Parallel Algorithms for Track Reconstruction in the CBM Experiment

I. Kisel, I. Kulakov*, M. Zyzak
(for the CBM Collaboration) 
* I.Kulakov@gsi.de

Tracking Challenge in the CBM Experiment

- Conventional KF
- Open MP
- Connect according to the track
- Single
- Task level (ITBB, parallelization between cores)
- MIMD
- Strong scalability on many
- Square root KF
- KF track smoother
- Highly parallelized code
- The algorithms are highly parallelized both on data and on task level.
- Single precision
- Headers
- The algorithms are effective and fast (up to 13 ns/track/node for KF track fitting and up to 200 ms/event/core for CA track finding)
- High reconstruction efficiency
- Reconstruction in the CBM experiment is based on the Kalman Filter and Cellular Automaton algorithms
- The algorithms are effective and fast (up to 13 ns/track/node for KF track fitting and up to 200 ms/event/core for CA track finding)
- High reconstruction efficiency
- Reconstructed track; conventional; analytic formula; single precision

Kalman Filter (KF) Based Track Fitter

- Track fit: 1. Start with an arbitrary initialization 2. Add one hit after another 3. Improve the state vector 4. Get the optimal parameters after the last hit
- High optimized code with respect to time
- Single precision
- Highly parallelized code

KF Library

- Tracking tools:
  - KF track filter
  - KF track smoother
  - Deterministic Annealing Filter
- KF approaches:
  - Conventional KF
  - UD-Fitting
  - Square root KF
  - Carlson
  - Potter
- Track propagation:
  - Runge-Kutta
  - Analytical formula
- Parallelization: data level:
  - Headers
  - WC
  - AVB
- Precision: single
- task level:
  - ITBB
  - ArBB
  - Open MP
- High track parameters resolution at all stations

SIMDization with C++ Header Files

- Header files overload SIMD instructions implementing all operators and inflining basic arithmetic and logic functions, that makes a code compact and easily readable.
- Overloading of + and * operators, example:
  - friend fvec operator(const fvec &a, const fvec &b) { return _sm_mad(p_a(_sm_mad(p_a(a,b)),b)); } return _sm_mad(p_a(_sm_mad(p_a(a,b)),b)); }
  - Line function, example:
    - w/s headers
    - _vec y = _vec + b;
    - a SIMDized code with the headers
    - is as simple as a scalar code

Conclusions

- Reconstruction in the CBM experiment is based on the Kalman Filter and Cellular Automaton algorithms
- The algorithms are effective and fast (up to 13 ns/track/node for KF track fitting and up to 200 ms/event/core for CA track finding)
- The algorithms are highly parallelized both on data and on task level.
- The algorithms scales linearly with number of cores.