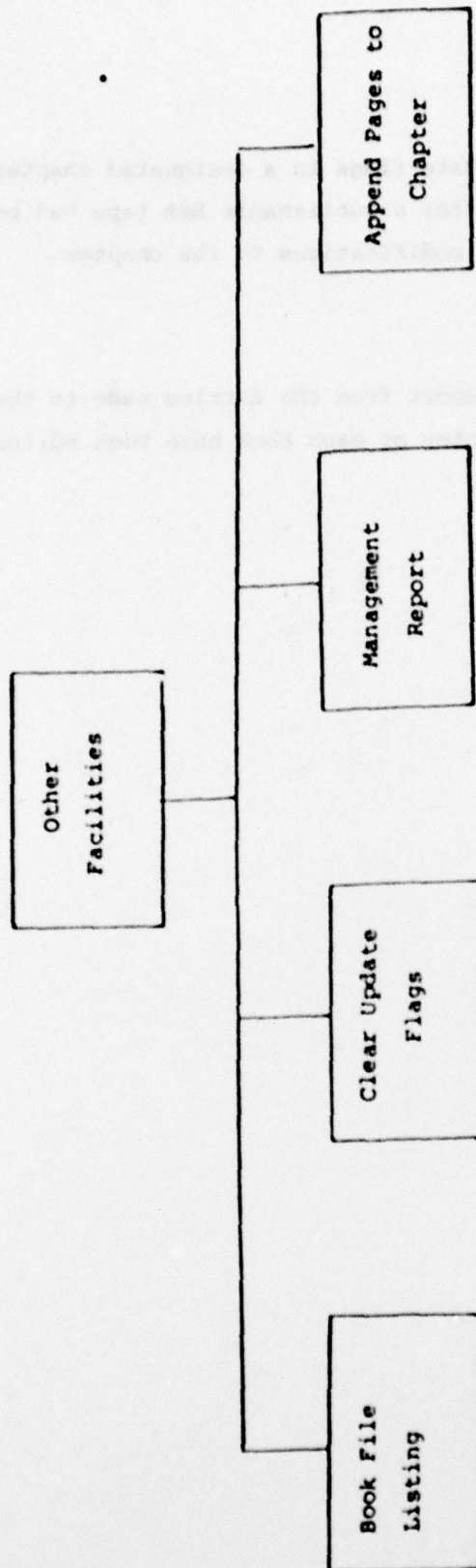
Figure 78   HELP

Figure 79   Other Facilities

3.4.8.2 Clear Update Flags

This module will clear all update flags in a designated chapter/version (Figure 80). This would be used after a publishable EBR tape had been generated and prior to any further modifications to the chapter.

3.4.8.3 Management Report

This module will generate a report from the entries made to the log indicating which pages of each chapter of each book have been edited and proofed.
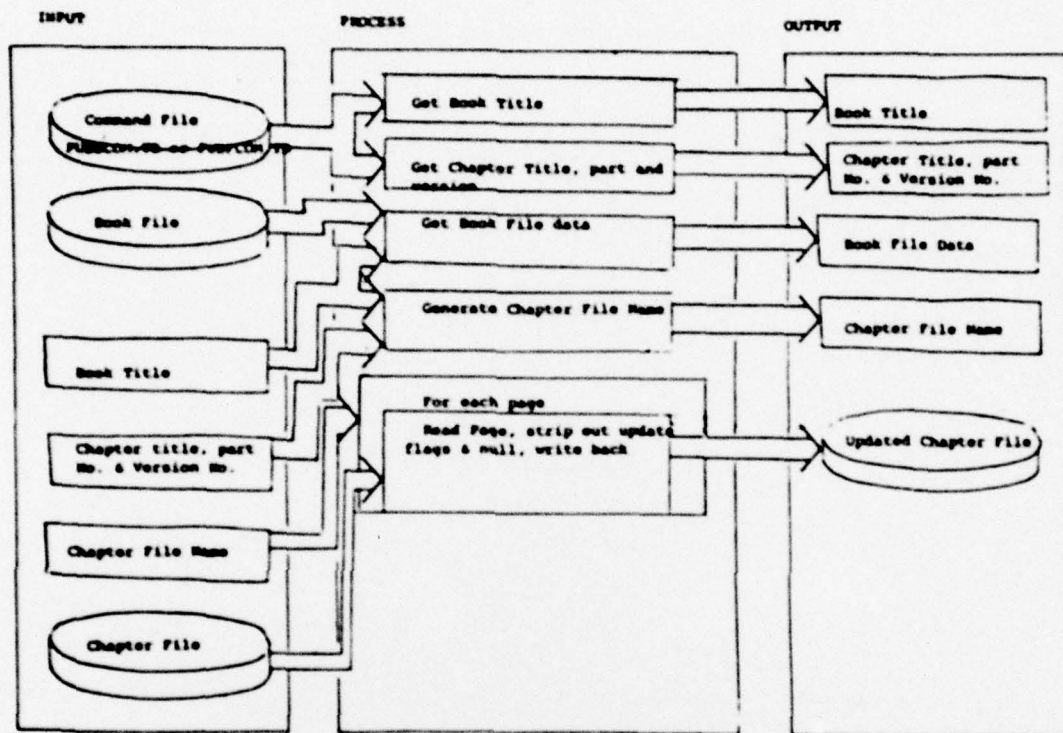
Figure 80   Clear Update Flags

SECTION IV

CONCLUSIONS AND RECOMMENDATIONS

4.0 General

The AAIPS Publishing Subsystem as delivered and accepted will produce all the publications it was designed to produce. Use of a digital data base will provide maximum flexibility to enable changes in format with a minimum of description of production. The ability to automatically repaginate and to merge graphics into the publications results in a finer quality product which requires less labor to produce.

4.1 Conclusions

The AAIPS Publishing Subsystem has met all design objectives and has proven itself to be a cost effective modern way to maintain FLIP publications.

4.2 Recommendations

The system as delivered will produce the required publications. The following software/hardware is recommended to further optimize performance throughput and human engineering in a production environment:

a. Specification, acquisition and interface to a viewer model terminal incorporating extra function keys; Datagraphix 132B.

b. Specification, acquisition and interface to a proofing printer capable of reproducing upper/lower case special characters in several font styles and sizes.

c. Improvement or provision of such software as would improve hyphenation, AGEAR entry, block deletion and right justification within a field of data.

d. Continued analysis of production requirements and possible throughput enhancements that may be found to be cost effective.

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.