GPU integration in the Atlas framework

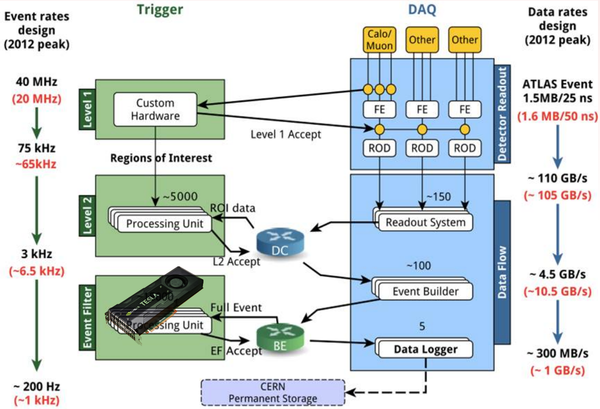
M. Bauce, A. Messina,
S. Giagu, M. Rescigno

January 12, 2014

SAPIENZA
UNIVERSITÀ DI ROMA

GAP RT

Atlas High Level Trigger

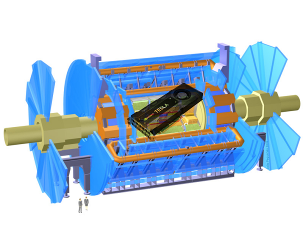


- Atlas has implemented a three stage Trigger and DAQ system
- Run II data taking conditions will be demanding for the data processing:
  - Considering new technologies to include after the upgrade
  - GPUs are good candidates to be exploited in L2/EF trigger reconstruction

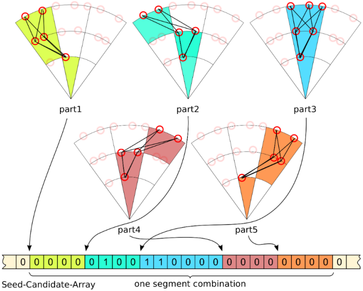
Atlas is interested in this R&D activity:

- Possibility to join a proposal for GPU application in Phase 2 Atlas High Level Trigger

Atlas GPU-related activities



Track seeding

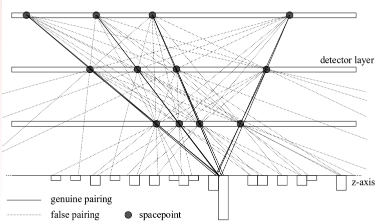


0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Seed-Candidate-Array

one segment combination

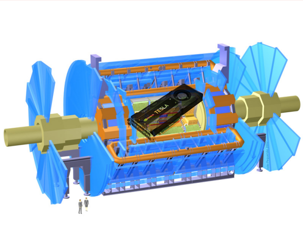
Z-Fider algorithm



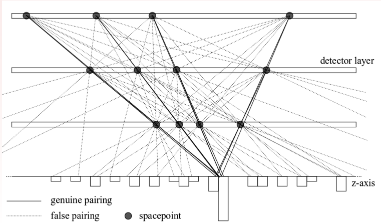
Several improvements from the implementation of parallel computing devices:

- Vertex Position (∼35x)
- Track-seeds identification (∼50x)
- Track Fitting (∼5x)

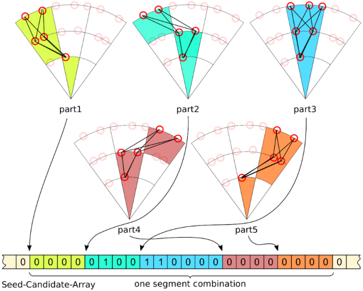
Atlas GPU-related activities



Z-Fider algorithm



Track seeding



0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

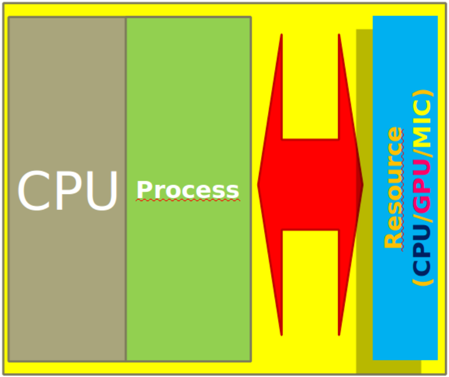
Several improvements from the implementation of parallel computing devices:

- Vertex Position (∼35x)
- Track-seeds identification (∼50x)
- Track Fitting (∼5x)

Crucial point is the homogeneous implementation of this devices in the Atlas DAQ and Processing framework.

