

Acnet 15 Hz Correlated Data

Related data across front ends

Tue, Nov 30, 2004

Introduction

This note deals with providing Acnet client support for correlated 15 Hz data collected from multiple front ends. The difficulty is that Acnet console applications do not operate in accelerator 15 Hz time, and the data they collect can arrive at various times from the front ends. In order to sort out what data relates to what, a key can be included with the data to identify the 15 Hz cycle to which it belongs. Note that the 15 Hz cycle key relates to the beam of that cycle, not the time. Linac beam data is known by, say, 10 ms after Linac beam time, whereas Booster beam loss data is only known after Booster extraction, say, 40 ms after Linac beam time. Even though these data are measured at quite different times, they relate to the same beam, and in that sense, they are correlated.

Correlation key

A natural key for tagging 15 Hz data is a 15 Hz cycle counter, one whose value can be known by every front end. Such a globally accessible counter is included in the 15 Hz ethernet multicast message that includes all Tevatron clock events that occurred since the last such message. This datagram is sent out soon after clock event 0x0F, or about 16 ms ahead of Booster reset event (0x11, 0x12, etc) time. For this discussion, the low 16 bits of that cycle counter are used as a tag to identify any 15 Hz data that is measured on the following cycle, beginning at Booster reset time.

Note that is not necessary for every front end to monitor the multicast event message in order to obtain this key. It is enough for one front end to monitor the message for that purpose, then multicast the cycle counter key value to other front ends by means of another multicast message, but only occasionally. In between, each front end increments its own copy of the key every 15 Hz cycle. Several years ago, in order to support correlated access to Booster BLM waveform data, this scheme was implemented in front end node06C3, which multicasts this key every 17 seconds. As a result, if a front end reboots, it must wait up to 17 seconds before it has obtained the common key value.

Tagging data with the key

When an application makes a RETDAT request for normal analog channel data readings in IRM front ends or in PowerPC front ends used in Linac, it asks for 2 bytes of data. But if it asks for more than 2 bytes of data, this can be interpreted as a request for tagged data. If the request is for 4 bytes, it can mean the data returned should consist of the 2 bytes of reading followed by the 2 bytes of cycle counter key. This key can then be used to correlate similarly obtained data values from other front ends. Use of the length parameter in this way can help to avoid having to create a new Acnet device to obtain tagged data for an existing signal.

Recent Cycle data

As an amplification of the tagged scheme described, a front end that maintains a 15 Hz data poll can keep multiple copies of such data pool readings, so that one can obtain 15 Hz tagged data readings from multiple recent cycles. This allows collecting such data at slower rates, say, at 1 Hz. If a RETDAT request is received for a device in which the length parameter is 32 bytes, and for which the FTD indicates 1 Hz replies, each reply will include 15 readings plus the 2-byte key. The key relates to the first reading, which is the present cycle reading, the same as that obtained by a 2-byte request. Successive readings in the reply data relate to ever-earlier cycles. This is an extension of the case of collecting 15 Hz tagged data readings.

An implementation of the above scheme was done for the IRM front ends, allowing for access

to as many as 32 recent 15 Hz cycle readings. This allows for one-shot requests for (very) recent historical data. If some strangeness occurs, such as a signal going bad, one can issue a one-shot request for a set of readings covering the previous 2 seconds.

Such data can also be requested to be returned on a clock event, in which one would get the recent 15 Hz cycle readings that occurred before and on that cycle. As a variation of this, one can “invent” a clock event. This was done for MiniBooNE to mark the cycle after a string of 0x1D cycles by the aid of a local application. With such a locally understood event, a data request specifying that invented event number allows capturing data measured for the string of 0x1D events, which might be 10 events, say.

Application support

How can an application use this support for continuous correlated 15 Hz data? Support for correlating data values using the key described here can be a bit messy. But this mess can be sorted out by suitable Acnet support once, easing the job for many application programs. An application would “open” its interest in the 15 Hz readings of a set of devices. It could call a “next” routine to ask for the oldest set of data for which all matching keyed data values have been received from whatever front ends are involved. It could then process this data and loop to ask for another set. When a status return indicates none is available, it can await its next cyclic execution. When the application no longer desires such data, it can “close” its interest. Things might be messy “behind the scenes,” but it should be easy for the application to monitor continuous samples of correlated 15 Hz data readings.

As an aid in processing such correlated 15 Hz data, it may be useful to include in the data of interest the device B:BREVNT, whose reading value is the current Booster reset event number, such as 0x001D.

Example of reply data layout

5 Hz example 8 byte request

reading	current cycle
reading-1	previous cycle
reading-2	two cycles ago
cycle number	current cycle#