MSE | MASTER OF SCIENCE IN ENGINEERING

Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

Master Thesis

Atlas Tracking Optimization on GPU

Luis Domingues

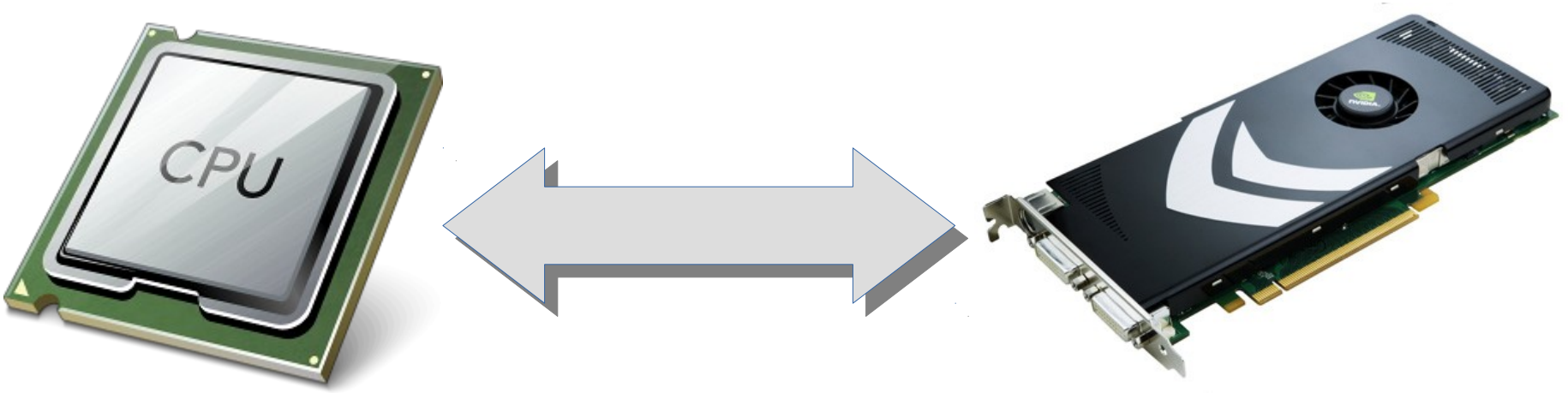
Professor: Frédéric Bapst
Supervisors: Paolo Calafiura
Wim Lavrijsen

