Hall-D Offline Software Status

GlueX Collaboration

June 1, 2012

Contents

1 Introduction **3**

2 Simulation **4**
2.1 Event Generators 5
2.2 The Geometry Description—*HDDS* 7
2.3 The core simulation code—*hdgeant* 7
2.4 Detector response simulation—*mcsmear* 9
2.5 Parametric Simulation—*HDParSim* 9

3 Reconstruction **10**
3.1 JANA Framework 10
3.2 Charged Particle Tracking 10
3.3 Calorimetry 12
3.4 Event Reconstruction 13
3.5 Event Viewer 14

4 Physics Analysis **15**
4.1 Event Selection 15
4.2 Amplitude Analysis 16
4.3 A Test Case: $\gamma p \to \pi^+\pi^-\pi^+ n$ 17

5 Calibration **18**
5.1 Calibration and Conditions Databases 20
5.2 Detector Calibration and Alignment 20

6 Software Coordination and Organization **21**
6.1 The GlueX Collaboration 22
6.2 The Offline Working Group 23
6.3 Manpower 24

6.4 Subversion Repository 25
6.5 3rd party Software Packages 27
6.6 Bug Tracking 28

7 CPU, Storage, and Bandwidth Requirements 28
7.1 CPU Requirements 28
7.2 Tape Storage Requirements 29
7.3 Disk Use 31

1 Introduction

Hall-D at JLab was designed primarily for a single experiment, GlueX, with the design and development of both the physics program and the facilities being driven by the user community. The primary goal of the GlueX experiment is to map out the spectrum of exotic hybrid mesons. The GlueX detector in Hall-D has nearly 4π acceptance for both charged particles and photons. In order to carry out its primary physics program, it must be able to fully reconstruct final states involving many particles in a high-rate environment. The primary physics reactions are the photoproduction of mesons off protons using linearly-polarized photons with energy from 8.5 to 9 GeV. Figure 1 shows a cut-away rendering of the photon beam and detector.

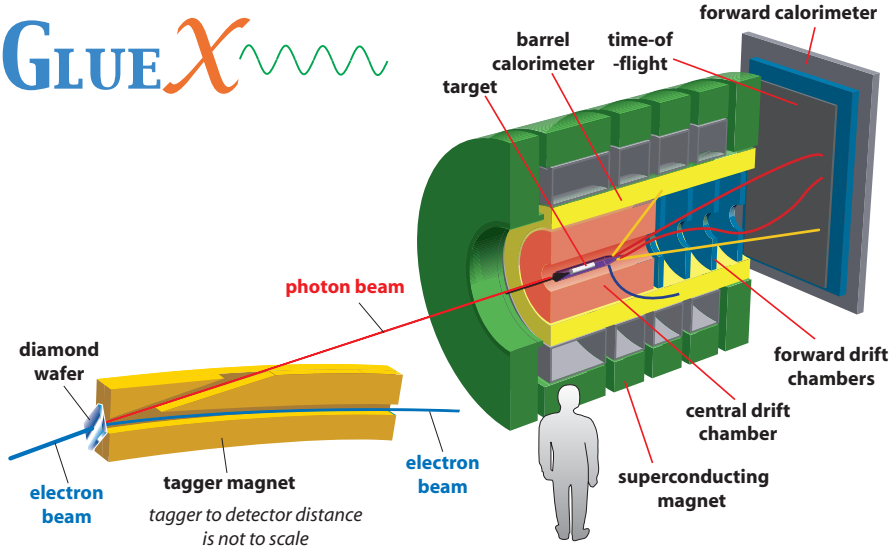

Figure 1: A schematic of the GlueX detector and beam in Hall-D.

The photon beam is derived from 12 GeV electrons impinging on a 20 μm thick diamond crystal that has been accurately aligned to produce linearly polarized $\sim$9 GeV photons via coherent bremsstrahlung. The paths of the recoil electrons are bent in the tagger magnet, and the recoil electrons are then detected in a fine hodoscope which tags the energy of the produced photon. The tagged photons proceed down an 80 m long beam line and pass through a 3.4 mm diameter collimator before entering Hall-D. They then proceed into the GlueX detector where they interact in a 30 cm long liquid hydrogen target. During the initial phases of GlueX running, we anticipate about $10^7\,\gamma/s$ from the coherent bremsstrahlung incident on the target. During later running periods, this rate will be increased towards a $10^8\,\gamma/s$ design limit of the experiment.

Photons interacting in the target will produce final states involving several charged

particles and photons as well as a recoil nucleon. The reactions of interest are discussed further in in Section 4. The GlueX detector is in a 2.2 T solenoidal field that allows us to momentum-analyze the charged particles coming from the interactions. Charged particles pass through a thin scintillator "start counter" just outside the target that is used to provide a start time for the event. These particles are then tracked through the "central drift chamber" which is a 28-layer straw-tube based detector. They then move downstream into the "forward drift chambers" which are multi-plane drift chambers where both the anodes and cathodes are read out to provide space points along the tracks. Finally, the charged particles are detected in a down-stream "time-of-flight" wall. Photons from the decay of mesons such as π^0 and η are also measured by GlueX. Those emitted at angles larger than about 10° are detected in the "barrel calorimeter" which is a lead-scintillating fiber calorimeter with readout on both ends. Those going more forward are seen in the "forward calorimeter" which is an array of lead-glass blocks.

Events with signals in all of these detectors must be fully reconstructed and then given to the physics analysis. The development of reconstruction and analysis software by the GlueX collaboration began in 1998[1]. This has led to a mixture of elements in the software repository, some of which may be considered legacy now (e.g. GEANT3) and some developed to address the current, modern landscape of multi-core computers (e.g. JANA). Experience with other experiments has shaped the collaboration to emphasize software as a major component of the GlueX experiment with considerable effort and resources applied to it early on. The data volumes[2] GlueX will produce are unprecedented at JLab and are comparable to most LHC experiments.

This document presents the computing plan for GlueX/Hall-D and covers both software and the computing hardware needed to analyze the expected data. The document is organized roughly the way that the software is currently used, followed by a description of the software management structure and, finally, estimates of the needed hardware. We first describe the Monte Carlo simulation of the experiment. This is followed by the core reconstruction code and then physics analysis. Finally, calibration procedures are discussed. This document does not cover the online systems (e.g. Data Acquisition).

2 Simulation

The simulation package for GlueX is constructed within a framework that wraps the core simulation code inside of a larger package that handles event generation, geometry and calibrations, and production of "hits" as a part of the simulated data stream. It is written in such a way that multiple core generation packages could be plugged into the same framework. This allows for exactly the same geometry and digitization to be used. The current production system uses GEANT-3 as its core simulation code, but the collabora-

[1] CVS repository entry

[2] estimated at 3PB/yr of raw data

tion has started to transition into using GEANT-4. The entire package of simulation and reconstruction is known as the *sim–recon* package. Within the package, the geometry description is known as *HDDS*[1], the core Monte Carlo is *hdgeant*, and the detector response is *mcsmear*.

2.1 Event Generators

GlueX has several Monte Carlo event generators available for use as part of the *sim–recon* package. These provide a variety of events that are useful for testing software, studying detector efficiency, understanding backgrounds and carrying out physics analysis. Each of these will be described in the following sections.

In addition, the *hdgeant* package has the built-in ability to overlay electromagnetic background on top of simulated events. It does this by simulating photons sampled from the coherent bremsstrahlung spectrum for a configurable time range and a configurable beam rate. Most of these "beam" photons will pass right through the target, but a few will interact with material in the beam line (dominated by the target) giving the correct electromagnetic background.

A tunable particle gun

The simplest event generator is a "built-in" particle gun that allows one to shoot single particles through the detector simulation. The particle type, vertex location, momentum range and angular range can all be controlled by external parameters. This generator is particularly useful for studying particular parts of the detector, or particular event classes.

A PYTHIA-based event generator—*bggen*

The photon beam in GlueX contains not only the coherent photons, but also an incoherent component that extends from zero up to the $12\,GeV$ electron energy. Any photon with energy larger than that needed for single pion production can undergo a hadronic interaction in the GlueX target. The purpose of our "PYTHIA-based" event generator is to simulate all of these hadronic interactions for the GlueX photon beam. Because PYTHIA does not accurately simulate these reactions below $E_\gamma = 3\,GeV$, a modified generator has been built that simulates eleven photoproduction reactions. Above the $3\,GeV$ energy, standard PYTHIA is used. Figure 2 shows the contribution of these channels and the total hadronic cross section as a function the photon energy. This generator is known as *bggen*.

