Published: 01/17/68

Identification

The eplbsa command
N. Adleman

Purpose

This section describes the eplbsa command for Initial Multics. The user invokes this command to produce standard text, link, and symbol segments from a segment containing an eplbsa symbolic source program. The eplbsa command may also be invoked by the epl command (MSPM BX.14) following an epl compilation.

Usage

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

```
eplbsa alpha [option_command(s)]
```

the eplbsa command is invoked by the shell (MSPM BX.2). The symbol "alpha" may be either a path name or an entry name and specifies that segment <alpha.eplbsa> is the source program. If alpha is any entry name it is assumed to be in the current working directory. If alpha is a relative or absolute path name it is assumed to be relative to the current working directory or root directory respectively (see BX.0.01 for further information). The remaining symbols are interjected option commands. Currently the eplbsa 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.
list	on	-	segment <alpha.list> will be produced by this command.
	off	-	(default setting) segment <alpha.list> will not be produced by this command.

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

Method

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

   If a pointer to the segment <alpha.eplbsa> cannot be created, error type 1 is inserted in the user's error file. (See the discussion on errors below.)

2. The settings of the brief and list options are then retrieved by calls to the read_opt procedure. These settings are then examined and pertinent internal indicators are established within the eplbsa command.

3. The eplbsa command then invokes the unique_chars procedure (MSPM BY.15.01) to construct the following names:

   unique_name

   unique_name.link

   unique_name.symbol

   unique_name.error

   unique_name.list

4. Successive calls to the smm$set_name_status procedure (MSPM BD.3.02) are then executed to create the following empty data segments in the process directory:

<unique_name>	to contain the equivalent binary text for the source program in <alpha.eplbsa>
<unique_name.link>	to contain the linkage information for alpha.
<unique_name.symbol>	to contain the symbol table for alpha.
<unique_name.list>	to contain the equivalent octal and symbolic listing for the source program in <alpha.eplbsa>.

If any of the above segments cannot be created, error type 2 is reported by the eplbsa command.

5. The assembler is then invoked by the following epl statement:

```
call eplbsa_1;
```

Note that eplbsa_1 has been previously created by the gecos_seg command (MSPM BX.17.01).

6. Upon return from the eplbsa_1 procedure, the eplbsa command calls the delete_entry procedure (MSPM BY.2.01) to delete unique_name.list if the list option is off.

7. The entry_status$type procedure (MSPM BY.2.10) is invoked to determine the status of the following segments:

<alpha>

<alpha.link>

<alpha.symbol>

<alpha.error>

<alpha.list> (if the list option is on)

Note that the above five segments may have been available prior to the current request for the assembly of the source program in <alpha.eplbsa>.

If any of these "old" segments exists as a link or as a directory branch in the file system, error type 3 is reported by the eplbsa command.

If the "old" segment didn't exist at all, the change_name procedure is invoked to change the name of the uniquely named segment just created by the assembler. For example, if the assembler just created <unique_name.error>, the change_name procedure will rename it to <alpha.error> to be returned to the user.

If the "old" segment exists as a branch, the following steps are executed to return the desired segment (i.e., the new one) to the user:

(a) The truncate_seg procedure (MSPM BY.2.01) is invoked to discard the contents of the "old" segment.

(b) The move_file_procedure is invoked to move the contents of the "new" (e.g., unique_name) segment to the empty "old" segment.

(c) The delete_entry procedure (MSPM BY.2.01) is invoked to delete the uniquely named segment.

8. The contents of <alpha.error> is then typed by a call to the write_out procedure (MSPM BY.4.01).

If for some reason, the contents of <alpha.error> cannot be typed, error type 5 is reported.

Errors

The eplbsa command uses the standard error handling mechanism (MSPM BY.11) to report any abnormalities that may be encountered. The condition "eplbsa_err" is signalled for the following errors:

Error No.	Meaning
1	error in attempt to obtain a pointer to the following segment: <"segment_name_filled_in">
2	error in trying to create the following segment: <"segment_name_filled_in">
3	error in trying to delete the following segment: <"segment_name_filled_in">
4	error in trying to move or rename the following segment: <"segment_name_filled_in">
5	error in trying to type the contents of the following segment: <"segment_name_filled_in">