Welcome
Thoughts & wishlist

Jérôme LAURET,

BROOKHAVEN NATIONAL LABORATORY

Welcome …

- 3rd workshop of its kind, follow-on workshop after the successful events at GSI (2010) and CERN (2011)
- These workshops are focused on event reconstruction *and analysis* in the High Level Trigger of modern experiments, especially those built for heavy-ion interactions by adapting the software to modern many-core CPU/GPU computer architectures
  - Working from the core-tasks from HENP communities (tracking, analysis being the biggest task) finding a solution and working outward
- The workshop is supported by the Goethe University of Frankfurt, Frankfurt Institute for Advanced Studies (FIAS), Hessischen Ministerium für Wissenschaft und Kunst, Helmholtz International Center for FAIR (HIC for FAIR), GSI Helmholtzzentrum für Schwerionenforschung and the CBM experiment.
  - Chairs: Prof. Kisel, Ivan, Prof. Hoehne, Claudia, Dr. Jarp, Sverre, Dr. Lauret, Jerome

Other workshops trying to address the multi-core, parallelization problem

At this stage of maturity, we need to pass information across … and broaden the base …

ACAT 2011 – not a commercial for it (but in my view, very relevant)

- ACAT: Advanced Computing and Analysis Techniques in Physics Research
- A workshop with many interests, many fields:
  - Track 1: Programming languages, software quality, IDE and User Interfaces / Distributed and parallel computing / New architectures, many-cores / Virtualisation / Online Monitoring and Control, HLT
  - Highlights / plenary – Sverre Jarp "*Where do we go from here? – The next phase of computing in HEP*", Kate Keahey "*Building an outsourcing ecosystem for science*", Patrick Fuhrmann "*NFS 4.1/pNFS, the final step*", Anar Manafov "*Computing On Demand: Analysis in the Cloud*", David De Route "*SALAMI project*"
- Strong focus on cross-technology and solutions
  - **Not only multi-core counts but how does it fits with IO, Grids, Cloud, other fields data mining …**
  - Lots of conclusions from there …

What did we hear from the start?

- From Sverre → setting the moc
  - Has been warning us for a while to get prepared
    - The "7 dimension" of speed increase
      - **Multiple computer nodes** – embarrassingly parallel
      - **Multi-core** – one program uses as many cores
      - **Multi-sockets** – provides more hardware parallelism (but hard to program due to NUMA, Non-Uniform Memory Access)
      - **Pipelining** – instruction pipelining
      - **Superscalar** (MIMD Multiple Instructions, Multiple Data)
      - **Vector widths/SIMD** (Single Instructions, Multiple Data)
      - pseudo-dimension – **hardware multi-threading**
    - **Perhaps even an 8th :** Precision – quadruple precisions and beyond …

  - Hardware vectors keeps growing – community not using much of it. Assumed to be 10% at most of a machine capability – **Many trying but beating one or couple dimension only – collaborative effort across the dimensions**

- Tera-Flops machine and Exa-scale coming and many architecture and hardware believed to become tomorrow commodities (at least some of it): Xeon, Atom, Tilera, … CPU, GPU, …

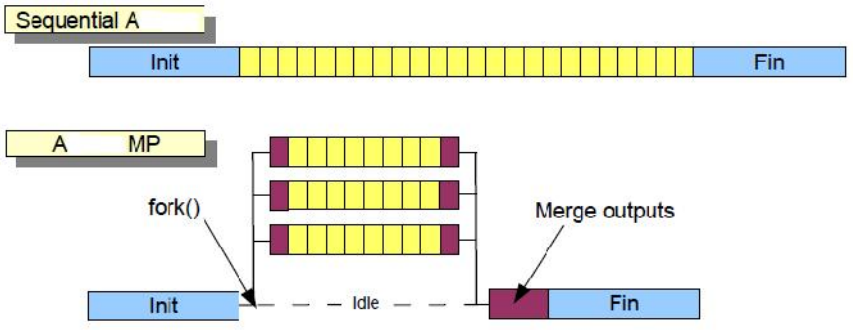
What are the tools? APIs?

