

# EPICS Database Updates

Kay Kasemir

Many slides from Andrew Johnson,  
APS/ANL

Jan. 2022

# Comparably New Database Features

- Wildcards for 'patch' databases
- Long Strings
- JSON Notation for links and subscriptions

# 'Patch' Database

Database files can **modify fields** of existing records or **add** new fields/records

## ps.db

```
record(ai, "voltage")
{
  field(DTYP, ... INP ...
  field(SCAN, "2 second")
  field(FLNK, "current")
}

record(ai, "current")
{
  field(DTYP, ... INP ...
  field(FLNK, "power")
}

record(calc, "power")
{
  field(INPA, "voltage")
  field(INPB, "current")
  field(CALC, "A*B")
}

record(bo, "ps_enable")
...
```

## limit.db

```
record(ai, "voltage")
{
  field(SCAN, ".1 second")
}

record(calc, "power")
{
  field(FLNK, "limit")
}

record(calcout, "limit")
{
  field(INPA, "power")
  field(CALC, "A < 100")
  field(OUT, "ps_enable")
}
```

## st.cmd

```
dbLoadRecords "db/ps.db"

dbLoadRecords "db/limit.db"
```

May help to organize records.  
"ps.db" can be used standalone,  
or with "limit.db".

Order of loading \*.db files matters!

# 'Patch' Database ...

Note that the record type cannot be changed.

Must know the **type** of record, or use **"\*"** to patch any type

## ps.db

```
record(ai, "voltage")
{
  field(DTYP, ... INP ...
  field(SCAN, "2 second")
  field(FLNK, "current")
}

record(ai, "current")
{
  field(DTYP, ... INP ...
  field(FLNK, "power")
}

record(calc, "power")
{
  field(INPA, "voltage")
  field(INPB, "current")
  field(CALC, "A*B")
}

record(bo, "ps_enable")
...
```

## limit.db

```
record(ai, "voltage")
{
  field(SCAN, ".1 second")
}

record("*", "power")
{
  field(FLNK, "limit")
}

record(calcout, "limit")
{
  field(INPA, "power")
  field(CALC, "A < 100")
  field(OUT, "ps_enable")
}
```

In this example, the patch database doesn't care if "power" is a calc, calcout, .. type of record

# Long Strings

- Unfortunately, EPICS 'STRING' was initially defined as CHAR[40]
- Channel Access uses DBF\_STRING = CHAR[40]

- Record names, INP links can now be much longer

```
field(INPA, "SomeReallyLongRecordNameThatExceeds40Characters PP MS")
```

..but you cannot read/modify them via channel access

# Old Workaround

Use

```
record(waveform, "long_string")
{
    field(FTVL, "CHAR")
    field(NELM, "200")
}
```

.. and (!) tell client that CHAR[] is not meant to be BYTE[] but long STRING.

caget -S .....

Format: String in operator interface tools

# New Workaround

- IOC turns any string for VAL, INP, DOL into long-string CHAR[] if PV name ends in '\$'
- Long-String input and output records "Isi" and "Iso" default to VAL of type STRING, but VAL\$ becomes CHAR[]
- Clients still need to be told if CHAR[] is BYTE[] or long string!

# JSON Notation for INP links

- New support for JSON notation allows providing more detail
- First practical use: {const:...}

```
record(lsi, "lsi_init")
{
  field(INP, {const:"Hello, this is a string and it can be really long!"})
  field(PINI, "YES")
}
```

```
record(waveform, "array")
{
  field(FTVL, "CHAR")
  field(NELM, "200")
  field(INP, {const:[65, 66, 67, 68, 69]})
  field(PINI, "YES")
}
```



# JSON: “JavaScript Object Notation”

- Standard text representation of data
  - Can represent “anything”
  - More compact than e.g. XML

```
{  
  number: 3.14,  
  text: "Hello!",  
  substructure: {  
    numbers: [1, 2, 3, 4]  
  }  
}
```

- EPICS starts to use it to provide details for
  - Links, see <https://epics.anl.gov/base/R7-0/6-docs/links.html>
  - Subscriptions, see <https://epics.anl.gov/base/R7-0/6-docs/filters.html>
  - .. More to come

# JSON Notation for Subscription Filters

- In one window, run

```
cd /ics/examples/01_first_steps  
softIoc -d ramp1.db
```

- In other window, compare the following

```
camonitor limit  
camonitor 'limit.{ts:{}}'
```

```
camonitor ramp  
camonitor 'ramp.{dbnd:{abs:1.9}}'
```