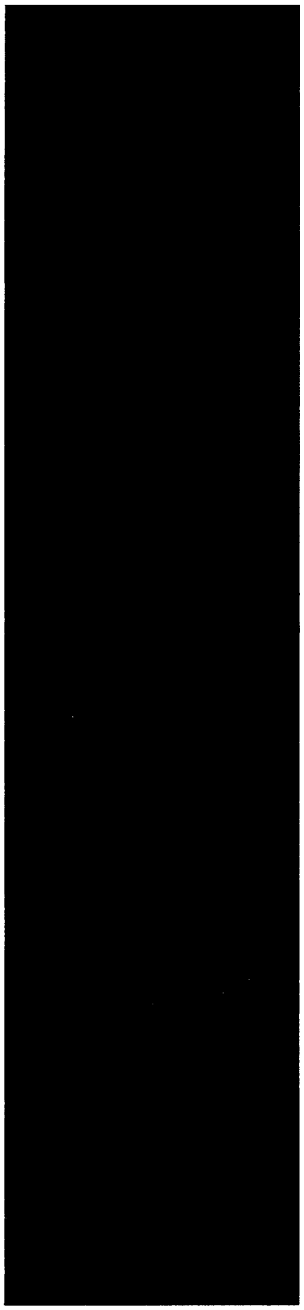
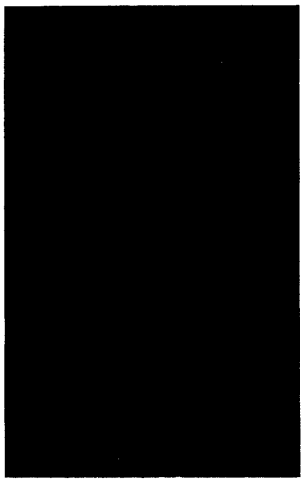


FLOATING POINT
SYSTEMS, INC.



LED100
Manual
Release C.1

860-7469-001A

Publication No. 860-7469-001
August, 1980

NOTICE

This edition applies to Release C.1 of FPS-100 software and all subsequent releases until superseded by a new edition.

The material in this manual is for informational purposes only and is subject to change without notice.

Floating Point Systems, Inc., assumes no responsibility for any errors which may appear in this publication.

Printed in USA

by FPS Technical Publications Staff

LED100
Manual
Release C.1

860-7469-001A

Publication No. 860-7469-001
August, 1980

NOTICE

This edition applies to Release C.1 of FPS-100 software and all subsequent releases until superseded by a new edition.

The material in this manual is for informational purposes only and is subject to change without notice.

Floating Point Systems, Inc., assumes no responsibility for any errors which may appear in this publication.

Printed in USA

PURPOSE

This manual documents the FPS-100 Library Editor (LED100), which is designed to function with the Floating Point Systems array processor (FPS-100). Throughout the remainder of this manual the array processor is referred to as the FPS-100.

SCOPE

This manual provides software information necessary to understand and use LED100. Information on the FPS-100 itself is not included. If more information is needed, refer to the FPS manuals listed in Table 1-1.

Table 1-1 Related Manuals

MANUAL	PUBLICATION NO.
FPS-100 Loader (LOD100) Reference Manual	FPS 860-7423-001
FPS-100 Assembler (ASM100) Reference Manual	FPS 860-7428-002

OVERVIEW

LED100 is a library editor which can manipulate FPS-100 libraries or object modules so that they can be handled more efficiently by the FPS-100 Loader (LOD100). LED100 converts libraries and object modules into a form that can be loaded from one to five times faster than a standard library. The sole function of LED100 is to streamline libraries for the loader.

The input libraries are similar to those created by ASM100 (using $LIB and $ENDLIB pseudo-ops) or the host editor.

The standard library format can still be created and used; however, it may be more efficient, especially in the case of large libraries, to use LED100.

