What has been done since the last ROD meeting

✔ Busy logic is implemented (DSP responsibility)

✔ DMA scheduling simultaneously with event processing (sustained rate)

✔ Used the DSP JTAG port for debugging

✔ PU boards now equipped with TMS C6202 (instead of TMX) chips.

✔ 2nd iteration of the PU board (solved the power-on problem)
Main event loop

The main program (written in C/C++)

- schedules the DMA transfers
- implements the BUSY logic
- checks for offline commands sent to the PU.
- calls the event processing and histogramming routines:

```c
void evtproc26(int* data_in, int* data_out);
```

Energy, time, and chi2 calculation, currently based on loop26 from S.Böttcher. Also uses the calibration coefficients and a division table, whose addresses are provided via global symbols.

```c
double emask(int* data, float El);
```

To flag the channels above a given energy threshold.
int filterHisto(int* data, double mask, float E2, float E3,
    int chi2_cut, float Offset);

• Fills energy histograms;
• Suppresses the time/chi2 words from the event output, as specified by mask;
• Fills the time histograms for channels above E2 energy threshold;
• Fills a second channel mask and flag channels with E > E3 or chi2 > chi2_cut
  (for dumping ADC samples)

The function returns the number of channels for which the time/chi2 information is present.

The energy and time histograms are filled separately for each gain.

The number of bins can be configured separately for each set of histograms. The bins are 32-bits, with saturation at $2^{32}-1$ in case of overflow. Each histogram has underflow and overflow bins. The addresses of the energy and time histograms are provided via global symbols.
**BUSY logic**

The BUSY signal is active when there are at least $N_{BusyOn}$ events waiting in the dual-port memory ($N_{BusyOn} < 128$). The BUSY signal is off when there are less than $N_{BusyOff}$ events in the dual-port memory ($N_{BusyOff} < N_{BusyOn}$).

**DMA scheduling**

Uses 2 input buffers (ping-pong) and 4 output buffers (circular accessing). The DMA and the CPU can access simultaneously the internal DSP memory. Input DMA at 83 Mwords/s, output DMA at 62.5 Mwords/s. The DMAs are active during $\leq 4 \, \mu s/event$. 
**Time budget (C6202, 250 MHz)**

<table>
<thead>
<tr>
<th>Where</th>
<th>CPU Time per event (for 64 channels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>loop26 (E, t, chi2 calculation)</td>
<td>3.5 µs/64 channels</td>
</tr>
<tr>
<td>emask (mask generation)</td>
<td>0.15 µs/64 channels</td>
</tr>
<tr>
<td>filterHisto (output formatting, fill histograms, second mask generation)</td>
<td>24 ns/selected channel typ. = 0.15 µs (f=10%)</td>
</tr>
<tr>
<td>Event loop, DMA scheduling, BUSY logic, etc.</td>
<td>0.8 µs (C code)</td>
</tr>
<tr>
<td>Total time measured</td>
<td>4.8 µs (f=10%)</td>
</tr>
</tbody>
</table>

Note: 128 channels already possible with this board (4.8 µs * 2 = 9.6 µs)

**Upgrade path:**

Replace the C6202 by the C6203 running at 300 MHz (TMX devices available now)

Upgrade to faster dual-port memory and output FIFO.
Input DMA reading from DPRAM (active low)

Event processing (active high)

Output DMA writing to the event FIFO (active low)
C6202 boards being delivered now
To Univ. of Geneva, MPI, and BNL.

What remains to be done
Complete the hardware documentation. First draft already available at
http://www.nevis.columbia.edu/~simion/DSP6202.html

Distribute and document the DSP and offline source code and utilities library.