asynMotor: Support For Coordinated Multi-Axis Motion In EPICS

Mark Rivers

University of Chicago

THE UNIVERSITY OF CHICAGO

Argonne NATIONAL LABORATORY

Motivation

- Traditional step scanning overheads:
  - Start and stop motors & mechanical systems
  - Arm and read detectors.
- Typically much more efficient to collect data on-the-fly as the motors are moving.
  - Better detectors, more flux with APS Upgrade, don't waste photons and time
- However, on-the-fly scanning with EPICS has been subject to limitations.
  - Typically limited to simple single-motor scans because EPICS did not provide a way to program motor controllers to perform synchronized motion of multi-axis controllers.
    - Exception to this was the Trajectory Scanning software I wrote for the Newport XPS and MM4005 motor controllers.
    - Not a clean solution: EPICS SNL program talking directly to the controller, "behind the back" of the motor record driver.
    - PMAC does have motion control programs – good for fixed, simple geometry
  - New solution that incorporates a coordinated motion API in the motor driver layer

asynMotorController

```
class epicsShareFunc asynMotorController : public asynPortDriver {
  asynMotorController(const char *portName, int numAxes, int numParams,
                      int interfaceMask, int interruptMask,
                      int asynFlags, int autoConnect, int priority, int stackSize);

  virtual asynMotorAxis* getAxis(asynUser *pasynUser);
  virtual asynMotorAxis* getAxis(int axisNo);

  virtual asynStatus startPoller(double movingPollPeriod, double
     idlePollPeriod, int forcedFastPolls);
  virtual asynStatus wakeupPoller();
  virtual asynStatus poll();
  void asynMotorPoller();

  virtual asynStatus initializeProfile(size_t maxPoints);
  virtual asynStatus buildProfile();
  virtual asynStatus executeProfile();
  virtual asynStatus abortProfile();
  virtual asynStatus readbackProfile();
```

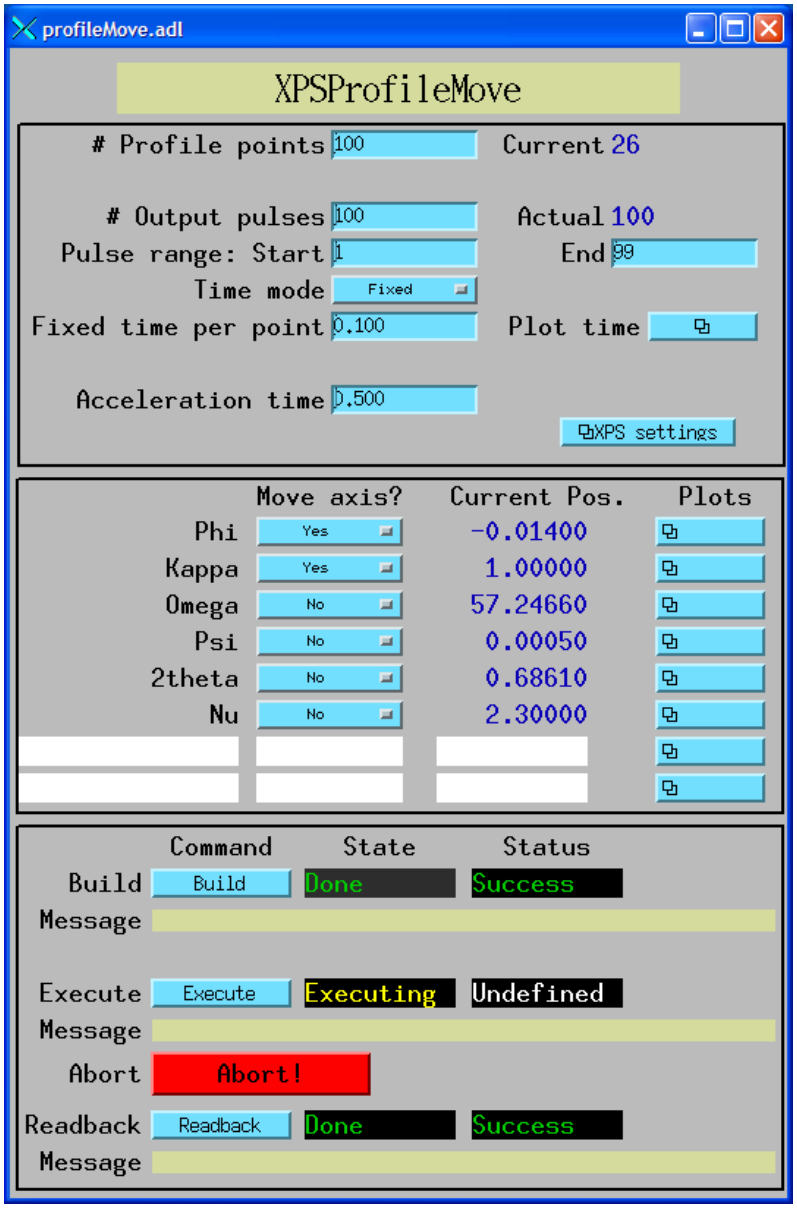
asynMotorAxis

```
class epicsShareFunc asynMotorAxis {
  virtual asynStatus move(double position, int relative,
            double minVelocity, double maxVelocity, double acceleration);
  virtual asynStatus moveVelocity(double minVelocity, double maxVelocity,
            double acceleration);
  virtual asynStatus home(double minVelocity, double maxVelocity,
            double acceleration, int forwards);
  virtual asynStatus stop(double acceleration);
  virtual asynStatus poll(bool *moving);
  virtual asynStatus setPosition(double position);

 virtual asynStatus initializeProfile(size_t maxPoints);
  virtual asynStatus defineProfile(double *positions, size_t numPoints);
  virtual asynStatus buildProfile();
  virtual asynStatus executeProfile();
  virtual asynStatus abortProfile();
  virtual asynStatus readbackProfile();
```