A coherent bremsstrahlung event generator

The coherent bremsstrahlung event generator simulates the coherent photon beam which is produced at the bremsstrahlung target. These photons can then be propagated through

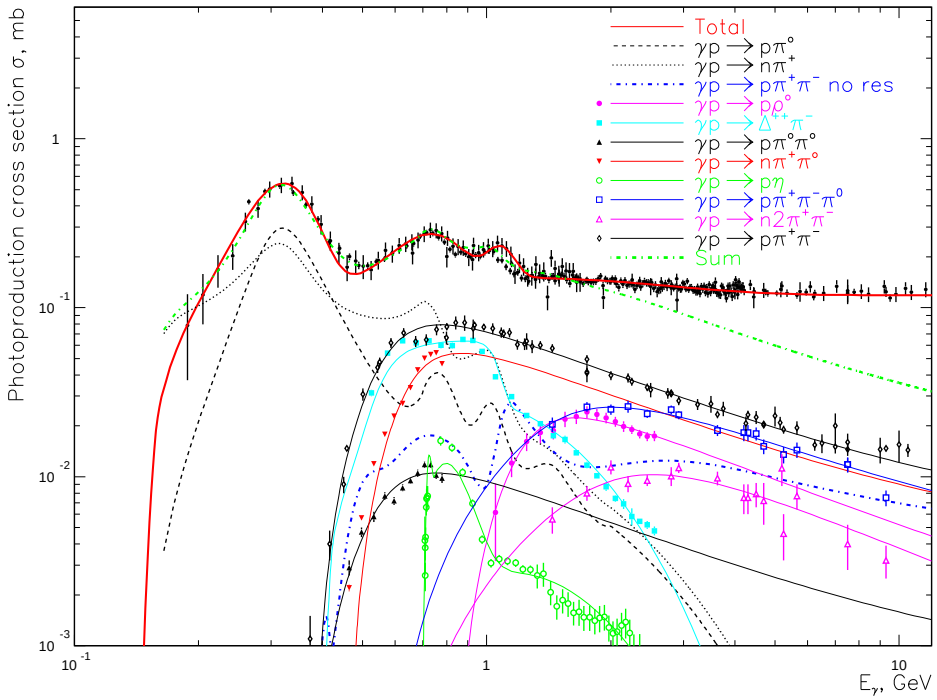

Figure 2: Low energy γp cross-section data along with the distributions produced by the *bggen* generator. Below $E_\gamma = 3GeV$, a mixture of the 11 reactions listed is used. Pythia is used for higher photon energies.

the beam-line simulation to simulate and understand the beam-related and electromagnetic backgrounds that will be observed in the detector.

A simple t-channel process event generator—*genr8*

The physics processes that are of interest in GlueX are t-channel production of mesons. The *genr8* event generator allows the user to specify a photon beam energy and a t-slope for the photoproduction. This is then used to generate a specified meson, which is then allowed to decay via a user-specified decay chain. The decays account for the mass of the particles, but do not include any spin or angular momentum information. For a given value of s and t, it is essentially a phase-space generator.

Physics event generators based on amplitude analysis tools

Going beyond the simple *genr8* generator, it is possible to use the physics analysis tools discussed in Section 4 to produce a set of Monte Carlo events that are weighted by any desired physics amplitude. This generator is needed to be able to test the amplitude analysis as it allows one to fully include spin, angular momentum in the decay of a resonance. It also allows for the inclusion of quantum mechanical interferences between two or more resonances that all decay to the same final state.

2.2 The Geometry Description—*HDDS*

The geometry definition of the GlueX detector for use in both simulation and reconstruction is maintained using the HDDS system. This is a set of XML files baed on the ATLAS AGDD format. In addition multiple tools have been developed that parse the XML and output the information in other formats. Specifically, in GEANT3 compatible FORTRAN code as well as ROOT compatible C++ code. An example of the format can be seen in figure 3.

The reconstruction code also has access to the HDDS geometry via a JANA (see Section 3) interface. The interface allows extraction of any attribute from the XML using xpath[3] formatted strings.

Numerous wiki pages exist that document the HDDS format and describe how to use HDDS. They can be found on the GlueX wiki in the Offline Software HOWTO pages linked from main Offline Software wiki page.

2.3 The core simulation code—*hdgeant*

As noted above, *hdgeant* is the core detailed Monte Carlo that tracks particles through the detector. It allows for the particles to interact with material in the detector, and to record the energy and timing of signals in the active areas of the detector elements.

The GEANT-3 code

The main work-horse of current Hall-D/GlueX simulation is a detailed GEANT-3 code known as *hdgeant*. In order to run this code, it is linked to the geometry subroutines written by the HDDS system, and then reads in events from one ore more event generators. The code includes not only a detailed description of all the detector elements and material, but also a detailed three-dimensional map of the Hall-D solenoid both inside the magnet volume and in the region from the down-stream bore of the magnet to the lead-glass forward calorimeter.

[3]See http://www.w3schools.com/xpath for info of path

ForwardTOF_HDDS.xml

<section name = "ForwardTOF" version = "2.0" date = "2006-11-22" author = "M. Shepherd, R.T....

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DOCTYPE HDDS>

  Hall D Geometry Data Base: Forward TOF counter
  ***********************************************

     version 1.0: Initial version   -rtj

<HDDS specification="v1.0" xmlns="http://www.gluex.org/hdds">
-->

<section name        = "ForwardTOF"
         version     = "2.0"
         date        = "2006-11-22"
         author      = "M. Shepherd, R.T. Jones"
         top_volume  = "FTOF"
         specification = "v1.0">

<!-- Origin of ForwardTOF is center of entrance plane of detector -->

<!-- upstream plane has vertical oriented bars -->
<!-- downstream plane has horizontal oriented bars -->

<!-- Fri Jan 16 08:14:30 EST 2009                                              -->
<!-- changed order of paddle counting: 1-40 bottom to top and south to north   -->
<!-- the half paddles are counted separately 41,43 north halfs, 42,44 south halfs -->

  <composition name="ForwardTOF">
    <posXYZ volume="forwardTOF" X_Y_Z="0.0  0.0 1.27" rot="0 0 -90">
      <plane value="0" />
    </posXYZ>
    <posXYZ volume="forwardTOF" X_Y_Z="0.0  0.0 3.85" rot="0 0 0">
      <plane value="1" />
    </posXYZ>
  </composition>

  <composition name="forwardTOF" envelope="FTOF">
    <posXYZ volume="forwardTOF_bottom" X_Y_Z="0.0 -66.0 0.0" />
    <posXYZ volume="forwardTOF_north" X_Y_Z="+66.0 0.0 0.0" />
    <posXYZ volume="forwardTOF_south" X_Y_Z="-66.0 0.0 0.0" />
    <posXYZ volume="forwardTOF_top" X_Y_Z="0.0 +66.0 0.0" />
  </composition>

<!-- the attribute 'row' is synonymous with 'bar' -->
  <composition name="forwardTOF_top" envelope="FTOT">
    <mposY volume="FTOC" ncopy="20" Z_X="0.0 0.0" Y0="-57.0" dY="6.00">
      <row value="21" step="1" />
      <column value="0" />
    </mposY>
  </composition>
  <composition name="forwardTOF_bottom" envelope="FTOB">
    <mposY volume="FTOC" ncopy="20" Z_X="0.0 0.0" Y0="-57.00" dY="6.00">
      <row value="1" step="1" />
      <column value="0" />
    </mposY>
  </composition>
  <composition name="forwardTOF_north" envelope="FTON">
    <mposY volume="FTOH" ncopy="2" Z_X = "0.0 0.0" Y0="-3.0" dY="6.0">
      <row value="42" step="2" />
      <column value="1" />
    </mposY>
  </composition>
  <composition name="forwardTOF_south" envelope="FTOS">
    <mposY volume="FTOH" ncopy="2" Z_X="0.0 0.0" Y0="-3.0" dY="6.0">
      <row value="41" step="2" />
      <column value="2" />
    </mposY>
  </composition>