Expert: Mathieu Monney

02/25/2015

BERKELEY LAB

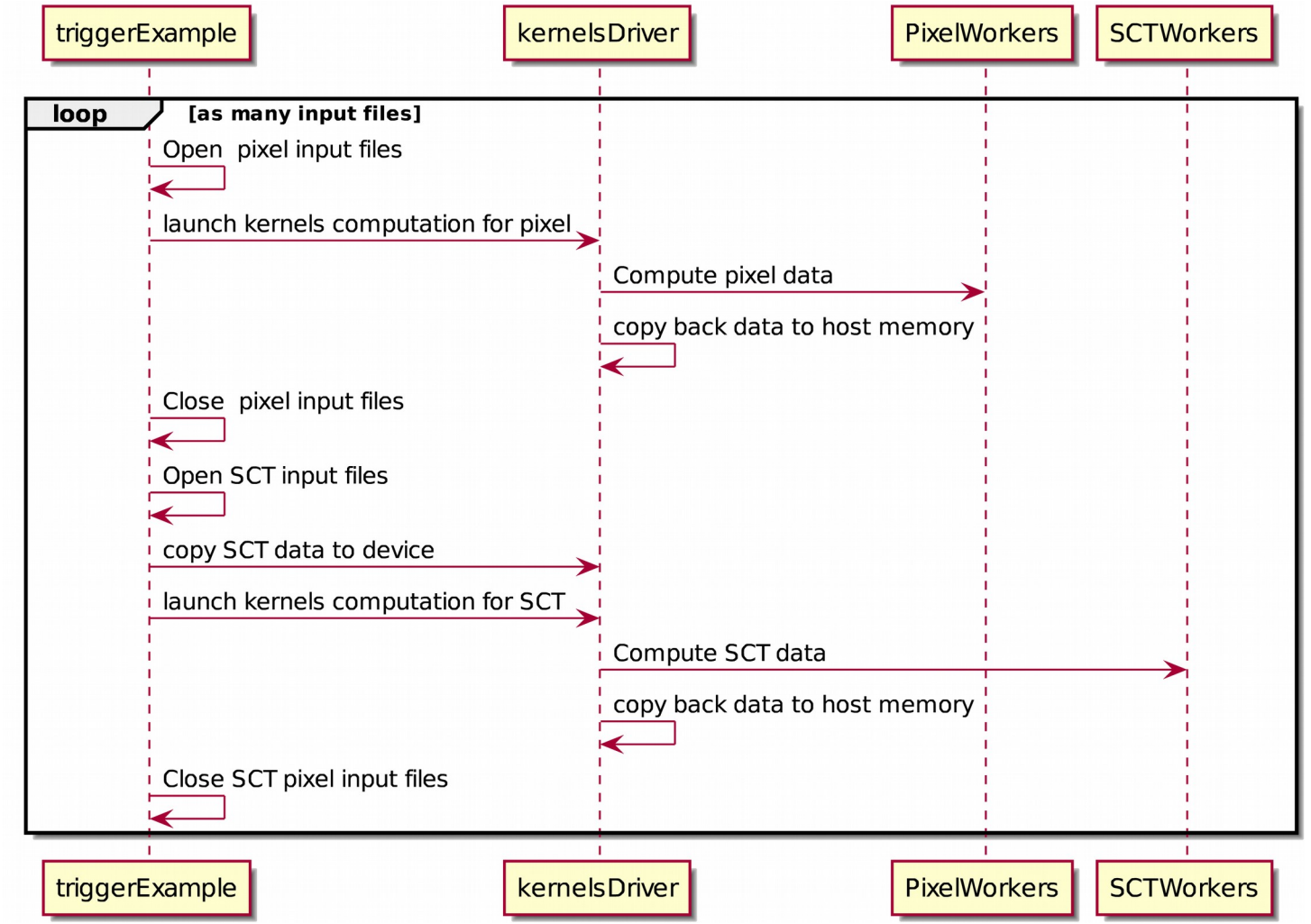
Target



Code we started from

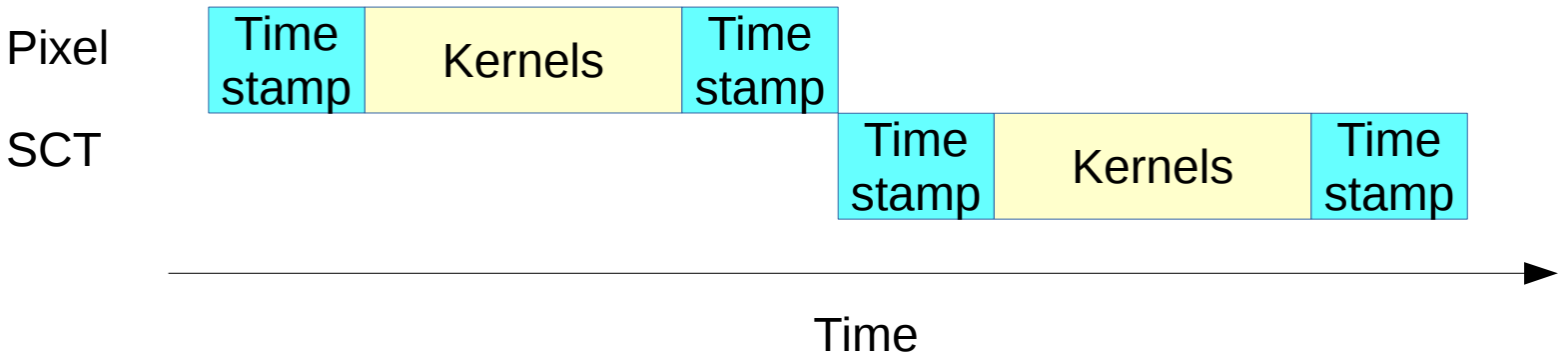
- Demonstrator of ATLAS trigger on GPUs
- Basic host side
  - Take data
  - Send and compute data on GPU
  - Sleep waiting the response