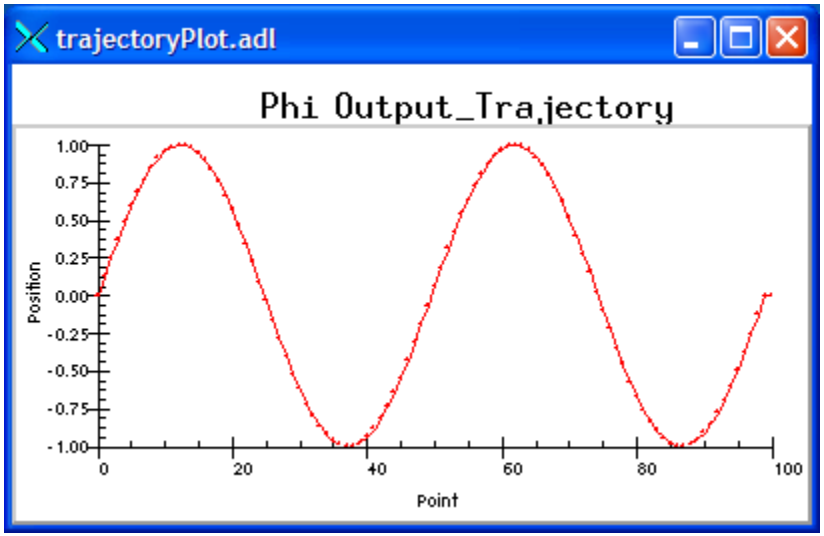
Coordinated Motion
(Now called ProfileMove, previously TrajectoryScan)

- Create arrays of target locations of each motor in a multi-axis controller
- Create array of time per element
- Define times/locations to output synchronization pulses to trigger detectors
- Execute coordinated move
- Optionally read back encoder positions when each synchronization pulse was output
- Many controllers have this capability
  - Newport MM4005, Newport XPS, Delta Tau PMAC, Pro-Dex MAXv, Parker ACR series, etc.
- We now have an API to take use this capability in a consistent way
- Now done in same driver as for motor record, eliminates conflicts

