TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Jungfrau-Photoncounter

Florian Warg, Jonas Schenke, Sebastian Benner
Dresden, Januray 31 2017

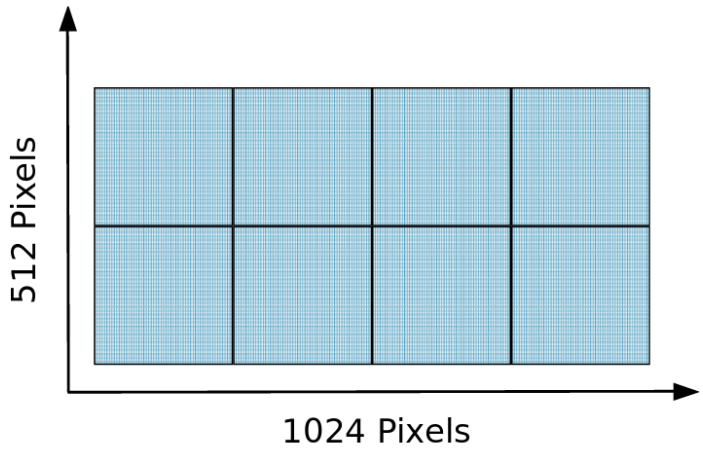
Outline

- Task
- Implementation
    - Concept
    - Optimization Concepts
    - Results

Task

- Conversion
  charge → number of photons
- Module:
  - 2000 fps
  - 16 Bit / Pixel
  - ~2 GB/s

