EIGER meeting / Baden / 24-25 Jan 2013

Agenda

24/01/13

Christian Broennimann Welcome
*Miroslav Koba*s Agenda and goals of the workshop
Clemens Schulze-Briese DECTRIS EIGER product family
Bernd Schmitt PSI EIGER system
Heiner Billich Challenges of EIGER for beamline IT. Move towards a container-based file format. HDF5 developments by HDF5 Group, Dectris, PSI, Desy
Mark Koennecke NeXuS - what is it good for?
Herbert Bernstein CBF/HDF5/NeXuS integration
Nick Rees High speed data storage / Real-time processing / HDF5 plans of Diamond
Yukito Furukawa Development of DAQ system for SACLA (XFEL) facility. HDF5 data handling at SACLA
Miroslav Kobas Workshop instructions
Michael Rissi HDF5/NeXuS for high-speed single photon counting detectors. Compression algorithms
Markus Mathes DECTRIS EIGER system - concepts for detector integration and data transfer
Peter Trueb Specific detector corrections: rate correction, virtual pixel correction, flatfield correction, sensor efficiency correction, parallax correction, geometric distortion
Marcus Mueller Challenges for crystallographic data reduction packages
Volker Pilipp ALBULA 2.0 - status and future plans
Kay Diederichs Basic thoughts/concerns/obstacles of HDF5
Workshop 1 - Control Software EIGER Interface and HDF5/NeXuS
Workshop 2 - Application Software HDF5/NeXuS and detector corrections

25/01/13

Mark Rivers EPICS for SPC: interface for EIGER DAQ
Sebastian Petitdemange LIMA for SPC: interface for EIGER DAQ
Alain Buteau TANGO for SPC: interface for EIGER DAQ
Philipp Duval TINE for SPC: interface for EIGER DAQ
Daniel Franke SAXS for SPC: current status, limitations, outlook
Thomas White CrystFEL for SPC: current status, limitations, outlook
Pierre Thibault Ptychography for SPC: current status, limitations, outlook
Mathias Meyer Crysalis for SPC: current status, limitations, outlook
Gwyndaf Evans, Nick Sauter DIALS for SPC: current status, limitations, outlook
Alexander Popov BEST for SPC: current status, limitations, outlook
Clemens Vonrhein AutoProc for SPC: current status, limitations, outlook
Wladek Minor HKL for SPC: current status, limitations, outlook
Andrew Leslie MOSFLM for SPC: current status, limitations, outlook
Wolfgang Kabsch XDS for SPC: current status, limitations, outlook
Workshop 1 - Control Software EIGER Interface and HDF5/NeXuS
Workshop 2 - Application Software HDF5/NeXuS and detector corrections
Workshop 1 - Control Software EIGER Interface and HDF5/NeXuS
Workshop 2 - Application Software HDF5/NeXuS and detector corrections

Heiner Billich Summary Workshop Control Software
Herbert Bernstein Summary Workshop Application Software
Miroslav Kobas Event summary and Outlook
Christian Broennimann Farewell

Synopsis

The most important points from the meeting are summarised here:

- EIGER's data rate *requires* the move to a container format (HDF5).
- The DECTRIS EIGER will transfer data at 40 Gb/s. That means 3 kHz frame rate for the 1M, or 187 Hz for the 16M
- The PSI EIGER is different (fully parallel → 3kHz even for a 16M), but not commercialised.
- Generally thought that to use HDF5 format you will be using the HDF5 library for access. Wolfgang Kabsch extremely resistant to this.
- Next release of HDF5 (v1.8.11), ~May 2013, will give sustained high volume acquisition at high rate (by parallel compression external to the library then writing in the compressed chunks).
- The HDF5 'format' will consist of a 'master file' consisting of metadata for the whole experiment, plus one or more data files consisting of blocks of images. This is to support parallel read (also possible to have the whole dataset concatenated to a single HDF5 file).
- General agreement that each data file should contain the basic details required for integration (e.g. by Mosflm) as data files can get separated from the master.
- Majority acceptance of proposed format with HDF5/NeXus master file and multiple data files, with some remaining dissent.
- Strong support for need to simultaneously read and write header information. Not entirely clear how this will be done.
- LZ4 compression (applied twice) found to be the best compression scheme. Very fast, good results and BSD license.
- The HDF5 group is under-funded and needs money from the facilities for necessary developments.
- Nick Sauter wants to use the stream API to access data, while Nick Rees says this will be much more complicated than we imagine at the moment. NKS to think about a trial run for this.
- DECTRIS will give option for raw data so corrections can be made in software
- NeXus uses the McStas coordinate system. It is similar to *but not the same as* imgCIF.
- The MX community must agree Application Definitions for MX NeXus files. Strong statement that Diamond will not write NeXus files unless they pass NXValidate. So this agreement has to come in advance of use.
- CBF and NeXus will ultimately be merged.
- The main action for MX is to now get the file + metadata specification right. HJB will head the discussion, *via* email at first then another meeting later in the year.
- Miro Kobas is leaving DECTRIS very soon.

