

Offline Software Update

- Offline overview
- Recent work:
 - Raw data format (Lebedev/Messier)
 - Online monitoring (Messier/Tatar)
 - Configuration (Messier)
- Immediate plans

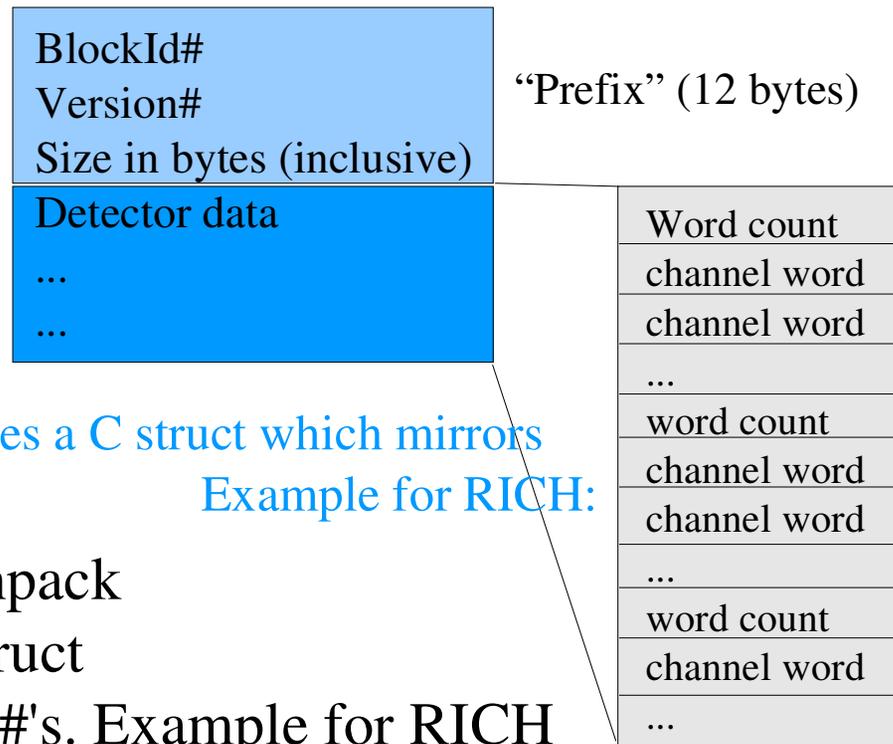
MippIo (Lebedev/Messier)

Defines online “raw” data format

Basic object is a MippIoBlock.

Each detector produces a MippIoBlock (MippIoRICHBlock, MippIoTPCBlock,...)

Representation in memory:



DetectorBlock (MippIoRICHBlock) defines a C struct which mirrors the packing of this data.

Example for RICH:

Block class contains methods to unpack

[1] buffer in memory to VME C struct

[2] VME words to logical channel #'s. Example for RICH

-VME Address-|--FEB # -----|--Channel # -----

15 14 13 | 12 11 10 9 | 8 7 6 5 4 | 3 2 1 0 |

Files and Events

FileHeader (Run#/Date/Trigger...), and EOF are also Blocks

File format:

FileHeader
EventBlock
EventBlock
...
...
EOFBlock

Event format:

EventPrefix
Block (eg. EventHeader)
Block (eg. RICHBlock)
Block (eg. DCBlock)
...