Jungfrau-Sensor



Task

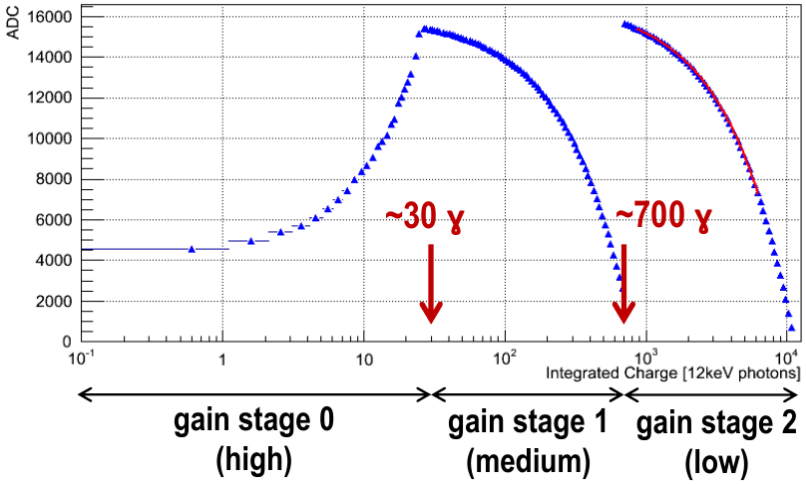
- 3 gain stages

- Offset – pedestal maps

G_0: $C_{corr}[x,y] = D_{ADC}[x,y] - P_0[x,y]$

G_1: $C_{corr}[x,y] = P_1[x,y] - D_{ADC}[x,y]$

G_2: $C_{corr}[x,y] = P_2[x,y] - D_{ADC}[x,y]$



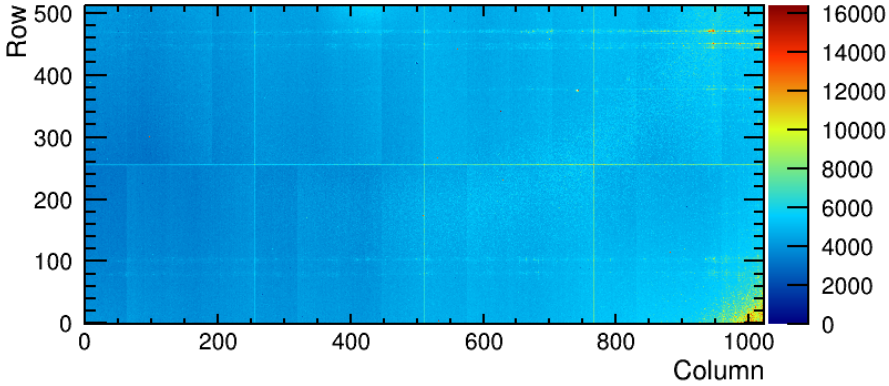
Task

- Gain Maps

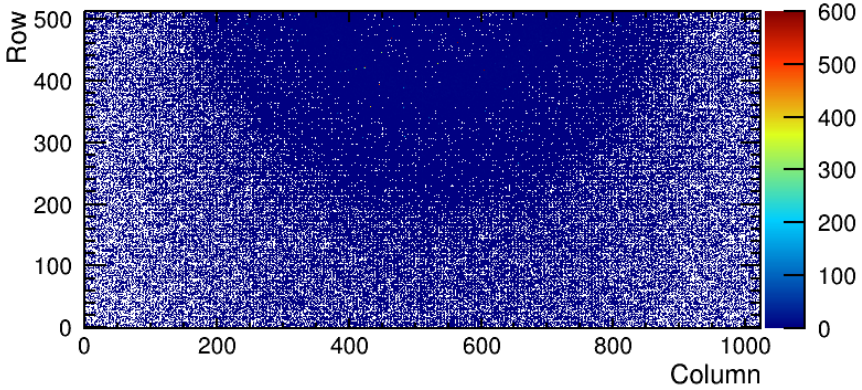
G_0: $E_{cal}[x,y] = C_{corr}[x,y] * G_0[x,y]$

G_1: $E_{cal}[x,y] = C_{corr}[x,y] * G_1[x,y]$

G_2: $E_{cal}[x,y] = C_{corr}[x,y] * G_2[x,y]$



- Data
- Offset removed
- Multiplied by gain map
- NOP[x,y] = $E_{cal}[x,y]/E_{Beam}$

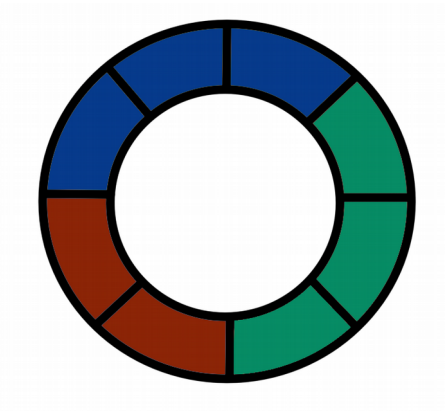


Implementation - Concept

Upload function

- uploads data from buffer to GPU
- asynchronous kernel calculation
- creates CUDA callback

