The PHENIX Event Builder

Brian A. Cole

Columbia University

and

J. Haggerty, X. He, A. Jhavieri, B. Kothari, S. Lin, J. Nagle, C. Witzig

for the PHENIX collaboration

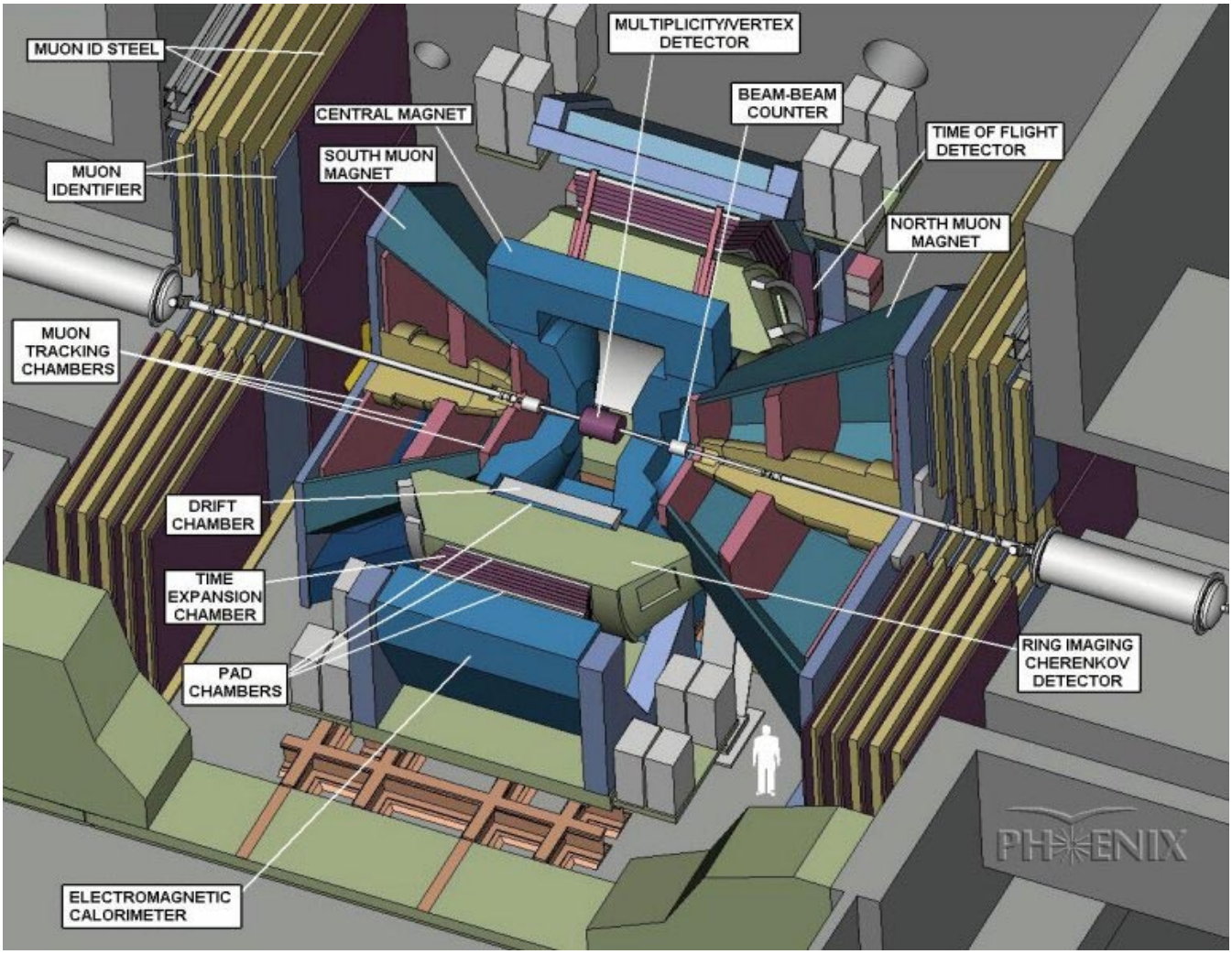
In collaboration with D. Calvet, J-P. Dufey, P. Le Du, I. Mandjavidze, and M. Nomachi

