OAK RIDGE National Laboratory

PyDevice – New Features and Updates

Klemen Vodopivec

vodopiveck@ornl.gov

September 22th, 2022

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

PyDevice Brief Introduction

- Extend EPICS records functionality with the power of Python

- Integrates into existing IOCs
  - Leverage IOC core functionality
  - Coexist with EPICS extensions
  - Quick and robust adoption

- Broad compatibility
  - Python2 & Python3
  - EPICS 3.14.12.x – EPICS 7.0.x

http://github.com/klemenv/PyDevice

```
# Keep running values in Python array
record(ao, "BeamPower:Cache") {
  field(DTYP, "pydev")
  field(DOL,  "BeamPower CP")
  field(OUT,  "@beampower.append(VAL)")
  field(FLNK, "BeamPower:Avg")
}
# Using Python built-in functions for average calc
record(ai, "BeamPower:Avg") {
  field(DTYP, "pydev")
  field(INP,  "@sum(beampower)/len(beampower)")
  field(FLNK, "BeamPower:Std")
}
# numpy one-liner for standard deviation
record(ai, "BeamPower:Std") {
  field(DTYP, "pydev")
  field(INP,  "@numpy.std(beampower)")
}
```

Supporting More EPICS Base Records

waveform

- Python excels at processing arrays
- Support for most element types
  - (U)CHAR, (U)SHORT, (U)LONG, FLOAT, DOUBLE, STRING
- Input and output arrays may differ in size but not type

lsi / lso

- EPICS 3.15 and later
- String size inherited from EPICS base, default 65,535
- Useful for file paths, URLs

New pycalcRecord

- Similar to aSub record but invokes Python for processing inputs
- Passing multiple parameters to Python expressions:
  - Functions
  - Formulas
- Selectable input and output field types and size
- Python code unaware of record semantics

Generating 1000 samples of quadratic function

```
record(pycalc, "Quadratic:X") {
  field(INPA, "Quadratic:X:LowLimit")
  field(INPB, "Quadratic:X:HighLimit")
  field(CALC, "list(np.linspace(A,B,1000))")
  field(MEVL, "1000")
}
record(pycalc, "Quadratic:Y") {
  field(INPA, "2")
  field(INPB, "-5.0")
  field(INPC, "7")
  field(INPH, "Quadratic:X CP")
  field(CALC, "list(A*np.array(H)**2+B*np.array(A)+C)")
  field(MEVL, "1000")
}
```

Escaping Fields is Now Optional

Before

```
pydev("VAL=834.2")

record(ao, "Number:Print") {
  field(DTYP, "pydev")
  field(VAL,  "419.5")
  field(OUT,  "@print(%VAL%)")
}
```

After

```
pydev("VAL=834.2")

record(ao, "Number:Print") {
  field(DTYP, "pydev")
  field(VAL,  "419.5")
  field(OUT,  "@print(VAL)")
}
```

Does it print 834.2 or 419.5?

More pitfalls

```
record(ao, "Equal") {
  field(DTYP, "pydev")
  field(VAL,  "419.5")
  field(OUT,  "@VAL=%VAL%")
}
```

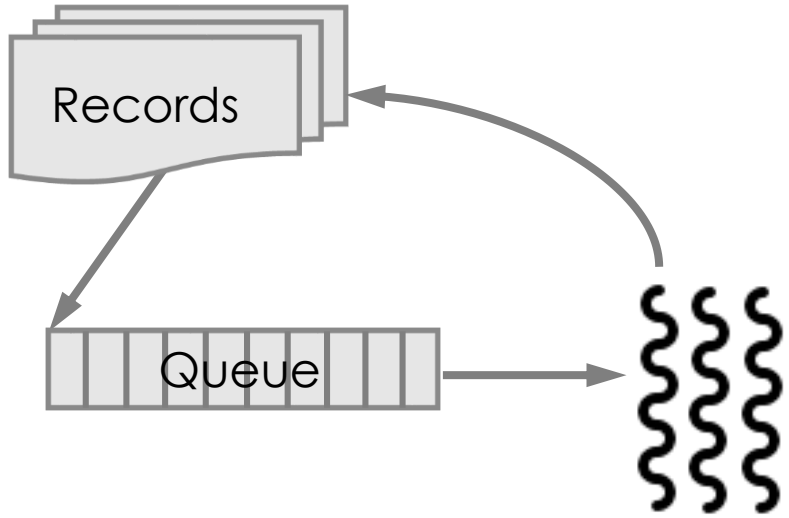
```
record(ao, "Macros") {
  field(DTYP, "pydev")
  field(OUT,  "@print($(HIGH))")
}
```

Concurrent Record Processing

- Allow processing multiple records concurrently
  - as long as CPython code releases GIL,
  - or runs out of allocated time slice*
  - Generally, it runs pretty well!

