What is an EPICS Database?

Andrew Johnson
APS
Outline

- Records
- Fields and field types
- Record Scanning
- Input and Output record types
- Hardware support
- Links
- Chaining Records together
- Protection mechanisms
- Alarms, deadbands, simulation and security
Database = Records + Fields + Links

- A control system using EPICS will contain one or more IOCs
- Each IOC loads one or more Databases telling it what to do
- A Database is a collection of Records of various types
- A Record is an object with:
  - A unique name
  - A behaviour defined by its record type (class)
  - Controllable properties (fields)
  - Optional associated hardware I/O (device support)
  - Links to other records
Records are active — they can do things:
- Get data from other records or from hardware
- Perform calculations
- Check values are in range & raise alarms
- Put data to other records or to hardware
- Activate or disable other records
- Wait for hardware signals (interrupts)

What a record does depends upon its record type and the settings of its fields

No action occurs unless a record is processed
How is a Record implemented?

- A ‘C’ structure with both data storage and pointers to record type information
- A record definition within a database provides
  - Record name
  - The record’s type
  - Values for each design field
- A record type provides
  - Definitions of all the fields
  - Code which implements the record behaviour
- New record types can be added to an application as needed
One view of a Record

The small CapFast symbol for an Analogue Output record
Another view

1999/ Ph 514: What is an EPICS Database?
The IOC’s view

The full .db file entry for an Analogue Output Record

```
record(ao,"DemandTemp") {
  field(DESC,"Temperature")
  field(ASG,"")
  field(SCAN,"Passive")
  field(PINI,"NO")
  field(DTYP,"VMIC 4100")
  field(DISV,"1")
  field(SDIS,"")
  field(DISS,"NO_ALARM")
  field(PRIO,"LOW")
  field(FLNK,"")
  field(OUT,"#C0 S0")
  field(OROC,"0.0e+00")
  field(DOL,"")
  field(OMSL,"supervisory")
  field(OIF,"Full")
  field(PREC,"1")
  field(LINR,"NO CONVERSION")
  field(EGUF,"100")
  field(EGUL,"0")
  field(EGU,"Celcius")
  field(DRVH,"100")
  field(DRLV,"0")
  field(HOPR,"80")
  field(LOPR,"10")
  field(HIHI,"0.0e+00")
  field(LOLO,"0.0e+00")
  field(HIGH,"0.0e+00")
  field(LOW,"0.0e+00")
  field(HHSV,"NO_ALARM")
  field(LLSV,"NO_ALARM")
  field(HSV,"NO_ALARM")
  field(LLSV,"NO_ALARM")
  field(HYSV,"NO_ALARM")
  field(ADEL,"0.0e+00")
  field(MDEL,"0.0e+00")
  field(SIOL,"")
  field(SMIL,"")
  field(SIMS,"NO_ALARM")
  field(IPOA,"Continue normally")
  field(IVOV,"0.0e+00")
}
```

This shows only the design fields, there are other fields which are used at run-time
Fields are for...

Defining

- What causes a record to process
- Where to get/ put data from/ to
- How to turn raw I/ O data into a numeric engineering value
- Limits indicating when to report an alarm
- When to notify value changes to a client monitoring the record
- A Processing algorithm
- Anything else which needs to be set for each record of a given type

Holding run-time data

- Input or output values
- Alarm status, severity and acknowledgements
- Processing timestamp
- Other data for internal use
Field types

- Fields can contain
  - Integers
    - char, short or long
    - signed or unsigned
  - Floating-point numbers
    - float or double
  - Strings
    - max length 40 characters or less
  - Menu choices
    - select one from several strings
    - stored as a short integer
  - Links
    - to other records in this or other IOCs
    - to hardware signals (device support)
    - provide a means of getting or putting a value
  - Other private data
    - not directly accessible
All Records have these fields

### Design fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>28 Character unique name</td>
</tr>
<tr>
<td>DESC</td>
<td>28 Character description</td>
</tr>
<tr>
<td>ASG</td>
<td>Access security group</td>
</tr>
<tr>
<td>SCAN</td>
<td>Scan mechanism</td>
</tr>
<tr>
<td>PHAS</td>
<td>Scan order (phase)</td>
</tr>
<tr>
<td>PINI</td>
<td>Process at startup?</td>
</tr>
<tr>
<td>PRIO</td>
<td>Scheduling priority</td>
</tr>
<tr>
<td>SDIS</td>
<td>Scan disable input link</td>
</tr>
<tr>
<td>DISV</td>
<td>Scan disable value</td>
</tr>
<tr>
<td>DISS</td>
<td>Disabled severity</td>
</tr>
<tr>
<td>FLNK</td>
<td>Forward link</td>
</tr>
</tbody>
</table>