Sept. 22, 1997

Outline

1. The PHENIX experiment
2. PHENIX DAQ system overview
3. PHENIX event builder design
4. Event Builder implementation
5. ATM on a Windows-NT platform
6. Winsock2 performance
7. Summary

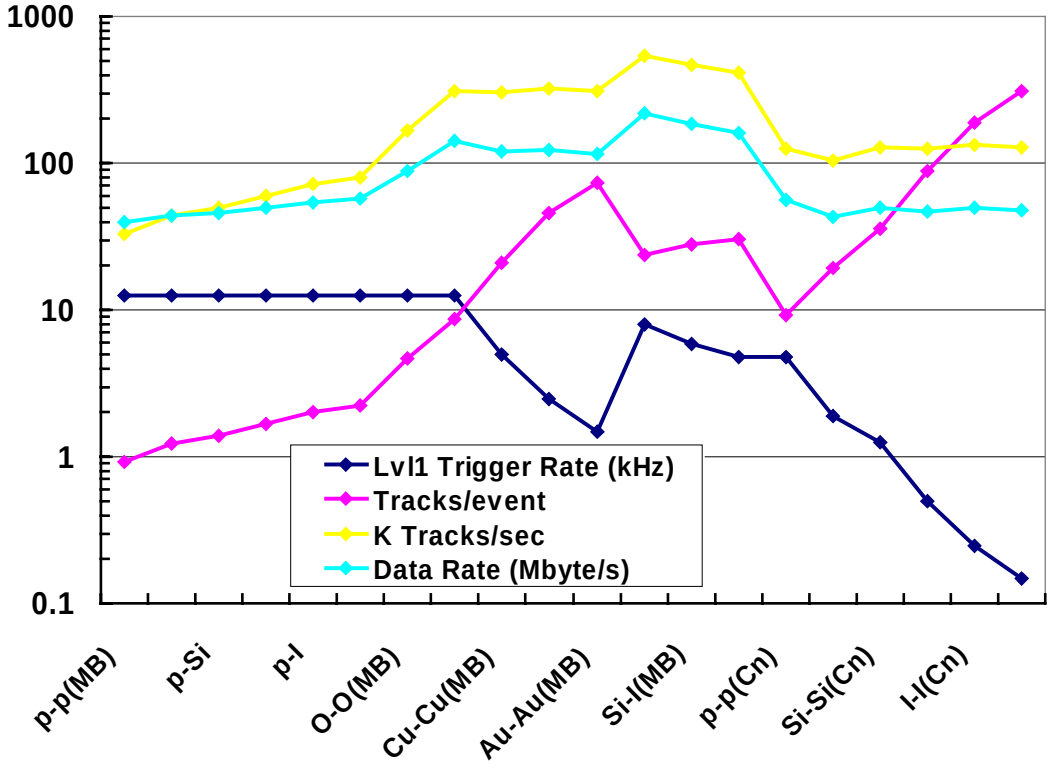
The PHENIX Experiment at RHIC



- Two “central” arm spectrometers at 90°
- Two “forward” muon spectrometers
- **High-rate, rare probe experiment**

PHENIX Event/Data Rates

RHIC (Design) Luminosity & Collision Rate vs species



- Large dynamic range in
  - Collision rate: 100 Hz (Au+Au central) → 50 kHz (p-p)
  - Event Size: 5 kbyte/event → 350 kbyte/event
- BUT ! Approximately constant ***data*** rate for all species
- PHENIX Online Performance specs
  - Baseline: **12.5 KHz Lvl-1 rate**, **500 Mbyte/s EvB bandwidth**
  - Upgrade: **25 KHz Lvl-1 rate**, **2 Gbyte/s EvB bandwidth.**

Triggers in PHENIX

Physics Considerations

- Signals "requiring" triggers
  - single or di-muons
  - high-pt single photons
  - high-p_t single electrons
  - high M_{inv} electron pairs
  - electron-muon coincidence
  - unusual "event topology"

