

# CA Channel vs. IOC Record vs. Python CAS

Kay Kasemir, August 2018

# What is a Process Variable?

Good question!

"A named piece of data with attributes"

# Channel Properties (= Data in a Process Variable)

Each channel comes with properties:

- Value
  - String, double, int or ...
  - Scalar or array
- Time stamp
  - Up to nanosecond precision
- Severity code
  - OK, MINOR, MAJOR, or INVALID
- Status code to qualify the severity
  - OK, READ error, WRITE error, at HIGH limit, ...
- Units, suggested display range, control limits, alarm limits.

Client uses 'request' type to select what it needs

# What is a PV (Channel)?

Whenever there's a CA server out there which decides to respond to a search request, that's a PV!

- IOC responds to "record.field"
  - Almost every field of every record is a PV
  - There's a mapping from record fields to channel properties  
(you might need to read the source code of the specific record for full detail)
- With pccaspy, your python code creates the PVs and sets the channels' properties
  - Nobody will know what you decide to put there

# What is a Process Variable?

## Analog Record (ai, calc, ...)

### Fields

- VAL
- DESC
- EGU
- PREC
- LOPR, HOPR
- LOLO, LOW, HIGH, HIHI
- TIME



## Channel

### **DBR\_CTRL\_DOUBLE**

- value
- status/severity
- time stamp
- units
- precision
- display limits
- warn limits
- alarm limits
- ctrl limits

## My Python Program

```
from pcas import Driver
```

```
...
```

# Consider this Record

```
record(calc, "t1:calcExample")
{
    field(DESC, "Sawtooth Ramp")
    field(SCAN, "1 second")
    field(CALC, "(A<10)?(A+1):0")
    field(INPA, "t1:calcExample.VAL")
    field(PREC, "2")
    field(EGU, "steps")
    field(LOPR, "0")
    field(HOPR, "10")
    field(HIGH, "8")
    field(HIHI, "9")
}
```



**t1:calcExample as DBR\_CTRL\_DOUBLE**

- value = VAL
- status/severity = STAT/SEVR
- Units = EGU
- Precision = PREC
- display limits = LOPR/HOPR
- warn limits = LOW/HIGH
- alarm limits = LOLO/HIHI
- ctrl limits = LOPR/HOPR

**t1:calcExample as DBR\_TIME\_DOUBLE**

- value = VAL
- status/severity = STAT/SEVR
- time stamp = TIME

**t1:calcExample.DESC or CALC as DBR\_TIME\_STRING**

- value = DESC
- status/severity = STAT/SEVR
- time stamp = TIME

**t1:calcExample.SCAN as DBR\_CTRL\_ENUM**

- value = SCAN
- status/severity = STAT/SEVR
- labels = [ "Passive", .., "10 second", .., ".1 second" ]

# Example: AI record "fred"

- PV "fred" or "fred.VAL"
  - value property of channel = VAL field of record.
    - Type double, one element (scalar).
  - time property = TIME field
  - status = STAT
  - Severity = SEVR
  - units = EGU
  - Precision = PREC
  - display limit low, high = LOPR, HOPR
  - control limit low, high = LOPR, HOPR
  - alarm limits = LOLO, LOW, HIGH, HIHI
- Makes a lot of sense.
  - GUI can display the value together with units, formatted according to the precision, as e.g. "12.37 volts".

# Example: AI record "fred"

- PV "fred.SCAN", read as a number
  - value property of channel = Enum index of record's SCAN value
    - 0 for "Passive", 1 for "Event", .., 6 for "1 second", ..
  - time property = TIME field
  - status = STAT
  - Severity = SEVR
  - units = EGU
  - Precision = 0
  - display limit low, high = 0, ??
  - control limit low, high = 0, ??
  - alarm limits = 0, 0, 0, 0
- Makes some sense, but
  - Units don't really apply to the SCAN field.
  - Its value range is really limited by the available SCAN choices, not 0..??.



# Channel Access

vs.

# PV Access

- Original EPICS network protocol
- Typically used with IOCs & records
  - You get what the records provide
- The request types are fixed.
  - Predefined "DBR\_..." types
    - Just value.
    - Value with status and severity.
    - Value with status, severity and time stamp.
    - "Everything:" value, units, time, status, limits, ...
  - Client always gets the full requested DBR\_.. Data
  - Cannot ask for custom combination like value, units, seconds of time stamp.
- With your own CA server, you cannot support new properties like 'color'.

- Alternate network protocol since ~2015
- Can be used with same IOCs and records
  - You get what the records provide
- You can request anything
  - Suggested "Normative Types"
    - Same concept as DBR\_.. types
  - Optimized: Under the hood, only changes are sent to client
  - Can ask for custom combination like value, unit, seconds of time stamp.
- With your own PVA server, you can support new properties like 'color'.

# Key Points

- Channel != Record
  - IOC maps fields of records to properties of channel
  - This separation allowed development of generic clients (displays, alarm tools, archives) independent from IOCs
- There is a growing number of non-IOC CA servers
  - pccaspy, ...
  - They provide channels “x” with value, units, precision, alarms, time.. but that doesn't mean you can read/write “x.EGU”, “x.PREC”, ...  
There is no record!
- PVAccess allows custom data types
  - But to remain compatible, try to support the Normative Types