Integrate EPICS 7 with MATLAB Using PVAccess for Python (P4P) Module

K. Kim – ktkim@slac.stanford.edu , E. Williams, J. Bellister, K. Kim, M. Zelazny
SLAC National Accelerator Laboratory, Menlo Park, California, USA.

ABSTRACT

MATLAB is essential for accelerator scientists engaged in data analysis and processing across diverse fields, including particle physics experiments, synchrotron light sources, X-ray Free Electron Lasers (XFELs), and telescopes, due to its extensive range of built-in functions and tools. Scientists also depend on Experimental Physics and Industrial Control Systems (EPICS) 7 to control and monitor complex systems. SLAC has developed Matpva, a Python interface to integrate EPICS 7 with MATLAB. Matpva utilizes the PVAccess for Python (P4P) module and EPICS 7 to offer a robust and reliable interface for MATLAB users that employ EPICS 7. The EPICS 7 PVAccess API allows higher-level scientific applications to get/set/monitor simple and complex structures from an EPICS 7-based control system. Moreover, Matpva simplifies the process by handling the data type conversion from Python to MATLAB, making it easier for researchers to focus on their analyses and innovative ideas instead of technical data conversion. By leveraging Matpva, researchers can work more efficiently and make discoveries in diverse fields, including particle physics and astronomy.

PROBLEMS & SOLUTION

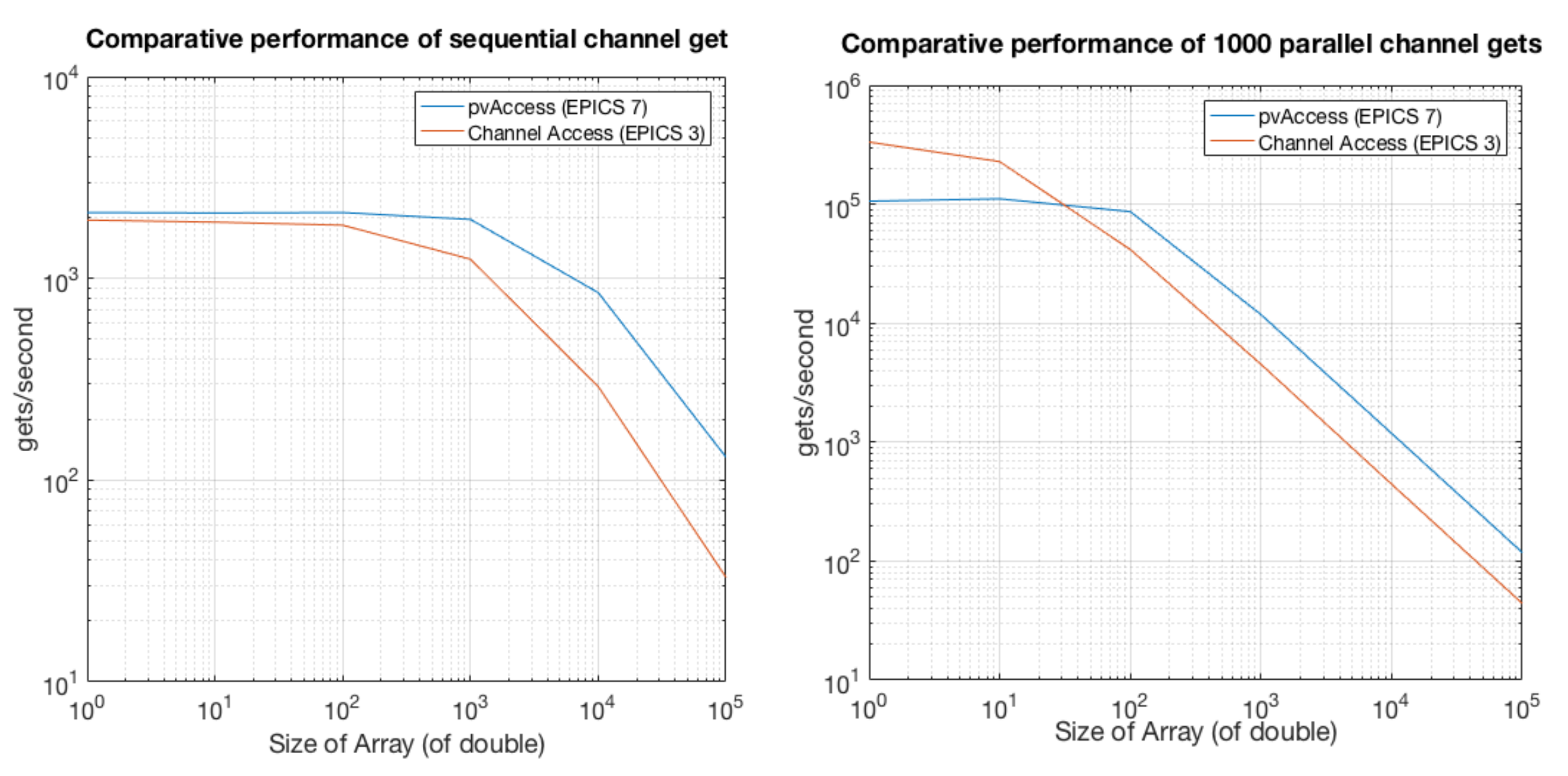
Problems:

- labCA and MATLAB CA (MCA) for EPICS Channel Access (CA) interface in MATLAB. => Not fully support PVAccess
- Integration of PVAccess into MATLAB using epicsCoreJava based on Java.
- Compatibility issue: MATLAB relies on Java 1.8.x, EPICS Java PVAccess defaults to Java 11.
- End of support: Oracle JDK 8 (used by MATLAB) ended support in March 2022, OpenJDK 8 to end support in November 2026.

