Published: 09/08/67

Identification

EPLBSA, Multiple Location Counters and Multi-Segment Assembly
J. D. Mills and D. B. Wagner

Purpose

Multiple location counters are required in EPLBSA to make the output code generated by EPL more efficient in execution. EPL produces output in a single stream with bits and pieces of prologue, main code, and epilogue intermingled. Without multiple location counters these bits and pieces are connected by transfer instructions increasing length of code by 8 to 10 percent and vitiating paging strategies.

A by product of multiple location counters is the ability to assemble into the link and symbol segments under programmer control. This is done by dedicating certain location counters to these segments with the _join_ pseudo-op herein described.

Conventions

The naive user can ignore multiple location counters and the related pseudo-ops _use_ and _join_. A default location counter will be used.

The following location counter names are predefined. Attempts to use these names as names for other location counters will give erroneous and unpredictable results.

Location Counter

Name	Use
.text.	the main (default) location counter for the text segment
.lkhead.	the linkage section header
.lksect.	the linkage section (links, entries, and mastermode calls)
.lit.	the literal pool
.defs.	external symbol and link definitions
.tv.	mastermode transfer vector
.ercall.	mastermode error call

.st.	the symbol table
.reltx.	relocation information for the text segment
.rellk.	relocation information for the link segment
.relst.	relocation information for the symbol segment

The use of "*" in an internal expression means use the current value of the current location counter. Current values of other location counters may be used in internal expressions by using the name of the location counter. However, if there is ambiguity due to having an internal symbol and a location counter with the same names, the value of the internal symbol will be used.

Usage

The basic pseudo-op is

```
     use name
```

If the name has not been seen before in this context, it is established as the name of a location counter and its value is set to zero. System location counters may be used.

The presence of odd, even, eight, and sixtyfour pseudo-ops is noted and the starting address of a region containing such a pseudo-op is adjusted so that the desired effect is not nullified. If gaps are generated by such location counter adjustment they are filled with nop's.

All instructions and data generated by EPLBSA statements between this line and the next use are assembled using the location counter named. All label prefixes are entered into the assignment table with the current value of name and with an indicator that this value is to be taken as relative to the absolute origin of name (this absolute origin is not known until the end of Pass 1 of EPLBSA).

The pseudo-op

```
     join /text/lct1, lct2, ...,lctm /link/ lcl1, ...,lcln/symbol/
     lcs1,...,lcsp
```

specifies the relative order in which the location counters listed are to receive their origins within the indicated segments. The only names permitted for segment names are text, link, and symbol. It is not necessary to include all three segments nor is any particular ordering of the segments necessary within the join statement. The location counter names must be defined by a <u>use</u> before their appearance in a <u>join</u>. A location counter may be joined only once. If the user includes more than one join statement, counters are first ordered by the order of the joins.

The only system location counters which may appear in a join statement are .text. and .st.

User defined location counters not appearing in a join pseudo-op will be joined with <u>text</u> segment location counters in the order of their appearance in use pseudo-ops and <u>before</u> other location counters. Joining is done at the end of Pass 1 of EPLBSA.

The domain and relative location of system location counters are tabulated below. User defined location counters are placed as indicated.

<u>Text</u>		<u>Link</u>		<u>Symbol</u>	
1.	.text., if unjoined				
2.	Unjoined lc's by order of appearance	1.	.lkhead.	1.	.st. and user defined symbol lc's.
3.	Joined, user-defined text lc's.	2.	user-defined link lc's	2.	.reltx.
4.	.tv.	3.	.lksect.	3.	.rellk.
5.	.ercall.	4.	.defs. (if defs in link segment)	4.	.relst.
6.	.lit.				
7.	.defs. (If defs in text segment)				

<u>Relocation</u>

The rules one would expect are followed in evaluating internal expressions. The values of symbols and of expressions may have either of two types: (1) absolute, and (2) relative to location counter <u>lc</u>. If it is relative to <u>lc</u> then it has a relocation type as defined in BD.2.01. The operands of the arithmetic operators are restricted to the combinations shown in the following list:

operator	operand 1	operand 2	result
+	absolute	absolute	absolute
	relative to lc	absolute	relative to lc
	absolute	relative to lc	relative to lc
-	absolute	absolute	absolute
	relative to lc	absolute	relative to lc
	relative to lc	relative to lc	absolute
	relative to lc1	relative to lc2	absolute*
*	absolute	absolute	absolute
/	absolute	absolute	absolute

Expressions which must be evaluated in Pass 1 (such as those appearing in the bss and org pseudo-ops) must be of type absolute.

The pseudo-op

org <u>inexp</u>

where <u>inexp</u> is an EPLBSA internal expression sets the value of the current location counter (as determined by the most recent <u>use</u>) to the value of <u>inexp</u>. The value of the location counter may be decreased by an org, but not overlaid. Gaps are filled with zero words as with <u>bss</u> and <u>bfs</u>.

* Permitted only if this expression is not evaluated in Pass 1. Also lc1 and lc2 must be in the same segment.