

Getting Started with EPICS Lecture Series

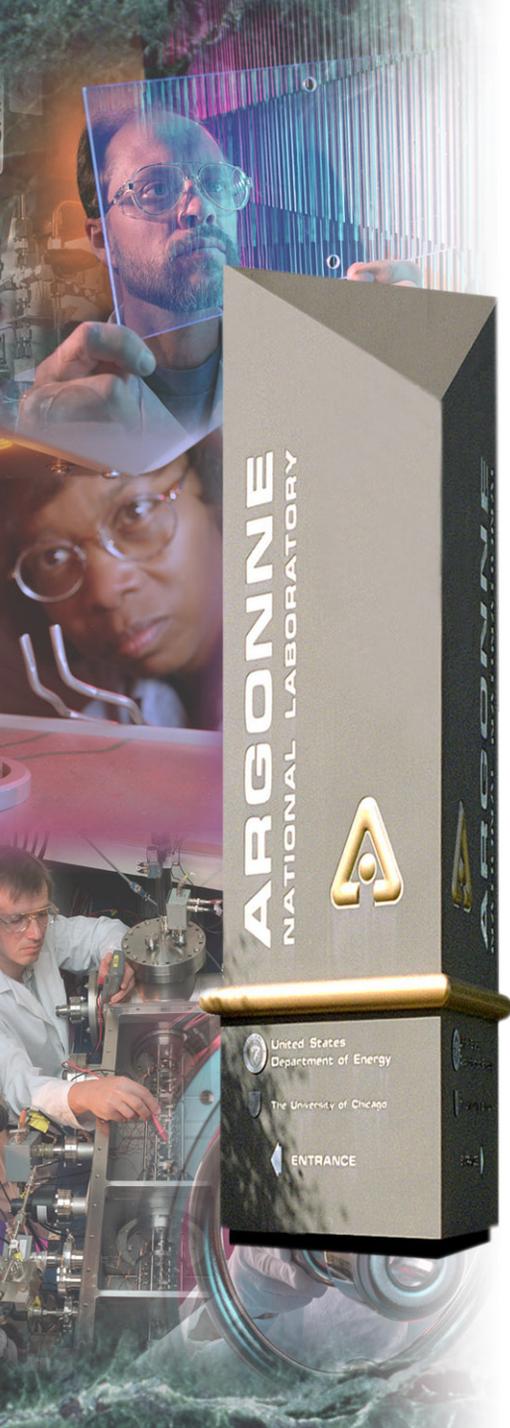
Introductory Session II

*John Maclean
8/16/04*

Argonne National Laboratory



*A U.S. Department of Energy
Office of Science Laboratory
Operated by The University of Chicago*



ARGONNE
NATIONAL LABORATORY



United States
Department of Energy

The University of Chicago

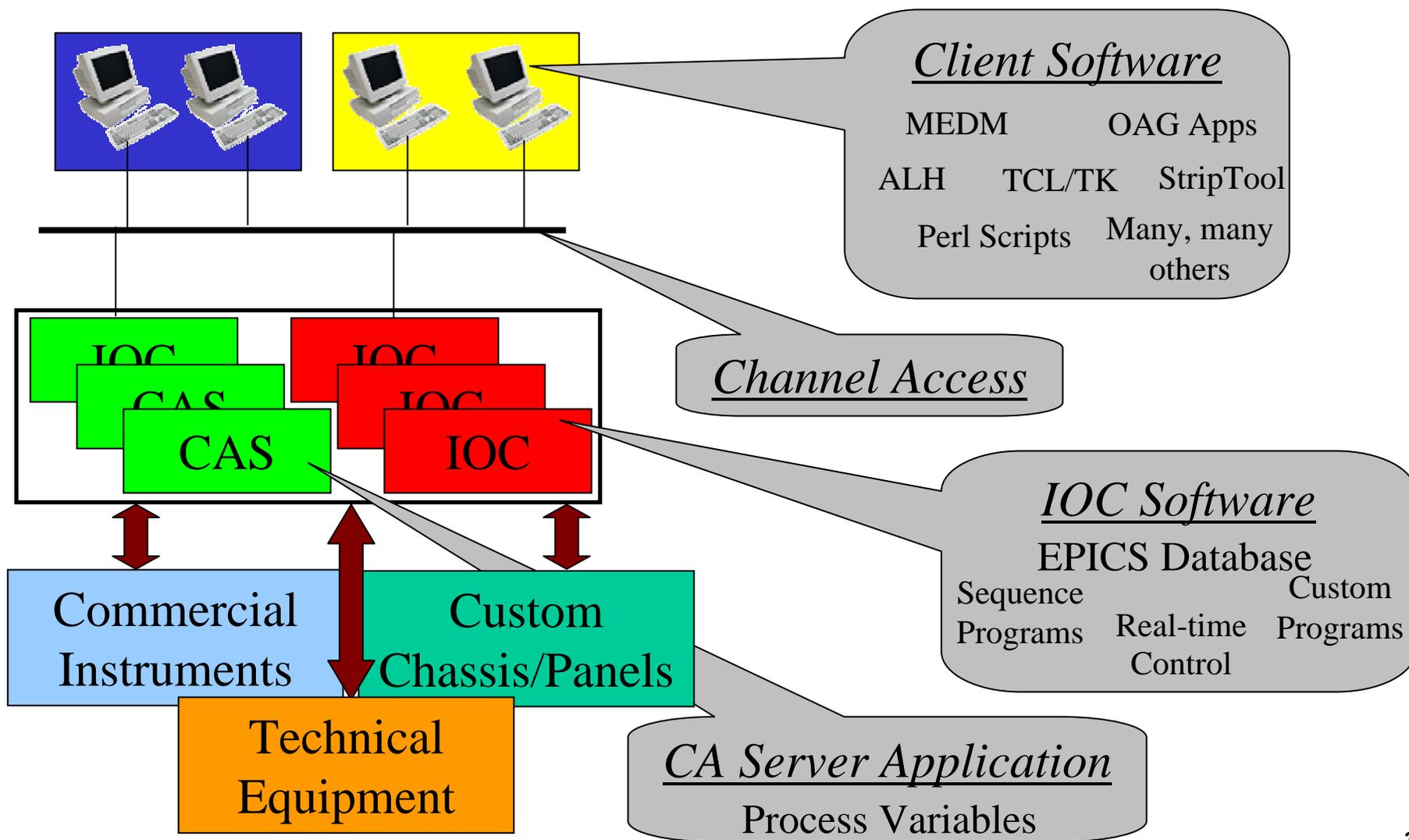
ENTRANCE

Overview

- Lay the foundation for understanding an **EPICS control system**
- **Introduce IOCs**
 - Channel Access (CA)
 - Database
 - Sequencer
 - Device Support
- **Choosing the correct tools for the job**
 - When to use a database
 - The sequencer, what is it good for?
 - Why write your own CA client program?
- **How fast is EPICS?**
- **How to find more information**
 - Website walk through
- **Virtual LINAC installation**



