Handling of rare events – Jürgen's ideas

Last updated: 8 April 2002

Most of the ALICE clusters won't have the advantage of multi-event buffering due to the fact that a significant number of sub-detectors contain only a single-event buffer in their front-end electronics. As a result, no trigger rate control algorithm of the *"leaky bucket"* type is applicable; the trigger class down-scaling remains the only available mechanism. The down-scaling is successful in "balancing" the share of different classes, but, unfortunately, it doesn't solve the problem of *rare events*.

The simulation shows that a relative inefficiency of the ALICE, caused by a predominant single-event buffer architecture, will affect the performance only in the first few minutes of a run; after that, the system rate will be restricted by the available throughput of the DAQ/HLT system.

Jürgen's proposal for handling of rare events is similar to the *Trigger Modifier* option (NA57); or the *Soft BUSY* approach; it's also similar to the *Trigger Priority* scheme, adopted by the ATLAS Central Trigger Processor. It operates in the following way.

Principle:

When the LDC buffers reach the *Nearly Full* status, all the trigger classes that do not contribute to rare events are disabled; from that moment on, only the rare events are taken. The rare event rate is certainly below the available throughput of the DAQ/HLT system and the LDC buffer occupancy moves towards the *Nearly Empty* level. When the level is reached, all the trigger classes are again enabled. The setting of the *Nearly Full* and the *Nearly Empty* levels must be done in such a way that the available throughput of the DAQ/HLT system is fully utilized, while the maximum length of time is *dedicated to the acquisition of rare events only*. The setting would be under software control, dynamically optimized during a run time.

Implementation:

Jürgen has suggested the introduction of an additional *DAQ BUSY* signal. The *DAQ BUSY,* described in the current version of the CTP URD, would remain a general "Trigger Enable/Disable" command, affecting *all* the trigger classes. The *DAQ BUSY (1)* signal, on the other hand, affects only a *selected subset* of trigger classes, the selection being a fully programmable option. If the subset embraces all the classes that do not produce rare events, and if the *DAQ BUSY (1)* is asserted when the *Nearly Full* level is reached and cleared when the buffer occupancy falls to the *Nearly Empty* level (see above), the time while the *DAQ BUSY (1)* is set is dedicate to the acquisition of rare events only.

Comment on implementation:

All the trigger classes already have an individual, dedicated bit – the *Class Mask*, controlled by the system software (see CTP Notes 1, *Class L0 trigger logic*). Turning a selected subset of those bits *on* and *off* is functionally identical to the operation controlled by the *DAQ BUSY (1)* signal. The use of *Class Mask* bits offers an additional flexibility: a number of *Nearly Full/Nearly Empty/DAQ BUSY(1)* partitions could be made to work in parallel; if appropriate, the option could be used to further "balance" the taking of rare events. No hardware modification/addition is required.

Another comment on implementation:

Doubts have been expressed about the feasibility of monitoring the occupancy of the LDC buffers and a too long latency in delivery of the status information. The large capacity of the buffers provides for an alternative implementation, not affected by the mentioned shortcomings.

The switching between the mode in which all the classes are active and the mode in which only the rare events are enabled could be controlled by a programmable time function. The *period* should be long enough to avoid too frequent switching; the *duty cycle* should be tuned to keep the occupancy of the LDC buffer fluctuating up and down, without ever/only rarely becoming *full* or *empty*. Temporary dwelling in either status only makes the system operation less than optimum (*full* - looses the precious rare event acquisition time; *empty* - contributes to the inefficiency); neither leads to a fatal error or a major degradation.

The dynamic software control of the time function (*period, duty cycle*) requires only infrequent interventions based on long(longish)-term observation of the process; the latency and the rate of monitoring of the buffer occupancy are not critical.

Action:

- Discuss the feasibility of the proposed scheme with the DAQ/HLT groups. Could the way be found to adequately monitor the occupancy of the LDC buffers/a sample LDC buffer? Is the latency in delivering the buffer status information acceptable (CPU to CPU transfer time)? Could the buffer status be successfully modeled?

- Use simulation to estimate the performance of the proposed scheme. Compare it with other solutions (if any).

- Consider the problems the scheme might cause for the cross-section calculation.