Published: 10/30/68

Identification

The epl command
N. Adleman

Purpose

The epl command invokes the epl compiler and the eplbsa command (MSPM BX.7.03) to translate a segment containing epl source code into corresponding segments containing standard text, link and symbol information. The segments created by the epl/eplbsa translation will appear in the user's current working directory.

Usage

When a user types a command line of the following form:

```
epl beta                                                    \
```

the epl command is invoked by the shell (MSPM BX.2). The symbol "beta" may be either a path name or an entry name and specifies that the segment <beta.epl> contains the source program to be translated. (If beta is an entry name, it is assumed to be in the current working directory.) The command line may also contain interjected option commands. Currently, the epl command observes only the following options:

option name	setting	meaning
brief	on	abbreviated comments will be produced by this command.
	off	(default setting) complete comments will be produced by this command.
listing	on	segment <beta.epl_list> will be produced by this command.
	off	(default setting) segment <beta.epl_list> will not be produced by this command.

For further details on options, see MSPM BX.12.01.

Method

1. Using the symbol "beta" as the input argument, the epl command invokes the entryarg procedure (MSPM BY.2.04) to determine the path name and entry name for <beta.epl> which contains the epl symbolic source program to be translated. A pointer to <beta.epl> is then established by calling the smm$initiate procedure (MSPM BD.3.02).

If a pointer to the segment <beta.epl> cannot be created, error type 1 is immediately reported. (See the discussion on errors below.)

2. The settings of the brief and listing options are then retrieved by calls to the read_opt procedure (MSPM BY.9.10). These settings are then examined and pertinent internal indicators are established within the epl command.

3. The epl command then invokes the working_segs$init procedure (MSPM BY.2.11) to create segments in the process directory as follows:

suffix name	maximum size	access setting	copy-relate setting	vacant setting	replace setting
.macros_1	65,536	"RWA"	"00"b	"0"b	"1"b
.macros_7	65,536	"RWA"	"00"b	"0"b	"1"b
.macros_2	65,536	"RWA"	"00"b	"0"b	"1"b
.eplbsa	65,536	"RWA"	"00"b	"0"b	"1"b
.epl_list	65,536	"RWA"	"00"b	"0"b	"1"b

(See MSPM BY.2.11 for a complete discussion of the above settings.)

If any one of the above segments can not be created in the process directory, error type 2 is reported by the epl command.

4. The compiler is then invoked by the following epl statement:

```
call epl_driver;
```

to translate the contents of segment <beta.epl> into the necessary macros and eventually produce a segment <beta.eplbsa> which contains the eplbsa symbolic equivalent of the epl symbolic code found in segment <beta.epl>.

5. The working_segs$finish procedure is then invoked to move all newly created segments from the process directory into the working directory. If any of the new segments can not be moved into the desired directory, the epl command reports error type 3 (see below).

6. The eplbsa command is then invoked to create segments containing text, link, and symbol information for the new segment <beta.eplbsa>.

Upon return from the eplbsa command, the epl command returns to the user.

Errors

To report any abnormalities that may be encountered a message is immediately written by calling the write_out procedure (MSPM BY.4.02). The error types listed below are not included as part of the message but merely serve as a convenient number for documentation purposes. The possible error messages are:

Type	Meaning
1.	unable to obtain a pointer to the following segment: <"segment_name_filled_in">
2.	unable to create the following segment in the process directory: <"segment_name_filled_in">
3.	unable to move the following segment into the desired target directory: <"segment_name_filled_in">

The following comments may also be produced by the epl command:

Reason	Message
(normal event:)	compilation successful, begin assembly.
(due to a previous error:)	assembly deleted.
(no end card:)	end card probably missing, compilation aborted.