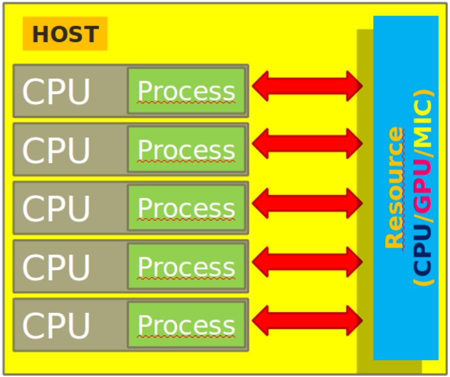
Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?



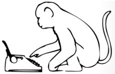
Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?



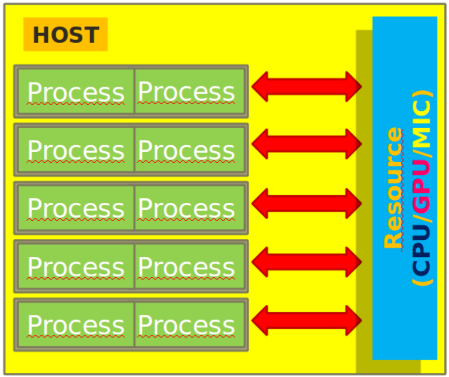
- Atlas is using multi-process model
  - Each process is unaware of the others

APE project



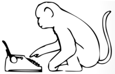
Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?



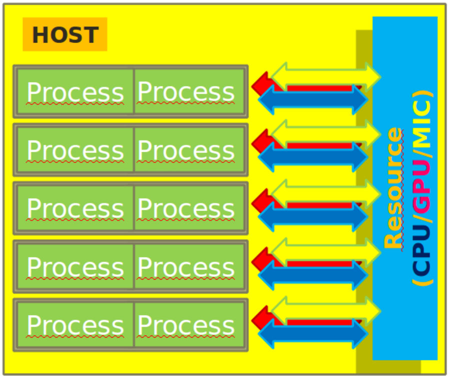
- Atlas is using multi-process model
  - Each process is unaware of the others
  - Each process access resources as a owner

APE project



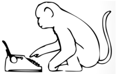
Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?



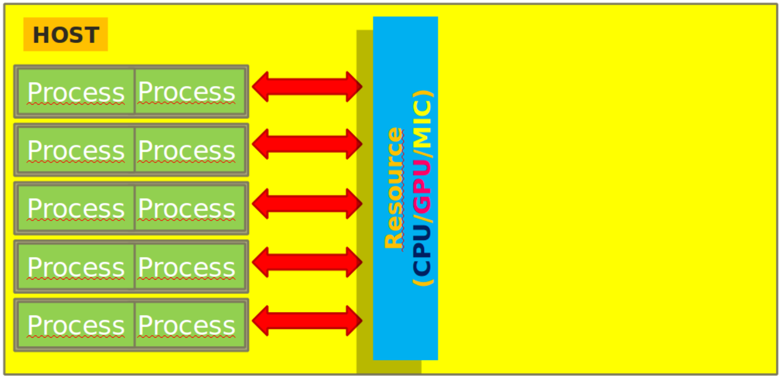
- Atlas is using multi-process model
  - Each process is unaware of the others
  - Each process access resources as a owner
  - Resources have their own code, pattern, algorithms

APE project



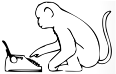
Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?



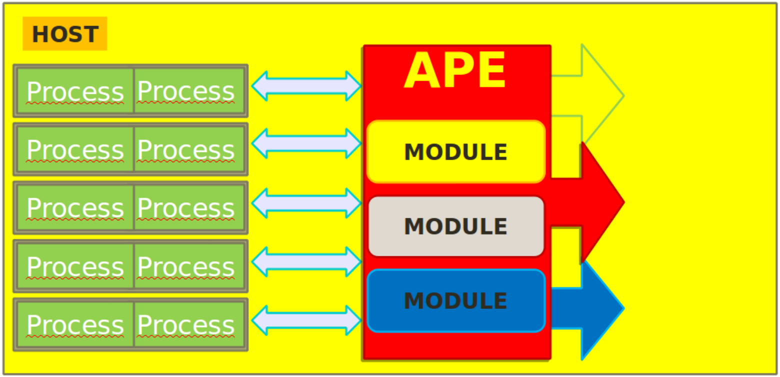
- Investigated solution: adopt a **Client-Server** architecture

APE project



Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?