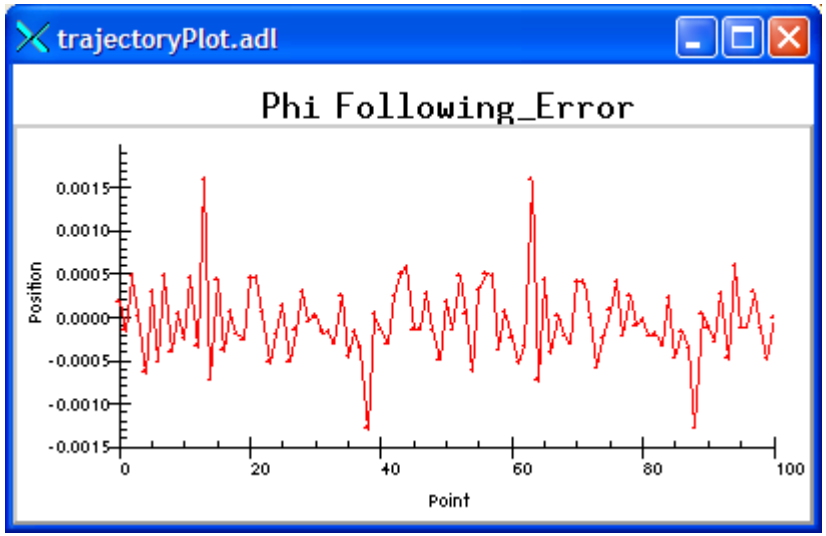
Coordinated Motion



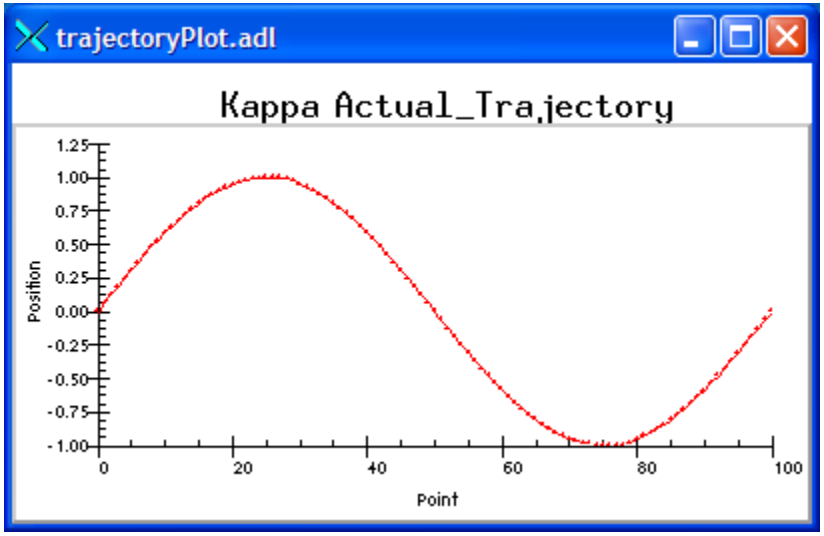

```
; IDL program to build, execute and readback profile
; This program builds a profile and executes it.

nelements = 100 ; 100 elements in the profile.
npulses = 100 ; 100 pulses in the profile.
naxes=2       ; We will move the first 2 motors (Phi and Kappa)

; Define array of positions
positions = dblarr(nelements, naxes)

; The Phi profile is a sin wave with two complete periods and an
; amplitude of +-1 degrees
positions[*,0] = 1.*sin(findgen(nelements)/(nelements-1.)*4.*!pi)

; The Kappa profile is a sin wave with one complete period and an
; amplitude of +-1 degrees
positions[*,1] = 1.*sin(findgen(nelements)/(nelements-1.)*2.*!pi)
profile = 'IOC:Prof1:'
group = 'GROUP1'

time = 0.1      ; Fixed time per profile point
status = profile_move(profile, positions, group=group, maxAxes=6, $
            build=1, execute=1, readback=1, time=time, $
            npulses=npulses, actual=actual, errors=errors)
end
```