### Run-time fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROC</td>
<td>Force processing</td>
</tr>
<tr>
<td>PACT</td>
<td>Process active</td>
</tr>
<tr>
<td>STAT</td>
<td>Alarm status</td>
</tr>
<tr>
<td>SEVR</td>
<td>Alarm severity</td>
</tr>
<tr>
<td>TPRO</td>
<td>Trace processing</td>
</tr>
<tr>
<td>UDF</td>
<td>Set if record value undefined</td>
</tr>
<tr>
<td>TIME</td>
<td>Time when last processed</td>
</tr>
</tbody>
</table>
Record Scanning

- **SCAN** field is a menu choice from
  - Periodic — 0.1 seconds .. 10 seconds
  - I/ O Interrupt (if device supports this)
  - Soft event — **EVNT** field
  - Passive (default)

- The number in the **PHAS** field allows processing order to be set within a scan
  - Records with **PHAS=0** are processed first
  - Then those with **PHAS=1**, **PHAS=2** etc.

- Records with **PINI=YES** are processed once at startup

- **PRIO** field selects Low/ Medium/ High priority for Soft event and I/ O Interrupts

- A record is also processed whenever any value is written to its **PROC** field
Input records often have these fields:

- **INP**: Input link
- **DTYP**: Device type
- **RVAL**: Raw data value
- **VAL**: Engineering value
- **LOPR**: Low operator range
- **HOPR**: High operator range

- **Analogue I/O records have these fields:**
  - **LINR**: Unit conversion control
    - No conversion, Linear, breakpoint tables...
  - **EGUL**: Low engineering value
  - **EGUF**: High engineering value
  - **EGU**: Engineering unit string
Periodic Input

- Analogue Input “Temperature”
- Reads from the Xycom XY566 ADC Card 0 Signal 0
- Gets a new value every 0.1 seconds
- Data is converted from ADC range to 0..120 Celsius

EPICS

 análógus bemenet "hőmérséklet"
 olvasható az Xycom XY566 ADC Card 0 Signal 0
 minden 0.1 másodperc alatt kap új értéket
 adat konvertálás a ADC tartománytól 0..120 Celsiusig
◆ Binary Input “VentValve”
◆ Reads from Allen-Bradley TTL I/O Link 0, Adaptor 0, Card 3, Signal 5
◆ Processed whenever value changes
◆ 0 = “Closed”, 1 = “Open”
◆ Major alarm when valve open
Output records often have these fields:

- **OUT**: Output link
- **DTYP**: Device type
- **VAL**: Engineering value
- **RVAL**: Raw output value
- **DOL**: Input link to fetch output value
- **OMSL**: Output mode select
  - Supervisory, Closed Loop
- **LOPR**: Low operator range
- **HOPR**: High operator range

➤ Analogue outputs also have these fields:

- **OROC**: Output rate of change
- **OIF**: Incremental or Full output
- **OVAL**: Output value
- **DRVH**: Drive high limit
- **DRVL**: Drive low limit
- **IVOA**: Invalid output action
- **IVOV**: Invalid output value
- **RBV**: Read-back value
Passive Output

Solenoid

DOL FLNK
SLNK VAL
SDIS OUT

DTYP: XY220
SCAN: Passive
PHAS: 0
ZNAM: Locked
ONAM: Unlocked

- Binary Output “Solenoid”
- Controls Xycom XY220 Digital output Card 2 Signal 12
- Record is only processed by
  - Channel Access ‘put’ to a PP field (e.g. .VAL)
  - Another record writes to a PP field (e.g. .VAL)
  - Forward Link from another record
  - Another record reads this with .PP
Links

A link is a type of field, and is one of

- Input link
  - Fetches data
- Output link
  - Writes data
- Forward link
  - Points to the record to be processed once this record finishes processing
Input and Output links may be...