  <box name="FTOF" X_Y_Z="252.0 252.0 2.55" material="Air" />
  <box name="FTOB" X_Y_Z="252.0 120.0 2.55" material="Air" />
  <box name="FTOT" X_Y_Z="252.0 120.0 2.55" material="Air" />
  <box name="FTON" X_Y_Z="120.0  12.0 2.55" material="Air" />
  <box name="FTOS" X_Y_Z="120.0  12.0 2.55" material="Air" />
  <box name="FTOC" X_Y_Z="252.0   6.0 2.54" material="Scintillator"
```

Figure 3: Example of HDDS formatted file defining the Forward TOF geometry.

The GEANT-4 code

As noted above, the simulation framework within GlueX allows for the simple replacement of the *hdgeant* code with a GEANT-4 based code. The HDDS package can can also generate geometry objects suitable for use in GEANT-4, so the exact same geometry can be run through both GEANT-3 and GEANT-4. As a collaboration, GlueX recognizes that cernlib may have a limited life as we move forward. There have already been issues of being unable to produce a working copy on one of our primary development platforms[4]. Thus, within the collaboration, work has started on the GEANT-4 based core which should allow us to migrate away from cernlib and other legacy codes that may not be supported in the future. At the time of this report, the work is about one-third done, and it is expected to be completed well in advance of the first beam in Hall-D.

2.4 Detector response simulation—*mcsmear*

Much of the digitization and detector resolution effects of the simulation have been placed in a separate program called *mcsmear*. This allows tuning of these effects in the *mcsmear* code without incurring the overhead of the full simulation at every cycle of the development. Things such as drift time resolution, cathode strip resolution, SiPM dark hits, etc., all are implemented in *mcsmear*. Eventually, this will also apply dead channel and efficiency maps to the simulated data to better reflect the actual detector conditions for a given run period. Such information will be derived from the calibration system, but the mechanism not been developed at this time. This package is common to both the GEANT-3 and GEANT-4 core code.

2.5 Parametric Simulation—*HDParSim*

Outside of the GEANT-based detailed simulation, the collaboration also has a very fast Monte Carlo for carrying out initial studies without incurring the CPU expense of a full GEANT simulation. The GlueX parametric simulation package is called *HDParSim* and has been is integrated into the base *sim-recon* package. *HDParSim* uses tables of resolutions and efficiencies to produce reconstructed parameters based on transverse momentum, θ angle, and particle type. The tables are generated from tracking results on GEANT3 simulated tracks. *HDParSim* is implemented as a plugin that can produce the same type of reconstructed data objects as full tracking reconstruction. This makes it easy to swap between the two Monte Carlo schemes without modification to the analysis code. *HDParSim* does not output full covariance matrices though so detailed analyses that include things like kinematic fitting are not possible with *HDParSim* produced data. This may be added in the future.

[4]We have not been able to build or find a working version of 64bit cernlib for Mac OS X 10.6 or 10.7.

3 Reconstruction

Reconstruction is one of the most manpower intensive parts of the software effort. Its primary function is to extract particle properties (charge, momentum, mass) from the raw data. It needs to combine information from various detector systems and *a priori* knowledge of the detector to calculate these properties in physical units with accurate covariance matrices. Physics analyses can begin only after reconstruction has been performed.

Many collaborators contribute to the reconstruction code base, so a framework is required that allows them to work independently while still maintaining a coherent reconstruction package. Standard software practices such as modularity and reusability are utilized in the Hall-D software. Reconstruction is done using C++ in the JANA framework[2, 3]. JANA is a framework developed at JLab for Hall-D. The framework is designed to allow multi-threaded event-level parallelism. Reconstruction is broken up into several modules called *factories*. Factories use data objects as inputs and produce other data objects as outputs. Figure 4 shows the current relationship between reconstruction factories.

3.1 JANA Framework

JANA implements a data-on-demand paradigm that can improve overall efficiency by limiting the algorithms run on a particular event to only those that are needed and guaranteeing that each unique algorithm is only run one time per event. JANA is designed to allow event-level parallelism via multi-threading using the pthreads package. Each processing thread contains a complete set of factory objects making it capable of completely reconstructing an entire event independent of of threads. JANA has been extensively tested to verify that the rate scales well with the number events on machines containing as many as 48 cores.

3.2 Charged Particle Tracking

Extensive work has been done on charged particle tracking software to date [4, 5, 6]. The goal of the charged particle tracking code is to use the raw hits in the Forward and Central Drift Chambers to determine the momenta of charged particles traversing the field of the solenoidal magnet. The first stage of the tracking reconstruction is the *track finding* or *pattern recognition* stage. Adjacent hits in successive layers of the forward drift chambers are associated together into segments and these segments are linked together to form *track candidates* using a helical model to determine initial guesses for the track parameters. Similarly, adjacent hits in successive CDC axial layers are linked together to form a seed for a circle fit from which an estimate for the transverse momentum can be determined. The seed is then extended into the stereo layers. At this stage the angle of the track relative to the beam line and the z-position at a particular reference radius are determined and a CDC track candidate is formed. In the angular range of $\sim 5 - 20^{\circ}$ with respect to the beam line, a charged particle will produce hits in both the CDC and the FDC.

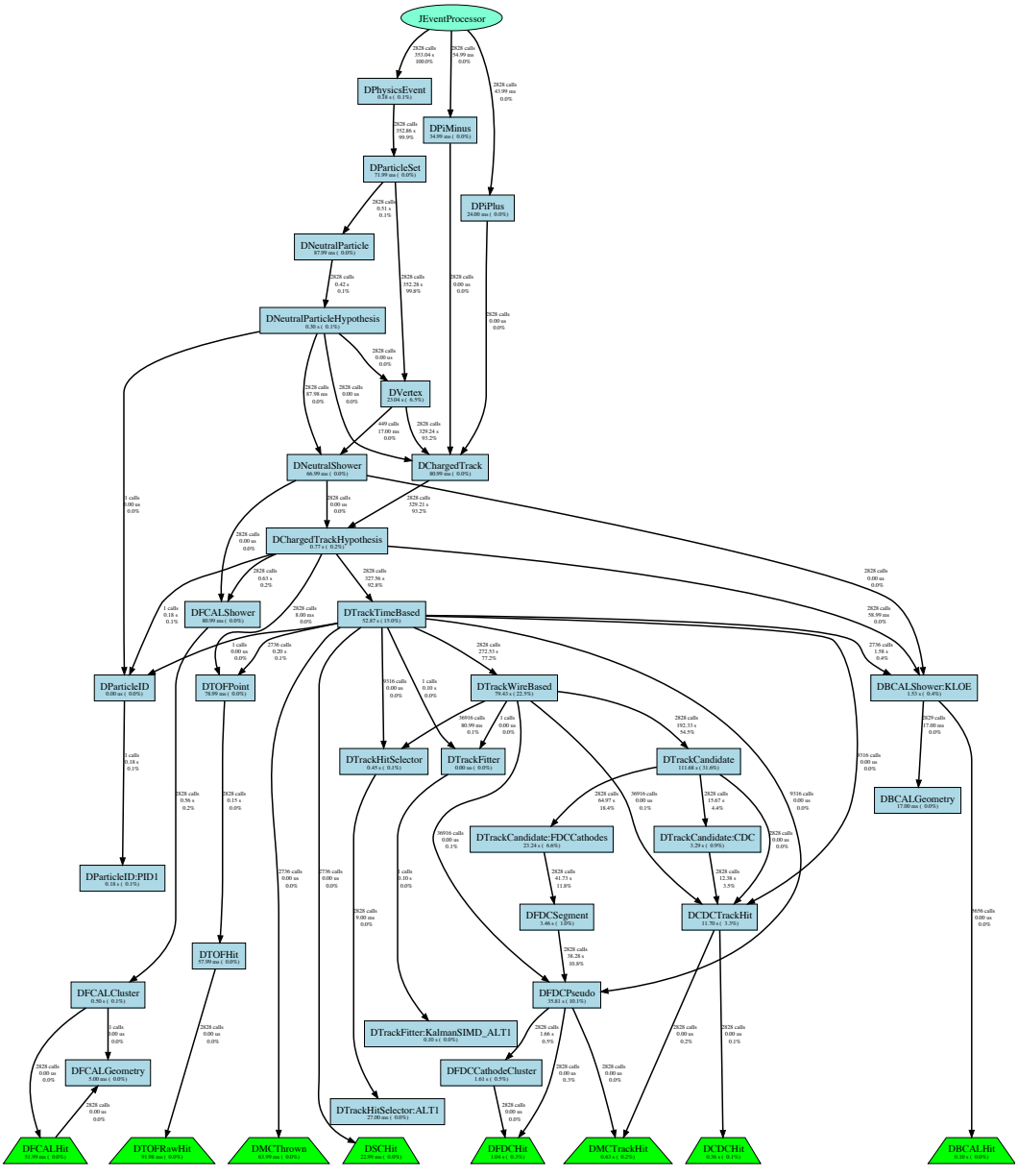

Figure 4: Call graph produced by reconstructing simulated $b_1\pi$ events.

The code finds candidates for FDC and CDC hits separately and matches them to form single candidates where possible. The list of track candidates provides the input to the second stage of the reconstruction: *wire-based fitting*. The rough track parameters (using a helical model) determined by the first stage are used as a seed for the fitting algorithm. We are using a Kalman Filter. At the wire-based stage we do not use the drift-time information from the CDC wires; nevertheless this stage provides an improved guess for the track parameters because we employ knowledge of the full magnetic field (as opposed to assuming a constant magnetic field everywhere – a condition implied by using a helical model at the earlier stage). The FDC provides very precise coordinates along the wires due to the cathode readout; we found that we do not need to use the drift time information from the wire readout to get good momentum resolution. The result of the wire-based stage provides the input to the final fitting stage: *time-based tracking*. Here we use the drift time information from the CDC wires. At this stage we have implemented *hit pruning* and *broken track recovery*. Because the track parameters determined by the earlier stage can be somewhat crude, sometimes hits due to delta rays or hadronic interactions can be associated with the track even though they may be fairly far removed from the "true" trajectory. These extra hits may cause the χ^2 of the track fit to become large or cause the fit to fail entirely. Another source of poorly-fit tracks is a kink in the trajectory due to hard scattering or particle decay. We prune the hits that are too far away from the projected position along the trajectory to be considered consistent with the current trajectory. Since we are primarily interested in the track parameters near the interaction point, if the code detects a kink in the trajectory, it attempts to recover these tracks by dropping the hits beyond the position of the kink and refitting the track with the reduced set of hits closest to the target.

3.3 Calorimetry

Reconstruction code has been written for both the Forward Calorimeter (FCAL) and Barrel Calorimeter (BCAL). The FCAL code is based on algorithms successfully implemented for a similar lead-glass detector used for the Rad-ϕ experiment at JLab. The original BCAL code was directly derived from the KLOE fortran code (converted into C)[7] since the KLOE calorimeter design is similar to that implemented for the GlueX BCAL. A new BCAL reconstruction package is currently under development that was written specifically for GlueX. The BCAL and FCAL packages reconstruct clusters independently with no attempt to combine information for single showers that may have sprayed particles from the end of the BCAL into the FCAL. The fringe field of the magnet is strong enough in that region and the gap between BCAL and FCAL large enough that it has been shown that such reconstruction would not be possible.

