Hall A Analysis Software & Computing Update

Ole Hansen

Jefferson Lab

Hall A Collaboration Meeting
February 10, 2022
Podd Status

- Current release: **1.7.0** (16 Nov 2021)
  - Many new features (already presented at previous meetings)
  - Additional improvements and bugfixes based on early SBS data taking
  - Significant speedup, primarily in decoder and database
  - Improved CODA 3 support
  - Dynamic raw data event buffer size
  - PID calculation based on Bayesian likelihoods
  - Requires **C++11** compiler and ROOT 6. Installed in counting house and on the farm.

- Priority development: **2.0-devel** (Summer 2022, delayed because of SBS work)
  - Multithreading
  - Will benefit SBS and Hall C, primarily for online replay
  - Requires **C++17** (e.g. gcc 9+, available on ifarm)
  - Existing code will need minor modifications

- Auxiliary development: **1.8-devel** (if time permits)
  - Add small new features missed in 1.7
  - Maintain system requirements and API of version 1.7 as much as possible
Podd: Profile-Based Code Optimization

Ole Hansen (Jefferson Lab)

Hall A Software & Computing Update

Hall A Collab Meeting, 10 Feb 2022
Podd Source Code & Documentation

JLab Redmine

GitHub
Podd: Building with CMake

Prerequisites:

- Install ROOT (ensure root-config is in PATH, or set $ROOTSYS)
  - Farm: run setroot_CUE.csh. RHEL: install from EPEL. macOS: install from Homebrew.
  - See also https://redmine.jlab.org/projects/podd/wiki/ROOT_Installation_Guide
- Ensure you have CMake ≥ 3.5 (cmake --version. cmake3 on RedHat)

Building & Installing Podd with CMake ≥ 3.15

$ git clone https://github.com/JeffersonLab/analyzer.git
$ cmake -S analyzer -B analyzer-build [-DCMAKE_INSTALL_PREFIX=/some/dir]
$ cmake --build analyzer-build [-j4]
$ ./analyzer-build/apps/analyzer
$ [cmake --install analyzer-build]
$ [/some/dir/bin/analyzer]

Notes:

- Installing recommended (cmake --install): Set CMAKE_INSTALL_PREFIX
- Will phase out aging SCons build system (too many limitations)
Pre-Installed Podd

farm/ifarm (works in Counting House, too)

$ module use /group/halla/modulefiles
$ module load analyzer
$ analyzer --version
Podd 1.7.0 Linux-3.10.0-1160.31.1.el7.x86_64-x86_64 git @e26c21d ROOT 6.22/06

Counting House (local installation, faster, safer)

$ module use /adaqfs/apps/modulefiles
$ module load analyzer
$ analyzer --version
Podd 1.7.0 Linux-3.10.0-1160.31.1.el7.x86_64-x86_64 git @e26c21d ROOT 6.24/06

The SDK is located in $ANALYZER/../src/SDK/
Podd 2.0

- Event-based parallelization/multithreading
  - Important for online replay
  - Reduced memory footprint compared to multiple individual jobs
  - Requires thread safe user code (→ only const or protected globals, statics)

- I/O improvements
  - Output system upgrade (full set of data types, object variables) — largely complete
  - TBD: HIPO or PODIO output file format support
  - TBD: EVIO 6 input format support (HIPO-like raw data files)
  - Goal: Make output easily usable with Python and Julia tools (e.g. uproot, UnROOT)

ETA: This summer. Delayed because of work on SBS.
Podd Parallel Processing Prototype

- https://github.com/hansenjo/parallel
- Small standalone toy analyzer with hand-crafted multithreading (std::thread)
- Mimics main components of Podd (e.g. decoder, analysis variables, output)
- A few example “detectors” included whose processing is intended to burn CPU cycles
- Exploring migration to TBB (Intel Thread Building Blocks)
Parallel Podd Performance Scaling Benchmark