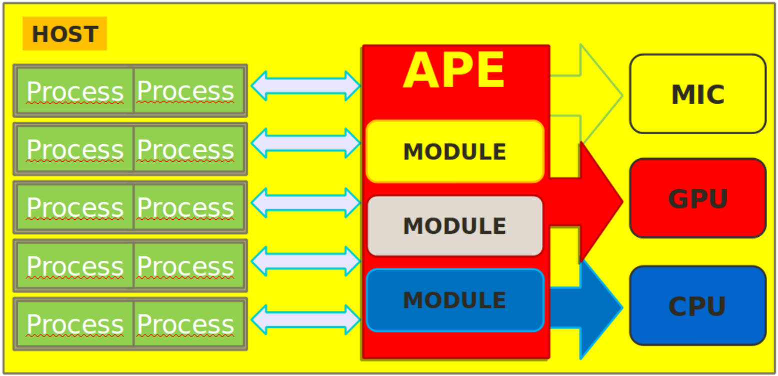


- Investigated solution: adopt a **Client-Server** architecture
  - Accelerator Process Environment modules:
    manage resources, group, schedule.

APE project

Atlas data analysis framework: ATHENA

- How to implement the Host-Device interaction for parallel computation in Atlas?

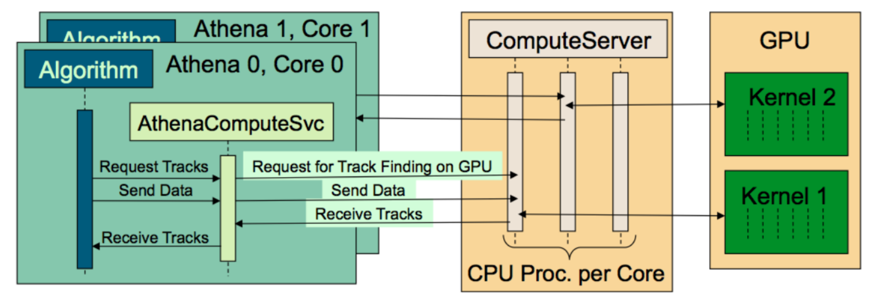


- Investigated solution: adopt a **Client-Server** architecture
  - Accelerator Process Environment modules:
    manage resources, group, schedule.
  - Flexible and compatible with different kind of computational devices.

APE project

Framework Architecture Implementation

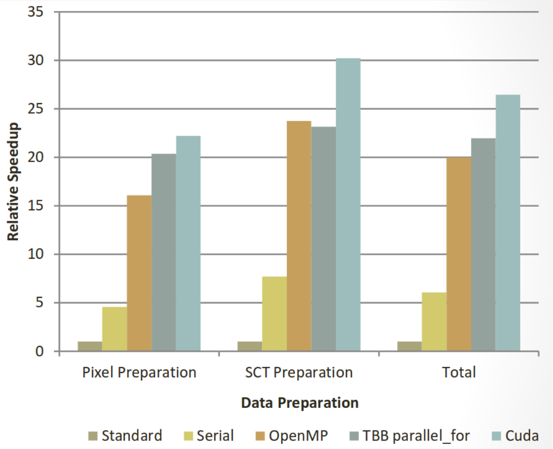
- Atlas ongoing development, in contact with the group that is willing to support us.



- **AthenaCompute SVC**: *patch* to include in Athena code, contains instruction for data formatting and trasfer
- **Compute Server:** manage all the query for parallel-computing
- **GPU-device:** contains instructions and CUDA kernels to be executed

Timing tests

- Client-Server architecture tested and properly working on some trigger algorithms
- IPC overhead is negligible compared to the improvements
  - ∼6× speedup with no additional optimization



CPU i7-2760QM2.40GHz 4 Cores (8HT) Cuda on GPU Tesla C2050 (Dedicated code)

Credits to S.Kama, D. Emeliyanov

Currently ongoing and prospects

Athena interface

Need to customize a pre-existing version (developed for different tasks)

APE server

Shared versione already available.
Tested and costantly improving.

