

# LA Static Memory Access

*Local application diagnostic*

Mon, Apr 28, 2003

Every local application instance allocates a block of memory to use for maintaining its own context as long as the instance is active. During initialization, it allocates this block, and during termination, it frees the block. A pointer to the block is kept in the online copy of the local application table (LATBL) entry for that instance. This note describes support for a new listype that would support access to this memory via an offset.

The PAGEPMEM application already allows printing out such memory. And if more than one instance is encountered, the requested portion of the static memory is printed out for each instance. Suppose we implement a new listype that uses an ident of the following format:

<i>Field</i>	<i>Size</i>	<i>Meaning</i>
node	2	Node#
nameLA	4	Local application name
offset	2	Offset into allocated static memory

The internal pointer format may simply be the entry number of the matching LATBL instance. The read-type routine, then, examines the indicated LATBL entry for validity and adds to its static memory pointer the offset value, and returns the memory data so indicated. If the entry is invalid, or if there is no current static memory pointer, the returned data could be all zeros. This approach allows it to work even if the LA instance is restarted, which often results in a static memory block that is located elsewhere.

What if more than one instance exists using the same LA name? For this case, we may have to rely on this not happening often enough to matter. It would not be possible to determine how many instances (maximum) the user desires, as one does not know how many bytes is needed for each instance. Let us assume that only a single instance of an LA needs to be accessed via this listype.

In order to support this listype via RETDAT, we have a problem, because the SSDN cannot hold such a large ident. One approach to allow it would be to use the new value of 3 in the low nibble of the first word, which would mean that the last three words of the SSDN, when concatenated with the offset word, would comprise the ident to be used.

Another approach would be an ident with the following format:

<i>Field</i>	<i>Size</i>	<i>Meaning</i>
node	2	Node#
indexLA	2	Local application instance index
offset	2	Offset into allocated static memory

Here, it is necessary to know the LATBL index of the target entry. If one needed to search in order to get it, we need another listype that uses a named ident (same as used by listype 86) that returns an array of the entry numbers matching that given name, where 0xFFFF could denote the end of the array. In this way, one could more easily set up an Acnet SSDN to address a specific part of the static memory for an LA instance. One could fix the offset in the SSDN, or one could let the given offset value be applied to the SSDN.