- Vector Class helps – this comes for "free" and should (**MUST**) be used (SIMD)
- Many API and approach
  - Old fork() or POSIX threads methods
  - OpenMP, Threading Building Blocks (TBB),
    … GPU – CUDA,
    … OpenCL, (lots of promises here … whole slew of needed functionalities)
    … Intel Cilk (array notations ↔ data parallelism, serial semantic, forward scaling, ..)
    - Not all methods are compatible with each other
    - thread synchronization issues

Very complex problem with many dimensions (hardware and APIs)

Not only a broad set of programing talents would be needed but the plethora of (a) API (b) architectures and devices and (c) strategies (Grid, Cloud, serial) and hidden layers (IO, …) tend to indicate the **need for a very broad collaborative effort and set of skills**

How the efforts in the tranches looks like …



- "Sequentiality" of a workflow (reconstruction) is killing everyone
  - Do we have a wining strategy? Amdahl Law (oft forgotten) is merciless in that regard.
  - Even IO can kill the whole effort – some already splitting workflows

- Lots of (heroic) efforts to speed up framework – heard factors of x4, x10, x200
  - This shows inconsistent comparative measurements and metrics across efforts/experiments
  - **We must learn on how to present our results consistently** – chase for the largest speed up does not help
  - **%tage gain should be CAREFULLY stated (memory copy overhead, comparative to the whole workflow) or numbers are not meaningful – we need to define a standard metric**

- Valuable investigative work– Where is this going? How to make this all work together?
  - Solutions from within the same experiment used fork(), OpenCL, OpenMP, CUDA, …
  - **It is good that many are investigating diverse approach – learning phase**
  - **However, no common code & efforts for CPU / GPU tracking.**

- A lot of step back from STL, C++ types for C-like - Provocative but … **Did we take the wrong turn with C++?**
  - Good mileage in software development in one hand BUT systematically appears and *presented as a show stopper to exploiting new architecture*
  - **Feel is that we need for "a" new language?**

Focus in the community

- We have enough experience in the community from all the tries and tests – we need to broaden knowledge. **Workshops, training, education, …**
- We do NOT have a global strategy across and within experiments – **where are our (your) software architects?**
- Improvements over standard approach not always clear – we MUST have standard metrics and consistent measurements to have meaningful comparisons**: we need to present and state:**
  - **relative %tage gain, absolute %tage gain over entire workflows and ALWAYS consider the time to copy data in/out of memory (or specify what enters in the "gain")**
  - **We may need to test on the same platform – is it "forward" to suggest a common test-bed to achieve consistent comparative results?**
- **We have to pay attention to new languages**
- **We MUST work together, develop common strategies**

Wish list for this workshop
What can we do?

- A very success set already – Pulling tracking interests from many experiments together.
  - **We have not yet achieved a common software/packages but getting there (CA package, KFParticles, …) - hope for a common "core" (documentation, coding standards, data structures, interface specification, …)**
- We should (**MUST**) discuss the long term maintainability and availability of packages (Vc, OpenCL, TBB?, …) and attempt to narrow down on strategies and evaluate timelines
  - **ROOT: Would the ROOT team support & maintain Vc? If so, when? We all rely on ROOT and need a reliable, sustainable, long term software provisioning mechanism – what else is "cooking"?**
  - **Intel: OpenCL – are all the standards converging? Is the OpenCL 1.2 specifications near final? Is this the preferred way? How does Cilk fits into this? Any other "goodies" / good tips / pearl of wisdom?**
- How to consolidate the "base" / expand / broaden the acceptance / coordinate with other project's development (ROOT, Geant, …). This means that to win, we MUST
  - **… achieve compromise across** – yes, you may prefer "that" other package … or "that" other approach
  - **… organize MORE workshops where we invite each other and keep each other's informed**
  - **An over-arching coordination may be needed (a global working group?)**
  - **Always need to bring industry – many thanks to the representative from past and current workshop**

Hope for a great and productive workshop … and let's the fun begin