Code we started from

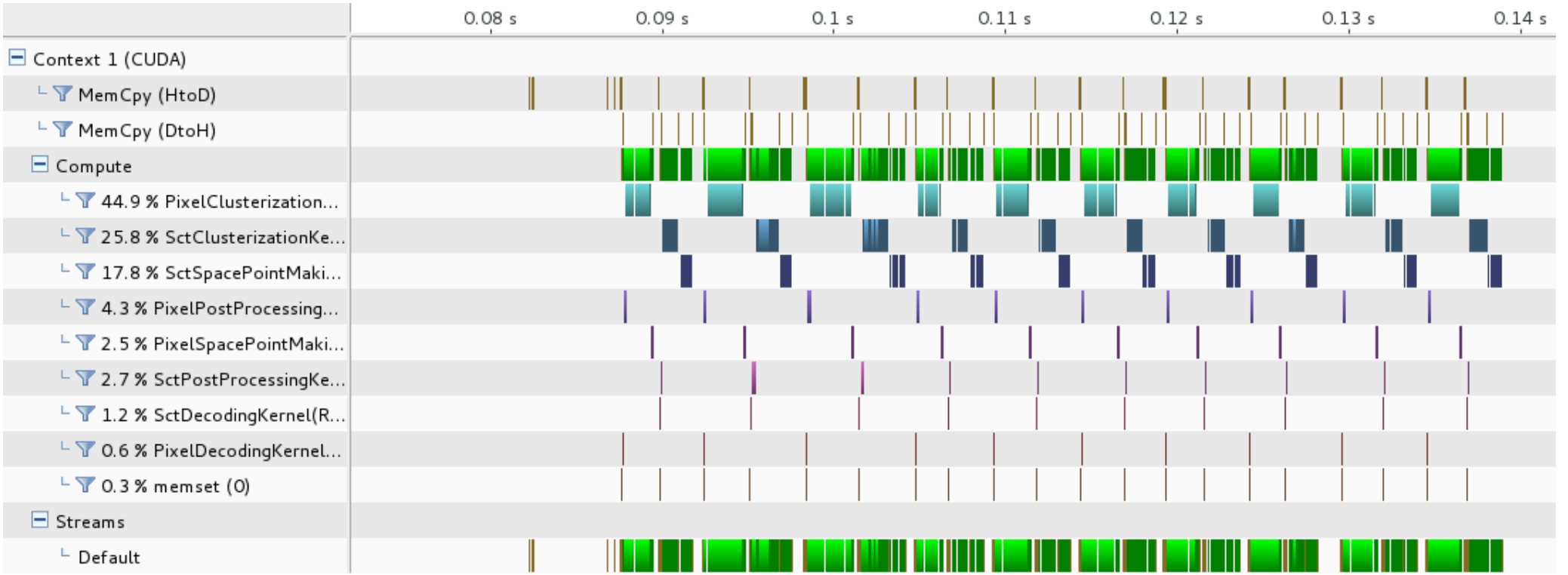


Overlapping pixels and SCT

- The pixel and SCT processing are done in sequence
- Same event, but sequential processing...

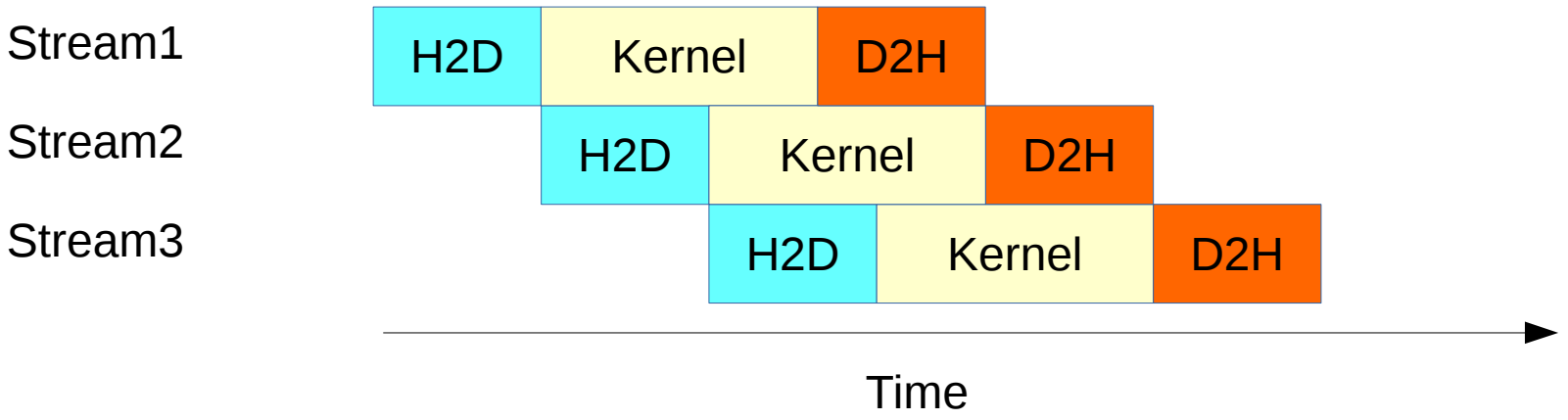


Overlapping pixels and SCT



CUDA Streams

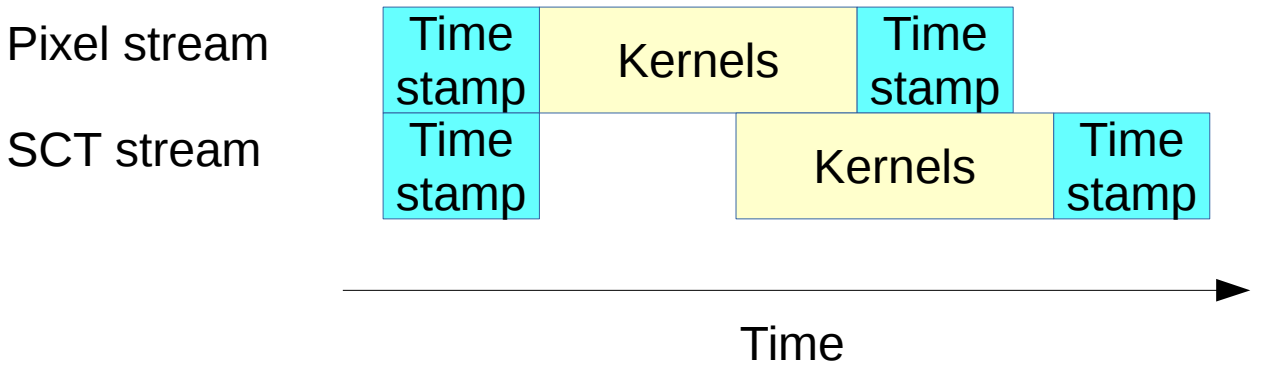
- A stream is a queue of execution
- Non-default streams can be executed in parallel