Canonical Form of an EPICS Control System



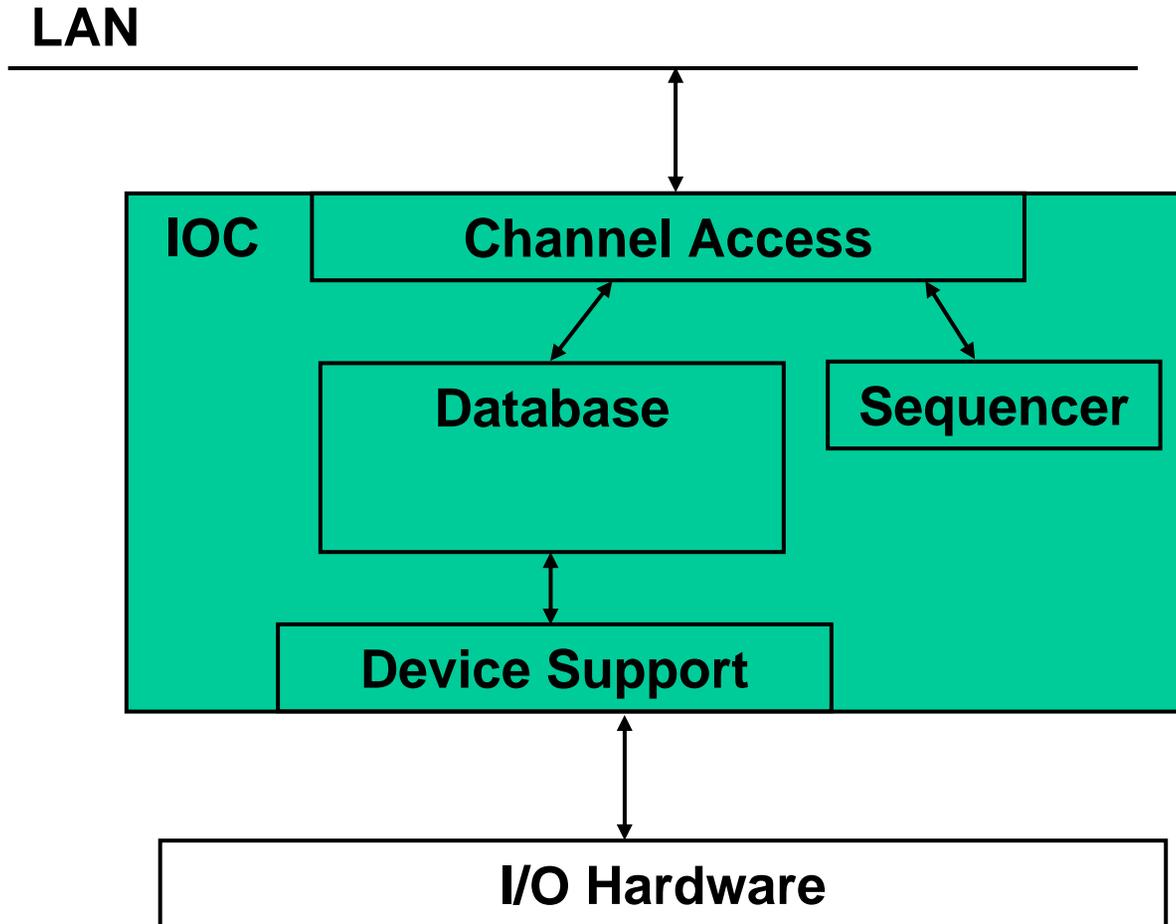
Introducing the IOC



- **Input Output Controller**
- **A computer running software called “*IOC Core*”**
- **The computer can be:**
 - VME based, running vxWorks (only choice until Release 3.14) or RTEMS
 - PC running Windows, Linux, RTEMS
 - Apple running OSX
 - UNIX Workstation running Solaris
- **Usually has Input and/or Output devices attached**
- **An EPICS control system must consist of at least one Channel Access Server (usually an IOC)**
- **An IOC has one or more *databases* loaded. The database tells it what to do**