Day 1 Talks

Christian Broennimann (DECTRIS)

All Pilatus sold to date are based on PSI development. Pilatus3 is DECTRIS development. 10^7 cts/pixel/s, 100 Hz, instant retrigger. EIGER is a step further: 40 Gbits/s, 3kHz frame, negligible deadtime.

Miroslav Kobas (DECTRIS)

Best possible data quality is a joint effort. EIGER **requires** move to container format. Normal hdf5 works with high volume data ac at slow speed. Parallel hdf5 works for high rate, but low volume (no compression). Next release HDF5 1.8.11: sustained high volume ac at high rate. Release ~May 2013. Data 'chunks' means images for DECTRIS. HDF5 will allow concurrent data writing and logging inside NeXus header, plus reading with a small delay.

Various detector corrections to be talked about. DECTRIS will give option for raw data so corrections can be made in software.

Want to get minimum requirement for automatic data processing and logging inside NeXus header.

Clemens Schulze-Brieze (DECTRIS)

EIGER family. The EIGER chip is licensed from PSI. Read-out electronics, software, other hardware from DECTRIS. Main difference between EIGER and PILATUS: 2* chip area, 5.3 times smaller pixel area, 11.3* greater pixel per readout chip (in eiger 256*256). Counter now 12 bit (4096) not 20 bit, but this is not a limitation (can do extreme fine slicing with onboard image summation). Same 2*10^6 X-rays/px/s. Continuous readout. Total data rate limited to 40 Gb/s.

PSI is also building an EIGER detector. PSI cf. DECTRIS EIGER modules are different. DECTRIS data rate is function of 1/n where n number of modules. I.e. frame rate **3 kHz * 1/n.** PSI is fully parallel. Frame rate remains 3 kHz no matter the size. Also PSI have on board memory, DECTRIS not. DECTRIS module much cheaper though, while PSI EIGER is a research project

EIGER 1M: 2 modules, 3 kHz frame rate. First 1M prototype is under test. It works. High frame rate good for ptychography and tomography applications. Room temperature data collection done quickly shows 3* increase in radiation tolerance (Robin Owen & Gwyndaf) of crystals.

Other EIGERs: 4M, 9M and 16M. All do 40 Gb/s. Frame rate decreases 750, 333, 187 frames/s. First 1M delivered end 2013. First 16M delivered mid 2015.

No options on sensor thickness, yet.

Wladek Minor question on dynamic range: pilatus will sum on the detector increasing dynamic range. Single photon counting with frame adding means you get increased apparent dynamic range. Users will see, e.g 32 bit image. It's not dynamic range that is the 12b limit, it is the counter depth on the readout chip.

Bernd Schmitt (PSI)

PSI EIGER system. Fully parallel. The PSI EIGER completely modular, scalable to any size. Dead time free mode of operation - shift data so can readout while collecting more. Have 500k single module and plan a 9M detector (with 906 Gb/s at 12 kHz possible!)

Trimming improves response uniformity between channels.