The reconstructed showers that are not matched to charged tracks are combined into a single list of DNeutralShowerHypothesis objects. A figure of merit is calculated for shower as being either a photon or a neutron. Neutrons are not reliably reconstructable in the

GlueX calorimeters so showers with a low photon FOM and a high neutron FOM will likely lead to the reconstructed shower (and possibly entire event) being dropped from the analysis.

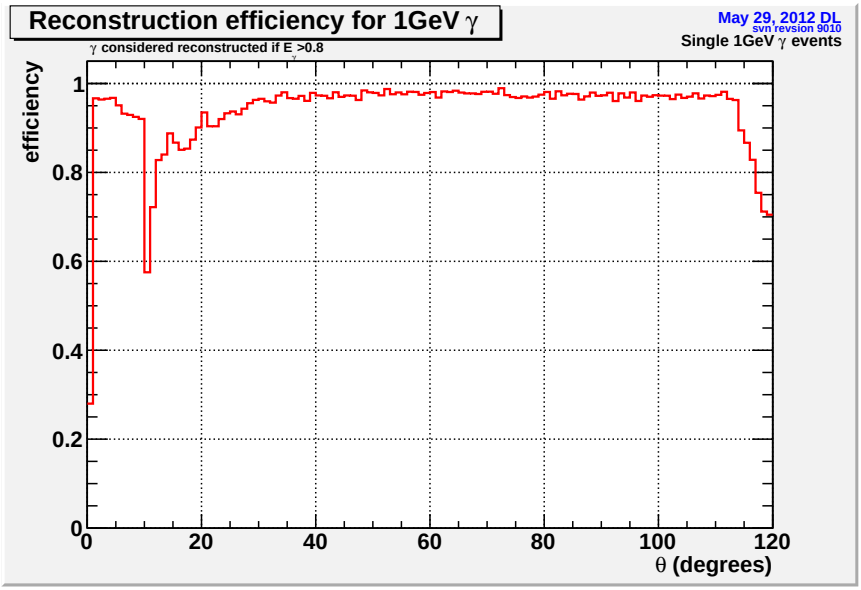

Figure 5: Reconstruction efficiency for 1GeV γs as a function of θ angle. The dip near 11° is due to the boundary between the BCAL and FCAL detectors.

3.4 Event Reconstruction

Full event reconstruction consists of numerous pieces:

- identifying all charged particles,
- identifying whether calorimeter clusters are due to photons or another type of particle,
- grouping particles together that come from the same vertex, and
- grouping vertexes together that belong to the same event.

A complete set of classes have been defined that should allow this information to be represented. Particle ID software generates a confidence level for each charged particle hypothesis and a figure of merit for each non-track-matched calorimeter cluster. Currently, the neutral particle FOM is calculated using the projected and measured time difference. This can be used to help distinguish photons and non-photons. Charged particles use time of flight and dE/dx from the wire chambers.

The default tracking code fits both π^+ and proton mass hypotheses for positive tracks but only π^- for negative tracks. The masses fit are taken from the configuration parameters MASS_HYPOTHESES_POSITIVE and MASS_HYPOTHESES_NEGATIVE which can be set at run time to include other particle types.

3.5 Event Viewer

GlueX has an event viewer that has been used to aid code development since 2005 (see figure 6). The viewer provides four 2-D views of the detector and couples directly into the JANA framework. Various options can be selected and deselected on the fly to inspect a single event. Alternate reconstruction algorithms can be displayed to make it easy to visually compare their output.

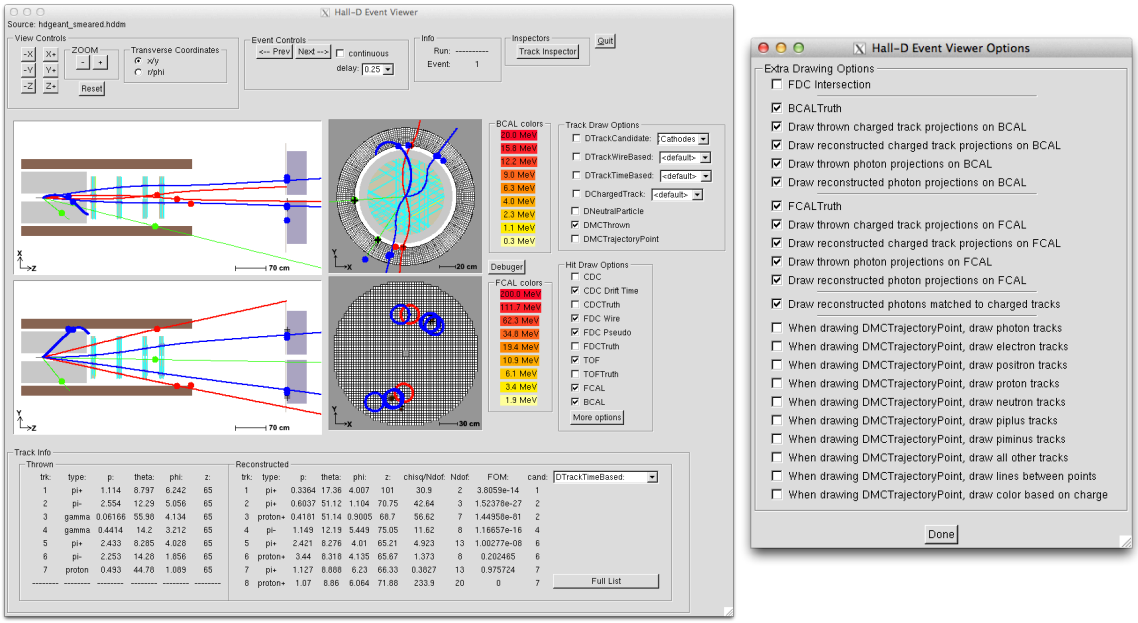

Figure 6: ROOT-based event viewer *hdview*2 and one of its options windows. This has been used extensively for code development, but we are currently exploring the option of using the CLAS12 developed framework *bCNU*.

A second generation event viewer is currently under development based on the CLAS12 event viewer framework: *bCNU*. This framework is written in Java and so cannot be compiled directly into the same executable as the JANA-based reconstruction (C++). However, an effort is being undertaken (summer of 2012) to develop a communication mechanism between the two that will tightly couple the reconstruction and viewer programs to give similar functionality as achieved with *hdview*2.

4 Physics Analysis

In the following section we present a conceptual overview of the envisioned plan for taking the outputs of reconstruction and simulation and conducting a physics analysis with them. Since the core focus of GlueX is spectroscopy and the search for exotic mesons, we will primarily discuss how such searches will take place. The same general analysis principles are applicable to other types of physics analyses.

Briefly, GlueX will study reactions of the type $\gamma p \to X^+ n$ or $\gamma p \to X^0 p$, where X is and intermediate resonance of interest that decays to a collection of stable hadrons. For the initial phases of GlueX running, the stable hadrons of interest are $\pi^\pm$, π^0, η, and η'. The goal is to isolate some collection of stable hadrons, *e.g.,* $\pi^+\pi^-\pi^+$, and then study the initial state and intermediate resonances. For example, the final state $\pi^+\pi^-\pi^+$ may be populated by decays $a_2^+ \to f_2\pi^+,\ f_2 \to \pi^+\pi^-$ or $\pi_2^+ \to \rho^0\pi^+,\ \rho^0 \to \pi^+\pi^-$. For any collection of final state particles we want to identify all such initial states X and measure the quantum numbers (total angular momentum J, parity P, and charge conjugation C) of X. The quantum numbers of the initial state are determined by performing and unbinned maximum likelihood fit to the angular distributions of the data, this is discussed in more detail below. In summary the analysis process has two key steps: (1) the selection of a signal-rich sample of events in which a particular collection of final state hadrons is produced, and (2) the subsequent "amplitude analysis" of the angular distributions of these events in order to extract intermediate resonances and their quantum numbers.

4.1 Event Selection

After the raw data events are reconstructed they will be classified into several categories. While the classification may become detailed at later stages of the experiment, we expect initial classification to require that there be a tagged incident photon that has energy that is approximately equal to the observable energy in the detector. Because of the fact that the level 1 hardware trigger for the experiment is very loose and many photons are produced outside the tagging range we expect that about 10% of all triggered events will satisfy this initial selection criterion, and these events will be the starting point for physics analysis by members of the collaboration. This sample of events is expected to be resident on disk and all analyzers will conduct their own skims of this data. We plan to retain the minimum amount of high level information in order allow complete analysis of the event while reducing the disk footprint.

The next stage in the analysis process will be to define selection criteria that isolate a particular set of final state particles. In order to avoid unintentional bias at this stage in the analysis we plan to use samples of inclusive Monte Carlo generated with the version of Pythia tuned for photoproduction, *bggen*. As results emerge from GlueX, it is expected that this generator will be refined to more accurately reflect the true background reactions. We will have a sample of Pythia-generated photoproduction events available for analysis

that is at least a factor of two larger than the data sample collected with the experiment. This sample will allow users to understand the dominant backgrounds for a particular analysis and develop event selection criteria to minimize those backgrounds.

At this stage of the analysis, it is expected that each analyzer may need to make up to ten passes through the entire sample in order to refine the selection criteria for a particular analysis. While the exact mechanism for conducting and managing these "skims" of the data has yet to be developed we would like to efficiently accommodate a variety of user preferences. We are examining systems such at EventStore [8], used by CLEO-c, which essentially provides a mechanism to efficiently index events and provide random access to lists of events. While this system would reduce disk footprint of a skim, it would require the user to perform analysis tasks, for example, generating a histogram of a variable, inside the JANA framework. On the other end of the spectrum would be a skim that produces a traditional n-tuple that be easily manipulated with a package such as ROOT. Such an approach results in duplication of data, but it tends to make subsequent analysis tasks easier for some users and would be ideal for sparse skims of the data. By having several options available, users may choose the technique that provides highest efficiency for a particular analysis.

4.2 Amplitude Analysis

Once an analyzer has decided on a set of event selection criteria to select a final state of interest, the amplitude analysis procedure can be started. The goal of amplitude analysis is to use all of the physical observables, *e.g.,* decay angles and invariant masses, to extract information about the intermediate resonances. In order to do this, one constructs a model probability density function that describes the density of events in the muti-dimensional phase space of observables. This model contains free parameters, which are typically production amplitudes, masses, or widths of various resonances that are determined through an unbinned maximum likelihood fit to the data. Typically three collections of events are input to the fit: (a) the actual data that pass all event selection criteria; (b) a sample of generated signal events, uniform in phase space, that is several times larger than the data; and (c) the sample (b) after it has been subjected to the event selection criteria used to select the data events. The Monte Carlo samples (b) and (c) are used to incorporate the acceptance of the detector and event selection algorithm into the physics model that describes the decay. It is expected that the amplitude analysis process will be repeated many times during the course of an analysis as the analyzer systematically tries different parameterizations of the physics model. Some analyses may require hundreds or thousands of fits to performed to the data to evaluate systematic uncertainties in the analysis.