Any block can/cannot appear inside an event in any order

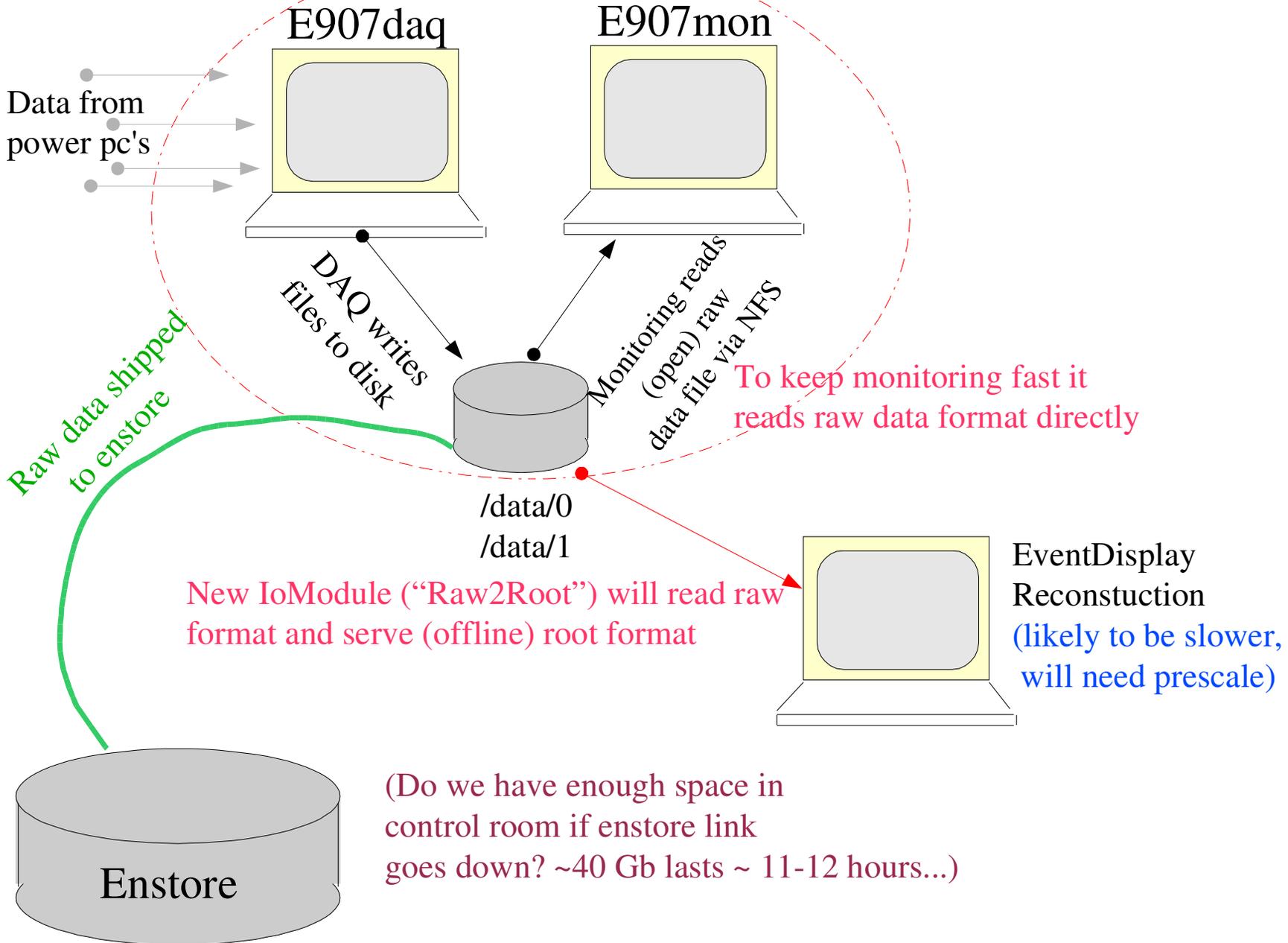
Utilities for debugging format: [MippIo/test/rawdump \[-h\] file.raw](#) – prints event data to screen