The following is a sample run of LED100. User input is underlined.

```
 OK, LED100
 LED100 VERSION DATE
 ENTER LIBRARY OUTPUT FILE =
BASLIB.FAST
 ENTER OBJECT OR LIBRARY INPUT FILE =
BAALIB
 ENTER OBJECT OR LIBRARY INPUT FILE =
BABLIB
 ENTER OBJECT OR LIBRARY INPUT FILE =
(null filename)

 LIBRARY EDITOR TERMINATION
```

The first file that LED100 request is the newly-formatted library output file. The library editor prompts for input files until the user terminates LED100 with a null filename.

LED100 alters the original form of the library in two ways. First, it inserts a new object block before each subroutine in the library. Second, it creates the file that contains the output using fixed-length records, direct access file methods, or some similar strategy whose goal is to facilitate easy movement throughout the library without causing excessive disk I/O. This second method is transparent to the user. Note, however, that the user cannot always modify these files using the host editor since this can interfere with the file accessing method.

INDEX BLOCK

The new object block constitutes the index block. This block is found only in libraries. The following illustrates the form of the new block:

header record:

```
14   entcnt   skpcnt   ***INDEX
```

data record:

```
title   ent(1)   ent(2)   ***   ent(entcnt)
```

where:

14 = the object block ID number (14 octal)

entcnt = the number of entry symbols found in the data record. This is always less than or equal to 6.

skpcnt = the number of records to skip to reach the next subroutine in the library. If this number is 0, then another index block follows this one. Therefore, it is possible for a subroutine to contain more than one index block only if it contains more than six entries.

title = the title of the subroutine that follows

ent(1) = an entry symbol that occurs in the following subroutine

ERROR MESSAGES

Error messages with LED100 take the following form:

```
message   (number type)
```

where:

message = the error message

number = the error number (refer to Table 1-2)

type = F (fatal error)
I (input ignored)
W (warning only)

Table 1-2 contains descriptions of error messages generated by LED100.

Table 1-2 Error Messages

MESSAGE NUMBER	DESCRIPTION
01 F	READ ERROR. A file read error has occurred. This usually is the result of a host operating system error.
02 F	BAD RECORD. The input library contains a bad record. That is, the object blocks being processed do not conform to the correct structure and form. For example, this may mean there was an error in the ASM100 assembly of this library.
03 I	BAD FILE NAME. The input file name supplied to LED100 does not exist or is inaccessible. The user input was ignored by LED100.
04 F	UNEXPECTED EOF. The input file does conform to the proper structure (refer to error 02).
05 F	UNABLE TO REWIND FILE. An error occurs while rewinding the temporary file.
06 F	ERROR WHILE CLOSING FILE. The file cannot be closed.
07 F	ERROR WHILE OPENING FILE. The file cannot be opened. The user may have misspelled the filename.

READERS COMMENT FORM

Your comments will help us improve the quality and usefulness of our publications. To mail: fold the form in three parts so that Floating Point Systems mailing address is visible, then seal.

Title of document _____

Name/Title _____ Date _____

Firm _____ Department _____

Address _____

Telephone _____

I used this manual...

- ☐ as an introduction to the subject
- ☐ as an aid for advanced training
- ☐ to instruct a class
- ☐ to learn operating procedures
- ☐ as a reference manual
- ☐ other _____

I found this material...

	Yes	No
accurate/complete	☐	☐
written clearly	☐	☐
well illustrated	☐	☐
well indexed	☐	☐

Please indicate below, listing the pages, any errors you found in the manual. Also indicate if you would have liked more information on a certain subject.

First Class
Permit No.A-737
Portland,
Oregon

BUSINESS REPLY

No postage stamp necessary if mailed in the United States

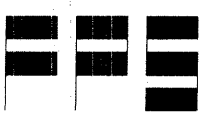
Postage will be paid by:

FLOATING POINT SYSTEMS, INC.

P.O. Box 23489

Portland, Oregon 97223

Attn: Technical Publications



FLOATING POINT
SYSTEMS, INC.

CALL TOLL FREE 800-547-1445
P.O. Box 23489, Portland, OR 97223
(503) 641-3151, TLX: 360470 FLOATPOINT PTL