With onboard memory, PSI eiger can do data processing in firmware (e.g. image summation, rate correction, pump probe series averaging, 2*2 pixel rebinning, SAXS ring intensity averaging, data compression)

Eiger control uses C++ detector class (all PSI detectors use the same). Client runs on beamline PC. Servers run embedded linux on each half module

New detector: Jungfrau is a 2D detector for Swissfel. "ajusting gain detector for the aramis user station". *Charge integrating*, starts with high gain, automatically moves into low gain at high flux. Same quality as SPC for low counts.

Heiner Billich (PSI)

EIGER and beamline IT. DECTRIS EIGER is in reach of current beamline IT capabilities (compression on detector system solves it). However PSI EIGER could be a significant challenge to use effectively.

Need lossless compression to stay in the reach of average beamline IT (10 Gb ethernet).

What's wrong with writing one file per frame? Inherent access pattern is sequential large block i/o. This pattern is well understood (HPC, video streaming, lustre...). Single file per image is completely at odds with these existing solutions. Latency of file create, open, close, stat becomes limiting factor. HDF5 does support large block sequential i/o very well. Next release will add parallel compression external to hdf5, dynamic load of custom filters (e.g. compression libraries at runtime).

Comparison of writing hdf5 chunks is comparable to raw, native sequential disk write especially for larger chunks such as 10 MiB.

HDF5

- writes compressed data at rate close to native file system
- reads compressed data sufficiently fast
- minimal overhead cf. native benchmark writes
- works well with archive, backup.

Kabsch: file overheads are not the issue. Response it is file *writes* that are the issue. It IS a problem if you want high sustained data rate. Kabsch: in MX you don't need such high data rate. So in that case, you don't need EIGER.

Andrew Leslie: what are alternative container formats to HDF5? Answer: there are none appropriate. It is the only game in town.

Mark Koennecke (on behalf of NIAC)

NeXuS - what is it good for?

- Full beamline description
- Extendable
- Self describing (discoverable)
- Easy plotting
- Platform independent

NeXus adds names and rules to hdf5. i/o achieved using either hdf5 or NeXus-API

HDF5 is efficient binary format, grouping support, on fly compression, reading/writing subsets of data, appendable dimensions, public domain access libraries in C, fortran 77, java, access in e.g. matlab, simple to read, widely used (NASA, boeing, hpc, weather forecasters...)

Hierarchy: Files; groups; scientific data sets; attributes; links (like symlinks).

Uses McStas Coordinate System. Not dissimilar to CIF one. Worth a look...

NeXus uses CIF way of storing translations, rotations etc. CBFlib to be updated to support.

NeXus also stores processed data. This means you can store complete workflow from raw data to reduced results in one file.

A few sensible rules for use, e.g. must store units, use supported names from dictionary.

Application Definitions define what has to be in a NeXus file for a certain application. Validated by NXValidate. Written in NeXus Definition Language (form of XML).

Various NeXus tools exist already, for e.g. extraction, validation, conversion, plotting.

Important to capture as much data as possible, to mitigate against unforeseen circumstances.

NeXus is already 'widely' used. Been around for a few years. Volunteer effort. Join the NIAC!

Conclusion. NeXus is the only game in town. HDF-5 is consensus. Work underway to collaborate / merge with CIF.

Herb Bernstein

We are going to merge CBF and NeXus! The transition will be as painless as possible.

How do you digest an EIGER and the NSLS-II: a Big Data problem? Issues of fraud. Must retain raw data. Lossy compression must also come! HDF5, NeXus, CBFlib and database integration must be pulled together. NSLS-II will be the world leading brightness synchrotron. 10^13 photons/s for MX.

IUCr exec committee decided that authors may provide a prominent and permanent link to raw data in articles, with view to making this formal requirement in time.

Figure out what data you need to keep. Don't keep absolutely everything! Use NeXus / HDF5 with compression. Lossless compression immediately, then lossy. "James Holton is right" 20:1 lossy compression gives the same structure at the end. Wavelet compression keeps the features (spots) essentially lossless, compresses background more.

CBF can take many files per image too. But HDF5 is tree-oriented. CBF is table-oriented, *which is essential for database access*. Combining CBFlib with HDF5 backend provides full database access to the metadata.