Ringbuffer



Vector
<DeviceData>

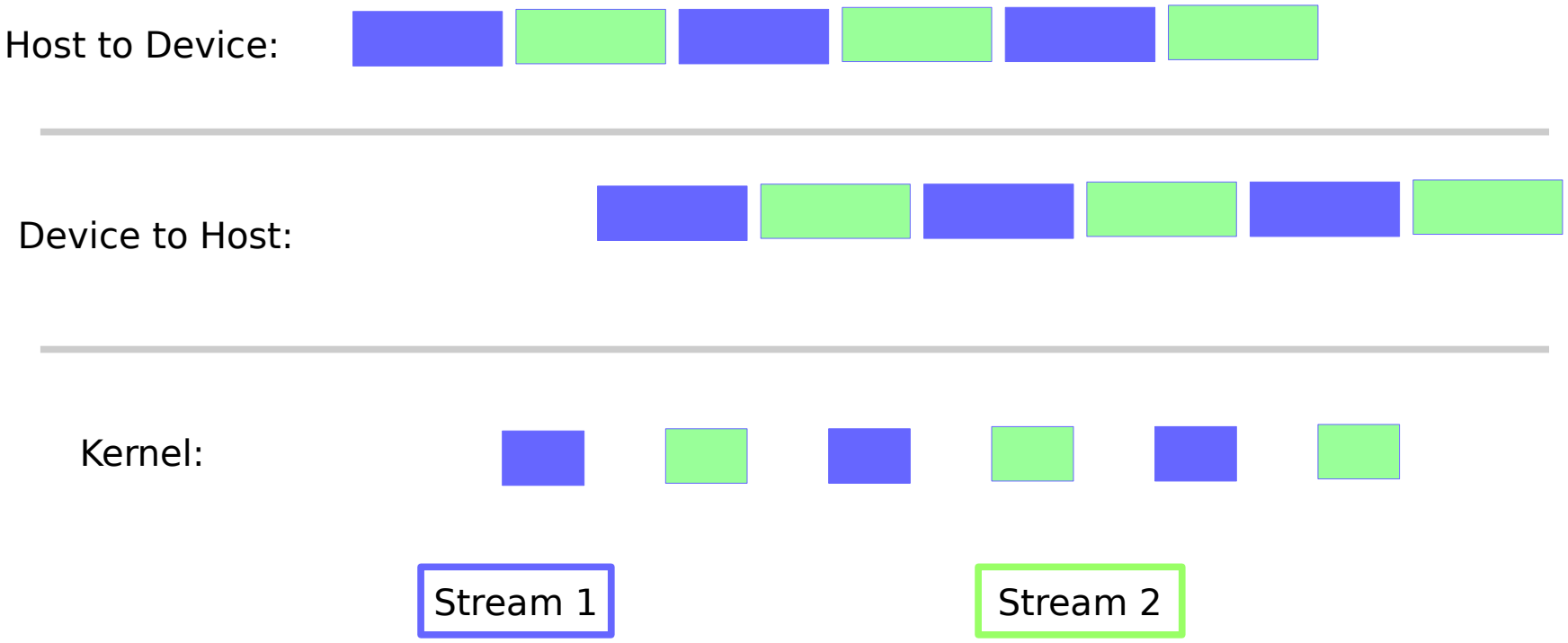
Download function

- copy data
- fill ringbuffer

DeviceData = streaminformationen; device- & host-pointer for data-, pedestal- and gainmaps; processing state

Implementation - Concept

Optimal Timeline:



Implementation – Optimization Concepts

- Split frame blocks onto multiple devices
- Calculate multiple frames in one kernel
- Use streams:
    - Use pinned memory
    - Asynchronous datatransfer
    - Asynchronous kernel call

Implementation - Optimization Concepts

Further improvements (fine tuning)

