Published: 10/13/66
(Supersedes: BE.8.04,1/14/66;
BE.8.04, 6/30/66)

Identification

Linkage building for ordinary slave procedures
in the pseudo-supervisor
D.H. Johnson

Purpose

Procedures are described which generate and examine linkage section information during the execution of a process. Using these procedures it is possible to construct links, entries, external symbol definitions, and link definitions (see MSPM Section BD.7.01 for details about linkage sections).

Linkage Section Examination

Given a specified external symbol and segment name, the following procedure determines whether the symbol is defined in the linkage section of the segment.

```
     pseudo_supervisor$sympr_ (name, symbol, answer, class,
                               error, code)
```

Where the arguments are:

identifier	attribute	meaning
1. name	character string(*)	segment name
2. symbol	character string(*)	external symbol
3. answer	bit string (1)	result of examination
4. class	fixed	class of symbol, if present
5. error	label	error return
6. code	fixed	error identifyinf code

If arg2 is defined in the linkage section for segment arg1, then arg3 is set equal to 1 and arg4 is set to equal the class of arg2. Otherwise, arg3 is set equal to 0 and arg4 is undefined.

Error code identification:

1. Segment arg1 not active in process.

2. Segment arg1 does not have linkage section.

Linkage Section Generation

There are four procedures defined below for generating linkage section information.

1.	linkmk	makes a normal link and link definition
2.	trapmk	makes a linkage for call before link type
3.	defmak	makes an external symbol definition and any necessary entries and links
4.	symmk_	adds n cells to a segment and makes an external symbol definitions in the linkage section

The normal link maker is called as follows:

```
     pseudo_supervisor$linkmk (seg, extype, base, name, exp,
                                  mod, point, error, code)
```

Where the arguments are

identifier		attribute	meaning
1.	seg	character string(*)	segment in whose linkage section to work
2.	extype	fixed	type of external reference
3.	base	fixed	base address register
4.	name	character string(*)	segment name
5.	symbol	character string(*)	external symbol
6.	exp	fixed	expression
7.	mod	fixed	address modifier
8.	point	pointer	pointer to link
9.	error	label	error return
10.	code	fixed	error identifying code

Arg1 defines the segment in whose linkage section to make the linkage information. Arg2 is the type of external reference (2, 3, or 4). Args3-4-5 have meaning based on the value of arg2. Aargs6-7 allow for full address flexibility and must be 0 if not required.

The procedure will search the linkage section of segment arg1 attempting to construct only that linkage information that is not already there. A pointer to the link is placed in arg8.

Error codes and their meanings:

1. Segment arg1 not active in process.
2. Error in attempting to create a linkage section for segment arg1.
3. External reference type not defined.
4. Error in attempting to grow linkage section to insert new linkage information.

A trap before linking type construction may be accomplished by calling

```
    pseudo_supervisor$trapmk (seg, extype, base, name,
                              symbol, exp, mod, point,
                              calnam, calsym, argtyp, argnam,
                              argsym, error, code)
```

Where the arguments are:

	<u>identifier</u>	<u>attribute</u>	<u>meaning</u>
	1-8 same as 1-8 for procedure linkmk		
9.	calnam	character string(*)	segment name of call before link
10.	calsym	character string(*)	ext. symbol name of call before link
11.	argtyp	fixed	type of link for argument of call before link
12.	argnam	character string (*)	segment name of argument of call
13.	argsym	character string (*)	external symbol of argument of call
14.	error	label	error return
15.	code	fixed	error identifying code

The first eight arguments are handled exactly as for procedure linkmk. Arguments 9-10 defined as type 4 external reference to be used as the entry point in the call before linking. Arguments 11-13 define the argument to be passed to the trapping procedure. It may be an external reference of type 3 or 4.

This procedure will search the linkage section of segment arg1 attempting to construct only the linkage information that is not already there. A pointer to the link for arguments 1-7 is placed in arg8.

Error codes and their meanings:

1. Error in constructing linkage information.
2. Link pair came back set. The reference has been made and used before.
3. Linkage section did not have space to insert trap word.

An external symbol definition and associated entries, links, and link definitions may be made by calling

```
     pseudo_supervisor$defmak (seg, symbol, value, class,
                               error, code)
```

Where the arguments are:

identifier	attribute	meaning
1. seg	character string (*)	segment in whose linkage to work
2. symbol	character string (*)	external symbol
3. value	fixed	value for definition
4. class	fixed	class of external symbol
5. error	label	error return
6. code	fixed	error identifying code

The linkage section for arg1 is searched for a definition identical to that given by arguments 2-4. If not found the definition is added to the linkage section along with any necessary entries, links, and link definitions if arg4 equals 1 (which means that the symbol is entry point).

Error codes and their meanings:

1. Segment arg1 not active in process.
2. Error in attempting to create a linkage section for segment arg1.
3. Error in attempting to grow linkage section to insert new linkage information.
4. Arg2 already defined in the linkage section of segment arg1.

To add n words to a segment, install an external symbol definition in that segment's linkage section, and obtain a pointer to the new block call

```
     pseudo_supervisor$symmk_ (seg, symbol, delta, point,
                                error, code)
```

where the arguments are:

identifier	attribute	meaning
1. seg	character string (*)	segment to work with
2. symbol	character string (*)	external symbol
3. delta	fixed	number of cells to add
4. point	pointer	pointer to new block
5. error	label	error return
6. code	fixed	error identifying code

This procedure adds arg3 words to the segment arg1. It places an external symbol definition of class 0 into the linkage section of arg1 with a value equal to the offset of the first word of the new block. A pointer to the new area of words is placed in arg4.

Error codes and their meanings:

1. Attempt to grow segment arg1 failed.
2. Attempt to create linkage section segment failed.
3. Attempt to grow linkage section segment failed.