As expected, performing an unbinned likelihood fit with many parameters to a large set of data presents a computational challenge. However, the problem lends itself well to parallel computing since evaluating the log likelihood at each fit iteration reduces to computing several large sums over the input event samples. Each term in the sum is essentially

the probability of producing a given event subject to the values of the fit parameters for that iteration. Once the event samples have been distributed to multiple host machines these sums can be computed in parallel. The only information exchange between the "fit manager" and the compute nodes is then the values of partial sums and updated sets of parameters. For large data samples, the fit time scales like $1/N$ where N is the number of nodes used to compute sums. Recently, graphical processing units (GPUs) have been utilized as an economical means of performing the parallel computing needed for amplitude analysis. Commodity GPUs available for several hundred dollars have provided one to two orders of magnitude increase in the speed at which amplitude analysis fits can be performed. Large data sets can be spread over multiple GPUs hosted by multiple CPUs. In such a configuration the popular Message Passing Interface (MPI) toolkit for parallel processing can be used to conduct fits on multiple GPUs simultaneously. The collaboration has developed an amplitude analysis framework for performing such fits. Given the easy access to GPU hardware, it is expected that most collaborating universities will be able to accomplish amplitude analysis tasks with a modest collection of GPU resources at their home institution. For each event, one only needs to input the four vectors of the final state particles into the amplitude analysis software; therefore, the disk size of the event samples is comparatively small (tens to hundreds of GB), which will facilitate easy analysis away from the centralized Jefferson Lab computing resources.

4.3 A Test Case: $\gamma p \to \pi^+\pi^-\pi^+ n$

In order to test the GlueX analysis framework, we conducted an amplitude analysis of mock data to study the $\pi^+\pi^-\pi^+$ system produced in γp collisions. The event sample corresponded to what we might expect to accumulate in several hours of data taking at beam intensities comparable to those planned for the first production physics runs at GlueX. The Pythia-based generator, *bggen*, was used to generate inclusive γp photoproduction at $E_\gamma = 9$ GeV. The signal events ($\gamma p \to \pi^+\pi^-\pi^+ n$) were generated at a level of about 2.5% of the total hadronic cross section. After optimizing all analysis criteria a signal selection efficiency of 25% and a signal-to-background ratio of 2:1 were achieved. About 20% of the total background originated from kaons misidentified as pions. The other backgrounds included protons being misidentified as pions or extra π^0's in the event that went undetected. This study, conducted in 2011, motivated a more detailed simulation of particle identification systems and tracking resolution along with enhancements in tracking efficiency. This work is still under development, and we expect that these enhanced algorithms along with improvements in analysis technique, such as kinematic fitting, will provide at least an order of magnitude further background suppression. Reducing the background to the percent level is essential for enhancing sensitivity in the amplitude analysis.

The sensitivity to small amplitudes that is provided by the GlueX detector acceptance and resolution was tested by performing an amplitude analysis on a sample of purely generated $\gamma p \to \pi^+\pi^-\pi^+ n$ events that has been subjected to full detector simulation and

reconstruction as discussed above. Several conventional resonances, the a_1, π_2, and a_2, were generated along with a small ($< 2\%$) component of exotic π_1. The result of the fit is shown in Figure 7. In such a fit, the data are divided into bins of 3π invariant mass. Each bin is fit independently, and the fit parameters are production amplitudes and phases of the different resonances that ultimately populate the 3π final state. In this mock data sample, all of the 3π resonances are modeled by a simple Breit-Wigner, and one can see that the both the Breit-Wigner lineshape and phase can be extracted from the small exotic wave decaying into 3π as well as the other, dominant resonances. This study indicates that with a pure sample of reconstructed decays, the GlueX detector provides excellent sensitivity to rare exotic decays. The analysis sensitivity will ultimately be limited by the ability to suppress and parametrize backgrounds in the amplitude analysis that arise from improperly reconstructed events, as noted above.

The analysis of $\gamma p \to \pi^+\pi^-\pi^+ n$ represents the first full end-to-end analysis of simulated GlueX data. While the statistics are only a fraction of what we expect to collect with the final experiment, the exercise has motivated continued refinement of the reconstruction algorithms. Similar studies with other physics channels are also underway by members of the collaboration, and the preliminary results demonstrate a notable improvement in efficiency and background rejection over that presented above. We expect this process to continue over the next three years, leading up to the first GlueX data. As the final production computing resources become available we plan to increase the scale of our mock analyses in order to ensure that we have an analysis framework that is capable of meeting the demands of the experiment.

5 Calibration

In order to be able to extract physics from the data collected in GlueX, it is necessary to have procedures in place to calibrate the detector elements. Calibration data must be stored in a robust calibration database that both allows easy access to the correct calibrations as well as a simple mechanism for updating these as a function of run conditions and run periods. While the database has been designed for GlueX, the actual calibration procedures are just starting development. These procedures require both good understanding of the actual hardware as well as a reasonable robust reconstruction code and framework. These have now reached a mature enough point that work on the calibration code can sensibly start. Developing the calibration software is estimated to be the largest remaining offline software effort to complete in terms of estimated FTE-years (see Section 6.3).

The JANA framework provides a well-defined API for accessing calibration constants from the reconstruction code[9]. In addition, a primitive command-line tool exists as part of JANA that allows one to browse the database using the same access mechanism as the reconstruction code. A simple ASCII based system was implemented that has been used for code development up to this point. A full-featured calibration database has been

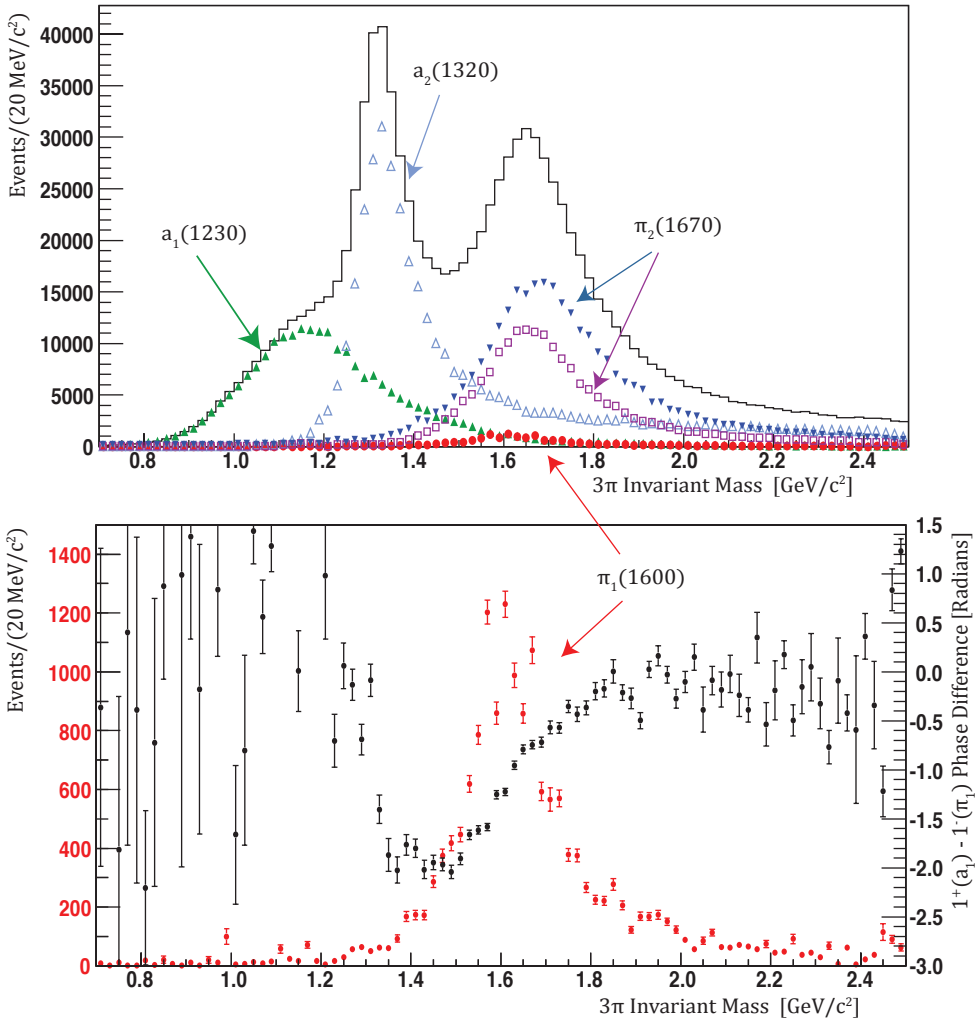

Figure 7: A sample amplitude analysis result for the $\gamma p \to \pi^+\pi^-\pi^+ n$ channel with GlueX. (top) The invariant mass spectrum as a function of $M(\pi^+\pi^-\pi^+)$ is shown by the solid histogram. The results of the amplitude decomposition into resonant components in each bin is shown with points and error bars. (bottom) The exotic amplitude, generated at a relative strength of 1.6%, is cleanly extracted (red points). The black points show the phase between the π_1 and a_1 amplitudes.

designed[10] and written. It is described in section 5.1.

5.1 Calibration and Conditions Databases

The Calibration and Conditions DataBase (CCDB) was designed based largely on experience with CLAS at JLab. The design accommodates the JANA API making it easy to begin using it with the sim-recon package. The CCDB can use multiple backends, but has been tested and will be used primarily with MySQL.

5.2 Detector Calibration and Alignment

Planning for the calibration procedure for each of the major detector systems is currently underway with the work being carried out within the relevant technical working group (see Section 6) dealing with the specific hardware. There is also activity within the software groups to develop tools that will be needed for these calibration and alignment procedures. In the following, we note the current state for the major detector systems.

Beam line

Members of the GlueX collaboration developed the beam-line calibration procedures for photon running in Hall-B during the 6-GeV era. This included tools to determine photon flux, photon energy and the degree of photon polarization. While we feel that it will be relatively straightforward to implement the older Hall-B procedures in GlueX, the relevant individuals have been focussed on improving hardware to eliminate some of the problems that hindered this work in Hall-B.

