

Acnet Error Log

Diagnostic addition

Thu, Jul 26, 2001

During installation of the upgraded PowerPC-based front-ends to support the Linac control system, a steadily-increasing number of errors are logged by Acnet clients. Before any nodes were upgraded, it was typical that no errors were reported. This note describes a diagnostic addition to the Linac front-ends that is designed to help "get a handle" on the cause of these reported errors.

The approach taken for this diagnostic is similar to many that have been designed in recent years, in that it is based upon the data stream support included in all such front-ends. A data stream is a formalized circular buffer scheme. Data requests can be made to capture the most recent records from a data stream, for example.

The new data stream to house the diagnostic records is called ACNETERR. (Every data stream has an 8-character name.) Each 16-byte record includes:

<i>Field</i>	<i>Size</i>	<i>Meaning</i>
tNode	2	Target node for Acnet reply with error status
tSize	2	Reply message size excluding Acnet header
eStat	2	Acnet status word in reply
eNode	2	Node causing error status
eTime	8	Date/time in usual BCD form including half ms in cycle

The target node is recorded as an Acnet node#, such as 0B23, which is dae04.fnal.gov. The target size does not include the Acnet header so it can be compared with the RETDAT request that specifies the number of bytes expected in the reply message. The error status word is in the usual Acnet form, with the error value in the high byte and the facility number in the low byte.

Nearly always, the error status is generated by a server node, based upon the lack of a timely reply from the front-end actually supplying the device data. Other error possibilities can stem from improper database entry, but they are not the focus of study here. These front-ends endeavor to support reliable and correlated 15 Hz data collection, so that the server node has but limited patience with replies from contributing nodes. The node causing the error status may not be the only one, since a single request may seek data from more than one node. The node written into the record is simply one node that caused the error; it may not be the only one.

Details of the implementation

During initialization of the ACReq Task, which supports Acnet protocols such as RETDAT and SETDAT, search for a defined data stream called ACNETERR. If one exists, remember its index# in the data structure of task variables. The logic that captures the diagnostic data should be placed into the routine ACUPSERV, at a point just after the server reply message has been successfully queued to the network via NETQUEUE.

There is a problem with this approach, in that the ACReq task is not running when ACUPSERV is invoked; rather, the Server task is running. This means that the ACReq task variables are inaccessible, which makes it difficult to find the data stream index that may have been captured at ACReq task initialization. In order to avoid searching the DSTRM table within ACUPSERV, use a spare field in the reply message block to retain a copy of this index, or -1 in the case that no data stream was found. When a server request is initialized by the ACReq task, and the server reply message block is initialized, the task variable holding the index is copied into a new field called ACNERRX. (It used to be called SPARE1.)

In `ACUPSERV`, then, a pointer to the reply message block queued to the network is all that is needed to build a diagnostic record. If the `ACNERRX` field in the reply message block is non-negative, then call the new routine `ERRORLOG` that may be included in the same source file with `ACUPSERV`.

`ERRORLOG` must first determine whether an error status is being returned for any of the devices whose data is sought. By using the `RQBLKPTR` field to find the request message block for this request, the array of server request blocks, one entry per device, is used to traverse the reply message to look for a nonzero error status word. If none is found, no diagnostic record will be written and `ERRORLOG` simply returns. But if a nonzero error status word is found, its value is saved, along with the `node#` that serves the offending device, plus the requester `node#` and the reply message size. The requester node will be a pseudo `node#` as seen in the Acnet header of the reply message, so it is useful to translate this value into an Acnet `node#`, which can be done by looking up the `NODENUM` field in the relevant `IPARP` entry use by the pseudo `node#`. (This field was captured when the request message was received.) The current data and time is also copied from `DATETIME`, with the 8th byte replaced by the relative time within the current 15 Hz cycle in half ms units. Call `DSWRITE` to write this record into the diagnostic data stream.

The contents of the data stream can be reviewed manually, say, via a Memory Dump page or the Print Memory page. It can also be accessed by any program using a `listype#` suitable for data stream access.

The diagnostic facility described here is suitable for nodes that are used as data server nodes. Similar logic could be used in the future for replies to non-server Acnet requests by adding the appropriate logic to `ACUUPDATE`. The reason for focusing on server requests is that the mysterious errors appearing in the Acnet Error web page are 36 -8 and 36 -7 errors, which are only returned via server nodes.