- Benchmark processing rate as function of number of analysis threads
- Run on aon11 (16 hyperthreaded cores, Intel Xeon E5-2650 v2 @ 2.60GHz), RHEL 7.9, idle
- Admittedly extreme example: maximally CPU-bound (negligible I/O & memory use)
**Remaining Podd Limitations**

- Algorithms and Data are closely coupled
  - More work to add new algorithms
  - Difficult to stream event data only
- No native event data I/O and API
  - Podd cannot take its own output as input
  - **One-pass analysis only:**
    - EVIO raw data → ROOT trees + histograms
  - Major limitation with large data sets
- **Addressing these would require complete re-write**

---

**Analysis Chain Example**
SBS Online Computing

- Traditional CODA3 DAQ for GMn: Single Event Builder host (new high-performance server), demonstrated $\geq 1 \text{ GB/s}$ peak raw data rate

- Plan to use CODA's scalable event stream parallelization to achieve up to $\approx 3 \text{ GB/s}$

- Online replay on aonlX systems (128 threads), 2014-vintage servers (to be upgraded)

- SBS is the first experiment to take full advantage of these systems, running 100 automated parallel analysis jobs.

- Online replay typically able to keep up with incoming data.
**SBS-GMn Data Volume in Comparison**

### JLab Raw Data to Tape Fall 2021

<table>
<thead>
<tr>
<th>Month</th>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>Hall D</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>2021-08</td>
<td>1,949</td>
<td>1,719</td>
<td>35</td>
<td>1,414</td>
<td>5,117</td>
</tr>
</tbody>
</table>

Fun fact #1: SBS-GMn took ≈ 5 times more data in these 6 months than all prior Hall A experiments combined in 25 years.

Fun fact #2: The entire SBS program expects to accumulate ≈ 25 PB raw data through 2024.
Scientific Computing Resources

- Farm/ifarm upgraded to CentOS 7.9. RHEL 8 clones being evaluated.
- Farm batch system has been transitioned to slurm and swif2. Legacy Auger/swif commands will stop working March 1, 2022. See https://scicomp.jlab.org/docs/FarmUsersGuide.
- Current farm resources
  - Disk: Lustre: 4.1 PB, Work: 1.4 PB (recent upgrade).
  - CPU: 14192 cores / 28384 threads. Total capacity 249 M-core-hours/year
  - Almost half the capacity is on AMD EPYC 7502 64C/128T systems (speed demons!)
  - 6 nodes with Nvidia TitanRTX GPUs dedicated for ML applications
- Mass storage system (as of Feb 2022)
  - Throughput $\approx 8$ GB/s (20 LTO-8 drives, uncompressed, theoretical)
  - $\approx 150$ PB capacity (LTO-8, uncompressed), $\approx 85$ PB used (23.4 raw, 26.7 rawdup).
  - Significant capacity headroom (more frames, LTO-9) with current silo, up to $\approx 325$ PB.
New Counting House Desktop Systems

- Clean separation of desktops and servers. → increased reliability and stability.
- Platform for browsers, editors, slow controls (some), remote logins
- Extensive use of VNC servers/clients, very successful
- No significant issues. Small updates planned (EPICS etc.)
- Feedback welcome (ole@jlab.org)
EPSCI group has offered support to deploy the Hall D Hydra system in Hall A for **automated data quality monitoring**.

Will tap into online histograms generated by panguin.

Currently being set up. Test version expected ≈ March–April.

One-time human review ("labeling") required. Volunteers welcome.
Summary

- “Podd” analysis software continues to be **actively maintained** and used by current experiments in Halls A & C.

- Significant **modernization** work (multithreading etc.) underway.

- The **large data volumes** from SBS are putting Hall A in the same league as Halls B & D in terms of computing resource needs. This will require **careful planning** going forward.

- Experience with the upcoming SBS mass replays on the farm will inform **future direction** of the Hall A software.