Central drift chamber

The central drift chamber (CDC) needs to be both accurately aligned with the other tracking elements, but also include accurate time-to-distance relations for converting measured drift times to coordinates along the track. We will also need to be able to track changing conditions within the detector, such as pressure and temperature, that may affect these calibrations. Such procedures were developed and used on a small prototype built in preparation for the final detector, and the results of this work have been published [11]. Work is just now starting on implementing these procedures for the CDC and should be ready to test when the chamber is installed in Hall-D in summer of 2013.

Forward drift chamber

The forward drift chambers (FDC) face similar issues as the CDC, and similar work has been carried out with prototypes. It is expected post-survey alignment will be done using photon beam with the magnetic field off. The resulting "straight-line tracks" can then

be used to align both the individual FDC packages to each other, but also the relative alignment of the CDC and FDC. The code to carry out this straight-line tracking has recently been written.

Forward calorimeter

Currently, the forward calorimeter (FCAL) is the most advanced in calibration; a full calibration has been performed using simulated data. The procedure is fairly standard in calorimeters, and will utilize π°s where both decay photons are observed in the FCAL. Constants for individual channels are then tuned to optimize the resulting two-pion invariant mass at the mass of the π°. The work remaining is to integrate it into the JANA framework and the calibration database.

Barrel calorimeter

Work on the the calibration of the barrel calorimeter (BCAL) is only beginning, although it is anticipated that a similar procedure as is done in the FCAL will ultimately be used. Studies need to be carried out to determine the best choice of photons to use.

Time of flight wall

The time of flight wall (TOF) is a scintillator a pair of hodoscopes, both of which are read out at both ends. Prototypes have been calibrated to the needed accuracy for GlueX, but the final procedure for the pair of scintillator arrays has not yet been worked out.

6 Software Coordination and Organization

This section presents the details of organization, staff resources, and development tools relevant to the offline software in GlueX and Hall-D. The efforts are coordinated within the GlueX collaboration, and, at present, nearly all groups are involved at some level in the software effort, either directly in its development or through physics analysis to test the performance and identify areas that need work. As will be noted, the collaboration has a good picture of what fraction of the various software elements described earlier in this document are done and what effort is needed to finish things. There also appears to be a reasonable match between the manpower available to complete these tasks and the expected effort needed to complete them. As will be seen, roughly one-half of all the software is completed, but some elaboration is needed here. The majority of the remaining work is in calibration procedures, while the development of simulation and reconstruction software is quite mature. Most of the remaining work requires modifications to multiple parts of the code to improve overall perfromance. Thus, the feeling of the collaboration is that the elements of software with the largest technical "risk" associated with them are

nearly complete, while the work that remains is for the most part implementation and integration of known calibration procedures.

6.1 The GlueX Collaboration

Software development is coordinated within the existing management structure of the GlueX collaboration. The collaboration is headed by an elected spokesperson who works closely with the Jefferson Lab Hall-D group leader. The spokesperson also chooses a deputy spokesperson who is then vetted by the collaboration. These three people form the *executive group* in the collaboration. In addition the the executive group, there is a six-member elected *collaboration board* that serves primarily in an advisory role. The actual work within the collaboration is carried out by the technical working groups. This structure including the current working groups is sketched out in Figure 8. Groups can be added or eliminated by the Spokesperson, with the current set reflecting the fact the the experiment is being built.

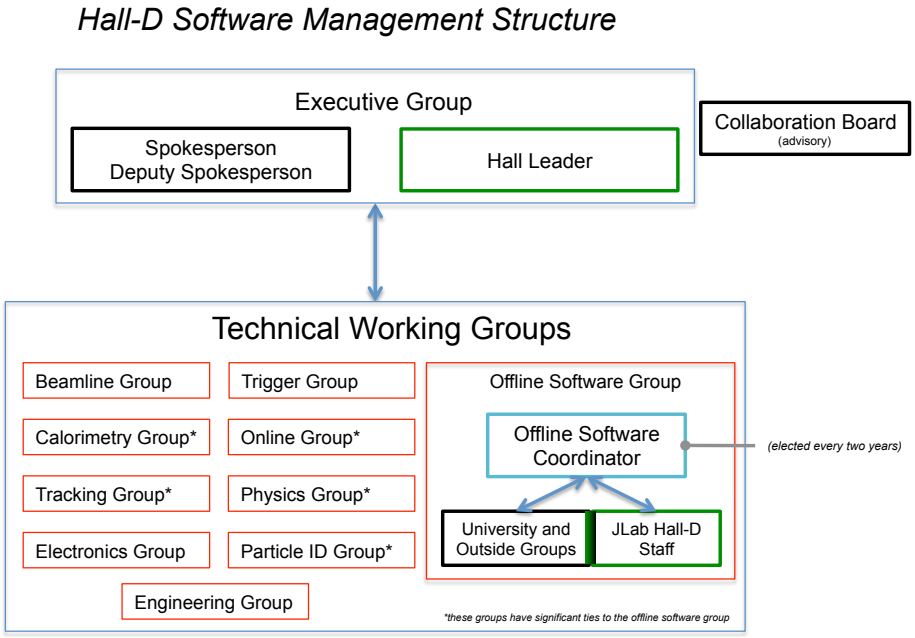

Figure 8: GlueX Collaboration Management Structure with the Offline Software Working Group emphasized.

While membership in the GlueX collaboration is formal, membership in any of the technical working groups by collaboration members is not. Any collaboration member wishing to participate in meetings and either speak or vote on working group decisions is welcome to do so at any time. The working groups have overlapping memberships with

most collaborators participating in two or more working groups. The collaboration is also in regular communication through video conferences at all levels. Each working group holds a video conference once every two weeks where detailed issues are discussed and critical decisions made. Most of these working group meetings also have at least one member from the executive group present. The collaboration holds a video conference every two weeks where all the technical working groups report. Finally, the collaboration holds meetings at Jefferson Lab three times a year where more detailed information and discussion occur. It is also possible to participate in these collaboration meetings via video conference as well.

6.2 The Offline Working Group

The primary offline activities are carried out in the "Offline Software Working Group" which is responsible for coordinating the development of the offline software and responding to issues that arise when it is used. The group also organizes workshops and tutorials on use of the GlueX software and maintains a wiki-based "how-to list". Because of these activities, the offline group works closely with several other working groups in the collaboration. In Figure 8, the offline group is called out in the larger box and the other groups that are involved in this effort are marked with an asterix ($*$).

The offline working group is led by the Software Coordinator who is an individual from the collaboration elected every two years by participants of the group. The responsibilities of the Software Coordinator are detailed as follows.

The primary responsibility is the overall coordination of the offline software effort. This includes coordinating with other working groups on software issues as well as the actual software development. The coordinator sets and enforces software related policies to maintain suitable standards. This occurs both by implementing "consensus policies" and making unilateral decisions in the case of unresolvable controversy. The coordinator is also responsible for watching the software repository and notifying responsible parties when build problems occur. The coordinator is also the primary contact for problems encountered in the software and maintains the offline software wiki page.

The coordinator organizes and chairs the biweekly offline meetings and reports of the groups activities at the larger collaboration-wide meetings, both the biweekly video conferences and the collaboration meetings. Finally, the offline coordinator is the primary person for maintenance of the software subversion repository. This work includes periodically checking out and building the software, and if this fails, getting the appropriate person to fix the issue. The Coordinator also creates and releases tagged versions of the software on a regular basis and maintains "hook scripts" and the "build systems" for the software.

6.3 Manpower

Manpower commitments dedicated to software from collaborating institutions have been gathered and condensed into a form that estimates the annual available manpower over the next 3 years (2012-2014). Many of the commitments are not backed by a formal MOU, but do represent the collaboration's good faith estimate of the software manpower that will be available. The estimates have been broken down into different categories of workers as seen in Figures 9 and 10. Figure 9 contains an "efficiency factor" as a means to normalize the units of FTEs into units of useful work. This accounts for the expectation that 1 hour of undergraduate time will generally not be as productive as 1 hour of a full professor's time due to the difference in experience levels. The efficiency factors are subjectively derived, but have an overall effect of lowering the total manpower estimates by 18%.[5] The first real physics beam is expected in 2015, so over the next three years, we estimate that about 23 FTE-years are available to work on software projects.