[MippIo/test/testWrite](#) – creates fake event and writes to file

- › Currently in use to write files of RICH test DAQ runs

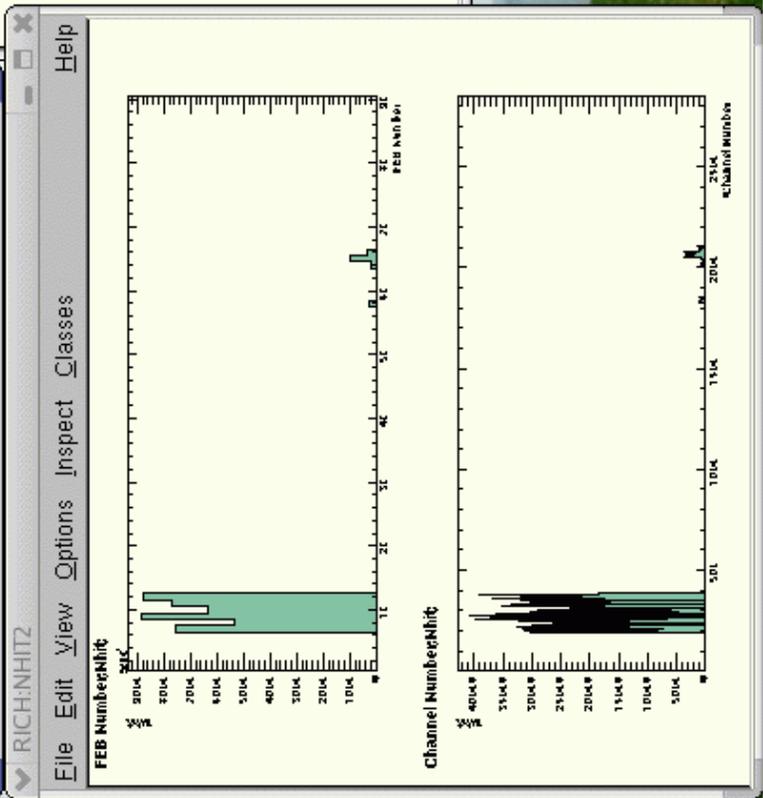
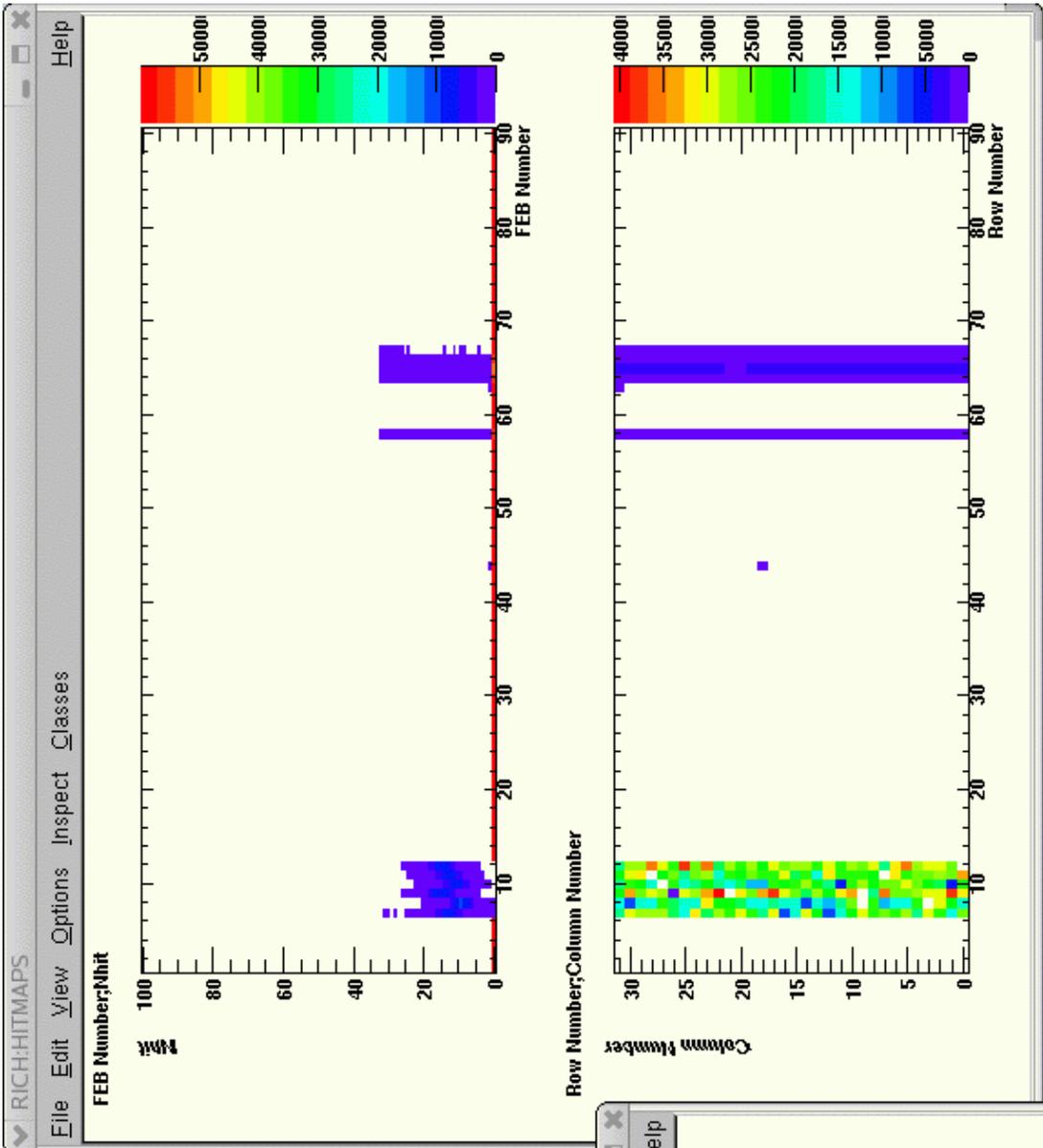
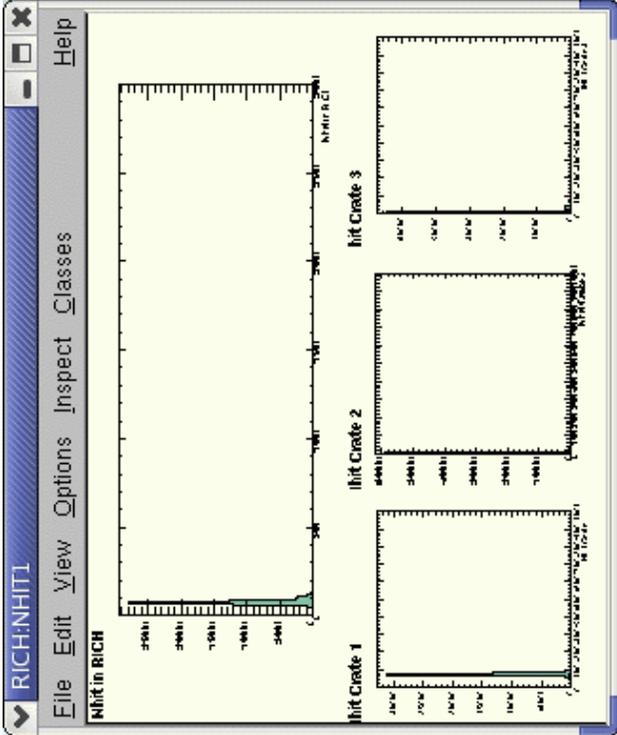
Online Data Handling