GPU algorithms

Convert serial trigger algorithms into CUDA Kernels

Summary and Conclusions

- Atlas is putting effort to include multithread computation devices in the DAQ/Trigger system:
  - GPUs are one of the technologies to investigate
- Implementation in the current framework through a *layer*:
  - already developed, tested and promising
  - developers are supporting us with the potential technical issues

Middle-term roadmap:

1. Setup and test the interface between Athena and GPU (dummy algorithms)
2. Work on the parallelization of L2-Muon algorithms (see next talk)
3. Perform benchmark measurements and compare with similar studies

BACKUP

GAP in Atlas Roma1

- Software Trigger Case Study: **Atlas Muon High Level Trigger**
  - Parallelization of the `MuComb` and `MuIso` trigger algorithms
  - Investigate the improvements from parallel computation, in particular in the high-luminosity regime
  - Improve the parameter resolution, to increase efficiency/purity of the selections

- Interesting opportunity:
  - Long-time involvement of the group in the Atlas Muon HLT
  - Profit from the existing expertise and know-how
  - Strenghten our role for future projects *(upgrade)*

Available in Rome

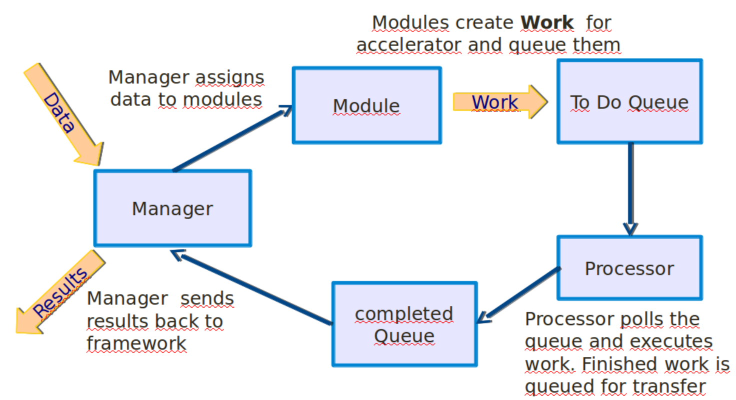
Setup a machine:

- CPU: Intel Xeon E5-2620
- GPU: Nvidia GTX Titan

Will be used for:

- Trigger performance tests
- Medical imaging processing

Processing Data Flow



Modules create **Work** for accelerator and queue them

Manager assigns data to modules

Data

Module

Work

To Do Queue

Manager

Processor

Results

Manager sends results back to framework

completed Queue

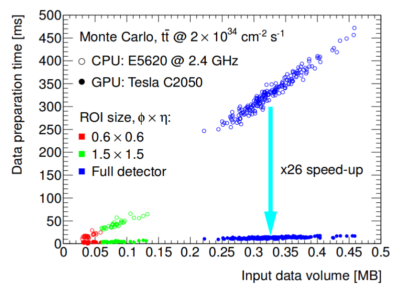
Processor polls the queue and executes work. Finished work is queued for transfer

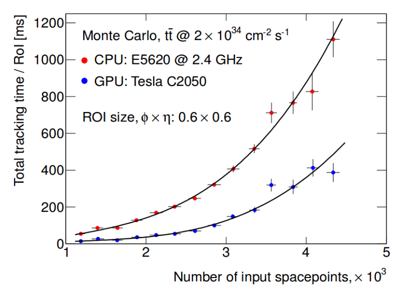
- Inter Process communications using yampl library: fast communication and support network transfers.
- Data converted to pure C-structures (no Athena data-model) and passed to APE-server
- Server-client communication through shared cache memory (may evolve in the future)

Atlas processing improvements

Data Preparation: conversion from detector bytestream to spacepoints (lightweight detector geometry for GPU)

Tracking: Track seeding, extrapolation, merging (`SiTrack` alg.)


Data preparation time [ms]
Monte Carlo, tt̄ @ 2 × 10^34 cm^-2 s^-1
○ CPU: E5620 @ 2.4 GHz
● GPU: Tesla C2050
ROI size, ϕ × η:
0.6 × 0.6
1.5 × 1.5
Full detector
x26 speed-up
Input data volume [MB]


Total tracking time / RoI [ms]
Monte Carlo, tt̄ @ 2 × 10^34 cm^-2 s^-1
● CPU: E5620 @ 2.4 GHz
● GPU: Tesla C2050
ROI size, ϕ × η: 0.6 × 0.6
Number of input spacepoints, × 10^3