* Python 2: sys.setcheckinterval()
Python 3: sys.setswitchinterval()

- Configurable number of worker threads
  - `epicsEnvSet("PYDEV_NUM_THREADS","10")`
  - All worker threads busy => requests queued in FIFO



David Beazley – Understanding GIL

Simpler Build Mechanism

PyDevice folder

Choose Python version in **configure/CONFIG_SITE**

```
PYTHON_CONFIG=python39-config
```

IOC folder

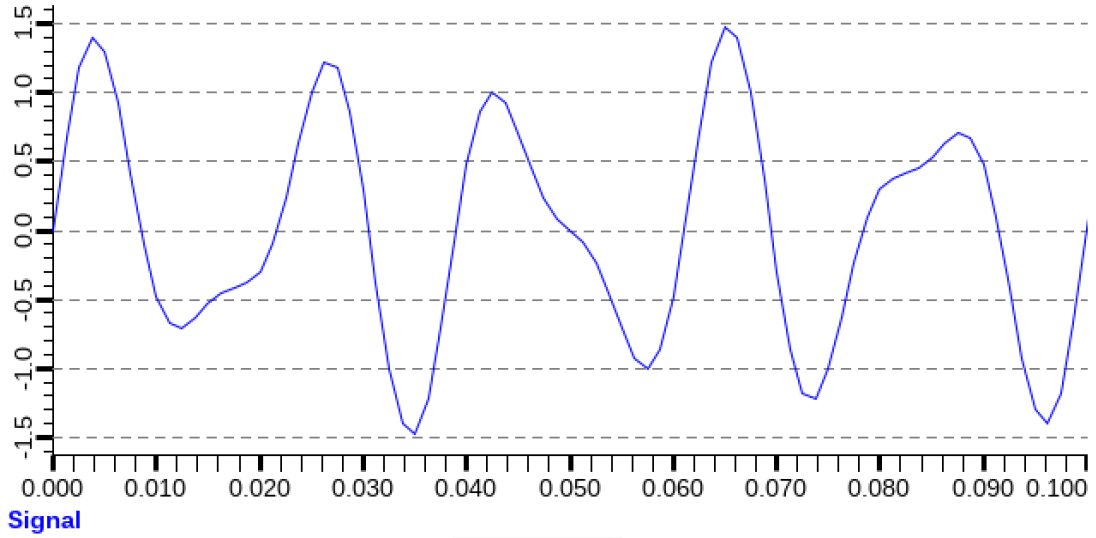
Add to IOC **configure/RELEASE**

```
PYDEVICE=/path/to/PyDevice
```

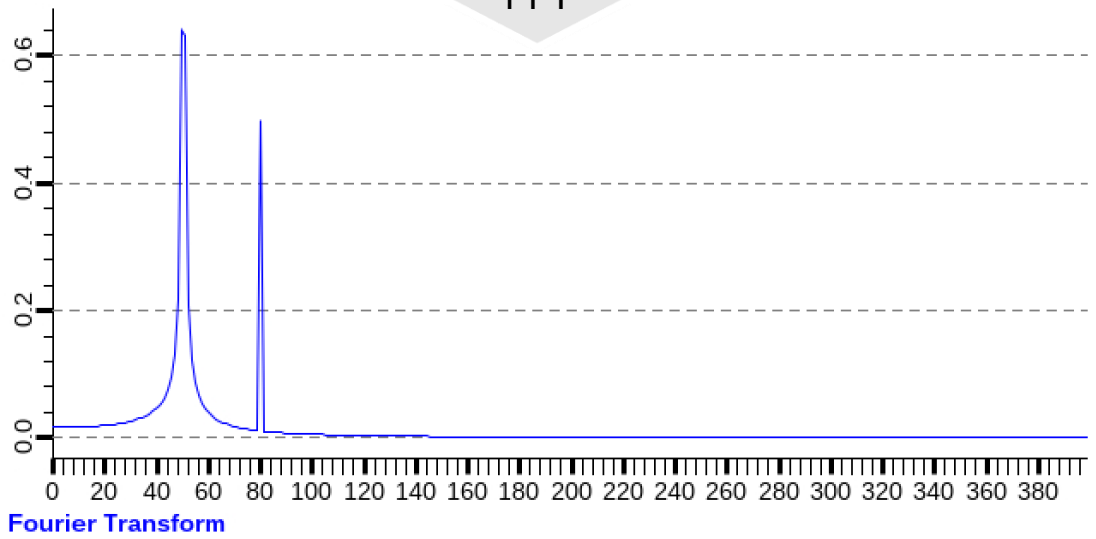
Add to IOC **iocApp/src/Makefile**

```
include $(PYDEVICE)/configure/CONFIG.PyDevice
ioc_DBD += pydev.dbd
ioc_LIB += pydev
```