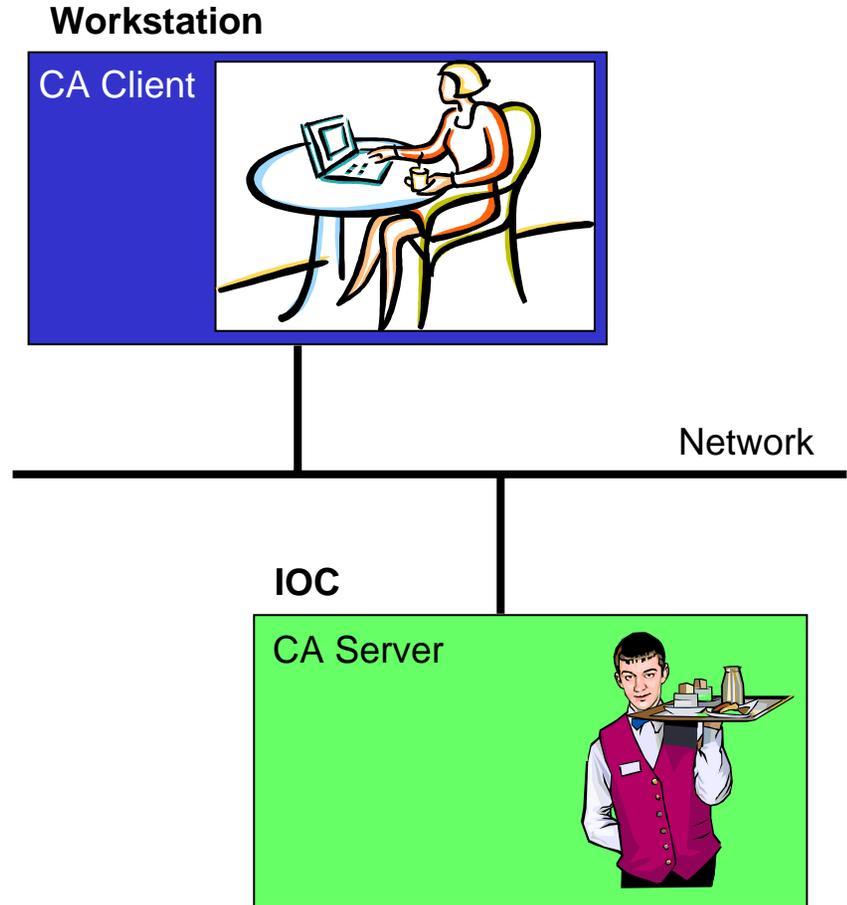
Inside an IOC

The major software components of an IOC (IOC Core)



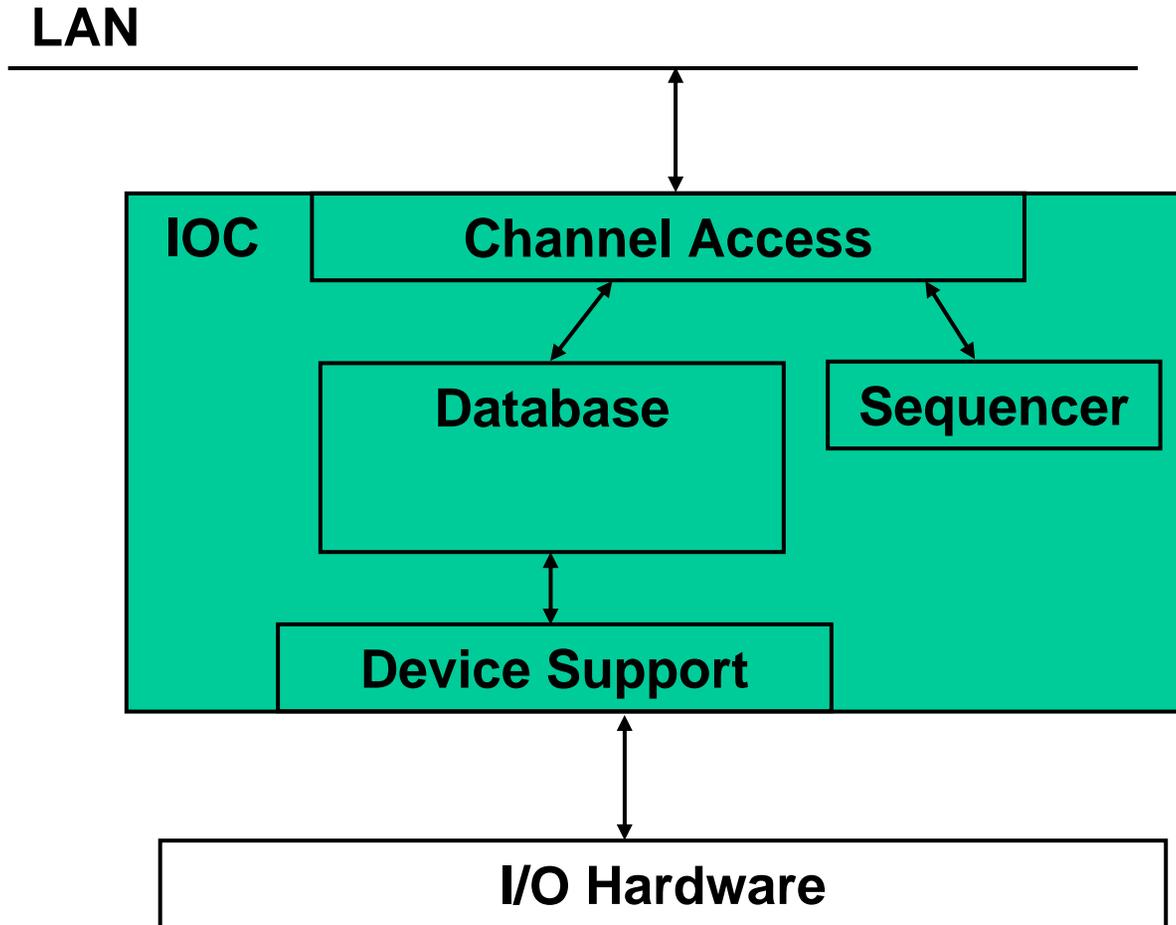
Channel Access

- Allows other programs (CA Clients) to see and change values of Process Variables in an IOC (CA Server)
- CA Clients may
 - Put (write)
 - Get (read)
 - Monitor
 data of Process Variables
- IOCs are both CA clients and CA servers. They can interact with data in other IOCs
- A CA Client can connect to many servers
- A CA Server may serve many clients
- A very efficient and reliable protocol



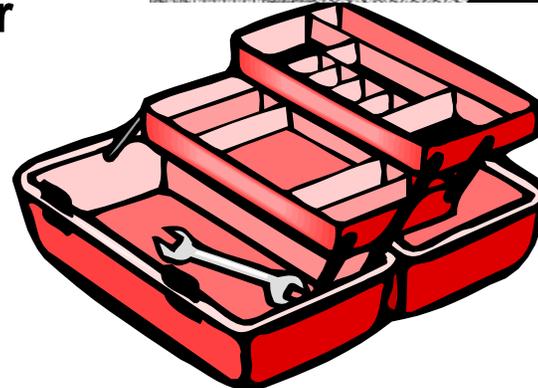
Inside an IOC

The major software components of an IOC (IOC Core)