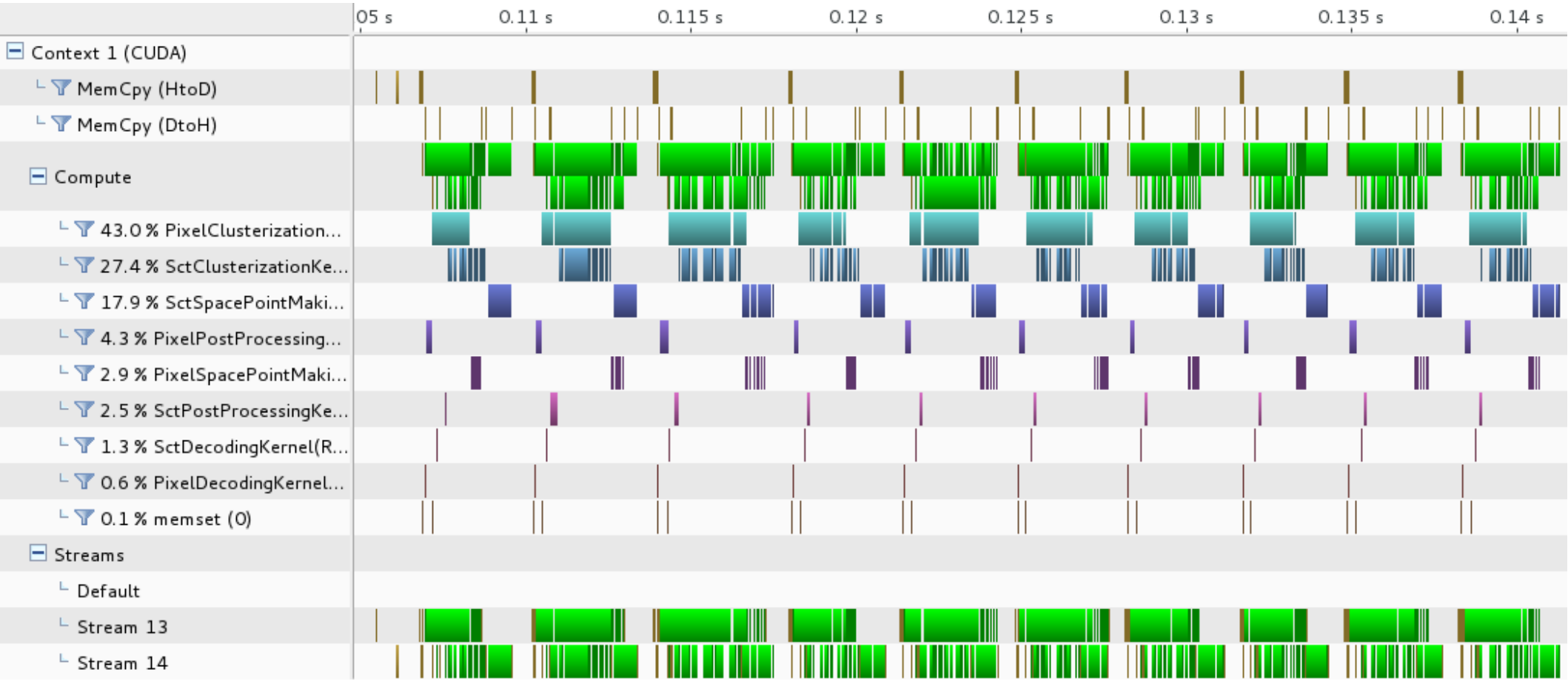
H2D = Host to device transfer
D2H = Device to host transfer

Overlapping pixels and SCT

- Use CUDA Streams
- Start the processing of SCT before pixels end



Overlapping pixels and SCT



Overlapping pixels and SCT

- For 2000 events, without overlapping
  - Avg Pixel: 2.03 ms
  - Avg SCT: 1.95 ms
  - Total avg: 3.98 ms
- For 2000 events, overlapping
  - Avg Pixel: 2.3 ms
  - Avg SCT: 2.5 ms

