

Booster Beam Loss Sums

Local application BLMS

Fri, Jun 30, 2000

Beam loss measurements from the BLMs in Booster are to be summed to get a total measurement representing relative beam loss during operations. The sums are to be done over time intervals. This note explores how the logic is to be done.

The sums are to be updated several times per long term period. For example, 5-minute partial sums might be accumulated, after which they are added to a 60-minute long term sum. It would be necessary to keep the most recent 12 partial sums in order to properly update the long term sum every 5 minutes, by subtracting the oldest partial sum from the long term sum and adding the new partial sum. The user would then see a long term sum that is updated every 5 minutes and always represents a sum over the last 60 minutes.

A disadvantage of the above logic is that one must wait for up to 5 minutes in order to see any beam losses reflected in the sum. While tuning, it would of course be useful to watch the beam loss over the last beam cycle of interest. (All sums are to be maintained separately for each possible Booster beam reset event.)

The sums over time intervals will naturally depend upon the frequency of a given Booster beam loss event as well as the beam loss specifically relating to that reset event. This will make it difficult to compare beam losses between different kinds of beam cycles. But the sums will reflect the beam loss over a period of time, such as 60 minutes, for each kind of beam cycle.

At the start, what will be the behavior of the long term sums—over the first 60 minutes? One approach would assume that the previous hour had no beam loss, so that the sums of beam loss would continually rise over the next hour to a true total sum, after which it would fluctuate as beam losses fluctuate. Another approach would, after 5 minutes, assume that the past hour has had the same sort of beam loss, so that the fluctuations would begin immediately, without the first hour of "ramping up" beam loss values.

Storage allocation

How shall the summation data structures be organized? In the main context block there will be the long term sums, which will be updated to entries in the new `FDATA` table as floating point raw data that is accessible via `RETDAT`. If there are 16 BLMs supported, there will be, for each clock event, 16 sums assigned to floating point readings of consecutive channels. The first 16 channels could represent clock event `0x11`, say. If there are 10 clock events of interest, there would be 160 channels needed.

In the main context block, let there be, for each clock event, a pointer to a partial sum block that would contain 12 sets of BLM partial sums. If the number of partial sums needed to form the long term sum changes, then there would be a different number of arrays of partial sums. Viewing these arrays in memory, one would follow a specific clock event to the partial sum block that builds up sums for that clock event. Of the 12 arrays, one could find one that would advance every time that clock event occurred.

An alternative would be to have a partial sum block for each partial sum. Inside each such block would be an array of BLM sums for each clock event. In this way, attention is paid on a given cycle to one particular partial sum block for 5 minutes, after which attention moves to the next partial sum block in sequence.