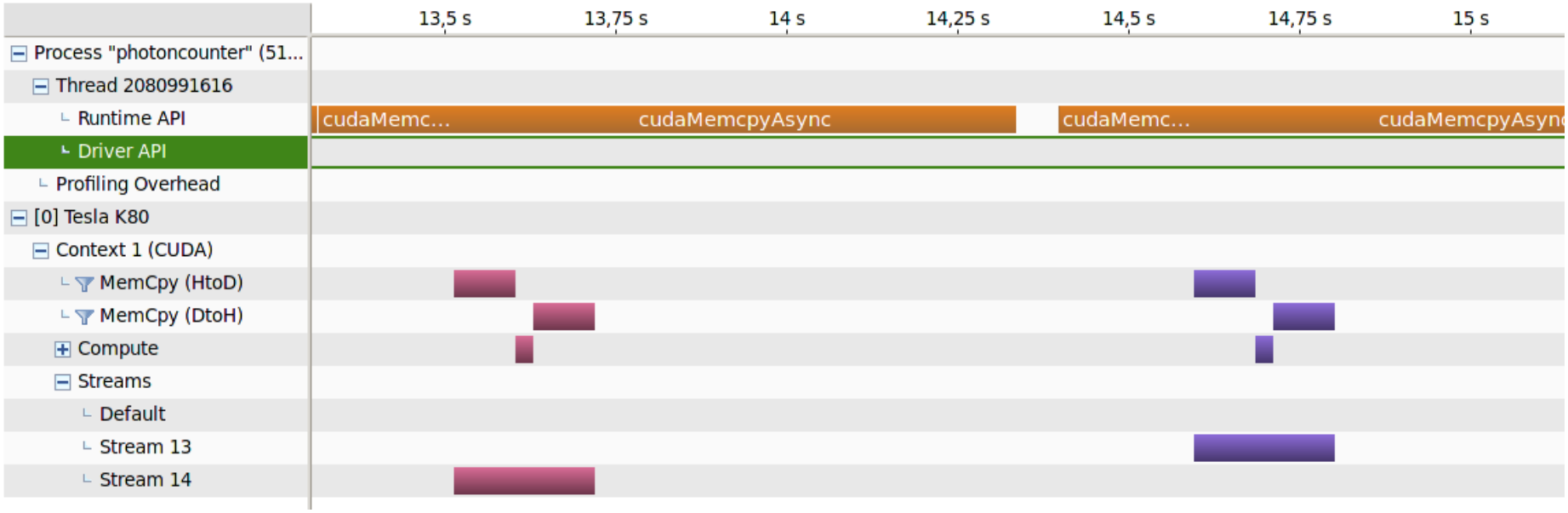
- Find optimal number of frames
- Find optimal proportion streams / device
- Remove paged memory
- Sequential reads inside of the kernel
- Switch X and Y when calling kernel
  - kernel<X, Y, ...>(...) → kernel<Y, X, ...>(...)

Implementation - Optimization Concepts

- Current state:
  - Working (without pedestal calibration)
  - Multiple GPUs with multiple streams
  - Shared memory