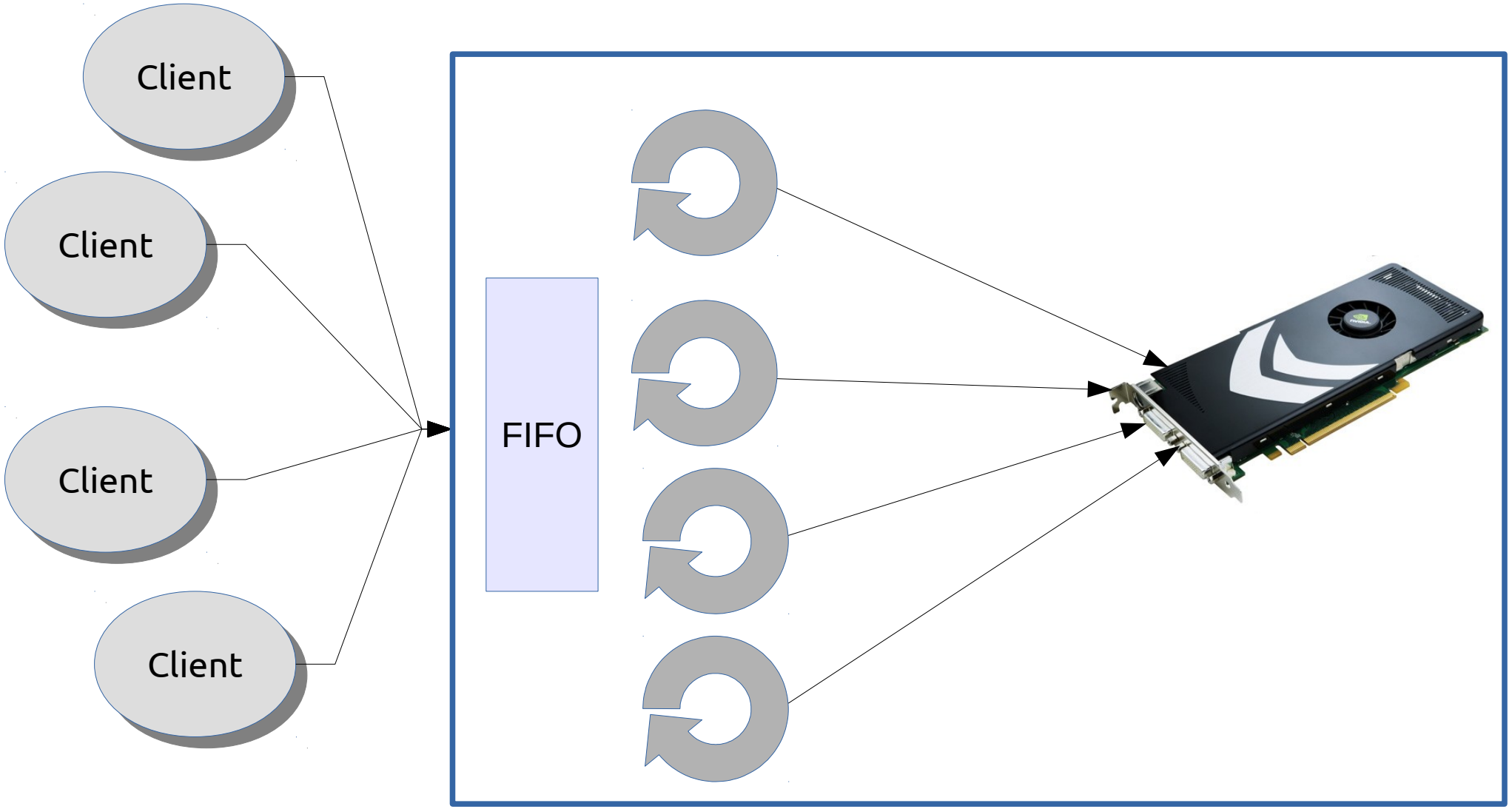
Overlapping pixels and SCT

- Total execution
  - Without overlapping: 8.65 s
  - With overlapping: 6.53 s

Multi-thread server side

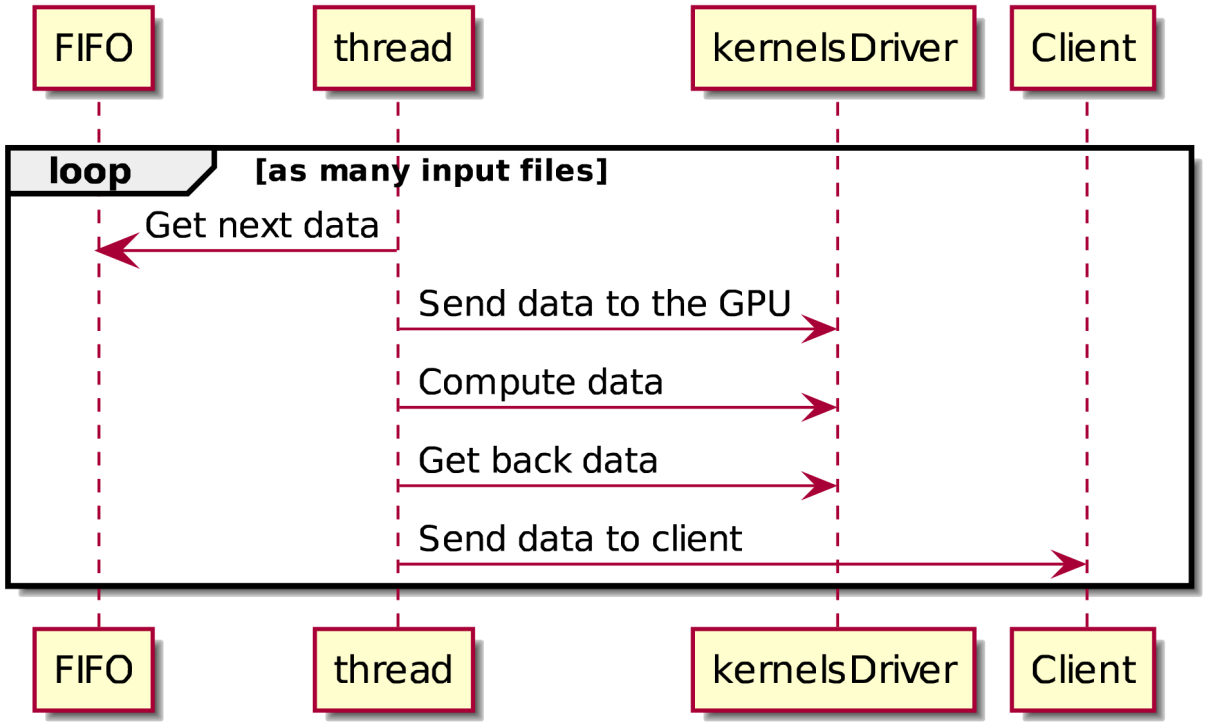
- Huge amount of “small” data
  - They do not fulfill the GPU
- Parallelize the “event” level processing with streams