Raw Totals

	undergrad.	grad. student	Post-doc	professor	staff scientist	technical assistant	research associate
2012	0.59	3.23	0.75	0.86	2.27	0.05	0.05
2013	0.59	4.28	0.75	1.03	2.18	0.05	0.09
2014	0.59	5.46	1.70	1.07	2.09	0.05	0.09
						total	**27.82**

Adjusted Totals

	undergrad.	grad. student	Post-doc	professor	staff scientist	technical assistant	research associate
2012	0.30	2.42	0.75	0.64	2.27	0.03	0.05
2013	0.30	3.21	0.75	0.77	2.18	0.03	0.09
2014	0.30	4.10	1.70	0.80	2.09	0.03	0.09
efficiency factor	0.5	0.75	1	0.75	1	0.75	1
						total	**22.92**

Figure 9: Spreadsheet summarizing the manpower contributions for software over three calendar years.

In addition to the available manpower, the progress on software tasks is also maintained in a spread sheet[6]. This is shown in Figure 11 where the basic tasks are listed. Also given

[5]The spreadsheet where these numbers are kept is maintained in the subversion repository and can be found here: https://halldsvn.jlab.org/repos/trunk/docs/offline/ProjectProgress/manpower_survey.ods

[6]The spreadsheet is maintained in the source code repository here: https://halldsvn.jlab.org/repos/trunk/docs/offline/ProjectProgress/OfflineComputingActivities2012.xlsx

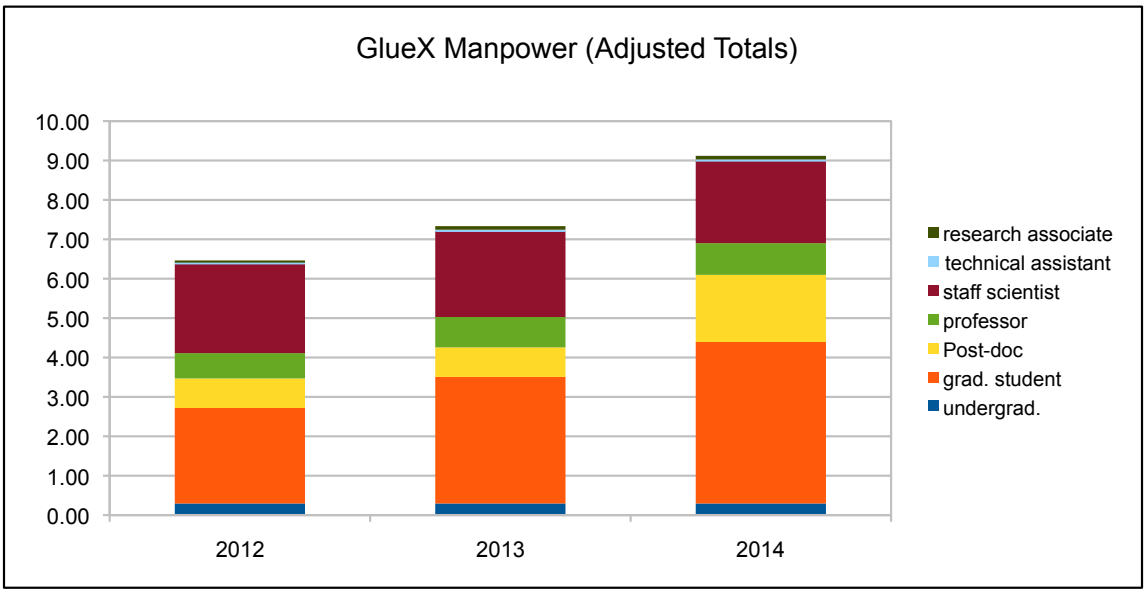

Figure 10: Bar chart summarizing the data in the "Adjusted Total" table shown in fig. 9. Only the adjusted totals are shown, where efficiency factors based on worker type are applied to estimate the effective available manpower.

are the estimated effort needed, the percent complete and the person responsible for the task. From the table, the total software effort is estimated to be 35.5 FTE-years and with 50% done, we anticipate needing an additional 18 FTE-years. This matches well with the 23 available within the collaboration. Figure 12 shows pie charts of both the total effort needed on each part of the software as well as what is remaining.

6.4 Subversion Repository

Hall-D software is stored in a subversion repository[7]. The repository uses SSL to provide secure, web-based access from anywhere in the world. The URL can be used by a web-browser to browse the code or a subversion client to access the repository. Anyone can check out the code anonymously, but a username and password to an active JLab CUE account that is a member of the "halld" unix group is required to check anything in. The JLab IT division maintains the filesystem holding the repository with regular backups. They also maintain the web server that provides access to the repository.

The structure of the repository is set up to keep the large core of offline software in a package called *sim-recon*. The detector geometry is maintained in a separate package called *hdds*. Early code development also uses calibration constants stored in text files and

[7]The repository is located at *https://halldsvn.jlab.org/repos*.

	Budgeted Labor Units (MW)	FTE-years	% complete	Responsible Institution	Responsible Persons	fraction of project
GEANT 3 simulation	88	2.0	100%	UConn	Richard Jones	5.6%
GEANT 4 simulation	88	2.0	0%			
DAQ to Detector Translation Table	44	1.0	5%	JLab		2.8%
Reconstruction	495	11.3	66%			31.7%
Reconstruction Framework	44	1.1	81%	JLab	David Lawrence	
CDC Reconstruction	33	0.9	78%	JLab	David Lawrence	
FDC Reconstruction	33	1.1	73%	JLab	Simon Taylor	
Track Finding	66	2.0	75%	JLab/CMU	Simon Taylor/David Lawrence	
Track Fitting	66	3.0	67%	JLab/CMU	S. Taylor/D. Lawrence/P. Mattione	
BCal Reconstruction	44	1.0	50%	IU/Regina	Matt Shepherd/Zisis Papandreou	
FCal Reconstruction	33	0.8	75%	IU/UConn	Matt Shepherd/Richard Jones	
TOF Reconstruction	33	0.8	50%	FSU	Paul Eugenio	
Tagger Reconstruction	33	0.8	0%	UConn/CUA	Richard Jones	
Start Counter Reconstruction	22	0.5	50%	FIU	Simon Taylor/Werner Boeglin	
Particle ID	44	1.0	75%	CMU/JLab	Paul Mattione	
Kinematic Fitter	44	1.0	95%	MIT/CMU	Mike Williams	
Calibration	242	5.5	23%			15.5%
Calibration Database	33	0.8	80%	MEPHI/JLab	Dmitry Romanov	
CDC Calibration	33	0.8	5%	CMU	Naomi Jarvis	
FDC Calibration	33	0.8	0%	JLab	Lubomir Pentchev/Simon Taylor	
BCal Calibration	33	0.8	0%	Regina	Zisis Papandreou	
FCal Calibration	33	0.8	80%	IU	Claire Tarbert/John Leckey	
Tagger Calibration	33	0.8	0%	UConn/CUA	Franz Klein	
Starter Counter Calibration	22	0.5	0%	FIU	Werner Boeglin	
TOF Calibration	22	0.5	0%	FSU	Alexander Ostrovidov	
DST Generation	132	3.0	11%			8.5%
Data format	44	1.0	33%	UConn/JLab		
Micro DST Writer	22	0.5	0%	UConn		
Job Control Reconstruction	33	0.8	0%	JLab/CMU/UConn		
Job Control/Database for Simulation	33	0.8	0%	UConn		
Analysis	220	5.0	54%			14.1%
PWA Development	132	3.0	90%	IU/CMU/MIT	Shepherd/Mitchell/Meyer/M. Williams	
PWA Challenge	44	1.0	0%	IU/CMU/MIT	Shepherd/Mitchell/Meyer/M. Williams	
Grid Implementation	44	1.0	0%	UConn	Richard Jones	
Misc.	341	7.8	50%			21.8%
Event Viewer (adapted from online)	22	0.5	50%	CNU/JLab	David Lawrence	
Documentation	88	2.0	40%	all	*multiple*	
MC Studies for Detector Optimization	132	3.0	95%	all	*multiple*	
Integration of Slow Controls	33	0.8	0%	JLab	Elliott Wolin/Hovanes Egiyan	
Integration/QC	44	1.0	0%	JLab		
Coordination	22	0.5	0%	JLab	Mark Ito	
	Man-weeks	FTE-years				
Total	**1562.0**	35.5				100.0%

Figure 11: Offline software activity schedule. This is a snapshot of the spreadsheet where this information is kept and tracked. See text for location where spreadsheet is maintained.

stored in the repository in a package called *calib*. All of these are required to build the simulation and reconstruction software for Hall-D. Versions of these are tagged separately (see Section 6.4).

The reconstruction code is written using the JANA framework described in section 3.1. The JANA framework is maintained in a separate repository that was set up to hold

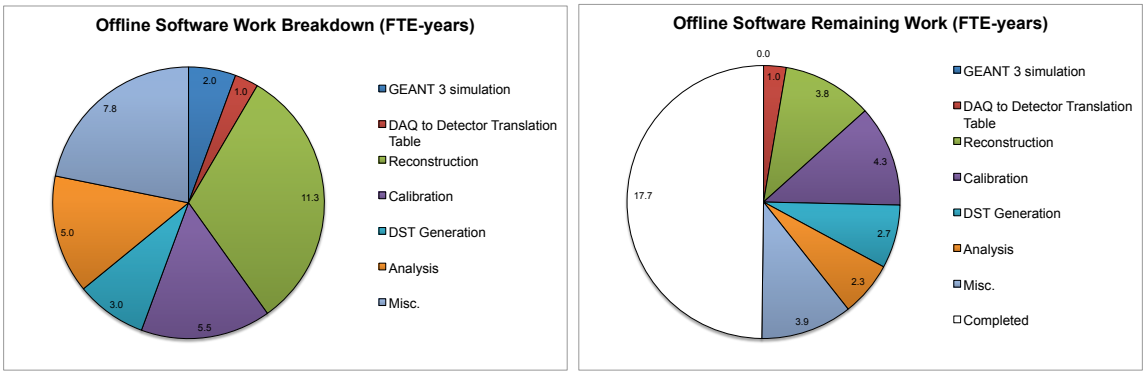

Figure 12: Software manpower needs for Hall-D. The chart on the left indicates the fractions of the overall project for the listed software categories based on the activity schedule shown in figure 11. The chart on the right separates work that has already been completed for each category indicating the breakdown of work still needed.

software used in common with other experimental halls[8]. Currently, only Hall-D uses JANA, but its design is kept free of Hall-D specific code to facilitate others use of it.

All major software packages in the repository are tagged periodically in order to maintain standard versions by which simulation studies may be compared. Tagged versions are created on an as-needed basis, but this tends to happen about once per month. Tagged revisions are named based on the date on which the tag was made. For example:

sim-recon-2012-03-12.

6.5 3rd party Software Packages

We utilize several 3rd party software packages in the Hall-D software base. These are:

- GEANT3
- XERCES (XML Parser)
- ROOT
- CLHEP

New packages are scrutinized carefully to try and ensure that they will be supported for the length of the GlueX experiment and will bring value to the code base. These packages are maintained by the physicists and users outside of the IT division.

[8]The common repository is located at *https://phys12svn.jlab.org/repos.*

6.6 Bug Tracking

To help track bugs and feature requests we utilize the web-based Mantis system[9]. Multiple projects are tracked using the system, but issues such as Offline software can be isolated using features of the Mantis system. The Mantis DB is reviewed at every Offline Software working group meeting.

7 CPU, Storage, and Bandwidth Requirements

7.1 CPU Requirements

The scale of the computing resources need to analyze GlueX data is set principally by the trigger rate, running time, the size of events, and the time it takes to reconstruct an event. In addition to the real data, simulated data sets will have to be generated to calculate efficiencies and study systematic effects in the real data. Any estimate of computing must take into account this simulation. Statistical errors from analysis of the simulated data must be comparable or preferably smaller that that coming from the real data, therefore the amount of simulated data is also driven by the raw data rate.

The GlueX hardware trigger is designed to accept the entire hadronic rate in the hydrogen target. The hardware trigger rate is therefore set by the beam intensity and the hadronic cross section. For Phase II and Phase III running, at 10^7 γ/s, this means a rate of about 20 kHz.

In later phases of running we will have higher beam intensity but will also have a Level-3 software trigger that will keep the rate being written to tape close to that of Phases II and III. As a result many of the assumptions that apply to these early phases will hold at least approximately for later GlueX running. For early phase running, the computing farm infrastructure for a Level-3 trigger farm will exist, but not with the computing power necessary for the high rate running in later phases when a software trigger decision must be rendered for every event. Initially the farm will be used for data monitoring and to prototype trigger algorithms in a mark-and-pass (non-cutting) mode.