Example – Fast Fourier Transform with scipy

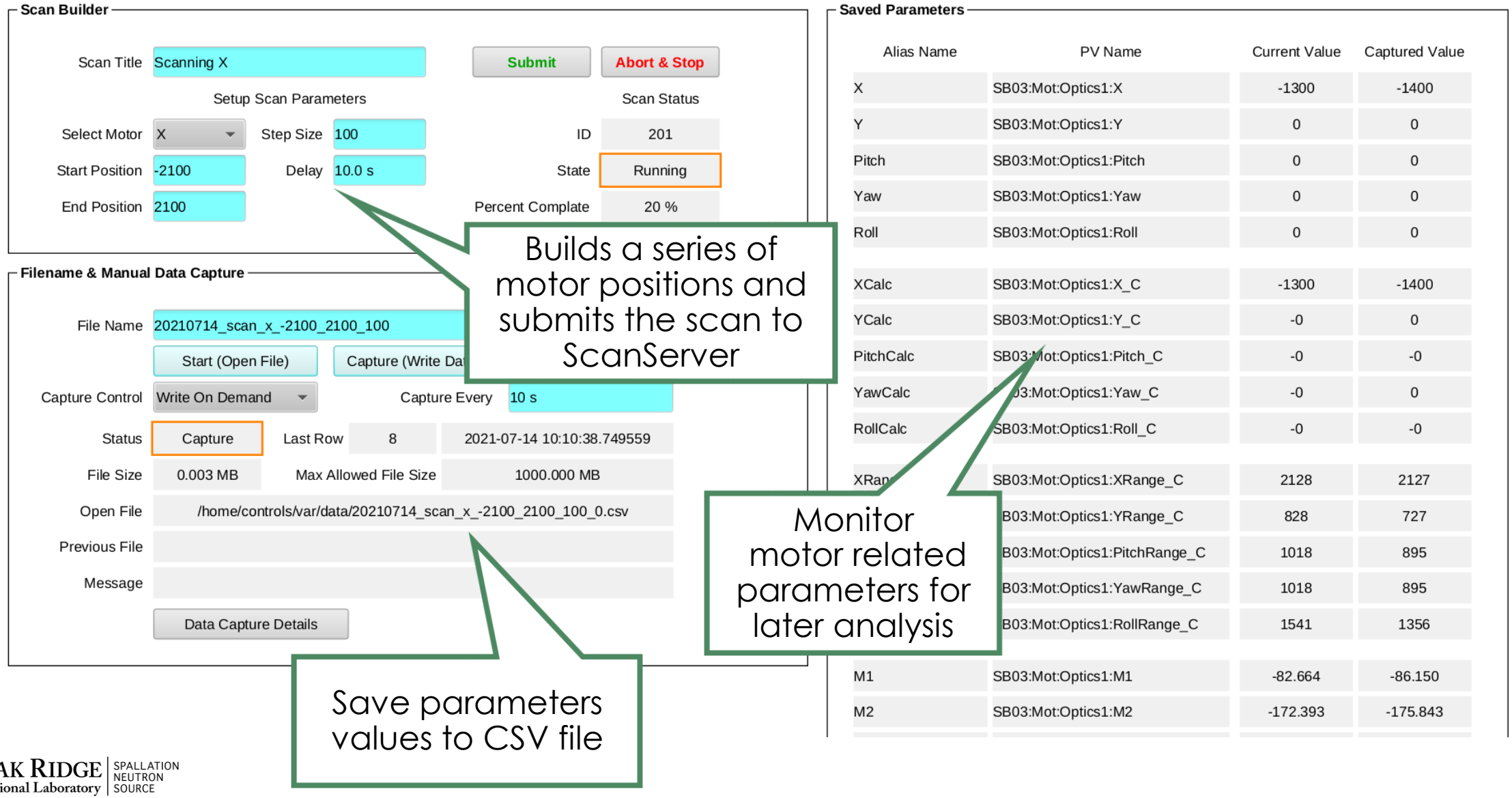


FFT

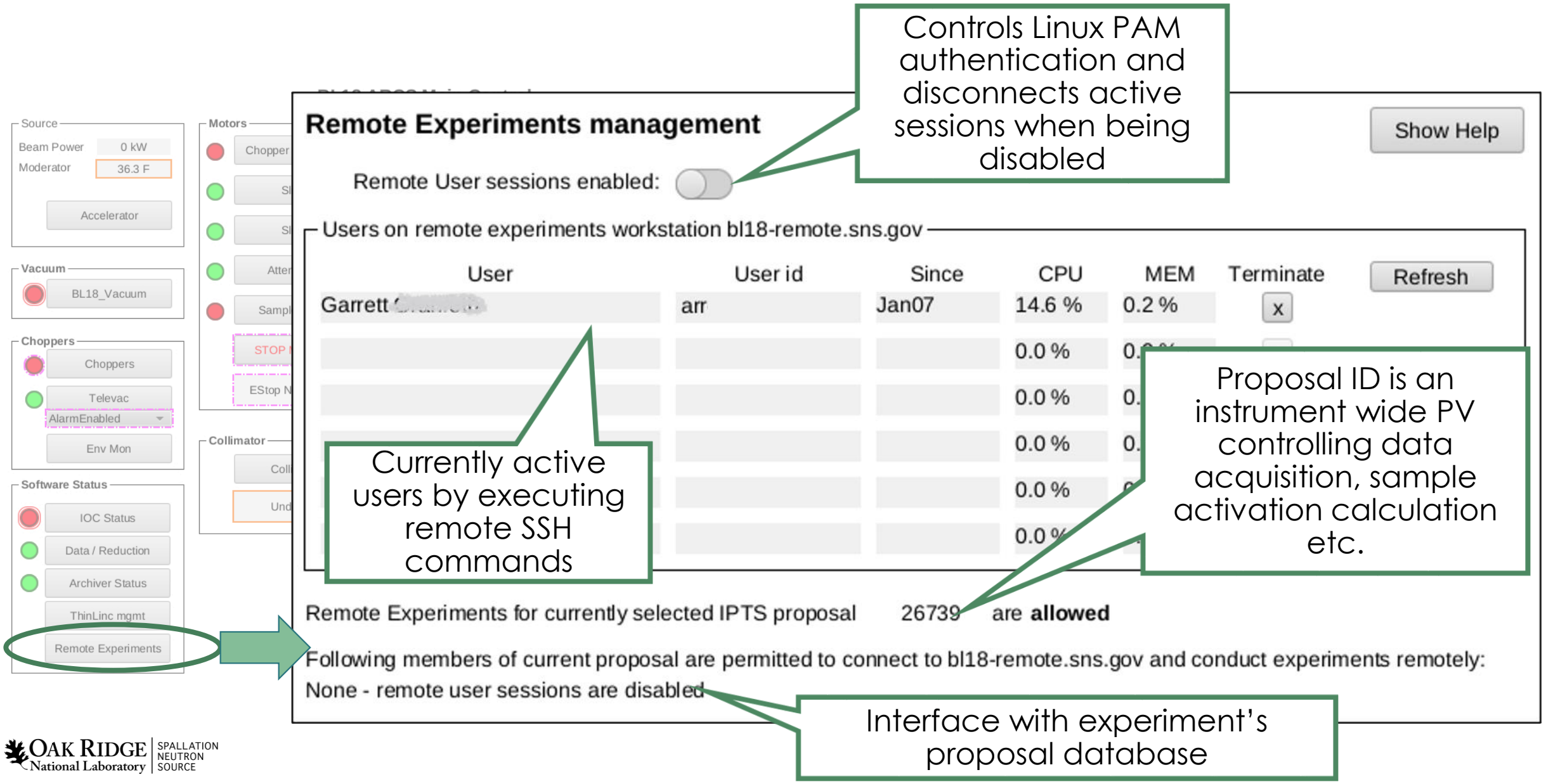



```
record(waveform, "Signal:X") {
  field(DTYP, "pydev")
  field(NELM, "600")
  field(FTVL, "DOUBLE")
  field(INP,  "@np.linspace(0, 0.75, 600, endpoint=False)")
}
record(pycalc, "Signal:Y") {
  field(INPA, "Signal:X")
  field(CALC, "np.sin(100*np.pi*A)+.5*np.sin(160*np.pi*A)")
  field(MEVL, "600")
  field(FTVL, "DOUBLE")
}
record(waveform, "Signal:FFT:X") {
  field(DTYP, "pydev")
  field(NELM, "300")
  field(FTVL, "DOUBLE")
  field(INP,  "@scipy.fft.fftfreq(600, 0.00125)[:600//2]")
}
record(pycalc, "Signal:FFT:Y") {
  field(INPA, "Signal:Y")
  field(MEA,  "600")
  field(CALC, "2.0/600 * np.abs(scipy.fft.fft(A)[0:600//2]")
  field(MEVL, "300")
  field(FTVL, "DOUBLE")
}
```

Example – Steerable Optics Evaluation



Example – Managing Remote Experiments



Time for Questions?
(and answers)