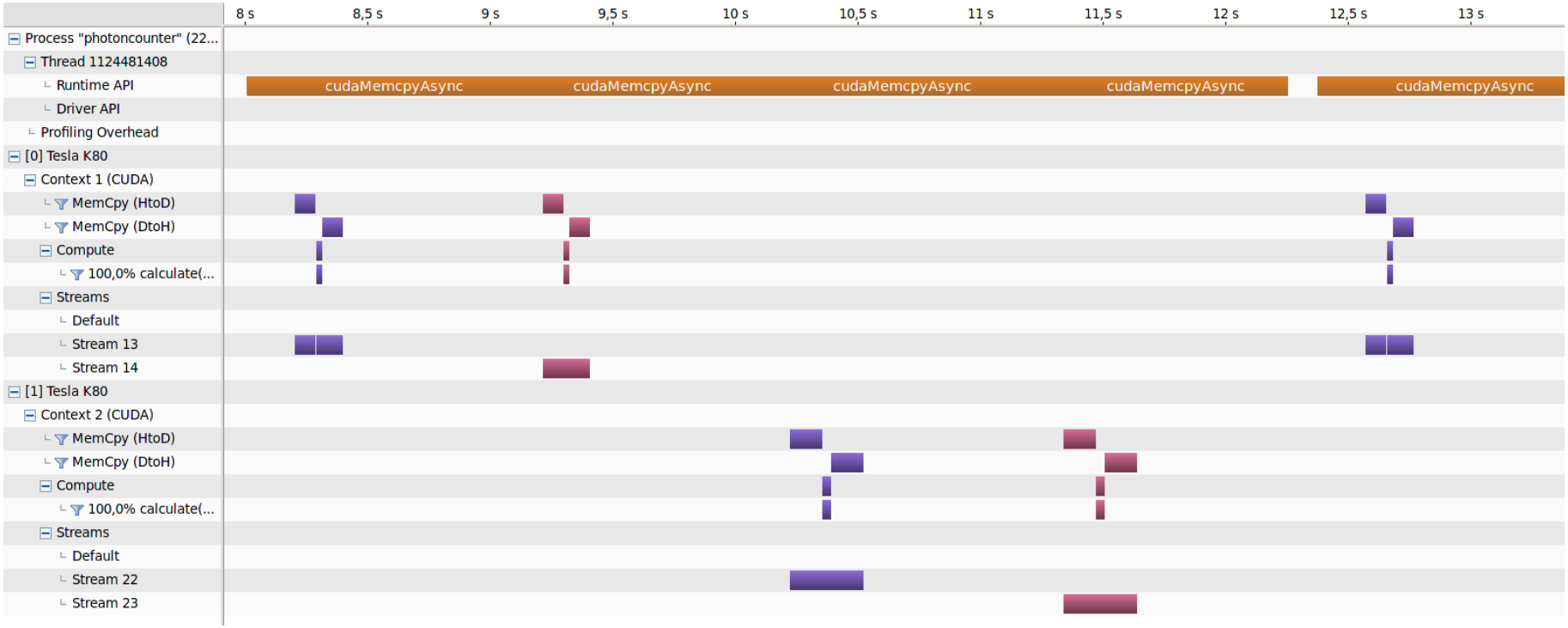
Implementation - Result

Current state:



Implementation - Result

Multiple GPUs:



Implementation - Result

Bandwidth Host to Device:

Name	Start Time	Duration	Size	Throughput
Memcpy HtoD [sync]	1,07 s	2,823 ms	12,583 MB	4,458 GB/s
Memcpy HtoD [sync]	1,073 s	410,202 µs	3,146 MB	7,669 GB/s
Memcpy HtoD [sync]	1,797 s	2,812 ms	12,583 MB	4,474 GB/s
Memcpy HtoD [sync]	1,8 s	412,219 µs	3,146 MB	7,631 GB/s
Memcpy HtoD [async]	3,724 s	84,735 ms	1,049 GB	12,375 GB/s
Memcpy HtoD [async]	4,659 s	84,727 ms	1,049 GB	12,376 GB/s
Memcpy HtoD [async]	8,206 s	84,717 ms	1,049 GB	12,377 GB/s
Memcpy HtoD [async]	9,216 s	84,737 ms	1,049 GB	12,374 GB/s
Memcpy HtoD [async]	12,574 s	84,735 ms	1,049 GB	12,375 GB/s
Memcpy HtoD [async]	13,585 s	84,74 ms	1,049 GB	12,374 GB/s
Memcpy HtoD [async]	16,792 s	84,735 ms	1,049 GB	12,375 GB/s
Memcpy HtoD [async]	17,801 s	84,71 ms	1,049 GB	12,378 GB/s
Memcpy HtoD [async]	21,162 s	84,707 ms	1,049 GB	12,379 GB/s
Memcpy HtoD [async]	22,174 s	84,708 ms	1,049 GB	12,379 GB/s

Implementation - Result

Bandwidth Device to Host:

Name	Start Time	Duration	Size	Throughput
Memcpy DtoH [async]	3,834 s	84,896 ms	1,049 GB	12,351 GB/s
Memcpy DtoH [async]	4,768 s	84,606 ms	1,049 GB	12,394 GB/s
Memcpy DtoH [async]	8,315 s	84,703 ms	1,049 GB	12,379 GB/s
Memcpy DtoH [async]	9,325 s	84,707 ms	1,049 GB	12,379 GB/s
Memcpy DtoH [async]	12,683 s	84,723 ms	1,049 GB	12,377 GB/s
Memcpy DtoH [async]	13,694 s	84,756 ms	1,049 GB	12,372 GB/s
Memcpy DtoH [async]	16,902 s	84,734 ms	1,049 GB	12,375 GB/s
Memcpy DtoH [async]	17,911 s	84,727 ms	1,049 GB	12,376 GB/s
Memcpy DtoH [async]	21,272 s	84,711 ms	1,049 GB	12,378 GB/s
Memcpy DtoH [async]	22,283 s	84,717 ms	1,049 GB	12,377 GB/s