The time to reconstruct a simulated event has been measured on a 2.8 GHz Nehalem processor to be 133 ms. The measurement was done on a sample of minimum-bias events, including all significant sources of hadron photoproduction on the proton. A simulated hardware trigger was applied to the generated data sample before inclusion in the sample.

To make an estimate of the amount of CPU power required we take a steady-state model, based on Phase III running assumptions. We assume that GlueX data is being produced at an average rate which takes into account running efficiency and machine shutdown periods and that this goes on *ad infinitum*. We enumerate all of the computing tasks that are generated from this incoming data stream, including the generation of simulated data and any repetition factors (see below) for real or simulated data processing, and calculate

[9]The Hall-D Mantis server can be accessed here: *https://halldweb1.jlab.org/mantisbt*.

the resulting rate of consumption of computing resources. Any offline compute complex must provide this rate of computing or the data taking will overtake the computing and an ever lengthening backlog will develop, year-over-year. Capacity higher than this rate means that there will be idle time on some compute nodes, but the latency for each step will be reduced. Note that we do not make any assumptions about the latency of any component step; these are set by requiring a rate of event processing that keeps up. We view this level of computing power as an acceptable lower limit on the size of the offline farm.

Commonly, large-scale reconstruction of the real data, as well as simulation and/or reconstruction of Monte Carlo data, is done more than once. Each iteration generates lessons for the next. We account for this possibility with a independent repetition factor for each step in our estimate, *e.g.,* every raw data event will have to be reconstructed twice. In other words, maintaining steady-state almost certainly means having enough computing to do some things more than once.

In addition to the main tasks of reconstruction and simulation, we account for other computing tasks:

Calibration We assume that some fraction of the data will need to be reconstructed for calibration purposes. The resulting data is not appropriate for physics analysis.

Skims/mini-DST production The production of skims for various topologies and the production of corresponding mini-DST files will require some resources.

Physics Analysis We account for the JLab-resident physics analysis effort. This is exclusive of GPU-based amplitude analysis (see Section 4).

The basic assumptions that we use here are for generic running in Phase III. These are shown in Table 1. Table 2 shows assumptions for each computing task and the size of the corresponding CPU requirement in this model expressed as a number of cores. The computing complex needed to keep up with all activities is then equivalent to about nine thousand cores.

Another view of these estimates is to ask how long some of these steps will take on a given compute farm. Table 3 shows the number of days required to complete a single pass at reconstruction and simulation for the three Phases, using the same assumptions as above, on a farm with 10,000 cores. Note that these times will only be obtainable if all cores are dedicated to these activities. For Phase III then, to do a complete cycle of reconstruction on the entire data set will take about two weeks, to do reconstruction and the required simulation about two months.

7.2 Tape Storage Requirements

The steady-state model described in the previous section implies a rate of events of various types being read and written to tape. Even if the data is meant to be accessed from disk, we

Parameter	**Value**
trigger rate	20 kHz
event size	15 kB
running time per year	35 weeks
time to reconstruct an event	133 ms
ratio of simulated events to real events	2
time to generate a simulated event	67 ms
time to reconstruct a simulated event	133 ms

Table 1: Basic assumptions for computing requirements. All computing times are for a single core.

Activity	**CPU-need, 1 iteration**	**Number of iterations**	**CPU-need**
Calibration	45	2	89
Reconstruction	894	2	1,789
Skims/mini-DST	89	5×2	894
Physics Analysis	89	10×1	894
Simulation	2,683	2	5,366
Total			9,033

Table 2: CPU needs. All needs are in terms of cores on a 2.8 GHz Nehalem processor.

	Phase I	**Phase II**	**Phase III**
Days of running	60	60	120
Trigger rate (kHz)	2	20	20
Number of events	5.18×10^{9}	5.18×10^{10}	1.04×10^{11}
Reconstruction time (days)	0.8	8	16
Simulation time (gen. + recon.) (days)	2.4	24	48
Recon. + Sim. time (days)	3.2	32	64

Table 3: Wait times for various steps on a 10,000 core farm for approved GlueX running.

Data Type	Rate to Tape (PB/year)
Raw data	3.2
Calibration	0.06
DST (Reconstructed)	1.3
Skims	0.6
Simulation DST	2.5
Total	7.7

Table 4: Average rate of writing data to tape.

intend to archive all data to tape, with the exception of the pre-reconstruction simulated data. We assume that the reconstructed DST data is 1.5 kB per event (a factor of 10 compression from the raw data) and that all events will be reconstructed. We also include repetition factors, despite the possibility that some of the tapes from early iterations may be recycled. The amount of data written to tape is summarized in Table 4.

The bulk processing being described also implies an average bandwidth to and from tape. To estimate this, the bandwidth for reading input as well as writing output is included. The sum of all activities, in steady state is 1.0 GB/s. Physics analysis is not included. Current tape technology in the JLab tape library can go at 100 MB/s. On average then, 10 drives will be need to support data analysis.

7.3 Disk Use

Disk storage is driven by the size of data sets necessary to support various analysis activities. The following classes of data will have to be permanently accessible from disk:

Calibration disk Disk space to support on-going calibration development and production.

Coherent-peak skim DST Reconstructed data selected from the coherent bremsstrahlung peak. This is the principal data set for GlueX.

Inclusive background simulation DST Simulation of minimum bias events with appropriate cross-section weighting. This represents the background for all physics channels of interest.

Individual analysis skim Skims of the coherent-peak DST and the inclusive simulation DST for individual analyses. These are used to study cuts and perform the physics analysis, tailored to a particular analysis. Events may contain additional analysis dependent data.

Mini-DST's for amplitude analysis "4-vector" files appropriate for amplitude analysis.

Data Type	**Phase II**	**Phase III**
Calibration disk	62	124
Coherent-peak skim DST	25	50
Inclusive background simulation DST	265	531
Individual analysis skims (10 analyses)	207	415
Mini-DST's for amplitude analysis	7	15
Total	567	1134

Table 5: Disk requirements for analysis in terabytes.

The total disk footprint for each are summarized in Table 5 separately for Phases II and III. (Phase I data is not a significant contribution.)

In addition to the disk space above, there will be a need for a general work disk of about 300 TB for staging files and scratch space. The total of all of these areas comes to 2.0 PB.

References

[1] R. Jones. Detector models for gluex monte carlo simulation: the cd2 baseline. GlueX-doc 732-v4, Univ. of Connecticut, 2007.

[2] D Lawrence. Multi-threaded event reconstruction with jana. *Journal of Physics: Conference Series*, 119(4):042018 (6pp), 2008.

[3] D. Lawrence. *Multi-threaded event processing with JANA*, number 062 in PoS, http://pos.sissa.it/archive/conferences/070/062/ACAT08_062.pdf, Nov 2008. SISSA.

[4] Simon Taylor. Track fitting in gluex: Development report v. Technical Report GlueX-doc-1812-v1, Jefferson Lab, 2011.

[5] David Lawrence and Simon Taylor. Track finding and fitting in gluex: Development report iv. Technical Report GlueX-doc-1004, Jefferson Lab, March 2008.

[6] David Lawrence. Track fitting in gluex: Development report iii. Technical Report Gluex-doc-762-v2, Jefferson Lab, 2007.

[7] C. Xu, M. Barbi, and G.Huber. The gluex bcal reconstruction code - preliminary studies. Technical report, Univ. of Regina, 2005.

[8] C. Jones, V. Kuznetsov, D. Riley, and G. Sharp. The new EventStore data management system for the CLEO-c experiment. *Int.J.Mod.Phys.*, A20:3868–3870, 2005.

[9] David Lawrence. The jana calibrations and conditions database api. *Journal of Physics: Conference Series*, 219 part 4:(6pp), 2009 doi: 10.1088/1742-6596/219/4/042011.

[10] Mark Ito and David Lawrence. Gluex calibration/conditions database specifcation. Technical Report GlueX-doc-1541-v6, Jefferson Lab, 2010.

[11] Y. Van Haarlem, C.A. Meyer, F. Barbosa, B. Dey, D. Lawrence, et al. The GlueX Central Drift Chamber: Design and Performance. *Nucl.Instrum.Meth.*, A622:142–156, 2010.