Multi-thread server side

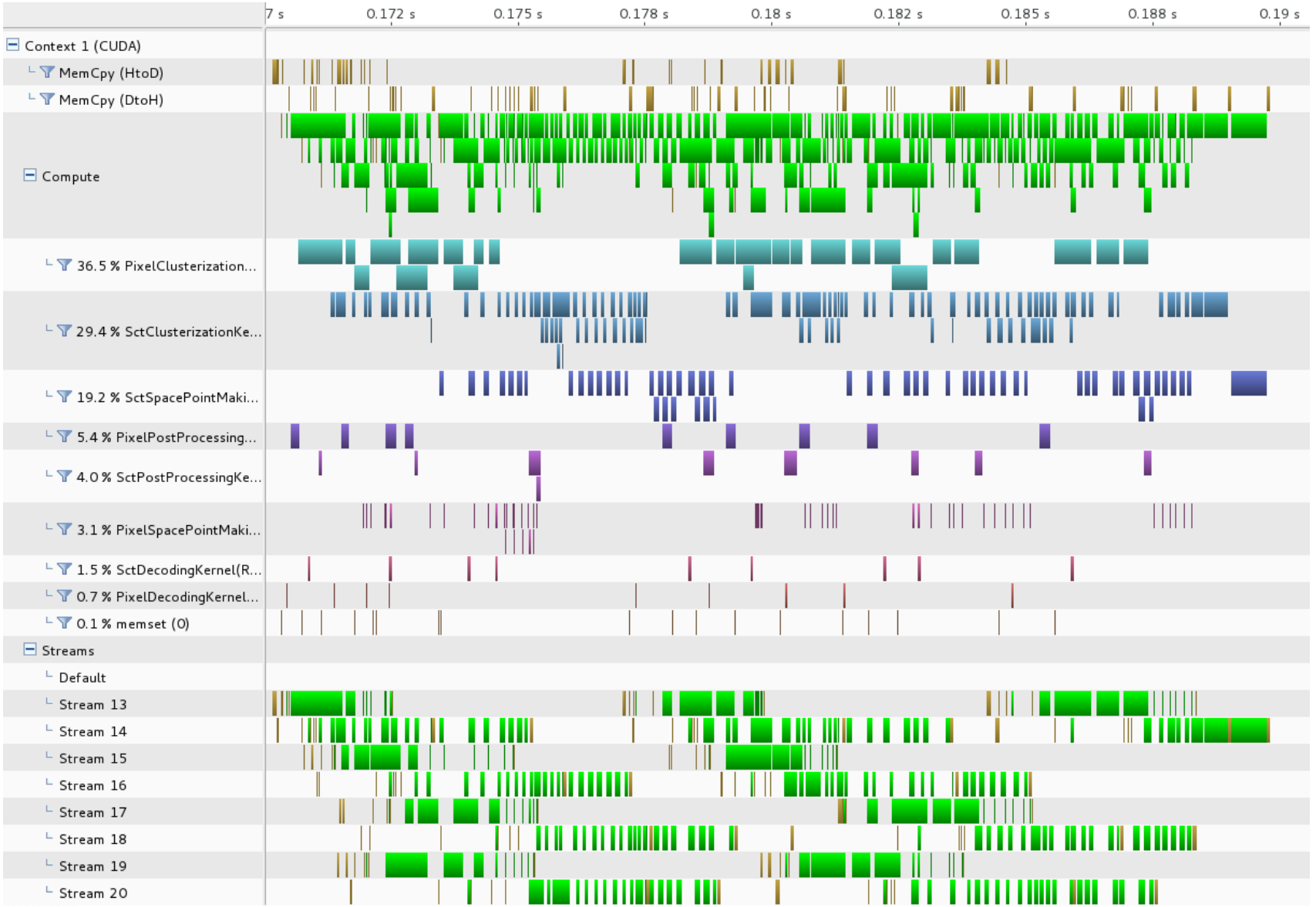


Multi-thread server side

- Life of a thread



Multi-thread server side



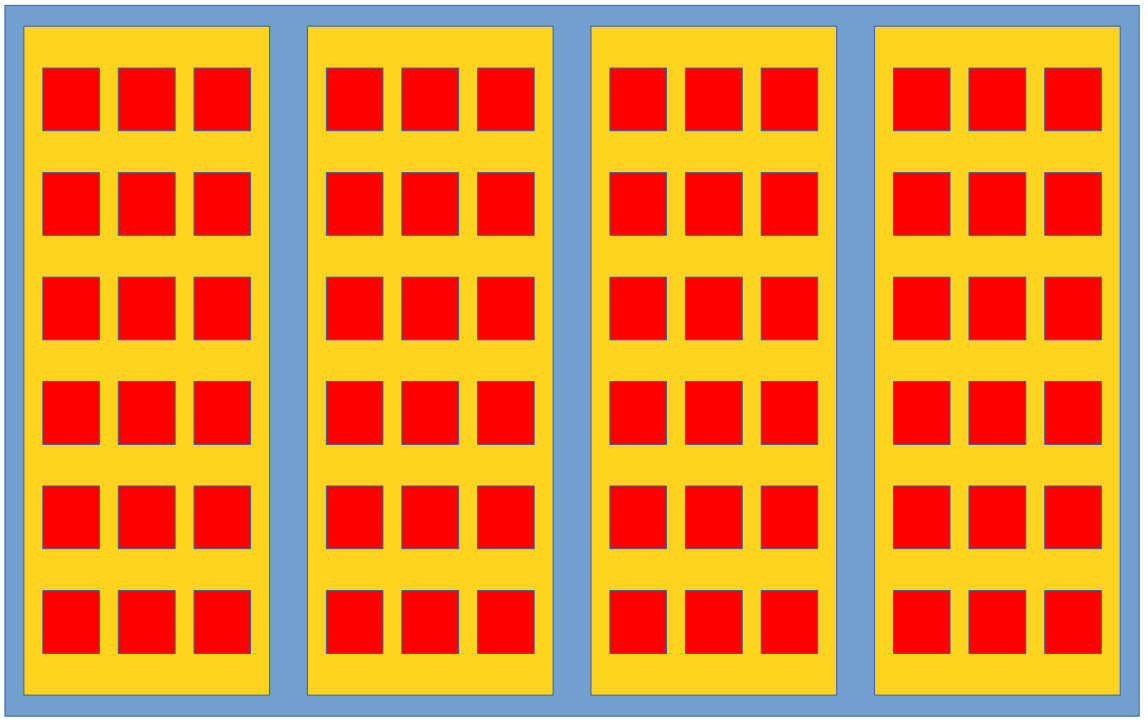
Multi-thread server side

- Executions time
  - Without overlapping: 8.65 s