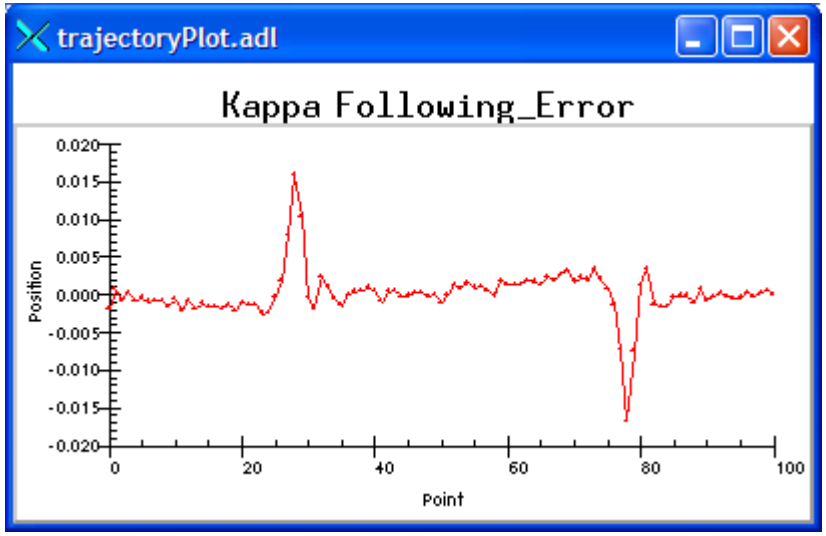
Phi output profile



Phi following error



Kappa readback profile

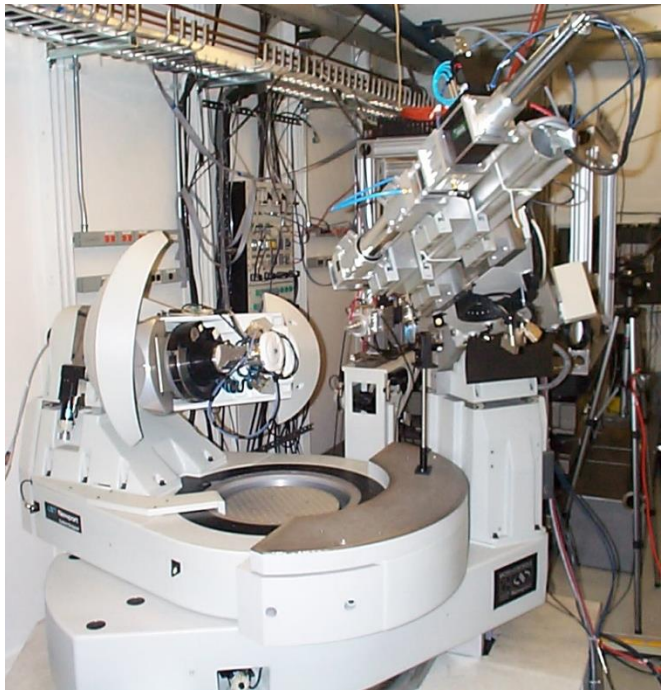


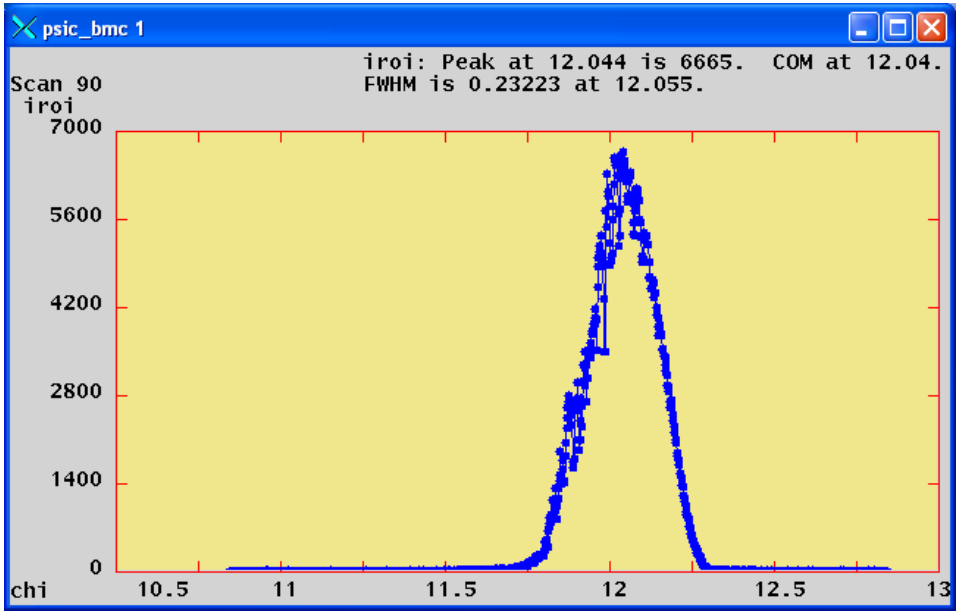
Kappa following error

SPEC Interface

- SPEC macros have been written to allow SPEC to utilize profile moves via EPICS interface
- Low level interface, all of SPEC's standard scans can be done "on-the-fly" with profile move software. Replacement macros for:
  - _ascan # Used by all ascan and dscan macros
  - Mesh
  - hklscan # Used by hscan, kscan and lscan
  - _hklmesh
