

Loss Monitor Averages

Local application

Mon, Jan 29, 2007

Introduction

There is a need to average beam loss monitor signals to eliminate the influence of spikes in the MI-8 line. These have log amplifiers, so it means computing the corresponding rads from the reading before performing the average summing. The 15 Hz data shows up from a sample and hold, which is triggered at (Booster extraction) event 0x0F plus 10 ms. Averaging is to be done on beam events, except for studies event 0x17. In addition, a TTL control line is to be controlled according to whether the average values are within limits. The latter can be done by using the usual alarm scan, but with the LA monitoring the state of the BAD bit in the alarm flags and operating the control line accordingly. If any of the averages is out of limits, the control line should be set lo; otherwise, it should be set hi.

Let the local application be called LMAV. Use the following set of parameters:

<i>Param</i>		<i>Size</i>	<i>Meaning</i>
ENABLE	B	2	Enable Bit# for LA
LOSSMON	C	2	Loss monitor Chan#
NCHANS		2	Number of LMs
NCYCLES		2	Number of beam cycles in average sum
LOSSAVG	C	2	Average loss target Chan#
ALERT	B	2	Alert one or more LMs out-of-range Bit#

Suppose we arrange to update the average readings for every new beam pulse. This means we need to keep the most recent NCYCLES readings for each loss monitor. At first, there are no historical readings, so the average can merely reflect the first reading. On each cycle after that, include successive readings until the arrays are full. From then on, the averages will reflect the full NCYCLES of readings. The maximum value for NCYCLES may be 32. A user needs to recognize that the larger this parameter, the slower will be the reaction to serious beam loss.

One can adjust the sum each beam cycle by subtracting the oldest term in the sum and adding the newest term. This may be good enough, although it is mathematically susceptible to accumulated round-off errors. An alternative is to sum up all terms in the sum after replacing the oldest with the newest. We need to examine the performance impact of this.

One can change the parameter NCYCLES while LMAV is running. The program will notice this the next time it reaches the end of its arrays using the current value for that parameter. It will then restart assuming no history. It is not expected that this parameter will change often.

A beam cycle for this program is marked by any Booster reset event besides 0x11, 0x12, or studies event 0x17. This is known by the code.

The conversion formula to get engineering units from the log amplifier voltage is:

$$R = 0.0015 * 10^{(v/1.688)}$$

$$R = 0.0015 * e^{(1.3641*v)}$$

The 68040 cpu does not include exponential function evaluation in its hardware. In other previous LAs, a simple function called EXPO has been used. We should compare the

performance of Expo with that included in the C library.

As diagnostics, include the last Booster event#, plus the last beam event#. Also keep the last non-beam raw values of the loss monitor readings.

It may be useful to subtract away a pedestal value for each loss monitor, perhaps based upon the most recent non-beam cycle, or the engineering units value of same.

We can also keep track of the largest beam loss values seen during the most recent series of NCYCLES.

Post-implementation notes

Logic for computing a pedestal value for each loss monitor is based upon the non-beam 0x12 event cycles. The number of cycles in the pedestal average is the same as that used for the loss monitor averages. Assuming 4 loss monitors, the time required for doing the average computations, for the maximum number of 32 beam cycles, is 0.55 ms. The time spent on the non-beam 0x12 cycles to build the pedestal sum is 0.28 ms. All these times seem short enough. If we use 20 beam cycles, the time to do the loss monitor averages is 0.40 ms.