- Constant numeric value, eg:
  
  0
  3.1415926536
  1.6e-19

- Hardware link
  
  A hardware I/O signal selector, the format of which depends on the device support layer

- Process Variable link — the name of a record, which at run-time is resolved into

  - Database link
    Named record is in this IOC

  - Channel Access link
    Named record not found in this IOC
Hardware links

VME_IO           #Cn Sn @parm
                  Card, Signal
INST_IO          @parm
CAMAC_IO         #Bn Cn Nn An Fn @parm
                  Branch, Crate, Node, Address, Function
AB_IO            #Ln An Cn Sn @parm
                  or  #Ln Pn Cn Sn Fn @parm
                  Link, Adaptor, Card, Signal, Flag
GPIB_IO          #Ln An @parm
                  Link, Address
BITBUS_IO        #Ln Nn Pn Sn @parm
                  Link, Node, Port, Signal
BBGPIB_IO        #Ln Bn Gn @parm
                  Link, Bitbus Address, GPIB Address
VXI_IO           #Vn Cn Sn @parm
                  or  #Vn Sn @parm
                  Frame, Slot, Signal
Database links

These comprise:

- The name of a record in this IOC
  
  `myDb:myRecord`

- An optional field name
  
  `.VAL` (default)

- Process Passive flag
  
  `.NPP` (default)

  `.PP`

- Maximize Severity flag
  
  `.NMS` (default)

  `.MS`

For example:

`M1:current.RBV .NPP .MS`

- NB: Get with `.PP` from record with asynchronous device support will not return the new value
Channel Access links

◆ Specified like a database link
◆ Name is not a record found in this IOC
◆ Use Channel Access protocol to communicate with remote IOC
◆ May include a field name (default .VAL)
◆ .PP Link flags are ignored:
  ◆ Input links always .NPP
  ◆ Output links follow PP attribute of destination field
  ◆ This behavior identical to all other CA clients
◆ .MS Link flags apply to Input links:
  ◆ Input links honour the .NMS or .MS flags
  ◆ Output links always .NMS
◆ Additional flags
  .CA Forces a “local” link to use CA
  .CP Process record when value changes
  .CPP Like .CP but only if Scan Passive
## Link flag summary

<table>
<thead>
<tr>
<th></th>
<th><strong>Input Links</strong></th>
<th><strong>Output Links</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DB Links</strong></td>
<td>.PP or .NPP</td>
<td>.PP or .NPP</td>
</tr>
<tr>
<td></td>
<td>.MS or .NMS</td>
<td>.MS or .NMS</td>
</tr>
<tr>
<td><strong>CA Links</strong></td>
<td>Always .NPP</td>
<td>.PP behavior of destination field.</td>
</tr>
<tr>
<td></td>
<td>.MS or .NMS</td>
<td>Always .NMS</td>
</tr>
<tr>
<td></td>
<td>.CA to force.</td>
<td>.CA to force.</td>
</tr>
<tr>
<td></td>
<td>.CP to process this record on change.</td>
<td>.CPP only process if SCAN=Passive</td>
</tr>
</tbody>
</table>

Pages 16 thru 23 of the IOC Application Developer’s Guide cover this topic.
Records do not access hardware directly
- The Device Support layer performs I/O operations on request
- A particular device support provides I/O for a single record type
- The **DTYP** field determines which device support to use
- The device support selected determines the format of the link (**INP** or **OUT** field) containing device address information
- Adding new device support does not require change to the record software
- Device support may call other software to do work for it (Driver Support)
Synchronous vs Asynchronous I/O

- EPICS rules do not allow device support to busy-wait (poll for results of slow I/O).
- Register-based VME cards usually give an immediate response: synchronous.
- When called, synchronous device support performs all I/O before returning.
- Serial & I/O-bus devices take a long time (>10ms) to return data: asynchronous.
- Asynchronous device support starts I/O when record calls it, flags it as incomplete by setting `PACT` true before returning.
- Once results are available (CPU interrupt), device support calls the record’s process routine which finishes the operation.
Soft Device Support

- Input and Output records are designed to perform hardware I/O via device support
- They can also access other records via DB or CA links, using soft device support
- 2 kinds of support are provided:
  - Soft Channel
    - Get/ Put VAL through link, no conversion
  - Raw Soft Channel
    - Inputs
      - Get RVAL via input link
      - Convert RVAL to VAL (device specific)
    - Outputs
      - Convert VAL to RVAL (device specific)
      - Put RVAL to output link
Forward links

