EPICS 7 Features Update

Michael Davidsaver
Osprey DCS

Work funded by ESSS

PVA in your IOC

- QSRV
  - PVAccess server
  - Automatically exposes all PVs
    - aka. same names as CA
  - Define “Group” PVs
    - Configure structures from regular/single PVs
    - Multi-locking guarantees consistency
    - Operations: get/put/monitor
    - Requires Base >= 3.16.1
  - Replaces pvaSrv module

Including QSRV

- Lives in pva2pva module
  - https://github.com/epics-base/pva2pva
- Provides softIocPVA executable
- Include like any support module

```
myioc_DBD += qsrv.dbd
myioc_LIBS += qsrv
```

- Starts automatically

NTScalar, NTScalarArray

- NT == Normative
  - aka. standardized/recommended
- Structure definition
  - Integer, floating point, or string
- Has fields, but is not a Record
- NTScalar w/ double float
  - ~= struct dbr_ctrl_double
  - aka. same information as CA

```
$ pvinfo test:scalar
epics:nt/NTScalar:1.0
    double    value
    alarm_t   alarm
        int       severity
        int       status
        string    message
    time_t    timeStamp
        long      secondsPastEpoch
        int       nanoseconds
        int       userTag
    display_t display
        double limitLow
        double limitHigh
        string   description
        string   format
        string   units
    control_t control
        double limitLow
        double limitHigh
        double minStep
```

NTScalar Example

```
cat << EOF > example.db
record(longout, “example”) {
        field(HOPR, “100”)
        field(DRVH, “99”)
        field(EGU, “arb.”)
}
EOF

./bin/linux-x86_64-debug/softIocPVA -d \
  example.db
```

```
$ pvget example
example                    42

$ pvget -v example
example
epics:nt/NTScalar:1.0
    int          value   42
    alarm_t      alarm NO_ALARM NO_STATUS NO_ALARM
    time_t       timeStamp 2017-10-06T14:37:23.290 0
    display_t    display
        double limitLow 0
        double limitHigh 100
        string   description
        string   format
        string   units arb.
    control_t    control
        double limitLow 0
        double limitHigh 99
        double minStep 0
```

Group PV Example

- NTTable
- Two columns: “A” and “B”
- Each column holds double

```
$ pvput TST:table1Tbl \
  value.A=[1,2,3,4]       \
  value.B=[5,6,7,8]
```

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Group PV Mapping Concepts

- Define mapping between structures
  - Process DB field(s)
    - eg. “record.VAL”
  - Group Structure(s)

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Group PV Example (1)

```
record(aai,  "TST:image1Labels_") { … }
record(aao, "TST:image1A") { … }
record(aao, "TST:image1B") { … }
record(bo,   "TST:image1Save") { … }
```

- Four records
- 1x Array of strings
- 2x Array of double
- 1x binary out
  - For side-effects

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Group PV Example (2)

```
record(aai,  "TST:image1Labels_") { … }
record(aao, "TST:image1A") { … }
record(aao, "TST:image1B") { … }
record(bo,   "TST:image1Save") { … }
```

```
record(aai, "TST:image1Labels_") {
    field(FTVL, "STRING")
    field(NELM, "2")
    field(INP , {const:["Column A", "Column B"]})
    info(Q:group, {
        "TST:table1Tbl":{
            +id:"epics:nt/NTTable:1.0",
            "labels":{+type:"plain", +channel:"VAL"}
        }
    })
}
```

```
$ pvget TST:table1Tbl
    string[] labels [Column A,Column B]
…
```

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Group PV Example (3)

```
record(aai,  "TST:image1Labels_") { … }
record(aao, "TST:image1A") { … }
record(aao, "TST:image1B") { … }
record(bo,   "TST:image1Save") { … }
```

```
record(aao, "TST:image1A") {
    field(FTVL, "DOUBLE")
    field(NELM, "10")
    info(Q:group, {
        "TST:table1Tbl":{
            "value.A":{+type:"plain",
                       +channel:"VAL",
                       +putorder:1}
        }
    })
}
```

```
$ pvget TST:table1Tbl
structure value
        double[] A [1,2,3,4]
…
```

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
        string[]  labels
        alarm_t alarm
            int    severity
            int    status
            string message
        time_t   timeStamp
            long   secondsPastEpoch
            int    nanoseconds
            int    userTag
        structure value
            double[] A
            double[] B
```

Group PV Example (4)

```
record(aai,  "TST:image1Labels_") { … }
record(aao, "TST:image1A") { … }
record(aao, "TST:image1B") { … }
record(bo,   "TST:image1Save") { … }
```

```
record(aao, "TST:image1B") {
    field(FTVL, "DOUBLE")
    field(NELM, "10")
    info(Q:group, {
        "TST:table1Tbl":{
            "":{+type:"meta", +channel:"VAL"},
            "value.B":{+type:"plain",
                       +channel:"VAL",
                       +putorder:1}
        }
    })
}
```

```
$ pvget TST:table1Tbl
structure value
        double[] B [5,6,7,8]
…
```

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Osprey DCS

EPICS

Group PV Example (5)

```
record(aai,  "TST:image1Labels_") { … }
record(aao, "TST:image1A") { … }
record(aao, "TST:image1B") { … }
record(bo,   "TST:image1Save") { … }
```

```
record(bo, "TST:image1Save") {
    info(Q:group, {
        "TST:table1Tbl":{
            "_save":{+type:"proc",
                     +channel:"VAL",
                     +putorder:2}
        }
    })
}
```

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

Group PV Mappings (2)

- TST:table1Labels_.VAL ↔ TST:table1Tbl.labels
- TST:table1A.VAL ↔ TST:table1.value.A
- TST:table1B.VAL ↔ TST:table1.value.B
- TST:table1B.VAL → TST:table1.alarm .timeStamp
- TST:table1Save ← (hidden)

```
$ pvinfo TST:table1Tbl
CHANNEL  : TST:table1Tbl
STATE    : CONNECTED
ADDRESS  : 10.44.0.1:5075
epics:nt/NTTable:1.0
    structure record
        structure _options
            uint queueSize
            boolean atomic
    string[]  labels
    alarm_t alarm
        int    severity
        int    status
        string message
    time_t   timeStamp
        long   secondsPastEpoch
        int    nanoseconds
        int    userTag
    structure value
        double[] A
        double[] B
```

PVAccess Simple(r) client API

- Simple, but not EZ

```
#include <iostream>
#include "pva/client.h"

int main(int argc, char *argv[])
{
    try {
        if(argc<=1) { std::cerr<<"Usage: "<<argv[0]<<" <pvname>\n"; return 1; }

        pvac::ClientProvider provider("pva");
        pvac::ClientChannel channel(provider.connect(argv[1]));
        std::cout << channel.name() <<" : "<< channel.get() <<"\n";
    }catch(std::exception& e){
        std::cerr<<"Error: "<<e.what()<<"\n";
        return 1;
    }
}
```

Blocks for I/O completion
(callback variation also available)

PV Access C++ Documentation

- Development version documentation
  - http://epics-base.github.io/pvAccessCPP/
- Released documentation (after release)
  - http://epics-pvdata.sourceforge.net/literature.html

- QSRV demo
  - https://github.com/epics-base/pva2pva/blob/master/iocBoot/iocimagedemo/table.db