LHCb Plans for Concurrency

M. Clemencic on behalf of the LHCb Collaboration

February 4, 2013

Outline

Outline

Hypothesis on the Future

- Golden Era of Moore Law is over
  - Power limitations
  - Topology limitations
- Silicon manufacturers getting imaginative
- Transistors organized in new configurations
  - Many small cores
  - Non-trivial memory bus (NUMA)
  - Specialized cores (GPUs)
  - Low power SOCc (ARM)
- Some old trends still pursued
  - Larger vector units
  - New, more complex instructions
  - More registers
  - Hardware threads

Hypothesis on the Future

- Golden Era of Moore Law is over
  - Power limitations
  - Topology limitations
- Silicon manufacturers getting imaginative
- Transistors organized in new configurations
  - Many small cores
  - Non-trivial memory bus (NUMA)
  - Specialized cores (GPUs)
  - Low power SOCc (ARM)
- Some old trends still pursued
  - Larger vector units
  - New, more complex instructions
  - More registers
  - Hardware threads

Hypothesis on the Future

- Golden Era of Moore Law is over
  - Power limitations
  - Topology limitations
- Silicon manufacturers getting imaginative
- Transistors organized in new configurations
  - Many small cores
  - Non-trivial memory bus (NUMA)
  - Specialized cores (GPUs)
  - Low power SOCc (ARM)
- Some old trends still pursued
  - Larger vector units
  - New, more complex instructions
  - More registers
  - Hardware threads

Hypothesis on the Future

- Golden Era of Moore Law is over
  - Power limitations
  - Topology limitations
- Silicon manufacturers getting imaginative
- Transistors organized in new configurations
  - Many small cores
  - Non-trivial memory bus (NUMA)
  - Specialized cores (GPUs)
  - Low power SOCc (ARM)
- Some old trends still pursued
  - Larger vector units
  - New, more complex instructions
  - More registers
  - Hardware threads

Getting Ready

- No real limitation hitting us yet
  - Old batch-style processing still feasible
- But we are wasting CPU cycles
- Memory limitations will happen (most probably)
- Need to address concurrency at several levels

Getting Ready

- No real limitation hitting us yet
  - Old batch-style processing still feasible
- But we are wasting CPU cycles
- Memory limitations will happen (most probably)
- Need to address concurrency at several levels

Getting Ready

- No real limitation hitting us yet
  - Old batch-style processing still feasible
- But we are wasting CPU cycles
- Memory limitations will happen (most probably)
- Need to address concurrency at several levels

Getting Ready

- No real limitation hitting us yet
  - Old batch-style processing still feasible
- But we are wasting CPU cycles
- Memory limitations will happen (most probably)
- Need to address concurrency at several levels

Outline

Multiprocessing

- Different possible approaches
  - Batch-like (independent processes)
  - Intercommunicating processes
  - Forking/cloning
  - Mixture
- Easier migration (no shared states)
- Not really efficient memory usage
- Communication overhead
- GaudiMP ready for LHCb (Nathalie's talk)

Multiprocessing

- Different possible approaches
  - Batch-like (independent processes)
  - Intercommunicating processes
  - Forking/cloning
  - Mixture
- Easier migration (no shared states)
- Not really efficient memory usage
- Communication overhead
- GaudiMP ready for LHCb (Nathalie's talk)

Multiprocessing

- Different possible approaches
  - Batch-like (independent processes)
  - Intercommunicating processes
  - Forking/cloning
  - Mixture
- Easier migration (no shared states)
- Not really efficient memory usage
- Communication overhead
- GaudiMP ready for LHCb (Nathalie's talk)

Multiprocessing

- Different possible approaches
  - Batch-like (independent processes)
  - Intercommunicating processes
  - Forking/cloning
  - Mixture
- Easier migration (no shared states)
- Not really efficient memory usage
- Communication overhead
- GaudiMP ready for LHCb (Nathalie's talk)

Multithreading

- Different levels
  - Event
  - Algorithm/Module
  - Sub-algorithm/module
- More efficient memory usage
- More difficult (shared memory)
- Integration of Whiteboard demonstrator in Gaudi (Benedikt's talk)

Multithreading

- Different levels
  - Event
  - Algorithm/Module
  - Sub-algorithm/module
- More efficient memory usage
- More difficult (shared memory)
- Integration of Whiteboard demonstrator in Gaudi (Benedikt's talk)