  - With overlapping: 6.53 s
  - Multi-threading server side: 4.7 s

CUDA Occupancy

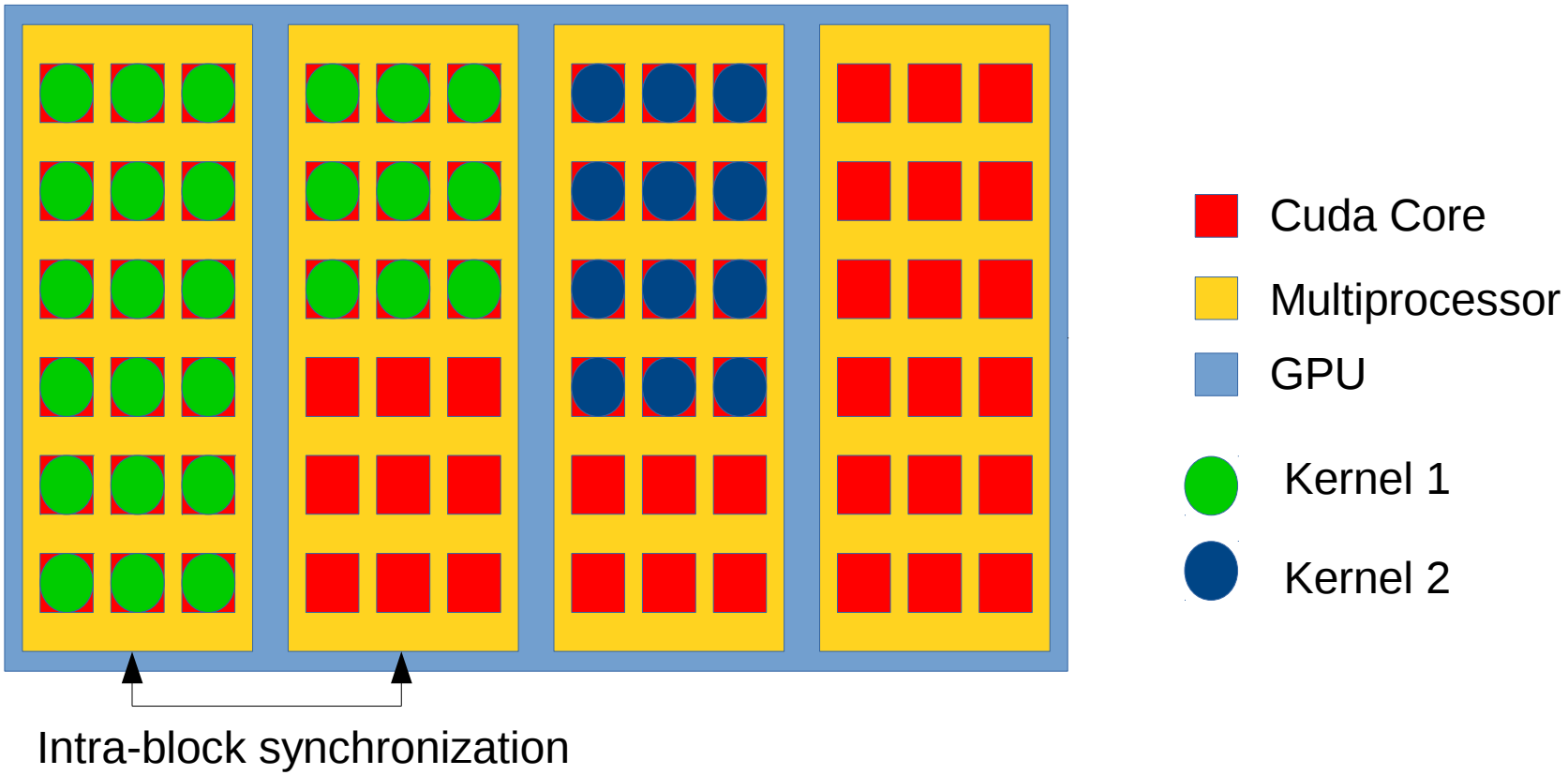
- A good setup of Grid/Block size in card can be significant
- CUDA offers an API to maximize the occupancy of the kernels

