Published: 11/03/67
(Supersedes: BY.13.03, 07/07/67)

Identification

link_change
D. L. Boyd, D. H. Johnson

Purpose

The link_change procedure generates linkage section information during the execution of a process. Using this procedure, it is possible to construct links, links with a trap, definitions, and definitions with a trap. (See BD.7.01 for details about linkage sections and definitions.)

Introduction

There are four entries to the link_change procedure: [make_link], [make_definition], [make_trap_link], and [make_trap_definition]. Each entry is described separately under Usage. All additions created by link_change are put in the next available free area in the appropriate linkage section. After each addition, the next free storage area is recalculated. In case of error, link_change uses the standard error handling mechanism (BY.11) for recording and signalling (BY.12.03) errors. The section Errors describes the errors which may be detected. If no linkage segment exists, <link_change> will create it. If, however, the linkage segment does exist, it must begin with a header as described in BD.7.01.

Usage

1. link_change$make_link

   This entry will make new links in a linkage segment. The call is:

```
call link_change$make_link (seg,extype,base,name,
                    symbol,exp,mod,point);

dcl (seg,name,symbol) char (*), (extype,base,exp,mod)
     fixed bin(17),point ptr;
```

The arguments are:

1.	seg	segment in whose linkage section to work
2.	extype	type of external reference
3.	base	base address register number
4.	name	segment name
5.	symbol	external symbol
6.	exp	expression
7.	mod	address modifier
8.	point	pointer to generated fault pair (link)

Argument one, seg, defines the segment in whose linkage section to make the linkage information. Argument two, extype, is the type of external reference (1, 2, 3, 4, or 5; see BD.7.01). Arguments three, four, and five have meaning based on the value of argument two. Arguments six and seven allow for full address flexibility and must be 0 if not required (see MSPM BD.7.01 for explanation of links). When argument two, extype, 3, argument four, name, is not needed and is ignored. The procedure constructs a link with all the necessary information and puts it in the next free area in argument one's linkage section (see BY.13.00). A pointer to the generated link is placed in argument eight, point. If the link is type 1 or 5 (arg 2), the link is set and the linker is not called later. This means that the trap pointer may be turned off if the type is 5 (see BD.7.01).

2. link_change$make_definition

This entry will add an external symbol definition and any associated entries to a linkage section. The call is:

```
call link_change$make_definition(seg,symbol,value,
     class);

dcl(seg,symbol) char(*), (value,class) fixed bin(17);
```

The arguments are:

1.	seg	segment in whose linkage section to work
2.	symbol	external symbol definition
3.	value	value for definition
4.	class	class of external symbol

Argument one, seg, defines the segment in whose linkage section the definition is to be added. Arguments two, three, and four describe the definition to be added to

the linkage section. A description of a definition is found in BD.7.01. If argument four, class, equals one, the symbol is an entry point. In this case, entry words (eaplp and tra instructions) are inserted into the linkage section as well as a link pair that points to the entry in the procedure segment.

The new definition is added to the next available free storage in argument one's linkage section and threaded onto the chain of definitions. The threading process is described in BY.13.0.

3. link_change$make_trap_link

This entry will create in a linkage section a link construction containing a trap.

The call is:

```
call <link_change>$[make_trap_link] (seg,extype,base,
                                     name,symbol,exp,
                                     mod,point,callname,
                                     callsym,argtype,
                                     argname,argsym);

dcl (seg,name,symbol,callname,callsym,argname,argsym)
     char(*), (extype,base,exp,mod,argtype) fixed
     bin (17), point ptr;
```

The arguments are:

1-8 are the same as 1-8 in entry [make_link]

9.	callname	segment name of call (trap) before link
10.	callsym	external symbol name of call before link
11.	argtype	type of link for argument of call before link
12.	argname	segment name of argument call
13.	argsym	external symbol of argument call

The first eight arguments are handled exactly as for entry [make_link]. Arguments nine, callname, and ten, callsym, are defined as a type 4 external reference that is to be used as the entry point in the call before linking. In this case, the m (modifier) and exp (expression) are both zero. Arguments eleven through thirteen define the argument to be passed to the trapping procedure. Argument eleven, argtype, must be type 1, 3, or 4 (see MSPM BD.7.01). Again, m and exp are zero. If the type number of argument eleven is 3, the thirteenth argument, argsym, is not necessary and it may be omitted from the calling sequence.

The new linkage information created by this entry is placed in the next free location in the linkage section of argument one. A pointer to the link for arguments one through seven is placed in argument eight, point.

4. link_change$make_trap_definition

This entry makes in a specified linkage section an external symbol definition and a trap to a specified procedure. This specified procedure will be executed before the definition is used by link_fault.

The call is:

```
call <link_change>$[make_trap_definition] (seg,symbol,value,
                                           class,callname,
                                           callsym,argtype,
                                           argname,argsym);

dcl (seg,symbol,callname,callsym,argname,argsym) char(*),
               value,class,argtype) fixed bin(17);
```

The arguments are:

1-4 are the same as for entry [make_trap].

5. callname segment name of call (trap) before link
6. callsym external symbol name of call before link
7. argtype type of link for argument of call before link
8. argname segment name of argument call
9. argsym external symbol of argument call

The first four arguments are handled exactly as for entry [make_definition]. Arguments five, callname, and six, callsym, are defined as a type 4 external reference that is to be used as the entry point in the call before definition. In this case, the mod (modifier) and exp(expression) are both zero. Arguments seven through nine define the argument to be passed to the trapping procedure. Argument seven, argtype, must be an external reference of type, 1, 3 or 4. Mod and exp are assumed zero. If the type number of argument seven is 3, the ninth argument, argsym, is not needed and it may be omitted from the calling sequence.

The new definition information is added to the next available free storage in argument one's linkage section and threaded onto the chain of definitions.

Errors

If link_change detects an error, it calls seterr (BY.11.01) to record information identifying the error in <error_out>.The condition "link_change_err" is then signaled with the option forbidding return. The following errors are detected:

Error Number	Meaning
1	bad linkage section in call to [make_definition]
2	bad linkage section in call to [make_link]
3	bad linkage section in call to [make_trap_definition]
4	bad linkage section in call to [make_trap_link]