Issues

- NeXus uses different axis conventions and tags from CBF
- Database access requires mapping trees into tables, but not good idea to have multiple copies of data. Must be able to use the links.
- new compression integrated with HDF5 are needed (not just plugins)

We should build the new system on the base of HDF5 as the agreed format

Axis definitions. NeXus Y is "up" whereas CIF Y completes a right handed set. Often coincide.

Proof of concept implementation of HDF5 backend to CBFlib exists already (in use at DLS). Can write HDF5 files with any of the CBF compressions.

Funding issues: general support needs "harder" APIs. Without funding it will be delayed. In the past not enough community support has been drawn.

Wladek: theoretically I agree. Key is implementation. CBF is actually many different frame formats. Kabsch just ignores header to get around this! HJB: the library handles this (the issue therefore seems to be that software developers don't use his library).

Nick Rees (Diamond)

High speed data storage and real-time processing HDF5 plans of Diamond. In 2008 Nick argued for centralised, Lustre system. Worked well (eventually) for MX. Problems with metadata and small files. 2011 extended. 6 GB/s, faster metadata. Used now for MX.

Read rate requirements are much higher off cluster than write rates. Don't focus on just the data coming out the detector. Got to multiply that out ~10 times to meet the read rate off the cluster.

Next challenges. Lustre client write speed inadequate ~400 MB/s with checksums. GPFS better. But for metadata speeds, Lustre is much better at every operation (create, open etc.) than GPFS except for file stat. A decision needed which file system (GPFS or Lustre) to move to for new hardware.

HDF5 plans of Diamond. GDA writes header data .nxs file. Detector writes its own hdf5. HDF5 links used to present one logical interface.

Tomography is example where the container format is essential. You need to view the sinogram perpendicular to the direction data is written. Solved by chunking the data and reading the chunks in the appropriate direction (imagine instead of an image stack, a 3D block made up of smaller blocks). This is not possible with 1 file per image mode of storage.

Michael Rissi (Dectris)

HDF5/NeXuS for high-speed single photon counting detectors. Compression algorithms.

HDF library introduces minimal write and read overhead.

Compression algorithms for EIGER. LZ4 found to be the best. Thaumatin dataset compressed by factor of 60, applying LZ4 twice. Compression 1.3 GB/s, decompress 1.7 GB/s with one process. New BSD license. HDF5 read performance for compressed data, 1.6 GB/s

New features of HDF5 - precompress images in parallel threads, write directly to hdf5 file. Filter plugin - read compressed datasets without recompiling application by dynamically loading filter.

NIAC and Dectris extended NeXus class for SPC PADs.

Want real time logging of system parameters. Yes, can do as separate header file from detector data file.

At the moment you can't do the image rechunking that Nick Rees discussed for tomography. Currently 1 image = 1 chunk.

Peter Trueb (Dectris)

Specific detector corrections: rate correction, virtual pixel correction, flatfield correction, sensor efficiency correction, parallax correction, geometric distortion.

Accurate data corrections essential, otherwise wrong by ~10%. DECTRIS apply corrections online: rate, flatfield, virtual pixel, efficiency. Then offline: parallax, geometry.

- Rate correction - for dead time of counting mechanism. Depends on detector gain, ratio of threshold energy to beam energy, bunch mode. Limitations: only works up to some cutoff rate. Rate corrected counts do not obey Poisson stats. Assumes constant rate during exposure.
- Flatfield correction - for impurities, chip/module boundaries. Depends on X-ray energy. Ratio of threshold to energy. Multiplicative correction. **That means the detector is not really photon counting** (the statistics of corrected images are not Poissonian)
- Virtual pixel correction - randomly split measured counts into virtual pixels of smaller size for some pixels that are twice the area of normal pixels (for fabrication reasons)
- Efficiency correction - corrects for top layer absorption, finite sensor thickness. Corrects counts for low DQE at non optimal energy. Multiplicative. Could be 10-20%
- Parallax - corrects measured position for penetration depth. Depends on energy, angle, thickness, material. Introduces correlation between neighbours if rebinning. Not applied online, up to the application software.
- Geometric - corrects for spatial displacements of module. Again would introduce correlations if pixel rebinning. Apply at same time as parallax correction. Better to take into account within the software when predicting spot position rather than rebinning.