Solution:

- Integrates EPICS PVAccess with MATLAB using Python.
- Matpva: A Python interface bridging EPICS 7 with MATLAB. Streamlining Data Conversion between Python and MATLAB seamlessly

PVACCESS VS CHANNEL ACCESS



Comparison of performance included in PVAccess compared to Channel Access [1]

- PVAccess: Faster for handling large datasets.
- Channel Access (CA) Faster for handling numerous small operations in parallel.

IMPLEMENTATION

- The PVAccess data type comprises Normative Types, which are defined as structures consisting of both required and optional fields.
- Matpva stands out for its ability to convert PVAccess Normative Types from Python to MATLAB.

```
% Bring P4P python module into MATLAB
MatP4P = py.p4p.client.thread.Context('pva',
pyargs('nt', false));
PV = MatP4P.get(pvname);

% Check the ID of PV
nt_id = string(getID(PV));
% Check the type of PV
t = struct(py.dict(type(PV))).value;

% To have timeStamp information in human readable form
TimeInSeconds =
double(int64(struct(struct(todict(PV)).timeStamp).second
sPastEpoch));
ts = datetime(TimeInSeconds, 'ConvertFrom', 'epochtime',
'Epoch', '1970-01-01', ...
    'Format', 'MMM dd, yyyy HH:mm:ss.SSS', 'TimeZone',
'UTC');
% To have alarm information
alarm.severity =
int32(int64(struct(struct(todict(PV)).alarm).severity));
alarm.status   =
int32(int64(struct(struct(todict(PV)).alarm).status));
alarm.message  =
string(struct(struct(todict(PV)).alarm).message);

if (contains(nt_id, "NTScalarArray"))
..
% NTTable data type PVs
elseif (contains(nt_id, "NTTable"))
    a           = struct(todict(PV)).labels;
    aa          = string(cell(a));
    numElement  = numel(aa);

    type_struct =
struct(py.dict(struct(py.dict(type(PV))).value));
    val_struct  = struct(struct(todict(PV)).value);

    for iElement = 1 : numElement
        e_type = type_struct.(aa(iElement));
        v      = val_struct.(aa(iElement));

        if (e_type == "ai")
            vv  = int32(py.array.array('i', v));
            vv2 = int32(py.array.array('i', v))';
```

Snippet of mpvaGet function as an implementation example

SYNTAX

```
% mpvaGet
% When PV is NTScalar or NTScalarArray type
[pv, ts, alarm] = mpvaGet(pvname)
% When PV is NTTable type
[NTTable, ts, alarm, NTStruct] = mpvaGet(pvname)
% Skip unwanted outputs using tilde(~)
[pv, ~, ~] = mpvaGet(pvname);

% mpvaPut
When PV is NTScalar or NTScalarArray type
mpvaPut(pvname, value, "mpvaDebugOn");
% When PV is NTTable type
mpvaPut(pvname, field1, value1, field2, value2, ..,
"mpvaDebugOn");
% When PV is NTTable type, MATLAB table or structure can
be used as an input value
mpvaPut(pvname, struct/table, "mpvaDebugOn");

% mpvaMonitor
mpvaMonitor(pvname);

% mpvaSetMonitor
PV = mpvaSetMonitor(pvname);

% mpvaNewMonitorValue
mpvaNewMonitorValue(PV)
```

Syntax of Matpva functions

Parameters:

- pv: Values of specified EPICS PV names
- pvname: Name of PV
- NTTable: Table values of EPICS PV names
- ts: Timestamp from EPICS records
- alarm: alarm status (HIHI, HIGH, LOW, LOLO) and severity associated with the PV
- NTStruct: Structure values of specified EPICS PV names under NTTable type
- value: Numeric, string, or logical data type
- field1, field2, field3, …: Fields of the specified EPICS NTTable PV
- value1, value2, value3, …: Numeric array, string array, or logical array data types for the corresponding fields
- PV: Instantiated Python class in mpvaSetMonitor.py for monitoring pvname

EXAMPLES

```
% mpvaGet
>> [PV, ts, ~] = mpvaGet("TEST:PVA:IntValue")

PV =
  int32

   10

ts =

  datetime

   Sep 29, 2023 00:23:46.233

% mpvaPut
>> ab = [true, true, false, true];
>> mpvaPut("TEST:PVA:BoolArray", ab, "mpvaDebugOn")
The old PV is
old_PV =

  1×4 string array

    "false"    "true"    "false"    "false"

The update PV is
updated_PV =

  1×4 string array

    "true"    "true"    "false"    "true"

% mpvaMonitor
>> mpvaMonitor("KTEST:PVA:IntValue")
NEW: KTEST:PVA:IntValue   Sat Sep 23 22:22:16 2023   36
NEW: KTEST:PVA:IntValue   Sat Sep 23 22:22:17 2023   2

% mpvaSetMonitor and mpvaNewMonitorValue
>> PV = mpvaSetMonitor("TEST:PVA:IntValue");
>> mpvaNewMonitorValue(PV)

ans =

  logical

   0
>> mpvaGet("TEST:PVA:IntValue")

ans =

  int32

   777
>> mpvaPut("TEST:PVA:IntValue", 10)
>> mpvaNewMonitorValue(PV)

ans =

  logical

   1
```

Matpva package example

REFERENCES

[1] G. White et al., *The epics software framework moves from controls to physics* [Slideshow], IPAC2019, May 2018.