Discussion and Planning for Calculating DAQ CPU Requirements

22 March 2000

Present: Ritz, Burnett, Wood, Wood, Suson, Russell, Lovellette, Williamson, Haller.

Goal is an estimate of the computing load to complete the DAQ functions, documenting all major assumptions. Where appropriate, bracket uncertainties.

Draft of steps:

1) Reading and unpacking of data:

- Briefly review the current fidelity of the simulation with regard to (a) volume and (b) format for TKR, CAL, ACD, Trigger, and Ancillary. Matching hardware output to CPU word sizes and padding.
- What should be added in the near term to get our answers? **We are OK for now – exact format fidelity is not critical for this study, so there is no reason to wait for a full implementation. The TKR format is already close. The work to finish bundling the data will be done at Stanford independently of this study.**
- How to calculate requirements for reading in and unpacking/reformatting data? An average event at orbit max after L1 is about 15-18 kbits. Must the whole event be read in (see item 2 below)? What CPU overheads are reasonably expected for memory management functions (both front and back end of processing)? **JJ will do this with benchmarks of toy algorithms and analysis for all data sources (TKR, CAL, ACD, Trigger&Ancillary) during the next 4 weeks.**

2) Processing strategy:

- Event reconstruction in stages ("L2" and "L3")?
- Track strategy: Hough and puff? Kalman? Other? Do this vs occupancy between $5x10^{-5}$ and 10^{-3}.
- Rudimentary CAL pattern recognition, noise issues.
- Selection primitives:

    number of tracks with quality cuts

    energy in CAL

    reconstructed gamma candidate direction

extrapolated gamma to ACD

# of hit ACD tiles

current s/c position and direction

comparison of gamma candidate direction to earth horizon direction

An allocation will be made for each of the primitives, but the major load will be the tracking which will be benchmarked as follows:
Toby will provide the Hough transform tracking algorithm (in C++) to Dan&Dan, who will

a) **compile it on the PowerPC (603) with documented compiler version and switches.**
b) **format the IRF output for downloading to the CPU board.**
c) **separately benchmark the unpacking and the pattern recognition steps.**

Benchmarking will be done on standard background files provided by Toby, with TKR noise occupancies of $5x10^{-5}$, $1x10^{-4}$, $5x10^{-4}$, and $1x10^{-3}$. Benchmarks will be mean times, along with the full processing time distributions, for the 4 components of the background mix files separately (chime, p albedo, gamma albedo, electrons by selecting on MC_Src_Code). Benchmarks to be complete by 4/27, with weekly updates on progress.
JJ will review the code (~2k lines) to search for obvious inefficiencies. He will also make an estimate of the additional overheads for robustness (range checking, etc.), to be reviewed also by Michael and Dan&Dan.
Toby will confirm the GCC compiler version can use the version of the STL used in his code. If it can't, we'll re-evaluate the strategy.
Steve will make an estimate of the CAL processing requirements, in consultation with the CAL group.

Selection strategy to be reviewed by Steve and Toby.

3) Housekeeping tasks:

- digital data quality
- subsystem tasks
- trigger housekeeping"
    - comparison of readout data with trigger data (subset of events, at least)
    - pass-thrus
    - rates of each type
    - distributions of variables used in selections (before cut!)
    - deadtime calculations

JJ will make estimates. Memory issues here.

4) Allocation of CPU and memory resources for onboard science analysis: (a) further cuts, and (b) analysis and alert message preparation.

The whole allocation will very likely be smaller than the uncertainties in the trigger analysis. An allocation will be made and circulated by JJ.

Impact on data handling of alternating X-Y tower orientation suggestion?

A complication for coding and testing, but not impossible; however, what does it really provide?