Trigger Hierarchy

- Level-1 triggers needed for "light" systems only
- Traditional Level-2 difficult in "messy" ion events.
- Decent rejection requires data from several detectors

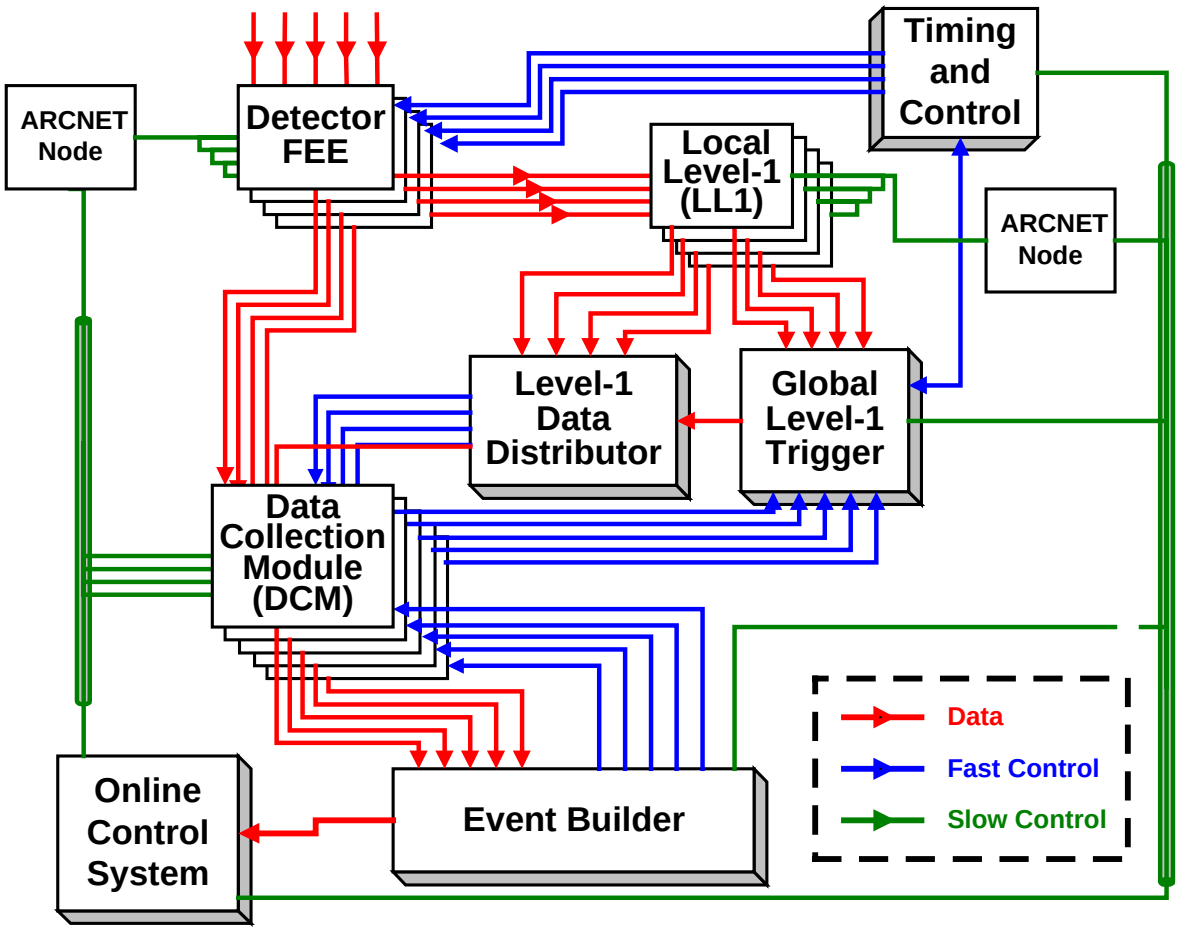
⇒ **Rely on "software" triggers**

Trigger Scheme

- Build full events at maximum Level-1 accept rate
- Run "software" trigger calculations
- Use level-1 trigger primitives to "target" algorithms
- Design in possibility of building partial events

⇒ **Event builder and Level-(?) trigger tightly coupled**

The PHENIX Online System



Pipelined, data-driven DAQ system:

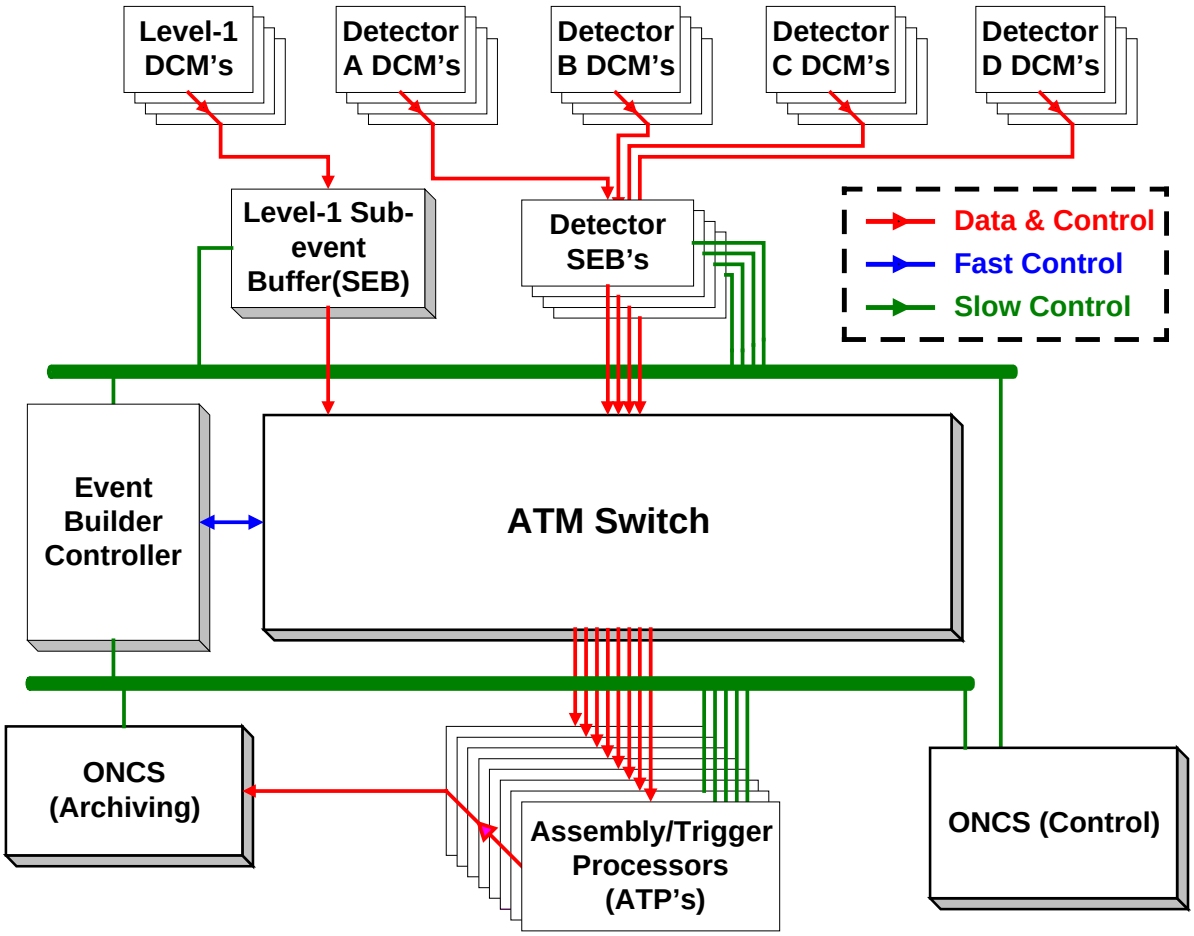
FEM's: Store signals in AMU/DMU's @ , digitized on Level-1.

DCM's: Read FEMs, perform zero suppression, and format data.

EvB: Reads DCM's, assembles event, runs trigger calc's.

ONCS: Receives, archives events from EvB, handles slow control.

