Publication Number S29-630R02

SOURCE UPDATER

USER GUIDE

PERKIN-ELMER

Computer Systems Division
2 Crescent Place
Oceanport, N.J. 07757

PAGE REVISION STATUS SHEET

PUBLICATION NUMBER S29-630

TITLE Source Updater User Guide

REVISION R02 DATE May 1979

PAGE	REV.	DATE	PAGE	REV.	DATE	PAGE	REV.	DATE
i/ii	R02	5/79						
iii	R02	5/79						
iv	R02	5/79						
1-1 thru 1-2	R02	5/79						
2-1 thru 2-21/2-22	R02	5/79						
Index -1	R02	5/79						
Index -2	R02	5/79						

PREFACE

This manual describes the Perkin-Elmer utility, Source Updater, that provides the operator with the capability to maintain, copy, and change source files on bulk storage devices. Configuration, logical unit assignments, and a complete description of each command are given.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER 1
INTRODUCTION TO SOURCE UPDATER

1.1 GENERAL

The Source Updater, a common mode utility program, provides the operator with the ability to create and maintain source files on bulk storage devices. These commands enable the operator to copy, verify, modify, or list source files. To position and access files, commands for positioning to a desirec name file starting with ** are provided. Although the Source Updater primarily processes source on magnetic media, any media can be processed by assigning the desired device.

1.2 CONFIGURATION

The Source Updater requires this configuration to execute:

- OS/16 MT2 or OS/32

- command input and list devices

1.3 LOGICAL UNIT ASSIGNMENTS

The Source Updater program uses these logical units (LUs) assignments:

AS 1 input source file (old master)
AS 2 output source file (new master)
AS 3 list device
AS 4 update source
AS 5 command input device
AS 6 error log device

LUs 3, 5, and 6 must be assigned by operator commands. LUs 1, 2, and 4 may be assigned by the operator or Source Updater commands.

Most error output goes to LU 3. If the commmand device is interactive and is not assigned to the same device as LU 3, error output also goes to LU 5.

If assigned, error output can also go to LU 6. If it is not assigned, error output defaults to LU 3 and LU 5.

1.4 DEFAULT DATA BLOCK SIZE (32-BIT ONLY)

The Source Updater allocates its output files with a default data block size of 5. If this block size is greater than the maximum established at SYSGEN time, the program repeatedly allocates a smaller block size until it is equal to the maximum established at SYSGEN time.

CHAPTER 2
SOURCE UPDATER COMMANDS

2.1 GENERAL

With the exception of the UPDATE command, all commands are input from LU 5. The UPDATE command causes temporary transfer of command input to a user-specified file descriptor (fd) that is assigned to LU 4.

Any invalid commands are rejected with a message; and, if not in interactive mode, the task is paused. If continued, a new command is requested. An invalid parameter also causes the command to be rejected and a new command to be requested.

Most commands require at least an input and output source file, denoted by "to" and "from" in the command format. Some commands use a label for positioning purposes, denoted by "label" in the command format. Label is from one to eight alphanumeric characters. The actual string that is searched for on the file is:

```
**label
```

Some commands require sequence numbers, denoted by "seqnbr", consisting of eight characters that are divided into an alphabetic portion and decimal portion. The alphabetic portion can be from 0 to 7 characters ending in an alphabetic character and the decimal portion can be from 1 to 8 characters. Any of these sequence numbers are valid:

```
ABC12345
A0000000
ABCDE000
00000000
D4E00000
```

The RESEQUENCE command increments only the decimal part of the sequence number. The alphabetic characters remain fixed.

Brackets, [], are used to enclose optional arguments of Source Updater commands.

Source Updater commands, which require file descriptors as arguments, can specify the currently assigned fd by specifying * cr by omitting the fc specification. If fd is specified, no action is taken if it is currently assigned; if fd is not specified, the LU is closed and the specified fd is assigned.

2.2 FIND COMMAND

The FIND command searches the source file for the specified label and positions the file immediately following the record containing the label.

Format:

```
FIND  label [,from]
                *
```

Parameters:

label — is a user-specified alphanumeric character string located in a record on the source file.

from — is a user-specified file descriptor indicating the input source file.

* — is the current input source file assigned to LU 1.

Programming Considerations:

When the label is found, the next command is requested. If the label is not found, a message is output to the error logging device, and no record positioning occurs before the search for the specified label starts.

If from is omitted or * is specified, the current input source (LU 1) is assumed.