Prototype of this is working



OnlineMonitoring

- State machine which continuously checks data directory for new files to analyze. When it finds new file, analyzes file and makes histograms of raw detector data
- Can read open files
- Each detector provides a “module” which extracts the detector data and fills histograms
- Histograms are arranged onto “views”
- Top level GUI will provide ways to browse views, report views with histograms out of spec. etc.
- Basic guts are working w/ one module (RICH). Lacks bells and whistles



Library: /usr/local/...
 [2003/08/09 00:26:57] Polling for file.
 [2003/08/09 00:27:01] Polling for file.
 [2003/08/09 00:27:02] Polling for file.
 [2003/08/09 00:27:03] Polling for file.
 [2003/08/09 00:27:06] Polling for file.
 [2003/08/09 00:27:07] Polling for file.
 [2003/08/09 00:27:08] Polling for file.
 [2003/08/09 00:27:12] Polling for file.
 [2003/08/09 00:27:13] Polling for file.
 [2003/08/09 00:27:14] Polling for file.

Configuration

Several packages in offline and online need a way to manage configuration parameters

- DAQ thresholds, delays, trigger settings (currently used by RICH)
- Reconstruction parameters, cuts
- User preferences

Single package (“Config”) for all of these

Client code register themselves and receive notification when configuration changes. Option to make configs RO

```
MyClass : public CfgObserver
```

```
MyClass::MyClass() { this->SetWatch(“MyClass”,”default”); }
```

```
MyClass::Update(const CfgConfig& c) {
```

```
    c(“fFloat”) . Get (fFloat); // Store parameter named “fFloat” as fFloat
```

```
}
```

The pool of configurations comes from database (ultimately) and from XML files

ConfigXML

XML for configuration looks like:

```
<config name="MyConfig" version="default" ro="1">  
  <param name="fFloats">  
    <float> 1.0 2.0 3.0</float>  
    This is an explanation of these parameters...  
  </param>  
</config>
```

The package to support this is called MippXML

- based on xerces-c parser (Apache, used by online also)
- Script to install package at site in SRT_MIPP/scripts/install-xercesc.csh
- To support new tags (eg. <config> and <param>) user has limited amount of code to write (a XMLtagBuilder) which is handled the tag attributes and text buffer
- Planned for use for configuring reconstruction and analysis jobs
- XML can be used to represent the database as flat text files
- More info see MippXML/test, Config/test, Config/xml/test

Immediate future

Get the Raw2Root IO module working. This will complete the cycle DAQ -> Online -> Offline

Put framework together for making reconstruction and analysis modules “pluggable”

Work out XML <-> database interface

Get the event display and online reconstruction working

These will complete the “framework” for the software (I think) and reconstruction work can ramp up.