

# PyDevice – New Features and Updates

Klemen Vodopivec  
vodopiveck@ornl.gov  
September 22<sup>th</sup>, 2022

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

## Isi / Iso

- 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))")  
}
```



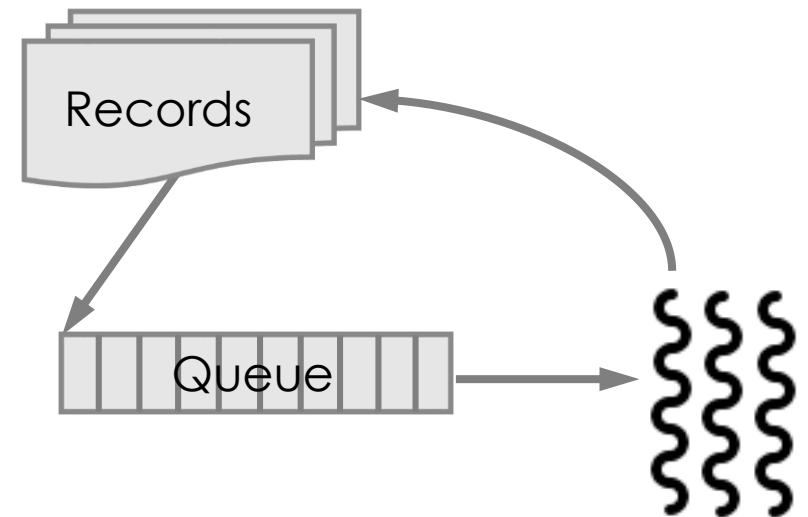
# Concurrent Record Processing

- 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



# Simpler Build Mechanism

## PyDevice folder

Choose Python version in **configure/CONFIG\_SITE**

```
PYTHON_CONFIG=python39-config
```

## IOC folder

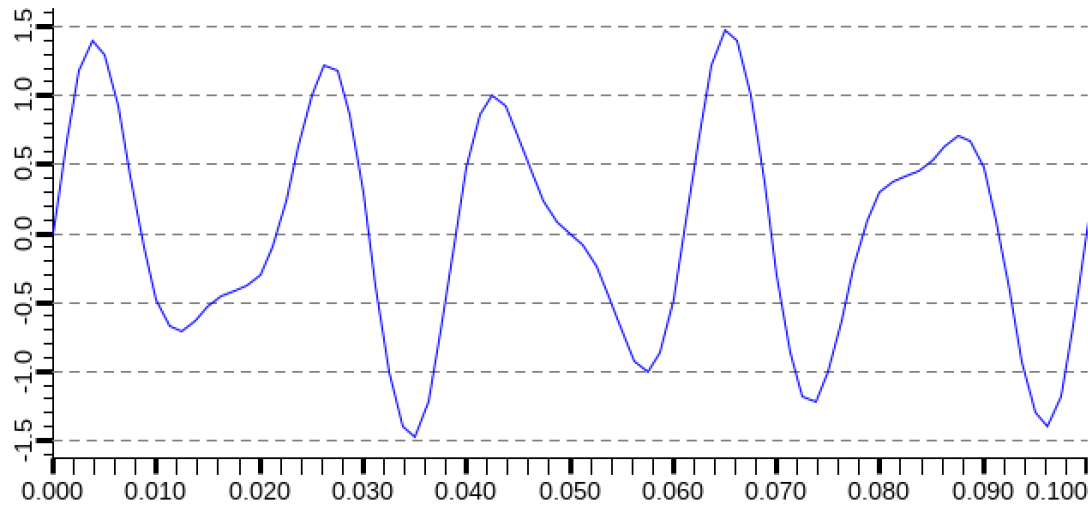
Add to IOC **configure/RELEASE**

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

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

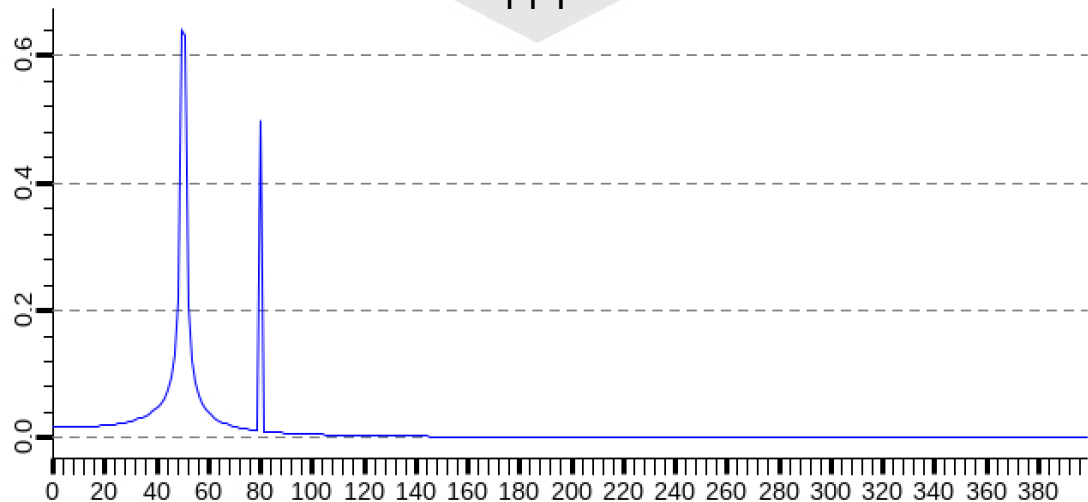
```
include $(PYDEVICE)/configure/CONFIG.PyDevice  
ioc_DBD += pydev.dbd  
ioc_LIB += pydev
```

# Example – Fast Fourier Transform with scipy



Signal

FFT



Fourier Transform