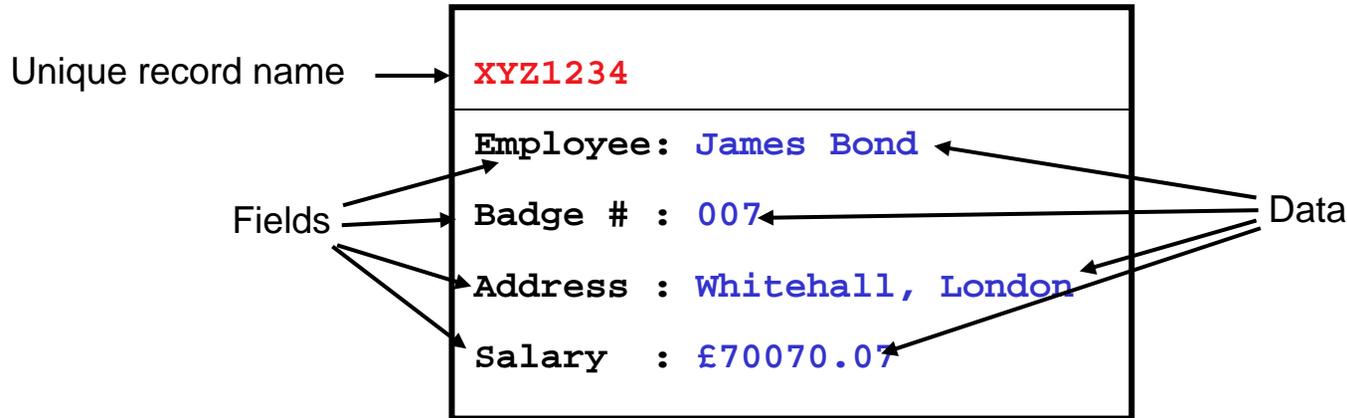
EPICS Databases – What are they for?

- **Interface to process instrumentation**
- **Distribute processing**
- **Provide external access to all process information**
- **Use common, proven, objects (records) to collect, process and distribute data**
- **Provide a common toolkit for creating applications**



What are records?

- **A record is an object with**
 - A unique name
 - Properties (fields) that contain information (data)
 - The ability to perform actions on that data
- **A personnel record in a relational database has a name, and fields containing data**

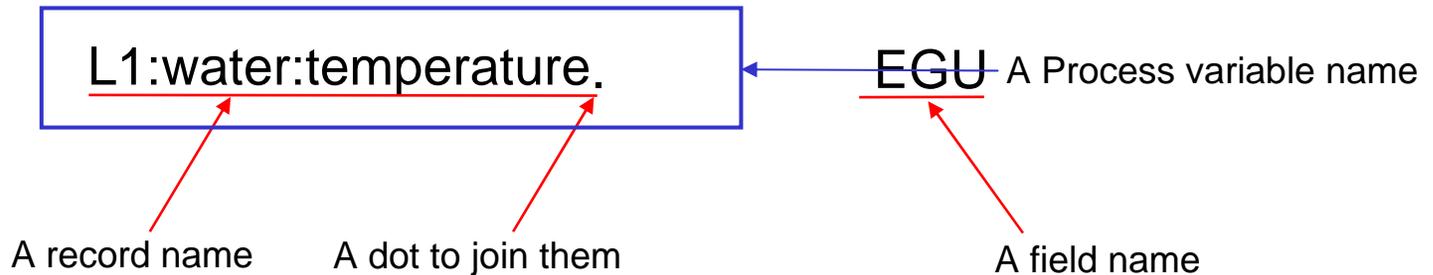


What are EPICS records?

- **A record is an object with...**
 - A unique name e.g. ***S28:waterPressure***
 - Controllable properties (fields) e.g. ***EGU***
 - A behavior - defined by its record type
 - Optional associated hardware I/O (device support)
 - Links to other records
- **Each field can be accessed individually by name**
- **A record name and field name combined give a the name of a process variable (PV)**
- **A Process Variable name is what Channel Access needs to access data**

A Process Variable name

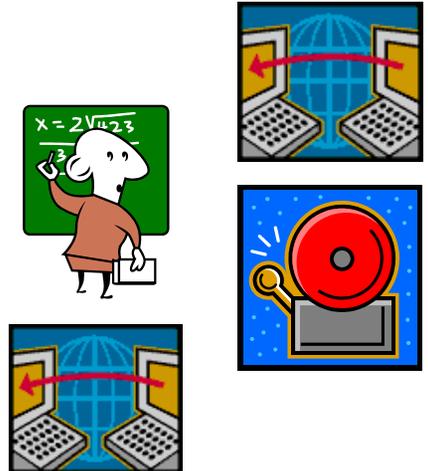
- A PV name is comprised of two parts
 - The record name, and
 - A the name of a field belonging to that record
- For example...



- Note that if no field name is given, Channel Access will default to using the .VAL field
- i.e. to CA, “L1:water:temperature” = “L1:water:temperature.VAL”

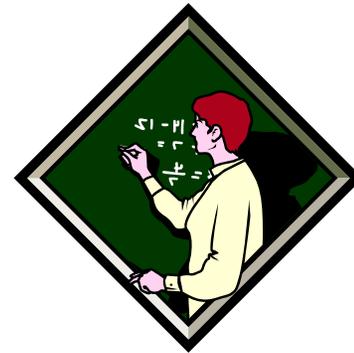
What do records do?

