Track and Vertex Reconstruction on GPUs for the Mu3e Experiment

Dorothea vom Bruch for the Mu3e Collaboration

GPU Computing in High Energy Physics, Pisa

September 11th, 2014



Physikalisches Institut Heidelberg



- The Mu3e experiment
- Readout and event selection
- Track fit on the GPU
- Current performance
- Outlook



The Mu3e experiment searches for...

... the charged lepton-flavour violating decay $\mu \to e^+ e^+ e^-$ with a sensitivity better than 10^{-16}



- Suppressed in Standard Model to below 10⁻⁵⁴
- Any hint of a signal indicates new physics:
 - Supersymmetry
 - Grand unified models
 - Extended Higgs sector
 - ····
- Current limit on branching ratio: 10⁻¹² (SINDRUM)





Signal

- Coincident in time
- Single vertex
- $\mathbf{\Sigma}\vec{p_i}=0$
- $E = m_{\mu}$

Signal versus Background







Signal

- Coincident in time
- Single vertex
- $\mathbf{\Sigma}\vec{p_i}=0$
- $\bullet E = m_{\mu}$

Combinatorial background

- Not coincident in time
- No single vertex

$$\bullet E = m_{\mu}$$

• $\Sigma \vec{p_i} \neq 0$

Signal versus Background









Signal

- Coincident in time
- Single vertex
- $\mathbf{\Sigma}\vec{p_i}=0$
- $E = m_{\mu}$

Combinatorial background

- Not coincident in time
- No single vertex
- $E = m_{\mu}$
- $\Sigma \vec{p_i} \neq 0$

Internal conversion background

- Coincident in time
- Single vertex
- $E \neq m_{\mu}$
- $\Sigma \vec{p_i} \neq 0$





- μ decays at rest \rightarrow $p_e < 53 \,\mathrm{MeV/c}$
- Resolution dominated by multiple Coulomb scattering (\propto 1/p)

Minimize material

- \blacksquare High Voltage Monolithic Active Pixel Sensors thinned to 50 $\mu{\rm m}$
- Ultralight mechanics

Detector Requirements





Excellent momentum resolution: < 0.5 MeV/c</p>

- Good timing resolution: 100 ps
- Good vertex resolution: 100 μ m

Graph: R. M. Djilkibaev, R. V. Konoplich, Phys.Rev.D79(2009)073004





























Beam and Statistics





- Beam provided by the Paul Scherrer Institut
- Currently: $10^8 \, \mu/s$
- In future: Up to $2 imes 10^9 \, \mu/s$

- Triggerless readout
- 1 Tbit/s data rate
- Online selection
 - \rightarrow Reduction by factor 1000

Readout Scheme





Readout Scheme









- GPU gets 50 ns time slice
- Full detector information
- Find 3 tracks originating from common vertex

Multiple Scattering Fit

- Ignore spatial uncertainty
- Multiple scattering at middle hit of triplet
- Minimize multiple scattering χ^2



Minimize
$$\chi^2 = rac{\phi^2_{MS}}{\sigma^2_{MS}} + rac{ heta^2_{MS}}{\sigma^2_{MS}}$$



Multiple Scattering Fit





- Describe track as sequence of hit triplets
- Non-iterative fit

GPU Specifications





- Use Nvidia's CUDA environment
- GeForce GTX 680
- 8 Streaming Multiprocessors

Image source: http://www.pcmag.com/article2/0,2817,2401953,00.asp



- Consider first three detector layers
- Number of possible track candidates $\sim n[1] \times n[2] \times n[3]$
- n[i]: # hits in layer i
- On GPU: Loop over all possible combinations
 - Geometrical selection cuts
 - Triplet fit
 - Vertex fit

\Rightarrow Goal: Reduction factor of \sim 1000



- On FPGA:
 - Sort hits
 - Copy hit arrays to global memory of GPU
- Currently: FPGA tasks are performed by CPU
- Within kernel / thread:
 - Apply geometrical selection cuts: For pairs of hits in layers [1,2] and [2,3] check proximity in x-y plane and in z
 - Do triplet fit
 - Cut on χ^2 and fit completion status
 - If all cuts passed: Count triplets and save hits in global memory using atomic function
- Copy back global index array



Some instruction

Within kernel / thread:

- Apply geometrical selection cuts: For pairs of hits in layers [1,2] and [2,3] check proximity in x-y plane and in z
- Do triplet fit
- Cut on χ² and fit completion status
- If all cuts passed: Count triplets and save hits in global memory using atomic function

Sharing the Work



Some instruction

Within kernel / thread:

- Apply geometrical selection cuts: For pairs of hits in layers [1,2] and [2,3] check proximity in x-y plane and in z
- Do triplet fit
- Cut on χ^2 and fit completion status
- If all cuts passed: Count triplets and save hits in global memory using atomic function

Grid Alternatives: One Fit per Thread



Grid Alternatives: Several Fits per Thread







Advantages

No idle threads in time-intensive fitting kernel Block dimension: Multiple of 32 (warp size)

Sep 11, 2014





\Rightarrow Choose separate kernel version

Compute Utilization: Fitting Kernel







Sep 11, 2014

Dorothea vom Bruch



Idea	Problem
Count triplets by using atomicInc on shared variable	Synchronization of threads and copying back to global memory takes too long
Compose grid of only one block and n[1] threads; load hit arrays into shared memory for quicker access	Amount of shared memory per Streaming Multiprocessor not enough # of blocks too small to hide latency



	One kernel	Separate kernels
Wall time of		
CPU & GPU $ ightarrow$	$7.6 imes10^9$ triplets/s	$1.4 imes10^{10}$ triplets/s
Run time		
measured over	> 11 days	> 15 hours

- Most time spent on selection cuts
- Can be improved by using FPGA for selection
- Currently: Fit performed on CPU and GPU to compare output
 - \rightarrow Contributes to computation time



- Searching for $\mu \rightarrow e^+ e^+ e^-$ with a sensitivity better than 10^{-16}
- Goal: Find 2×10^9 tracks/s online
- Achieved: Process 10¹⁰ triplets/s



- Include vertex fit or alternative vertex selection criteria
- Outsource pre-fit selection to FPGA
- Write data to GPU via Direct Memory Access from FPGA



Thank you for your attention!



Backup Slides

Wall time of	1.4×10^{10} triplets/s
fit on CPU	
GPU time only	
Average time per fit	26 µs

di O time only	
Average time per fit	26 μ s
Average time for fit & memory copying	30 μ s
Fit & copying	$1.1 imes10^7$ fits/s 1

¹Time measured by nvprof, includes profiling overhead.

Multiple Scattering





NVVP Profile: One Kernel



Kernel profile for one selected kernel



NVVP Profile: Separate Kernels



Kernel profile of one selected fitting kernel (without selection kernel):

• •	McKW House Profes			
No Vev Hep				
The second secon	and the factor (Tanana (TAP)			
Line Sher Court	Ga, hereasonha chin da analaintra la visa crudavidat nets sanarata kamal nu	Gar Cost Banarilla		
	A manufacture of a	Number Number 100	2	
244 245 245 246 246 247 246 240 240 250 251 252	and the second s	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		
Analysis IS Quarter	In Console Cli Settings			
2.2	stead of Analysis		a a a a a a a a a a a a a a a a a a a	





- Nvidia: API extension to C: CUDA (Compute Unified Device Architecture)
- Compile with nvcc and gcc → runs on host (= CPU) and device (= GPU)
- Very similar to C / C++ code
- Compatible with other languages





CUDA: special variables / functions introduced for

- Identification of GPU code
- Allocation of GPU memory
- Access to grid size

....

Options for grid launch

CUDA Grid

Grid: Consists of blocks Block: Consists of threads

CUDA Architecture (GTX 680 as example)



- One kernel per thread
- Up to 3 dimensions for block and thread indices
- Up to 1024 threads per block
- Max dimension of grid: 65535 x 65535 x 65535
- Access to thread & block index via built-in variable within kernel



Hardware Implementation (GTX 680 as example)

Mass

- All threads in grid execute same kernel
- Execution order of blocks is arbitrary
- Scheduled on Streaming Multiprocessors (SMs) according to
 - Resource usage: memory, registers
 - Thread number limit



Hardware Implementation: Warps

- After block is assigned to SM
 Division into units called warms
 - \rightarrow Division into units called warps
- On GTX 680: 1 warp = 32 threads



Warp Scheduling



Warps execute

In SIMD fashion (Single Instruction, Multiple Data) Not ordered











Warp Memory Access

128 bytes in single transaction