Implementation - Result

Kernel:

Name	Start Time	Duration	Grid Size	Block Size	Regs
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	3,809 s	24,899 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	4,744 s	24,793 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	8,29 s	24,811 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	9,301 s	24,736 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	12,659 s	24,81 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	13,67 s	24,905 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	16,877 s	24,791 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	17,886 s	24,776 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	21,247 s	24,821 ms	[1024,1,1]	[512,1,1]	32
calculate(unsigned int, unsigned short*, double*, unsigned short*, unsigned int, unsigned short*)	22,258 s	24,812 ms	[1024,1,1]	[512,1,1]	32

Implementation - Result

Speedup Shared Memory:

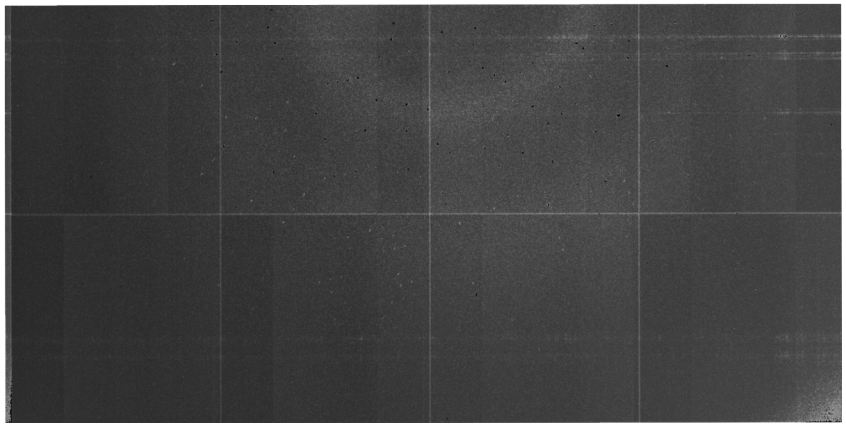
- Global Memory: ~30 ms
- Shared Memory: ~24.7 ms

→ Speedup: ~1.215

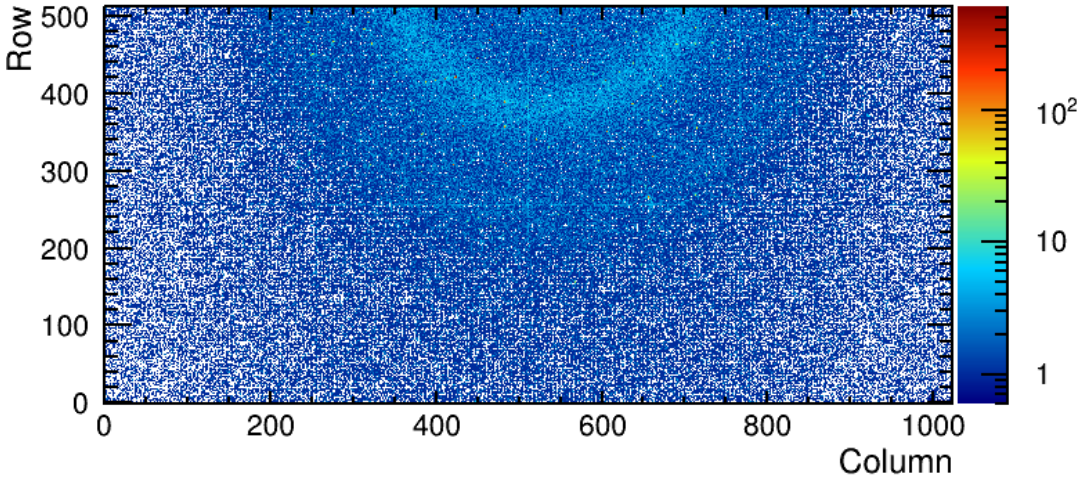
Implementation - Result

Output:

Result



Reference



Questions?