2.3 COPY COMMAND

The COPY command is used to COPY source files consisting of 80-byte records to an output file.

Format:

```
COPY [from] [,to] [,ending string [,column number]]
```

Parameters:

from — is the input source file.

to — is the output source file.

If both from and to are omitted or * is specified, the current file is selected.

ending string — if specified, the copy operation terminates upon copying the record that contains this ending string. When an ending string is specified, but not found in the text, a message to that effect is printeo.

column number — if specified, scanning the ending string starts at this column position in each input record, skipping leading blanks. If this argument is omitted, it defaults to 1.

Programming Considerations:

A // encountered in columns one and two on the input device is treated as an end-of-file marker.

The COPY operaticn terminates when end-of-file/end-of-medium (EOF/EOM) is reached or when a record containing the user-specified ending string is encountered.

2.4 LIST COMMAND

The LIST command provides a listing of a source file on the specified list device. Each page is numbered and contains up to 58 lines.

Format:

```
LIST [from] [,label]
```

Parameters:

from — specifies the input source file. If from is omitted, or if * is specified, the current input (LU 1) is used.

label — specifies the label that is to be found prior to beginning the list operation. If label is omitted, no positioning is performed.

Programming Considerations:

The list operation is terminated when EOF/EOM, or a statement containing // in columns one and two, is detected.

2.5 TABLE COMMAND

The TABLE command scans the specified input file anc lists all program labels on the list device. Thus, the source file's table of contents, which contains many labeled programs, can be obtained. Program labels are indicated by two asterisks in the first two character positions of a record.

Format:

```
TABLE  [from]
           *
```

Parameters:

from — specifies the input source file. If from is omitted or if * is specified, the current input source file (LU 1) is usec.

* — indicates program labels.

2.6 UPDATE COMMAND

The UPDATE command puts the system in the update mode and sets up various parameters for the following update commands:

- REPLACE
- INSERT
- DELETE
- MODIFY
- SELECT
- ENDUP

These commands are valid only when in UPDATE mode.

Format:

```
UPDATE  [update] [,from] [,to] [,ending string
        [,column number]]
```

Parameters:

update — indicates the file from which the update commands are to be read (LU 4).

from — indicates the old master; that is, the file to which the changes are to be made.

to — indicates the new master or file that is to be created.

ending string — if specified, the update operaticn terminates upon dating the record that contains this ending string. When an ending string is specified, but not found in the text, a message to that effect if printed.

column number — if specified, scanning the ending string starts at this column position in each input record, skipping leading blanks.

Programming Considerations:

If any of the first three parameters are omitted or if * is specified, the current file for that parameter is assumed.

If the fd specified in the third parameter does not exist, an indexed file by that name is allocated with a logical record length of 80 bytes.

A form feed is issued on the list device at the start of an update; anc if available, the time and date are printed at the top of the UPDATE listing. The UPDATE operation processes all commands from LU 4, updating the old master to the new master until an ENDUP command is encountered on LU 4 or until an EOF/EOM or ending string is encountered while being written to the new master (LU 2), whichever comes first. All updates should be made in order of ascending sequence numbers.

2.7 INSERT COMMAND

The INSERT command is used to insert one or more new statements into an existing source program. The records following the INSERT command are inserted in the new master until a record with a /* in columns one and two is read from the update logical unit.

Format:

```
    INSERT seqnbr
```

Parameters:

seqnbr — is the sequence number after which one or more new statements are to be inserted. The new statements are read from LU 4.

Programming Considerations:

To insert two statements after XXX00150, the following statements must be entered on the Update Logical Unit (LU 4):

```
    INSERT XXX000150
    - statement #1 -
    - statement #2 -
    /*
```

Error Messages:

If a sequence number is detected on the old master that is larger than seqnbr or an EOF/EOM or the ending string is detected, this error message:

```
    SEQ NBR NOT FOUND
```

is printed on the list device and the command input device, if that device is interactive. The operation is then atorted.

2.8 DELETE COMMAND

The DELETE command provides the capability to delete single statements or multiple statements from the input source file---old master (LU 1).

Format:

```
DELETE seqnbr1 [,seqnbr2]
```

Parameters:

seqnbr1 specifies the sequence number of the first statement to be deleted.

seqnbr2 if specified, statements between seqnbr1 and seqnbr2 inclusive are deleted. If seqnbr2 is omitted, only the statement specified by seqnbr1 is deleted.