  - _hkllline # Used by hkcircle, hlcircle, klcircle, hkradial, hlradial and klradial
  - _scanabort resume
  - _loop

SPEC Scan Using 6-Cicle Diffractometer





SPEC 4° scan of virtual chi axis, 1000 points at .02 seconds/point. Coordinated motion of the phi, kappa and omega axes.

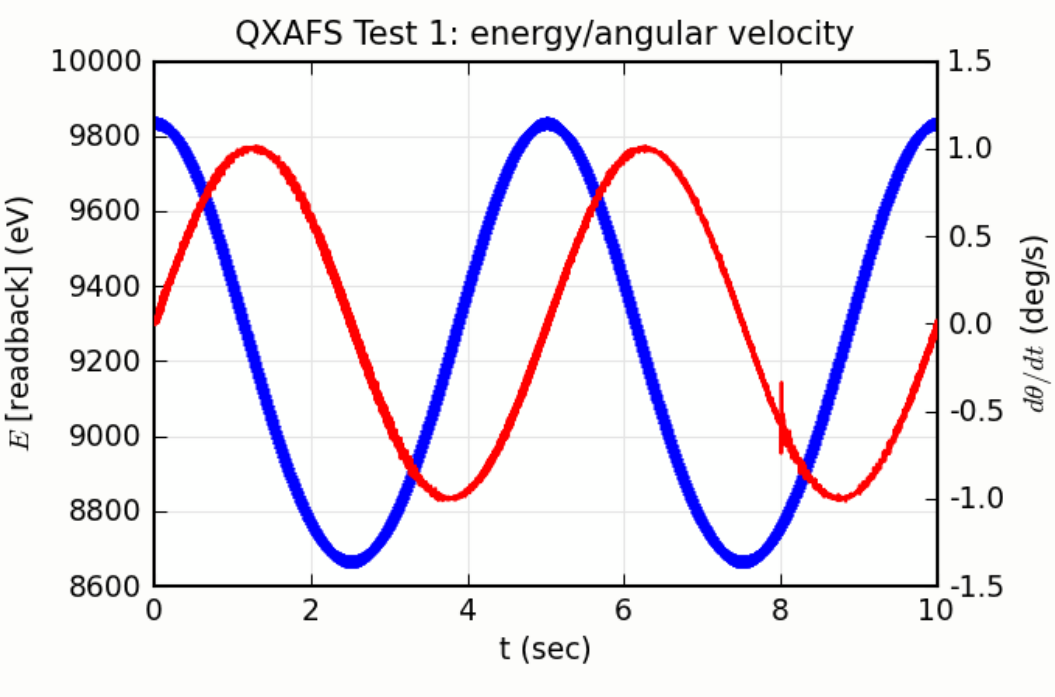
XPS outputs synchronization pulses to trigger Pilatus.

Execution time: theoretical 20.0 seconds, actual to completion 20.8 seconds.

Includes the to save 1000 images to disk (366 MB), for the Pilatus driver to read each file, correct the bad pixels and flat field, compute the ROIs, and post the ROIs to EPICS.