It is also possible to provide raw images and let the corrections be applied in software. This is the route to be taken by DIALS. Can avoid pixel rebinning that way and therefore avoid introduction of correlations. Also allows for potential refinement of corrections.

Marcus Mueller (Dectris)

Challenges for crystallographic data reduction packages.

Why HDF5? It is a necessity for EIGER data rate (so question is, do we need EIGER?)

HDF5 is not a challenge. Libraries exist for all major programming languages. Header format strictly complies with NeXus format (e.g. there will be no EIGER "miniNeXus").

Count rate correction, assumes constant count rate over image. Bad for wide images, e.g. fulls. Crystallographic software could do better count rate correction based on derivative of observed peak shape in phi. Integration software can do better virtual pixel correction based on profiles known from elsewhere. Geometrical corrections - errors within array of modules: 0.25 px translational error (sigma), < 1 mrad angular error. Refining residual vectors using crystallographic software would be more accurate.

Pixel values from corrected images do not follow Poisson stats. Can predict the right stats from knowledge of the corrections though.

Very high frame rates may show up high frequency modulations that were averaged out before.

Correct determination of background values for pixels with average count << 1 is tricky. That is why summed images give better reduction stats than treating the weak components separately.

Volker Pilipp (Dectris)

ALBULA 2.0 - status and future plans.

From version 2.0 ALBULA will be both a standalone program and a Python API for visualisation and data analysis of Pilatus and EIGER images. The main objective for this is to integrate viewer seamlessly into beamline infrastructure.

Gives RW access to image data by NumPy and meta data (NeXus). Basic statistic operations. Various filters.

Available ~March 2013

Kay Diederichs

Basic thoughts/concerns/obstacles of HDF5, with responses in red

- Cannot write compressed data in parallel. Elena Pourmal (HDF5): hdf5 needs funding to do parallel compression and writing
- Cannot run MPI-parallel and thread safe together There is an undocumented solution. Thread safety is possible but untested.
- Can you have concurrent access to a single dataset from multiple processes? Yes, but only if there are no processes also writing at the same time.
- Using single hdf5 file for a complete dataset has some problems - you can either write in parallel, or use compression, but not both. Solved by external parallel chunk compression.
- While hdf5 file is being written it cannot be read. You can't do online analysis really, it is post factum only. Discussion about 'SWMR' (concurrent Single Write Multiple Read). Inconclusive.
- HDF5 is not multi-threaded, nor thread safe (unless a very inefficient serial version is used).
- Doesn't work well with OpenMP.
- Parallel hdf5 uses MPI. Nobody wants to program MPI. It is far too complex and expands code size. HDF5 hides MPI from developers. You don't need to use it yourself.
- HDF5 does not provide better read performance than file system access.
- Using HDF5 shifts problem from file system and i/o layer (kernel mode) to the user software (user mode), which is less efficient.
- Possible non-hdf5 solution. Performance problem of using single image CBFs can be mitigated by using SSD drives. Herbert will provide CBF extract function, but reading directly from HDF5 file is easy and more efficient.
- Shifts bottleneck from data writing to data reading (processing), putting the burden for solving the problem on shoulders of developers of processing packages.

Mathias Meyer wants decompression of an image in parallel. Not easy with current 'chunking' scheme (1 image per chunk).

General agreement: organisations such as synchrotrons can and should pay for particular developments

Workshop on application software

Starting from Kay's talk, go through all the points raised.

Priority one: multiple reading in parallel.

You can state how many frames you want in each physical file. As soon as it is closed you can access. So you split full data collection over total_num_frames / frames_per_file. Say 10'000 frame dataset with 100 or 1000 frames per file. This is a practical way to avoid restriction on reading files while it is being written to. In fact, the latter *is* possible in upcoming version (see SWMR).

The master file contains the parameters required for processing (gonio direction etc, e.g. the frame header). It is written first during data collection, closed, then the app software can start reading. The master file IS the header.

Clemens Vonrhein: But it could be useful to have master file that shows the intention of the experiment, but also a copy that records what actually happens during to the experiment.