```
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

### Scan Builder

Scan Title:

Setup Scan Parameters

Select Motor:  Step Size:

Start Position:  Delay:

End Position:

Scan Status

ID:  State:

Percent Complete:

### Saved Parameters

Alias Name	PV Name	Current Value	Captured Value
X	SB03:Mot:Optics1:X	-1300	-1400
Y	SB03:Mot:Optics1:Y	0	0
Pitch	SB03:Mot:Optics1:Pitch	0	0
Yaw	SB03:Mot:Optics1:Yaw	0	0
Roll	SB03:Mot:Optics1:Roll	0	0
XCalc	SB03:Mot:Optics1:X_C	-1300	-1400
YCalc	SB03:Mot:Optics1:Y_C	-0	0
PitchCalc	SB03:Mot:Optics1:Pitch_C	-0	-0
YawCalc	SB03:Mot:Optics1:Yaw_C	-0	0
RollCalc	SB03:Mot:Optics1:Roll_C	-0	-0
XRange	SB03:Mot:Optics1:XRange_C	2128	2127
YRange	SB03:Mot:Optics1:YRange_C	828	727
PitchRange	SB03:Mot:Optics1:PitchRange_C	1018	895
YawRange	SB03:Mot:Optics1:YawRange_C	1018	895
RollRange	SB03:Mot:Optics1:RollRange_C	1541	1356
M1	SB03:Mot:Optics1:M1	-82.664	-86.150
M2	SB03:Mot:Optics1:M2	-172.393	-175.843

### Filename & Manual Data Capture

File Name:

Capture Control:  Capture Every:

Status:  Last Row:  2021-07-14 10:10:38.749559

File Size:  Max Allowed File Size:

Open File:

Previous File:

Message:

Builds a series of motor positions and submits the scan to ScanServer

Monitor motor related parameters for later analysis

Save parameters values to CSV file

# Example – Managing Remote Experiments

The screenshot shows a control interface for remote experiments. On the left, there are several status panels: Source (Beam Power: 0 kW, Moderator: 36.3 F), Vacuum (BL18\_Vacuum), Choppers (Choppers, Televac, AlarmEnabled), and Software Status (IOC Status, Data / Reduction, Archiver Status, ThinLinc mgmt, Remote Experiments). The Remote Experiments panel is circled in green with an arrow pointing to the main management window.

### Remote Experiments management

Remote User sessions enabled:

Users on remote experiments workstation bl18-remote.sns.gov

User	User id	Since	CPU	MEM	Terminate
Garrett	arr	Jan07	14.6 %	0.2 %	x
			0.0 %	0.0 %	
			0.0 %	0.0 %	
			0.0 %	0.0 %	
			0.0 %	0.0 %	
			0.0 %	0.0 %	

Remote Experiments for currently selected IPTS proposal 26739 are **allowed**

Following members of current proposal are permitted to connect to bl18-remote.sns.gov and conduct experiments remotely:  
None - remote user sessions are disabled

Callouts:

- Controls Linux PAM authentication and disconnects active sessions when being disabled
- Currently active users by executing remote SSH commands
- Proposal ID is an instrument wide PV controlling data acquisition, sample activation calculation etc.
- Interface with experiment's proposal database

# Time for Questions? (and answers)