- **Records are active, they do things**
 - Get data from other records or from hardware
 - Perform calculations
 - Check values are in range and 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 type and the values in its fields**
- **A wide range of records have already been created**
- **New record types can be added to a new application as needed**
- **A record does nothing until it is *processed***



Record types

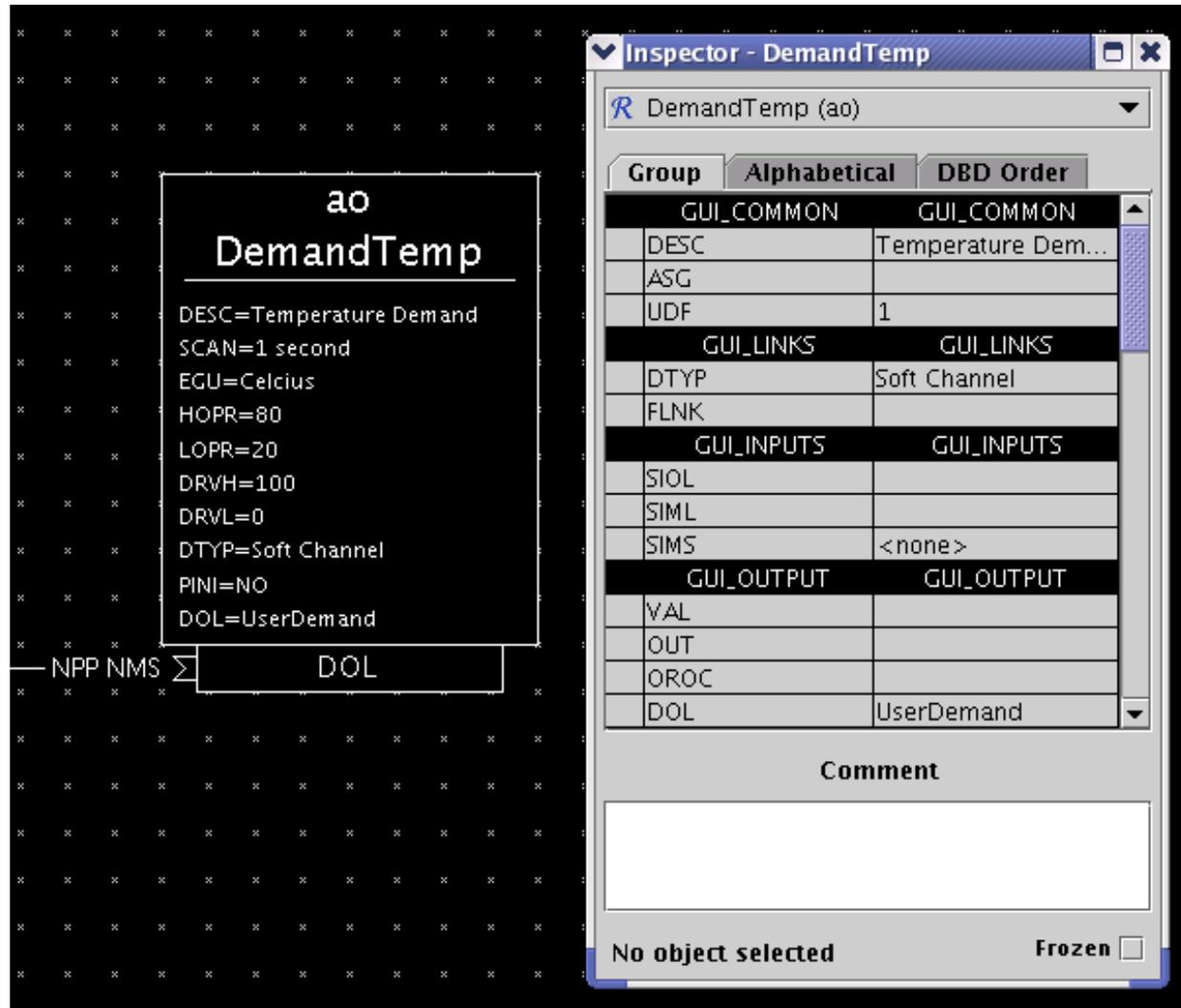
- **Classified into four general types**
- **Input: e.g.**
 - Analog In (AI)
 - Binary In (BI)
 - String In (SI)
- **Algorithm/control: e.g.**
 - Calculation (CALC)
 - Subroutine (SUB)
- **Output: e.g.**
 - Analog Out (AO)
 - Binary Out (BO)
- **Custom: e.g.**
 - Beam Position Monitor
 - Multi Channel Analyzer



Some record types

- Analog in
- Analog out
- Binary in
- Binary out
- Calculation
- Calculation out
- Compression
- Data fanout
- Event
- Fanout
- Histogram
- Motor
- Multi bit binary input
- Multi bit binary output
- PID control
- Pulse counter
- Pulse delay
- Scan
- Select
- Sequence
- String in
- String out
- Subarray
- Subroutine
- Waveform

Graphical view of a record



The image shows a graphical representation of an EPICS record and its corresponding Inspector window.

Graphical View (Left):

Record Name: **ao DemandTemp**

Parameters:

- DESC=Temperature Demand
- SCAN=1 second
- EGU=Celcius
- HOPR=80
- LOPR=20
- DRVH=100
- DRVL=0
- DTYP=Soft Channel
- PINI=NO
- DOL=UserDemand

Connections: NPP NMS Σ DOL

Inspector - DemandTemp (Right):

Record: DemandTemp (ao)

Sort: **Alphabetical** (Group, DBD Order)

| Group | Parameter | Value |
|------------|-----------|--------------------|
| GUI_COMMON | DESC | Temperature Dem... |
| | ASG | |
| | UDF | 1 |
| GUI_LINKS | DTYP | Soft Channel |
| | FLNK | |
| | | |
| GUI_INPUTS | SIOL | |
| | SIML | |
| | SIMS | <none> |
| GUI_OUTPUT | VAL | |
| | OUT | |
| | OROC | |
| | DOL | UserDemand |

Comment: [Empty text area]