HJB - how about you can ADD the actual diagnostics to the master file rather than a copy? Ok, this is fine. But we still need to write to this file (from diagnostics process) and read from it (from processing sw) simultaneously. HJB marks this down as a requirement.

Do we need additional data within each of the N image hdf5 files, to assist with processing? That tells you what each file is for. It also gives you a safety net if your master file is corrupted. Mosflm developers very keen, as are others who know that separate files can get "lost".

Do we want to post process all the files into a single hdf5 file once the dust has settled? Sort of an archiving operation. Most data processing runs are done long after beam time, so better not to impose the structure required for rapid online processing to the long term storage format. Inconclusive discussion.

Kay says the single file structure imposes on the software developers who actually have to do the work.

Day 2 Talks

Mark Rivers (APS)

EPICS for SPC: interface for EIGER DAQ

Distributed control systems for beamline control.

EPICS has an areaDetector framework.

Sebastian Petitdemange (ESRF)

LIMA for SPC: interface for EIGER DAQ

High speed data acq. At ESRF all detectors are interfaces with Lima library.

Alain Buteau (Soleil)

TANGO for SPC: interface for EIGER DAQ

Philipp Duval (DESY)

TINE for SPC: interface for EIGER DAQ.

TINE is a control system with emphasis on transport efficiency

Daniel Franke (DESY)

SAXS for SPC: current status, limitations, outlook. Solution SAXS. Do radial averaging of the images then don't look at the 2D image anymore, keep the 1D curve only.

Thomas White (DESY)

CrystFEL for SPC: current status, limitations, outlook.

Detector and data processing science with FELs. Applications: Coherent diffraction imaging (small crystals, viral particles etc). Single molecule diffraction (very low signals). Serial femtosecond crystallography.

Currently 2 hard X-ray FELs in the world. At LCLS, XTC file is a "tape recording" of the whole experiment (everything that passes thorugh DAQ). They analyse with software Cheetah and write out **one** HDF5 file per image classified as a hit. CrystFEL analyses using mutliple processes and writes into a text file called the stream with all analysis results.

Pierre Thibault (Munich university)

Ptychography for SPC: current status, limitations, outlook.

It is an imaging method using coherence. Ptychography (from greek to fold) was called thus to distinguish from holography. Coherent light gives you many copies of the same photon. That is, an extremely pure input, with which you can collect from very dirty samples. Collect far field intensities. Find illumination (probe) and sample functions consistent with these. Iterative projection algorithms and cost-function optimisation methods.

Can reconstruct illumination function easily: full complex-valued wavefield.

Mathias Meyer (Agilent)

Crysalis for SPC: current status, limitations, outlook

Data reduction with data from Dectris SPC. Relevant for CryAlis^{Pro} because Agilent users also do synchrotron experiments. Also, Agilent sells a system with the Pilatus 300K. Modifications to CryAlis^{Pro}: gap handling in strategy calcs, shutterless acquisition, positional correction. External users have synchrotron data mainly. CryAlis^{Pro} has a 3D lattice viewer. It has empirical corrections for absorption, detector sensitivity, time dependent decay. Shape based absorption correction from video feed of crystal (mainly for small molecule). Deconvolution of twin intensities.

Gwyndaf Evans (DLS), Nick Sauter (LBL)

DIALS for SPC: current status, limitations, outlook

DIALS is application sw for MX, and platform for research on data reduction methods.

Multiprocessing essential. Must have heavy parallel read access to HDF5. Nick is also interested in accessing the stream API rather than files.

Is it possible to marshall multiple crystal datasets in HDF5 as an organisational unit? Answer from HJB: very straightforward and highly recommended.

Extended requirements for logging in hdf5, beam details, crystal size/shape. Better knowledge for corrections and scaling

Alexander Popov (Soleil?)