Preliminary Quick-XAFS Tests at GSE 13-IDE

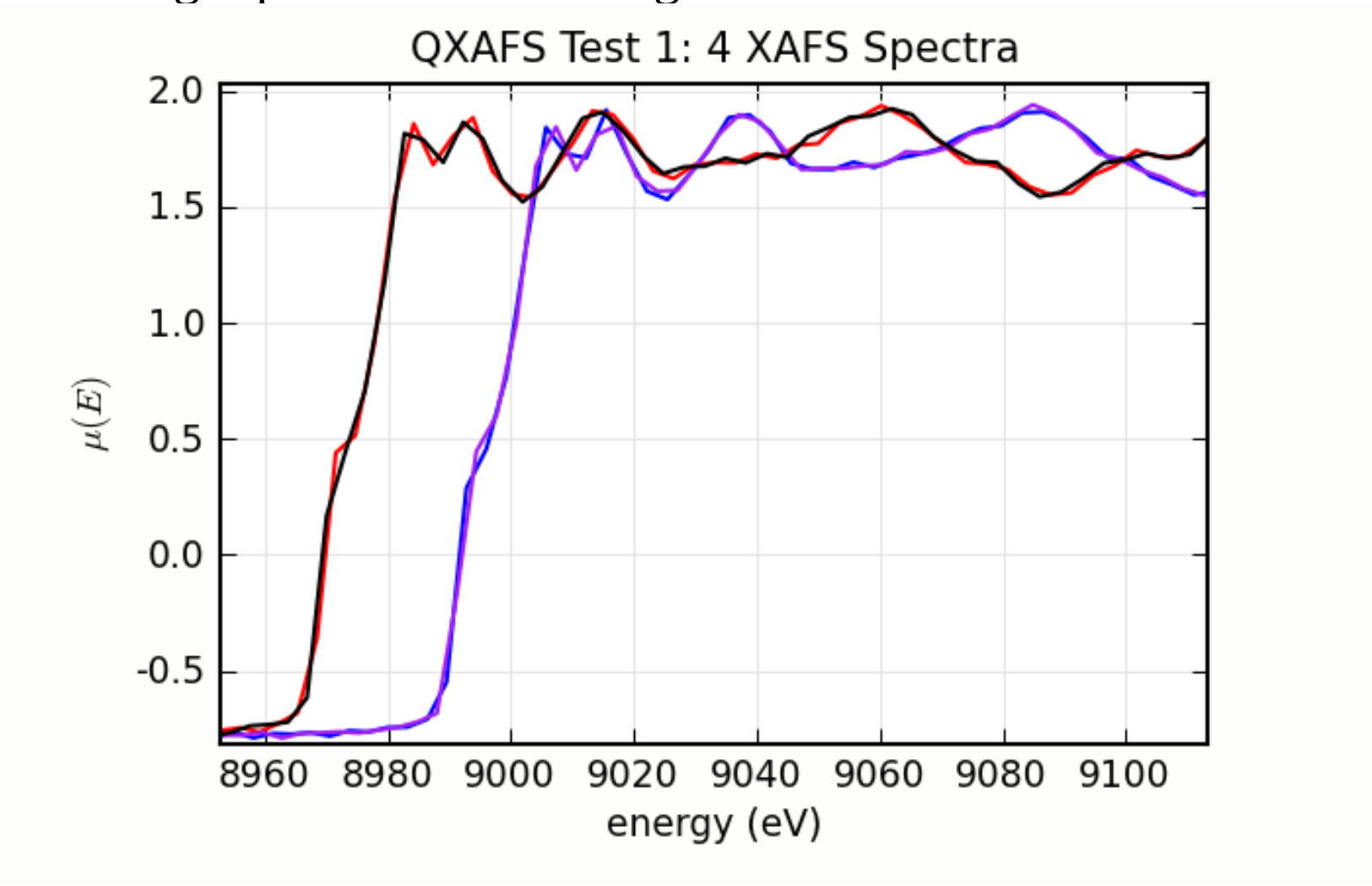
Si(111) mono, at Cu K-edge, 1 keV ~ 1 deg scan
Tapered undulator, but did not try to synchronize undulator and monochromator
Ran sinusoidal trajectory profiles with Newport XPS – can run many oscillations in a row.



The motor and controller can easily do 5 deg/s (probably up to 20 deg/s).
Full cycle scans of 1 degree in well under 1 seconds should not be a problem.

Preliminary QXAFS Tests at GSE 13-IDE

4 Cu K-edge scans in one 10 second sweep,
showing reproducible time-lag from ion-



Future Work

- Convert Pro-Dex MAXv, Aerotech Ensemble to Model 3
  - Tim Mooney already has trajectory scanning running in SNL program on each of these
- Support other controllers
  - PMAC, etc.
- Add additional features
- Work on replacement for EPICS motor record?