Programming Considerations:

To delete statements AAA01010 through AAA01500, this command is entered:

```
DELETE AAA01010,AAA01500
```

Error Messages:

If a sequence number is detected on the old master that is larger than seqnbr1 or seqnbr2, the error message:

```
SEQ NBR NOT FOUND
```

for seqnbr1 is printed on the list device.

The error message:

```
SECOND SEQ NBR NOT FOUND
```

is printed on the list device for seqnbr2. The operation is aborted when a sequence number is not found.

2.9 MODIFY COMMAND

The MODIFY command is used to delete a single statement and insert a new statement in its place.

Format:

```
    MODIFY seqnbr
```

Parameters:

seqnbr — indicates the sequence number of the statement to be modified.

Programming Considerations:

To use the MODIFY command to delete statement AAA01000 and replace it with a new statement, this sequence must be entered on the Update Logical Unit (LU 4):

```
    MODIFY AAA01000
    - new statement -
```

Error Messages:

If the sequence number cannot be found, this error message:

```
    SEQ NBR NOT FOUND
```

is printed on the list device; the command input device, if that device is an interactive device; and the error device, if it was assigned.

2.10 REPLACE COMMAND

The REPLACE command is used to modify a specified statement and to insert additional statements.

Format:

```
REPLACE seqnbr1 [,seqnbr2]
```

Parameters:

seqnbr 1 indicates the sequence number of the statement where the insert procedure is to begin.

seqnbr2 indicates, if specified, the second limit of a range of lines to be replaced by those entered in the insert lines. If seqnbr2 is omitted, only the line designated by seqnbr1 is deleted.

Programming Considerations:

If the sequence number cannot be found, an error message is generated.

REPLACE is terminated by a statement with a /* in columns one and two.

2.11 SELECT COMMAND

The SELECT command is used to modify or to insert scurce lines at selected locations specified by sequence numbers.

Format:

```
SELECT
Line 1   seqnbr1
Line 2   seqnbr2
Line 3   seqnbr3
:
:
/*
```

Parameters:

seqnbr — is a sequence number consisting of eight characters.

line — is a source line.

Programming Considerations:

The sequence numbers (seqnbr1, seqnbr2, etc.) start at column 73 in an 80-byte record.

When a sequence number is found in the old master that matches the sequence number specified, a modify is then in effect. If a sequence number is found that is larger than the one searched for, one line is inserted before the larger sequence number. When sequence numbers in the SELECT input file are out of order, a message is printed; but, processing is continued. SELECT is terminated with a record with /* in columns one and two.

2.12 ENDUP COMMAND

The ENDUP command terminates the UPDATE mode.

Format:

```
    ENDUP
```

Programming Considerations:

If the ending string was reached before the ENDUP, copying is terminated. If the ending string has not yet been encountered, the old master is copied onto the new master until it reaches the ending string or EOF/EOM.

2.13 RESEQUENCE COMMAND

The RESEQUENCE command is used to resequence a specified file. It increments only the decimal part of the sequence number.

Format:

```
RESEQUENCE from,start seqnbr[,incr][,to][,ending string
           [column number]]
```

Parameters:

from — is the file to be resequenced.

start — specifies the starting sequence number.

incr — is the sequence number increment; if omitted, decimal 10 is assumed.

to — specifies the output resequenced file.

ending string and column number — specifies an ending string anc column number for the start of the ending string.

Programming Considerations:

If to is omitted, resequencing is done in place; that is, resequencing is to the from file. If the to file is specified but does not exist, an indexed file is created with a logical record length of 80 bytes and a block size of 5 sectors. (See Section 1.4) If the from file is a magnetic tape medium, then resequence in place cannot be done. Resequencing .continues until a // in columns one and two, and EOF/EOM, or the ending string is encountered on the input file.

Specifying any ending string and column number for the start of the ending string allows the user to resequence cifferent parts of a file with different sequence numbers. However, when a resequence in place is done, the file is logically rewound at the start of each resequence command.

2.14 VERIFY COMMAND

The VERIFY command is used to compare one source file with another.

Format:

```
VERIFY [from][,to][,n][,ending string [column number]]
```

Parameters:

from and to — indicate the two source files to be verified. If the first or second parameter is omitted or * is specified, the current file assignment is assumed.

n — specifies a decimal number between 1 and 80 and indicates the number of bytes to be compared. If n is omitted, all 80 bytes are compared.

ending string — is optional and allows two files to be verified to an ending string.

column number — defines the column number for the start of the ending string; the default is column one. A // in columns one and two is treated as an end-of-file.