BEST for SPC: current status, limitations, outlook

At ESRF run inside EDNA

Clemens Vonrhein (Global Phasing)

AutoProc for SPC: current status, limitations, outlook

autoPROC is an expert system for data reduction and analysis. Single or multiple sweep experiments are supported. Multi sweep is very common for many reasons. No GUI, all CLI. Academic release available for free from globalphasing. Has a tool for reading all image header types and printing info. *Whatever is done with metadata must be tightly controlled*. Every Pilatus installation had a slightly different header! Response: that is not really Dectris' fault: it is what the *beamline* decides to write in via the DECTRIS Detector control software “camserver”.

In MX it is very common to reprocess data multiple times, as you learn each time. It must be set up so that images can be read when away from the beamline.

Interleaving (inverse beam e.g.) requires decoupling of dose information from "image number".

Dectris provides generic HDF5 with images only. Beamline must provide MX-specific additional information in structured, well organised way! Otherwise there is still a multiplicity of 'formats'. Who is to decide this? (Andrew's view: should be the processing software developers, they are the ones that choose to use it).

Wladek Minor (HLK Research)

HKL for SPC: current status, limitations, outlook

"High output" much more important than "high throughput". There is about 10-20 hours of beamtime per structure solution. About 2 times as many PDB deposits using HKL2000 than citations for it.

Andrew Leslie (LMB)

MOSFLM/iMosflm and the EIGER.

Main problems with Pilatus were very fine slicing and very weak diffraction. Needed algorithm changes. Now ok, and should be fine for EIGER.

Will Dectris provide an image reading library? Yes - there is on the webpage (but language requirements? There is a C++ library)

From user point of view moving files around is important. rsync, ftp, cp to disk. Whatever. Need robust ways to move around.

Do hot pixels depend on threshold energy? Their location seems to change. How are they flagged?

Will adequate header information be provided? Andrew lists about 20 items that should be included in *every file*. Andrew has no desire to look at the raw images. He wants the detector to apply corrections (as currently in Mosflm). Internal frame summation for visualisation and parameter refinement. Plans to do this. Currently focussing on multi lattice images.

Wolfgang Kabsch (MPI Heidelberg)

Essential preconditions for using XDS.

Size of rotation per image is arbitrary, though very fine slicing gives no advantage. Strongly suggest frame summation **within** the EIGER system (done most efficiently there).

XDS expects 1 file per image, though can accept a stream of images. Accessing via hdf5 routines has no advantages and makes XDS dependent on foreign library routines, plus legal issues. Suggest instead frame summation onboard EIGER and results stored on regular file system!

Comments on reconstruction of (full) 3d reciprocal space instead of the frames. There are many disadvantages: it requires **very accurate** knowledge of s0. Small errors → misindexing. Background and photons must be treated differently. Mixes data processing and data acquisition. So simply do not do integration in reconstructed reciprocal space.

Detector corrections. XDS assumes a flat field is applied. It calculates its own geometrical distortion. Assumes pixel values are proportional to true # quanta hitting pixel, and calculates proportionality constant (~1 for pilatus). Bad pixels must have -ve count.

XDS must have a very small amount of essential data in .inp file.

Parallelism by employing team of processors under openmp, or batch (different address spaces).

Discussion

Can we use stream i/o rather than file i/o to get access to the data faster than at all possible with an intermediate writes to disk? Nick Sauter is very interested in this. He will think about a trial for doing this.

Stream API from DECTRIS doesn't have all the metadata. Currently the beamline controls add metadata and then write to disk. How about controls adds the metadata and pumps out the new stream to application software? It is possible. But Miroslav prefers 2 streams, with controls pumping out the metadata stream separately and the application software reading both.

Nick Rees says it is much more complicated than we imagine. Determining the right protocols and standards for using the stream will be *really* difficult. Prototype it at perhaps SLS only, using DIALS only.

The community (application software developers?) need to share amalgamated requirements of metadata, to inform NeXus Application Definition for MX. The main action out of **this** meeting is to get the file + metadata specification right. The metadata spec must be put into proper NeXus structure and must pass NXValidate. It has to be general agreement. HJB will moderate this. Email communication first. At decision it then goes to NIAC for consolidation as application definition. In time for May.

Request for another workshop in 6 months.

Application workshop summary: general acceptance of proposed format with HDF5/NeXus master file and multiple data files. Strong support for need to simultaneously read and write header information