Status: No object selected | Frozen

IOC view of a record

```

record(ao,"DemandTemp") {
    field(DESC,"Temperature")
    field(ASG,"")
    field(SCAN,"Passive")
    field(PINI,"NO")
    field(PHAS,"0")
    field(EVNT,"0")
    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")

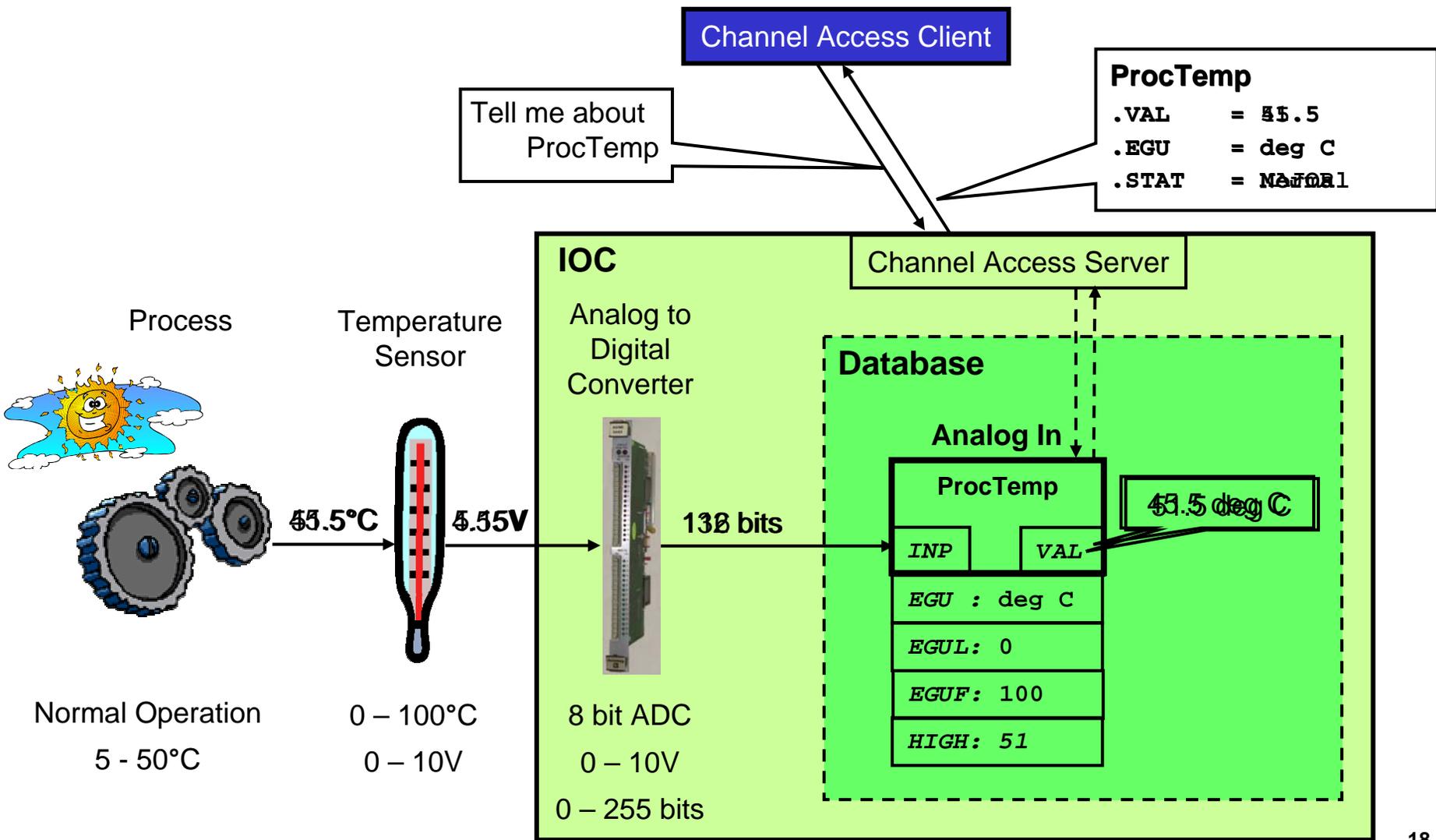
    field(DRVH,"100")
    field(DRVL,"0")
    field(HOPR,"80")
    field(LOPR,"10")
    field(HIHI,"0.0e+00")
    field(LOLO,"0.0e+00")
    field(HIGH,"0.0e+00")
    field(LOW,"0.0e+00")
    field(HHSV,"NO_ALARM")
    field(LLSV,"NO_ALARM")
    field(HSV,"NO_ALARM")
    field(LSV,"NO_ALARM")
    field(HYST,"0.0e+00")
    field(ADEL,"0.0e+00")
    field(MDEL,"0.0e+00")
    field(SIOL,"")
    field(SIML,"")
    field(SIMS,"NO_ALARM")
    field(IVOA,"Continue normally")
    field(IVOV,"0.0e+00")
}

```

EPICS Databases – What are they?

- **A collection of one or more EPICS *records* of various types**
- **Records can be interconnected and are used as building blocks to create applications**
- **A data file that's loaded into IOC memory at boot time**
- **Channel access talks to the IOC memory copy of the database**