Programming Considerations:

Verification is terminated normally when an end-of-file is detected. An abnormal termination is detected when a mismatch is detected. When a mismatch is detected, this message:

```
**VERIFICATION ERROR** - Record 1 - - Record 2 -
```

is printed on the list device (LU 3); the command input device (LU 5), if that device is an interactive device; and the error device, if assigned.

Record 1 is the source statement read from LU 1 and Record 2 is the statement read from LU 2.

2.15 EXCEPTION COMMAND

The EXCEPTION command allows the user to generate a listing that contains all source statements differing between two source files.

Format:

```
    EXCEPTION [from][,to][,n][,ending string [,column number]]
```

Parameters:

from anc to — indicate the two source files to be compared. If the first cr second parameter is omitted or * is specified, the current file assignment is assumed.

n — specifies a decimal number between 1 and 80 and indicates the number of bytes to be compared.

ending string — is optional and allows the two files to be compared to an ending string.

column number — defines the column number for the start of the ending string; the default is column one. A // in columns one and two is treated as an end-of-file.

Error Messages:

If a mismatch occurs, the message:

```
    *** EXCEPTION ERROR CN LUn **
```

is printed. Following this message, a list of the nonmatching records of the device, with the lower sequence numbers of the two, is output until the files are realizec or until an end-of-file occurs.

If the sequence numbers are identical, but the data in the remaining records does not compare, no attempt to align is made. The message:

```
*** EXCEPTION MISMATCH ***
```

is displayed and both offending lines are printed.

If an end-of-file occurs on one device and not on the other, the message:

```
*** EOF ON UNIT n
```

is displayed followed by a list of the remaining records.

2.16 BUILD COMMAND

The BUILD command puts the Source Updater into a moce that only allows INCLUDE commands to be processed. This command specifies the file to be used as the output build file (LU 2) in subsequent INCLUDE commands.

Format:

```
    BUILD fd
```

Parameters:

fd — if it does not exist, it is allocated on the system default volume as an indexed file with a logical record length of 80 bytes.

Programming Considerations:

A BUILD is terminatec by an ENDB statement.

2.17 INCLUDE COMMAND

The INCLUDE command causes source statements to be copied from a file specified in the operand of this command to a file specified by a BUILD. INCLUDE is terminated when a statement beginning with // or EOF/EOM is encountered.

Format:

```
    INCLUDE [fd][,label]
```

Parameters:

fd — if omitted, the current input file is assumed. The statement that terminated the INCLUDE is not copied to the BUILD file.

label — if specified, the include file is positioned to the statement after the label before copying.

2.18 ENDB COMMAND

The ENDB command writes a record containing an ENDUP statement to the build file, writes a filemark, terminates the scope of a BUILD, and causes the build file to be rewound.

Format:

```
    ENDB
```

REWIND
PAUSE
END

2.19 REWIND COMMAND

The REWIND command rewinds a specified file.

Format:

```
    REWIND fd
```

2.20 PAUSE COMMAND

The PAUSE command causes the program to relinquish control to the operating system via a pause SVC (SVC 2 Code 1).

Format:

```
    PAUSE
```

2.21 END COMMAND

The END command causes the Source Updater program to go to end-of-task via an SVC 3.

Format:

```
    END
```

INDEX

PUBLICATION COMMENT FORM

We try to make our publications easy to understand and free of errors. Our users are an integral source of information for improving future revisions. Please use this postage paid form to send us comments, corrections, suggestions, ect.

1. Publication number______________________________________

2. Title of publication______________________________________

3. Describe, providing page numbers, any technical errors you found. Attach additional sheet if neccessary.

   ____

   ____

   ____

4. Was the publication easy to understand? If not, why?

   ____

5. Were illustrations adequate? ____________________________

   ____

   ____

6. What additions or deletions would you suggest? ________________

   ____

   ____

7. Other comments: ______________________________________

   ____

   ____

From ____________________________ Date ____________________

Position/Title ________________________

Company ___________________________

Address ____________________________

____

____

STAPLE STAPLE

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 22 OCEANPORT, N.J.

POSTAGE WILL BE PAID BY ADDRESSEE

PERKIN-ELMER

Data Systems Group
106 Apple Street
Tinton Falls, NJ 07724

ATTN:
TECHNICAL SYSTEMS PUBLICATIONS DEPT.

FOLD FOLD

STAPLE STAPLE