PHENIX Event Builder Design



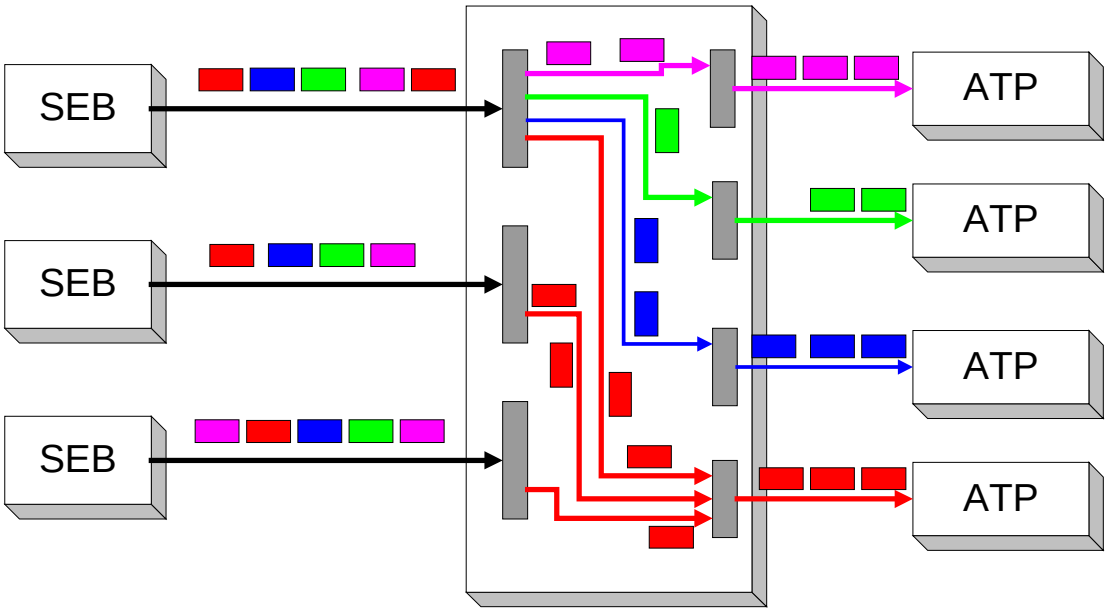
ATM Switch-based Event Builder

- **SEB's:** Receive, buffer data from DCM's, send data to ATP's.
- **Controller:** Allocates destination ATP's based on "availability".
- **ATP's:** Pull data, assemble events, run trigger algorithms, send accepted events to ONCS for archiving.

Event Builder Implementation

Data Flow

- Data will be “pulled” by ATP’s
  - Simplifies flow control, congestion avoidance, event assembly logic, implementation of partitioning
  - Allows partial event building before trigger
- Arrival of event “signalled” by presence of Level-1 data.
- Use ATM AAL5 PVC’s for messages, data.
- ACKnowledge, re-send control messages, but not data.
- Use VC-based source rate control.



Event Builder Implementation (2)

SEB's and ATP's:

- **PCI-based Intel** (and/or Alpha ?) processors
  - Ind.-PC packaging w/ motherboards or passive PCI+SBC's.
- Running (non) **Windows-NT**.
  - Use software from **VenturCom** to:
    - ⇒ Produce minimal NT systems on flash memory
    - ⇒ Provide null display, input drivers
    - ⇒ Provide hard+soft RT ext's.
- Use **DEC Pamette** PCI board for DCM ⇒ SEB link
  (see talk by C.Y. Chi and poster by J. Haggerty)

Controller:

- **Alpha** or **multi-processor Pentium** on PCI (64 bit ?)
- Also running **Windows-NT**.

Switch:

- **Fore ASX-1000**. 64 OC3 ports (fully equipped).