Our First Database

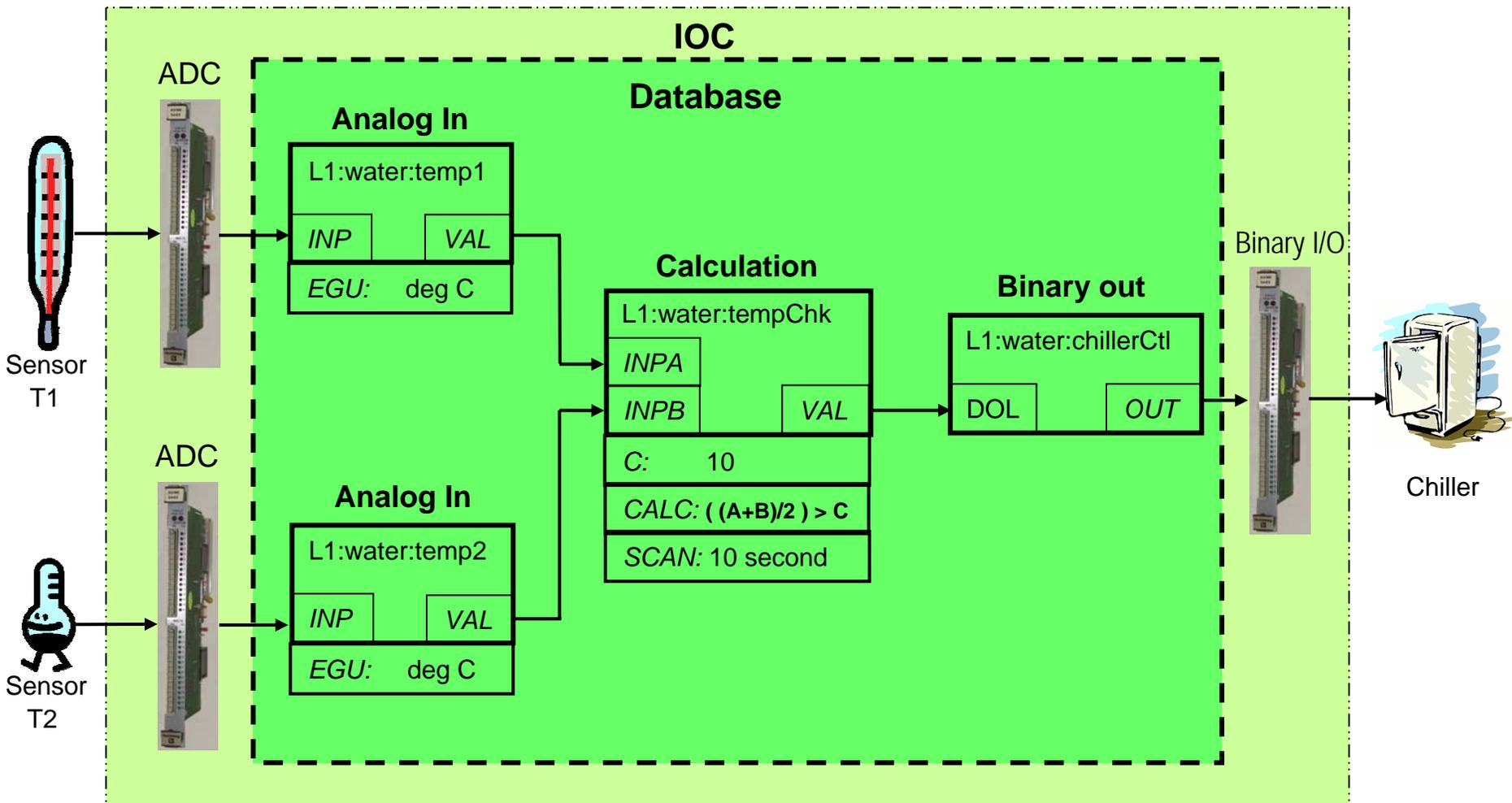


Record Processing

- **Record processing can be periodic or event driven**
- **Periodic: Standard scan rates are...**
 - 10, 5, 2, 1, 0.5, 0.2 and 0.1 seconds
 - Custom scan rates can be configured up to speeds allowed by operating system and hardware
- **Event driven: Events include**
 - Hardware interrupts
 - Request from another record via links
 - EPICS Events
 - Channel Access Puts

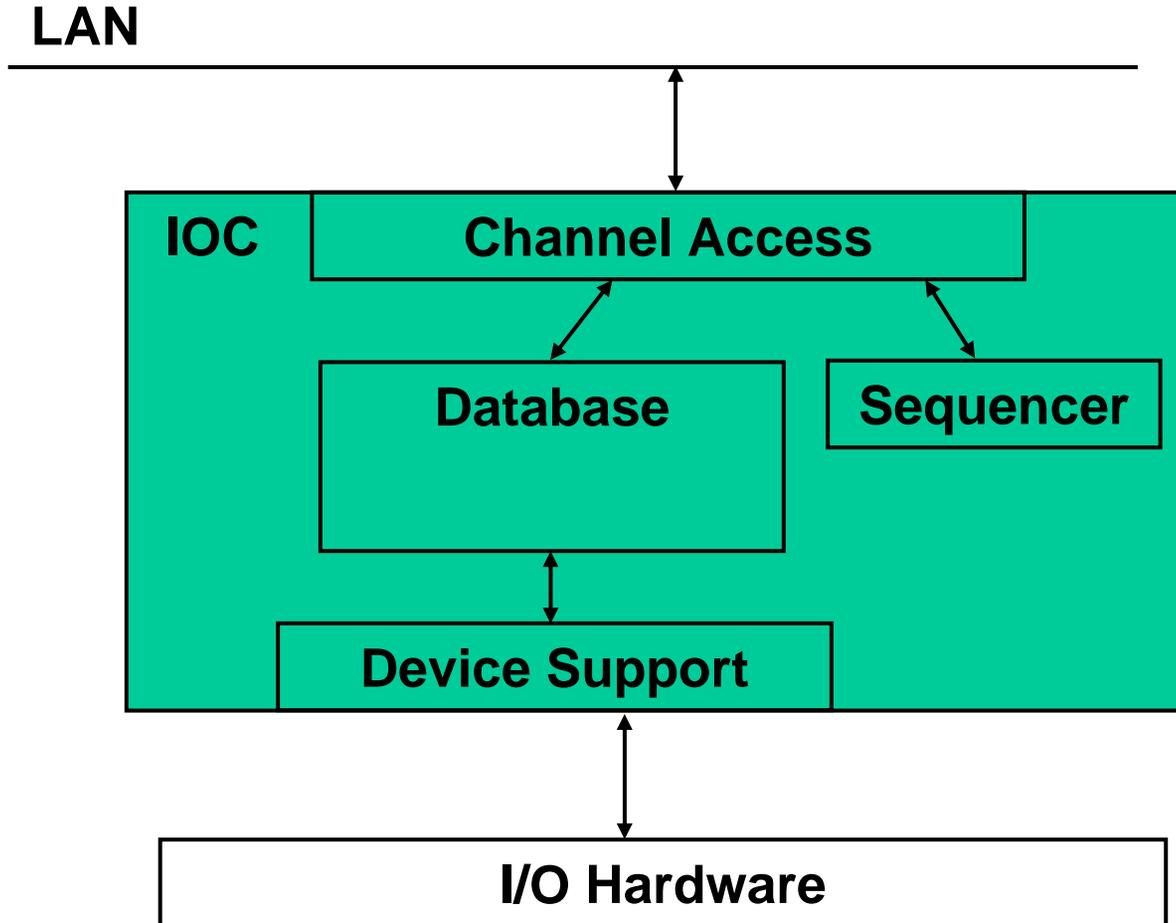


Database Processing



Inside an IOC

The major software components of an IOC (IOC Core)