- Usually a database link referring to a record in same IOC
- Channel Access links possible, must name the `PROC` field of the remote record
- No flags (`.PP`, `.NMS` etc)
- Destination record must have
  \[\text{SCAN} = \text{Passive}\]
  for it to be processed
- Does not pass a value, just causes subsequent processing
281999/ Ph 514: What is an EPICS Database?
Which record is never processed?
Which record is processed twice?
The PACT field

- Every record has a boolean run-time field called PACT (Process Active)
- PACT breaks loops of linked records
- It is set to ‘true’ early in the act of processing the record
  - PACT is true whenever a link in that record is used to get/put a value
- PACT is set to false after record I/O and forward link processing are finished
- A .PP link can never make a record process if it has PACT true
  - Input links take the current value
  - Output links just put their value
What happens here?
Preventing records from processing

- It is useful to be able to stop an individual record from processing on some condition.
- Before record-specific processing is called, a value is read through the SDIS input link into DISA.
- If DISA = DISV, the record will not be processed.
- A disabled record is alarmed by giving the desired severity in the DISS field.
- The FLNK of a disabled record is not triggered.
How are records given CPU time?

Several vxWorks tasks are used:

- callback (3 priorities) — I/O Interrupt
- scanEvent — Soft Event
- scanPeriod — Periodic
  - A separate task is used for each scan period
  - Faster scan rates are given higher vxWorks task priority
- Channel Access tasks use lower priority than record processing
  - If a CPU spends all the time doing I/O and processing, you will be unable to control or monitor the IOC via the network
What could go wrong here?
Lock-sets

- Prevent a record from being processed simultaneously from two scan tasks
- A lock-set is a group of records interconnected by:
  - Output Database links
  - Forward links
  - Input links which are .PP or .MS
  - Arrays
- Lock-sets are determined automatically by the IOC at start-up

You can split a lock set with
- Channel Access links, using .CA flag
- Database links which are .NPP .NMS
Every record has the fields

- **SEVR**: Alarm Severity
  - NONE, MINOR, MAJOR, INVALID
- **STAT**: Alarm Status (reason)
  - READ, WRITE, UDF, HIGH, LOW, STATE, COS, CALC, DISABLE, etc.

Most numeric records check **VAL** against **HIHI**, **HIGH**, **LOW** and **LOLO** fields after the value has been determined.

The **HYST** field prevents alarm chattering.

A separate severity can be set for each numeric limit (**HHSV**, **HSV**, **LSV**, **LLSV**)

Discrete (binary) records can raise alarms on entering a particular state, or on a change of state (COS)
Channel Access notifies clients which are monitoring a numeric record when:

- \texttt{VAL} changes by more than the value in field: \texttt{MDEL} Value monitors \texttt{ADEL} Archive monitors
- Record’s Alarm Status changes \texttt{HYST} Alarm hysteresis
- Analogue Input record provides smoothing filter to reduce input noise (\texttt{SMOO})
Breakpoint Tables

- Analogue Input and Output records can do non-linear conversions from/to the hardware data
- Breakpoint tables interpolate from given table
- To use, set the record’s LINR field to the name of the breakpoint table you want to use
- Example breakpoint table (in your .dbd file)

```plaintext
breaktable(attenuator1_1) {
    504  0
    795  1.25
    909  2.5
    1012 3.75
    ...
}
```
Input and output record types often allow simulation of hardware interfaces

- SIML: Simulation mode link
- SIMM: Simulation mode value
- SIOL: Simulation input link
- SIMS: Simulation alarm severity

Before using its device support, a record reads SIMM through the SIML link.

If SIMM=YES, device support is ignored; record I/O uses the SIOL link instead.

An alarm severity can be set whenever simulating, given by SIMS field.
A networked control system must have the ability to enforce security rules
  - Who can do what from where, and when?

In EPICS, security is enforced by the CA server (typically the IOC).

A record is placed in the Access Security Group named in its `ASG` field
  - DEFAULT is used if no group name is given

Rules for each group determine whether a CA client can read or write to records in the group, based on
  - Client user ID
  - Client IP address
  - Access Security Level of the field addressed
  - Values read from the database
Security rules are loaded from an Access Security Configuration File, for example:

```
UAG(users) {user1, user2}
HAG(hosts) {host1, host2}
ASG(DEFAULT) {
    RULE(1, READ)
    RULE(1, WRITE) {
        UAG(users)
        HAG(hosts)
    }
}
```

- If no security file is loaded, Security will be turned off and nothing refused
- For details and syntax, see Chapter 5 of the IOC Application Developers Guide