CUDA Occupancy



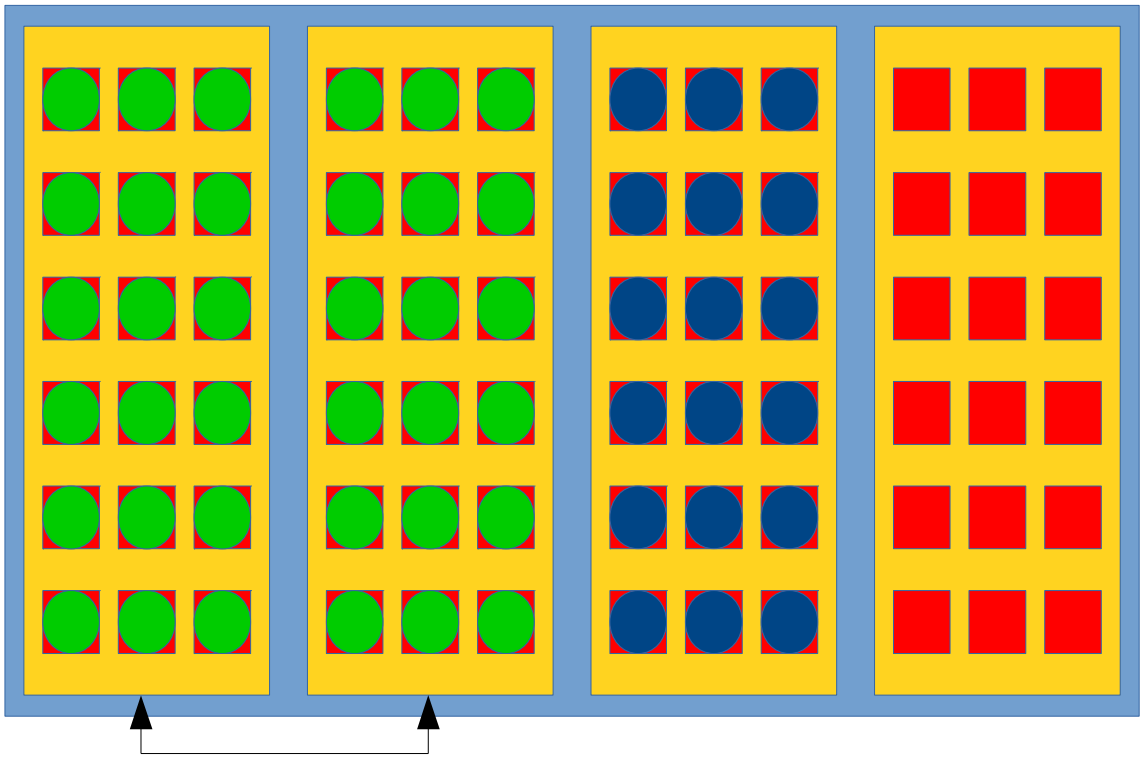
CUDA Occupancy

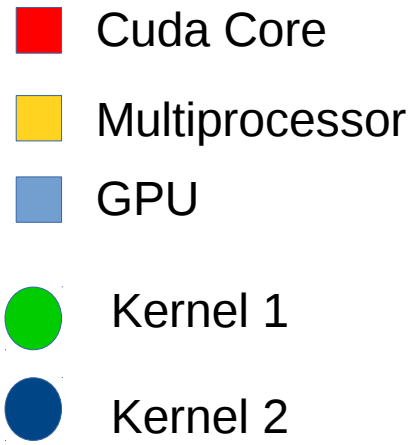
- Bad block size Setup



CUDA Occupancy

- Better block Setup





CUDA Occupancy

- Maximize the occupancy kills global performances
- Runs results for 2000 events
  - Big Blocks size: 10.88 s
  - Original configuration: 4.7 s
  - Small blocks size: 4.4 s

CUDA Occupancy

- Maximize the occupancy kills global performances
- Runs results for 2000 events
  - Big blocks size: 3 kernels in parallel (Max 5)
  - Small blocks size: 4 kernels in parallel (Max 7)

Conclusion

- Important points when using a GPU
  - Port of an algorithm to the GPU
  - Communicate with the GPU
  - Host side design

- Keep the GPU busy
- Big occupancy does not allow the GPU to schedule its tasks efficiently