Published: 10/13/66
(Supersedes: BE.8.03,1/14/66;
BE.8.03, 6/29/66)

Identification

Segment Management for the Pseudo-supervisor
D. H. Johnson

Purpose

The following procedures describe basic segment management functions such as creating and releasing segments, growing and truncating segments, and questioning the status of a segment.

Creating Segments

Segments may be created with the procedure

```
     pseudo_supervisor$newseg (name, max, error, code, mask)
```

where the arguments are:

identifier		attribute	meaning
1.	name	character string(*)	segment name
2.	max	fixed	maximum length of segment
3.	error	label	error return
4.	code	fixed	error identifying code
5.	mask	bit(9)	descriptor word bits 27-35 for segment

If arg5 is omitted, a default descriptor bit setting will be used. The default setting will create a segment which is paged, 64 word page size, write permit, slave access, and data.

This procedure will create entries in the descriptor segment, segment name table, and segment length table for segment arg1. It will not allocate memory to the segment. If the segment to be created is paged, a page table segment will also be created. The descriptor information for created segments will be saved and a directed fault will be placed in the descriptor entries. The procedure grow (see below) must be called to allocated memory to an active segment. Procedure grow will remove the directed fault and allocate memory to the segment and its page table segment.

Error codes and their meanings are:

1. Segment already exists for process.

For errors 2-4 all tables remain as they were before the call to newseg.

2. Descriptor segment full, segment cannot be created.
3. Segment name table full, segment cannot be created.
4. Segment length table full, segment cannot be created.
5. The attempt to restore the tables to the state that they were in before error 2-4 occurred failed. The segment name table and/or the segment length table will be out of phase with the descriptor segment if any more segments are created.

Releasing Segments

Segments may be released with the procedure

pseudo_supervisor$relseg (name, error, code)

where the arguments are:

identifier	attribute	meaning
1. name	character string(*)	segment name
2. error	label	error return
3. code	fixed	error identifying code

Error codes and their meanings are:

1. Segment does not exist in process.
2. Segment may not be released.
3. System or machine error in releasing segment arg1.
4. Page table for arg1 does not exist.
5. Free page list full before truncation could be completed. Segment arg1 has been truncated but not released.
6. System or machine error in passing arguments between pseudo-supervisor procedures.
7. System or machine error in releasing page table for arg1.
8. Free page list full before truncation could be completed. Segment arg1 has been release but its page table has not.

Growing Segments

Segments may be increased in length by calling the procedure:

pseudo_supervisor$grow (name, delta, begin, error, code) where the arguments are:

identifier		attribute	meaning
1.	name	character string(*)	segment name
2.	delta	fixed	number of words requested
3.	begin	pointer	pointer to first word added
4.	error	label	error return
5.	code	fixed	error identifying code

Procedure grow adds delta words to segment arg1. If arg1 has unused words in its allocated blocks, those are assigned first. If more pages are needed, the procedure newpag (see MSPM section BE.8.02) is called requesting a page from free storage. The procedure grow updates descriptor, page table, and segment length table entries. Arg3 is set to point to the beginning of the added words.

Error codes and their meanings are:

1. Segment does not exist in process.
2. Free storage exhausted before requested increase could be completed. Segment arg1 unchanged.
3. Maximum segment length exceeded; arg1 unchanged.
4. Segment is unpaged, was not just created, and is attempting to add a block of memory. Segment arg1 is unchanged.
5. Segment arg1 is paged, but page table does not exist; system or machine error.
6. Requested increase would exceed page table. Segment arg1 is unchanged.
7. Segment arg1 is unpaged, was just created, but free page list cannot supply enough contiguous pages of required size. Segment arg1 is unchanged.
8. Requested increase too large. Ask for fewer pages at one time.
9. System or machine error in passing arguments between pseudo-supervisor procedures.
10. User stack segment paged, but its page table could not be found in the segment name table; system or machine error.
11. Free page list overflowed in process of satisfying request.
12. Error in growing page table for newly created segment.

Truncating Segments

Segments may be truncated by the procedure

```
pseudo-supervisor$trunct (name, delta, error, code)
```

where the arguments are:

identifier		attribute	meaning
1.	name	character string(*)	segment name
2.	delta	fixed	number of words to truncate
3.	error	label	error return
4.	code	fixed	error identifying code

Procedure trunct removes delta words from the end of segment arg1. The segment length table entry is adjusted. If the truncation frees one or more pages, they are assigned to the free page pool. Descriptor boundary fields and page table words are revised.

Error codes and their meanings are:

1. Segment does not exist in process.
2. Segment may not be truncated.
3. Segment less than delta words. Segment arg1 will have no blocks (pages) assigned to it. If segment arg1 is paged, its page table also will be void. Descriptor and segment length entries have not been changed.
4. Page table for arg1 does not exist in process.
5. Free page list full before truncation could be completed.
6. System or machine error in passing arguments between pseudo-supervisor procedures.

<u>Segment Status Seeking</u>

The following questions may be asked about segments:

1. Is a segment active in a process?
2. What is the number of a segment?
3. What is the current and maximum length of a segment?

These questions may be answered by the procedures below.

To determine whether a segment is active, call

```
     pseudo_supervisor$segpr_ (name,answer)
```

where the arguments are:

identifier		attribute	meaning
1.	name	character string(*)	segment name
2.	answer	bit string (1)	result of segment search

If segment arg1 is in the segment name table, then arg2 is set equal to 1. Otherwise, arg2 is set equal to 0.

A segment number may be determined by the procedure:

```
     pseudo_supervisor$segman (name, number, error, code,
                                table ptr)
```

where the arguments are:

identifier	attribute	meaning
1. name	character string(*)	segment name
2. number	fixed	segment number
3. error	label	error return
4. code	fixed	error identifying code
5. table ptr	pointer	pointer to arg1 in segment name table

Arg5 is an optional argument.

This procedure searches the segment name table for segment arg1 and sets the segment number in arg2. If arg5 is present, a pointer to arg1 in the segment name table is returned.

Error code and its meaning:

1. Segment does not exist in process.

The current and maximum lengths of a segment may be obtained by calling

```
     pseudo_supervisor$length (name, current, max, error, code)
```

where the arguments are:

identifier	attribute	meaning
1. name	character string(*)	segment name
2. current	fixed	current segment length
3. max	fixed	maximum segment length
4. error	label	error return
5. code	fixed	error identifying code

The procedure length finds the length of segment arg1 and places the current and maximum lengths in arg2 and arg3 respectively.

Error code and its meaning:

1. Segment does not exist in process.