RDATA Table Counts

Page application

Robert Goodwin

Tue, Feb 15, 2011

Page application RDAT is a diagnostic aid for managing the Data Access Table (RDATA) entries. This note describes its functionality of listing the number of entry types used in one or more front ends.

The RDATA table comprises up to 128 entries of 16 bytes. The first byte is the entry type#. Valid type#s currently range 0x01–0x35. In addition, a type# 0x7F is used as a "period" entry to enable/disable interpretation of the following entries. Every 15 Hz cycle, the front end interprets the entries to update the local data pool of analog and digital data. For many more details on individual entry types, see the note, *Data Access Table Formats*, plus the note, *RDATA periodicity*.

The display layout is very simple:

```
4 RDATA STATS     02/15/11 1448
NODE<NLIN>-           LIST<0509>
```

Enter either a node# or a node data file name of the form 'Nxxx'. In this example, the data file name is DATANLIN, whose contents are merely an array of node#s, namely, all the Linac front end nodes. On the right is a place for the listing target node; the serial output will be "printed" from that node. Click anywhere on this line to start a listing of all the RDATA entry counts. There is a '–' that immediately follows the node number/file field. If the cursor is in that column, the listing selects only the disabled entries; otherwise, the listing shows only the enabled entries. (An entry is disabled by setting the sign bit of the entry# field.)

The following is a partial listing of the Linac front ends:

```
4 RDATA STATS     02/15/11 1426
     7F 1 2 3 4 5 6 7 8 9 A B C D E F10 1 2 3 4 5 6 7 8 9 A B C D E F20 1 2 3 4 5 6 7 8 9 A B C D E F30 1 2 3 4 5
 0600  1 . . . 1 . . . . . . . . . . . . . . 6 . . . . . . . 1 . . 1 . . . . . . . 2 . . . . . . . . . . . 1 1 1 . . . . .
 0601  1 . . . 1 . . . . . . . . . . . . . . 7 . . . . . . . 1 . . 1 . . . . . . . 1 . . . . . . . . . . . 1 1 1 . . . . .
 060A  1 . . 4 1 . . . . . . . . . . . . . . 3 . . . . . . . 2 . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . .
 060B  1 . . 5 1 . . . . . . . . . . . . . . 1 . . . . . . . 2 . . 1 . . . . . . . . . . . . . . . . . . . . . . . . . .
 0610  5 . . . 1 . . . . . . . . . . . . . .14 . . . . . . . 9 . . 1 . . 1 519 . . . 2 . . . . . . . 1 2 2 3 . . . . .
 0611  1 . . . 1 2 2 3 3 . . 1 . . . 1 . . . 9 . . . . . . . 2 . . 1 . . 3 318 . . . 4 . . . . . . . 8 3 7 . . . . .
 0612  1 . . . 1 2 1 1 2 . . 1 . . . . . . . 5 . . 1 . . . 1 . . 1 . . 1 1 8 . . . 2 . . . . . . . 7 2 7 . . . . .
 0613  1 . . . 1 2 1 2 2 . . 1 . . . . . . . 5 . . 1 . . . 1 . . 1 . . 1 1 8 . . . 2 . . . . . . . 7 2 7 . . . . .
 0614  1 . . . 1 2 1 2 3 . . 1 . . . . . . . 6 . . 1 . . . 3 . . 1 . . 2 210 . . . 2 . . . . . . . 7 2 7 . . . . .
 0615  1 . . . 1 2 1 1 2 . . 1 . . . . . . . 5 . . 1 . . . 1 . . 1 . . 1 1 8 . . . 2 . . . . . . . 7 2 8 . . . . .
```

The header line indicates the various RDATA entry type#s. A 2-byte count field is shown for each type. In case the count is zero, it is shown as " ." to make it easier on the eye. From this listing, one can see that many entry types are not used in these front ends. Examine the counts for entry types 0x20, 0x21, and 0x22. These are used to shepherd SRM-based data into the data pool. The type 0x20 entry sends out a request; the type 0x21 entry awaits the reply message, and the many type 0x22 entries specify the details of what part of the reply data goes where in the data pool.

In case an entry type is found that seems out of range, a message including the node#, the entry# and the invalid type# appears on the display. One does not expect any such.

When the program scans the contents of the RDATA table, it stops if it runs into 4 consecutive entries of type 0x00, which is a null entry. This logic is the same as that used when building the internal copy of RDATA that is actually used by the software. The copy is updated occasionally, so it tracks manual entry into the official nonvolatile version of RDATA. For more on this, see the note, *RDATA Scanning*.