NIC's:

- Currently unspecified Fore/Efficient//IDT... OC3

ATM API:

- Winsock2 (for now)
  - **Vendor independent, "standard" API on NT**

ATM on Windows NT

Windows-NT

- Preemptive, multi-threaded, multi-process OS
- “Intrinsic” support of asynchronous I/O (~ RT-11)
- Kernel-level implementation of synchronization objects

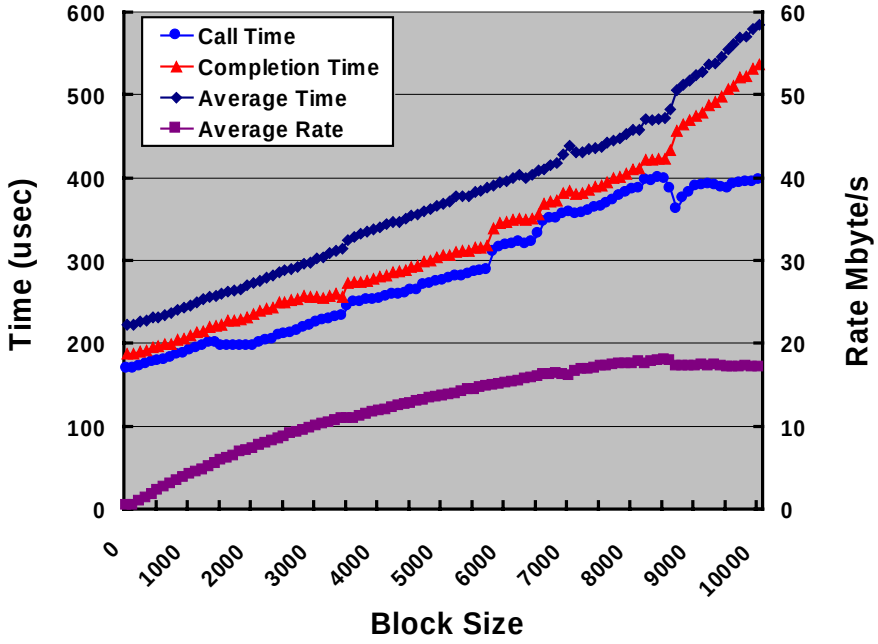
Winsock2

- Socket-based API extended from BSD sockets.
- “Transport independent” API for network I/O
- High-level interface to
  - ATM quality-of-service control
  - ATM “raw” AAL1/ 5 transports
  - ATM signalling
- Explicit support for
  - scatter-gather transfers
  - asynchronous I/O (using NT I/O services)
  - “pre-declaration” of receive buffers for “zero-copy” RX’s
- Uniform interface to NIC’s from different vendors
  - Allows same **executable** to run on different NIC’s
- **Significantly reduced ATM novice start-up time**

Winsock Performance

Method

- Measure send/receive rates on two different NIC's
- Use Saclay timer routines
  - cross-checked with vendor NIC monitoring programs
- Use asynchronous I/O and measure vs block size:
  - I/O call time
  - time to completion routine
  - average time to execute extended loop



Winsock Performance

Good News

- Software can saturate and sustain OC-3 line speed.
- Basically works “out of the box”
- No fussing with PCI issues (yet)
- Results are consistent for 2 NICs+Winsock SPI’s

Bad News

- Moderate overhead of 170 μs to every transfer.
  - Max TRX rate of 5 KHz
- Overlapped I/O shows odd NT behavior for small blocks.
- Results are consistent for 2 NICs+Winsock SPI’s

Future

- Existing SPI’s based on Intel example (according to Fore)
- Evidently Microsoft will be issuing their own “standard” Winsock ATM SPI
- Is this good news or bad news ?

Conclusion

- Performance is sufficient for now thru 1999 RHIC run.
- Advantages sufficient to adopt Winsock in PHENIX EvB.
- Let commercial market push the technology.

PHENIX EvB Status

Hardware

- Currently have & are using for tests:
  - 5 Gbit/s equipped ASX-1000 in hand.
  - Assorted NIC's.
  - 2 200 MHz desktop PC's
- This fall
  - Will purchase, install 2-4 IPC packaged processors
  - Start testing event building software on 2x2 + controller
- Currently developing DCM-SEB interface
  - DEC Pamette PCI board is "on the way"
  - daughter-board layout nearly complete

Software

- Collaborating with Atlas Architecture-C effort.
  - Ready to incorporate Winsock transport into "system"
  - Start with existing Architecture-C implementation
  - Augment with functionality that PHENIX needs
- BNL developing Corba (Orbix) control interface.
- "Embedded NT" using VenturCom software
  - Totally new technology to us