Multithreading

- Different levels
  - Event
  - Algorithm/Module
  - Sub-algorithm/module
- More efficient memory usage
- More difficult (shared memory)
- Integration of Whiteboard demonstrator in Gaudi (Benedikt's talk)

Multithreading

- Different levels
  - Event
  - Algorithm/Module
  - Sub-algorithm/module
- More efficient memory usage
- More difficult (shared memory)
- Integration of Whiteboard demonstrator in Gaudi (Benedikt's talk)

Vectorization

- One math operation per instruction $\rightarrow$ waste of transistors
- Different approaches
  - Compiler intrinsics
  - Auto-vectorization (icc, gcc 4.7, ...)
- Too technical for the average developer
- Should be hidden in libraries
  - Matrix operations (ROOT)
  - Helpers to produce vectorizable code

Vectorization

- One math operation per instruction $\rightarrow$ waste of transistors
- Different approaches
  - Compiler intrinsics
  - Auto-vectorization (icc, gcc 4.7, ...)
- Too technical for the average developer
- Should be hidden in libraries
  - Matrix operations (ROOT)
  - Helpers to produce vectorizable code

Vectorization

- One math operation per instruction $\rightarrow$ waste of transistors
- Different approaches
  - Compiler intrinsics
  - Auto-vectorization (icc, gcc 4.7, ...)
- Too technical for the average developer
- Should be hidden in libraries
  - Matrix operations (ROOT)
  - Helpers to produce vectorizable code

Accelerators

- Lot of fuss about GPUs and MIC
- Targeting controlled hardware environments
- We cannot control the hardware
- We need reproducible results (on CPUs too)
  - OpenCL may help
- Explicit memory management and small bandwidth
- Different approaches
  - One operation on several events
  - Several operations on one event
  - A mixture

Accelerators

- Lot of fuss about GPUs and MIC
- Targeting controlled hardware environments
- We cannot control the hardware
- We need reproducible results (on CPUs too)
  - OpenCL may help
- Explicit memory management and small bandwidth
- Different approaches
  - One operation on several events
  - Several operations on one event
  - A mixture

Accelerators

- Lot of fuss about GPUs and MIC
- Targeting controlled hardware environments
- We cannot control the hardware
- We need reproducible results (on CPUs too)
  - OpenCL may help
- Explicit memory management and small bandwidth
- Different approaches
  - One operation on several events
  - Several operations on one event
  - A mixture

Virtualization? Clouds?

- Virtualized computing centers more and more common
- Hardware abstraction
  - Vectorization still works
  - Memory bus layout?
  - Accelerators?
- Which of the parallelization exercises does make sense?

Virtualization? Clouds?

- Virtualized computing centers more and more common
- Hardware abstraction
  - Vectorization still works
  - Memory bus layout?
  - Accelerators?
- Which of the parallelization exercises does make sense?

Virtualization? Clouds?

- Virtualized computing centers more and more common
- Hardware abstraction
  - Vectorization still works
  - Memory bus layout?
  - Accelerators?
- Which of the parallelization exercises does make sense?

Outline

Requirements

- Compatibility with Gaudi
  - Allow use of current code
- Portability
  - Event Filter Farm
  - Grid
  - Laptops

Requirements

- Compatibility with Gaudi
  - Allow use of current code
- Portability
  - Event Filter Farm
  - Grid
  - Laptops

Status

- Multiprocessing with "late" forking (GaudiMP)
- Gaudi Whiteboard (GaudiHive)
- A few groups working on GPUs

Plans

- Multicores-aware grid jobs in productions (GaudiMP)
- Follow up with the GaudiHive exercise
- Evaluate auto-vectorization
- Evaluate GPUs
- Testing ARM servers

Outline

Conclusions

- What's the best strategy?
  - Multiprocessing
  - Multithreading
  - GPUs
  - ...
- A lot depends on external constraints
  - Cloud computing
  - Manycores-aware grid jobs
  - Memory
  - Layout (NUMA, etc.)
- Investigate different paths and see what happens

Conclusions

- What's the best strategy?
  - Multiprocessing
  - Multithreading
  - GPUs
  - ...
- A lot depends on external constraints
  - Cloud computing
  - Manycores-aware grid jobs
  - Memory
  - Layout (NUMA, etc.)
- Investigate different paths and see what happens

Conclusions

- What's the best strategy?
  - Multiprocessing
  - Multithreading
  - GPUs
  - ...
- A lot depends on external constraints
  - Cloud computing
  - Manycores-aware grid jobs
  - Memory
  - Layout (NUMA, etc.)
- Investigate different paths and see what happens