

Basic Status Property

Byte-swap bug option

Tue, Jul 14, 2009

The Acnet control system collects device status information via the Basic Status property. It may be 2 bytes or 4 bytes as described herein. Up to 32 status bits can be assigned text via the Extended Status property, along with a specification of how the bits are to be displayed in detail. This note describes how the status data is translated from the front ends to Acnet, especially as displayed on Acnet page D80. That page allows defining matching values and masks to recognize an ON state or an OFF state, for example, so it is important to understand how the data from the front end is actually presented. An existing bug complicates things.

Front end support for the RETDAT (or GETS32) protocol Basic Status property first assembles the status bytes as specified by the SSDN parameters. Then one or two 16-bit words of data are byte-swapped before being included in the reply data. This relates to the fact that Acnet is little endian based, whereas the front ends are big endian based.

Two byte case

Various SSDN specifications can be used to denote the status bytes to be assembled as reply data. As a simple example, one might specify "associated status" related to an analog channel. This allows for one or two status bits to be assembled in the upper two bits of a byte, according to information in fields of the analog channel-related ADESC table entry. If the request is for 2 bytes of such status data, these bits occupy the upper 2 bits of a 16-bit word. To be specific, say the status information thus assembled is 0x8000. When it is processed for inclusion in the reply data for a Basic Status request, it is byte-swapped, so that 0x0080 is returned. Viewed on page D80, this appears as 0x00000080, as D80 always presents basic status as 32 bits. One can then choose suitable attribute patterns to indicate the desired digital characteristics of the power supply, or whatever the status source.

Another SSDN format might collect the data from an analog channel whose reading houses "combined binary" status data. These values are constructed in the data pool according to specifications in the CSTAT nonvolatile memory table. Such specs include Byte#s, masks and shifts, allowing construction of a 16-bit result by sampling 1 or more bits from up to 8 different status Bytes and OR-ing them together. Each front end organizes its status data at a low level via Byte#s that typically range from 0x00 to 0x7F, or in some nodes to 0xFF. This allows for a digital data pool of up to 1024 bits or 2048 bits. This data pool occupies the BBYTE table, from which the CSTAT specs describe how to construct combined binary status words. This was done to mimic widely-used CAMAC power supply status words, in which all status bits are included in a single word. Anyway, whatever the 2 bytes of data, the same treatment of swapping the 2 data bytes is used before delivered the reply. If we have an example of a combined binary status word of 0x1122, then the word as delivered is 0x2211, and D80 displays it as 0x00002211.

Four byte case

Now consider the case when 4 bytes of status must be collected via the Basic Status property. If we assemble the status as 0x11223344, say, as seen in a big endian front end, then what we deliver as reply data is byte-swapped, so it is 0x22114433. But when D80 displays this data, it is word-swapped, so that it appears as 0x44332211. Effectively, it is shown as a little endian long word, byte-reversed from the original assembled data.

The bug

In 68K-based front ends only, there is a bug in the byte-swapping support for the Basic Status property data for 4-byte data requests, when only 2 bytes of status are actually present.

What is delivered is two words of both byte-swapped and non-byte-swapped versions of the 2 bytes of status. For the earlier example of data first assembled as 0x8000 in a 16-bit word, the bug results in returning the two words 0x0080 and 0x8000, which will be displayed (word-swapped) on D80 as 0x80000080. The bug causes the first word (0x8000) to be returned correctly, but the second word returned is a byte-swapped version of the first word. A user, faced with a display of 2 bytes of data both byte-swapped and not, has a choice in choosing attribute masks to characterize an ON state, for example. Either the left word or the right word can be used. But it is best to use the right half of the displayed data, since it will be displayed the same whether the request is for 2 bytes or 4 bytes. But more important, if the bug were fixed, the 2 bytes would appear only on the right, as 0x00000080.

The bug was discovered when we first ran into a case where we really had 4 bytes of status data to return, not merely satisfying a 4-byte data request for 2 actual status bytes. In order to deliver all 4 bytes of actual status data, the bug had to be avoided. But by that time, many Acnet data base entries had been made without recognizing the bug, and it was not obvious how to modify those entries; we merely wanted to make it possible to access 32 bits of status. To accomplish this, the system code was modified so that the bug could be "fixed" on an individual node basis by careful use of option switch settings. (See the note, *Option Switches*, for more detail.) With this scheme, for any individual node, a 4-byte request for the Basic Status property is fulfilled either with the bug in place or with the bug fixed, not a mixture. Again, for PowerPC based nodes, the bug is always fixed; the option switches are ignored.

Fix the bug

It would certainly be desirable to eliminate the bug completely in the 68K based systems. But before doing so, it is necessary to correct all the Acnet database entries that refer to the left half of the 4 bytes displayed when only 2 bytes of status actually exist. We need an algorithm to identify all such cases.

One class of such cases are those that specify listype 0x05, the one that refers to "associated status" of an analog channel. This only has 2 bytes of status, or actually, 2 bits only. Another class would be a single combined binary status word. But we must watch for a case in which two consecutive combined binary status words are accessed, using the last word of the SSDN of 0x0002, rather than 0x0000, as that would be a likely way to access 4 actual status bytes.