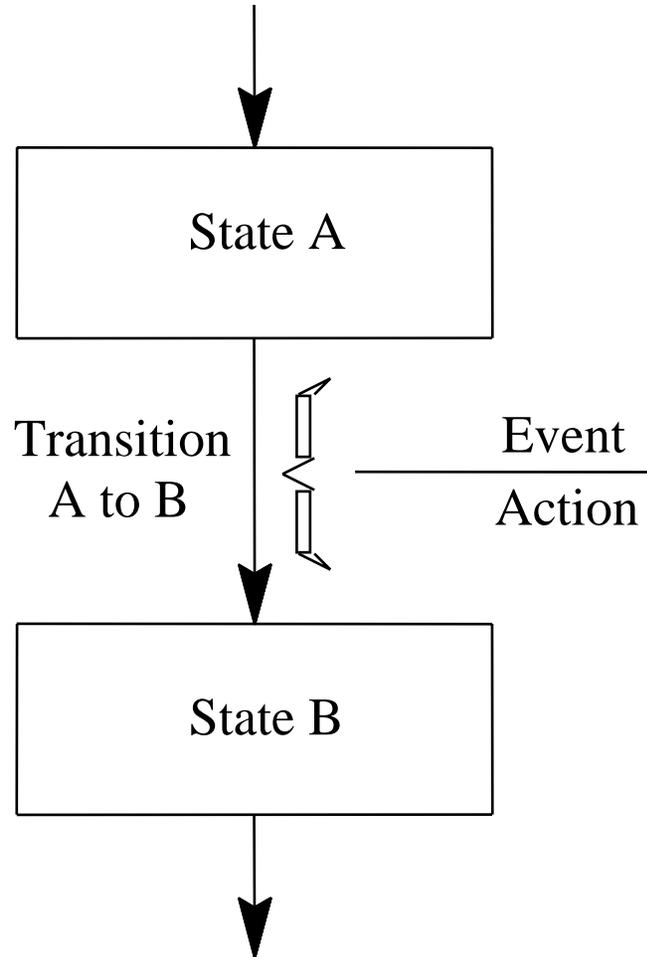


The Sequencer

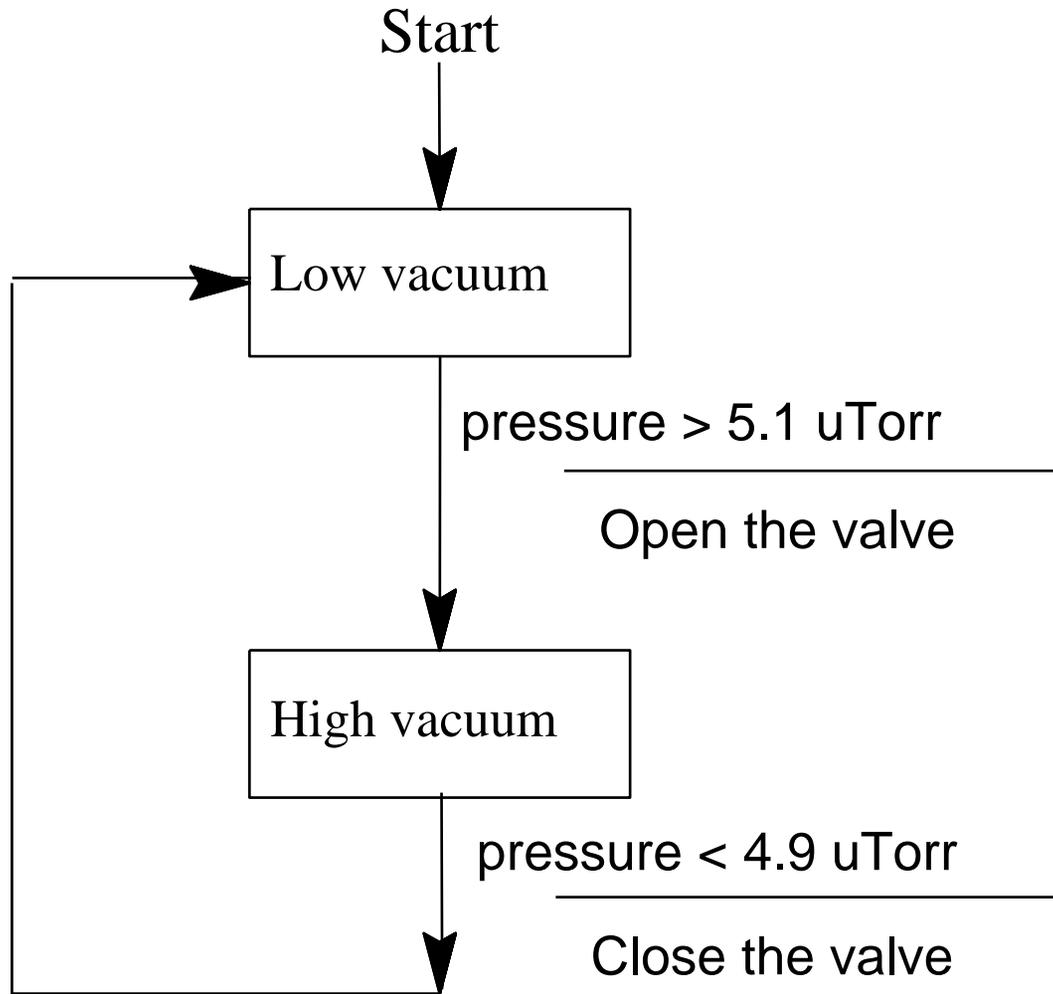
- **Runs programs written in State Notation Language (SNL)**
- **SNL is a 'C' like language to facilitate programming of sequential operations**
- **Fast execution - compiled code**
- **Programming interface to extend EPICS in the real-time environment**
- **Common uses**
 - Provide automated start-up sequences like vacuum or RF where subsystems need coordination
 - Provide fault recovery or transition to a safe state
 - Provide automatic calibration of equipment



SNL implements State Transition Diagrams

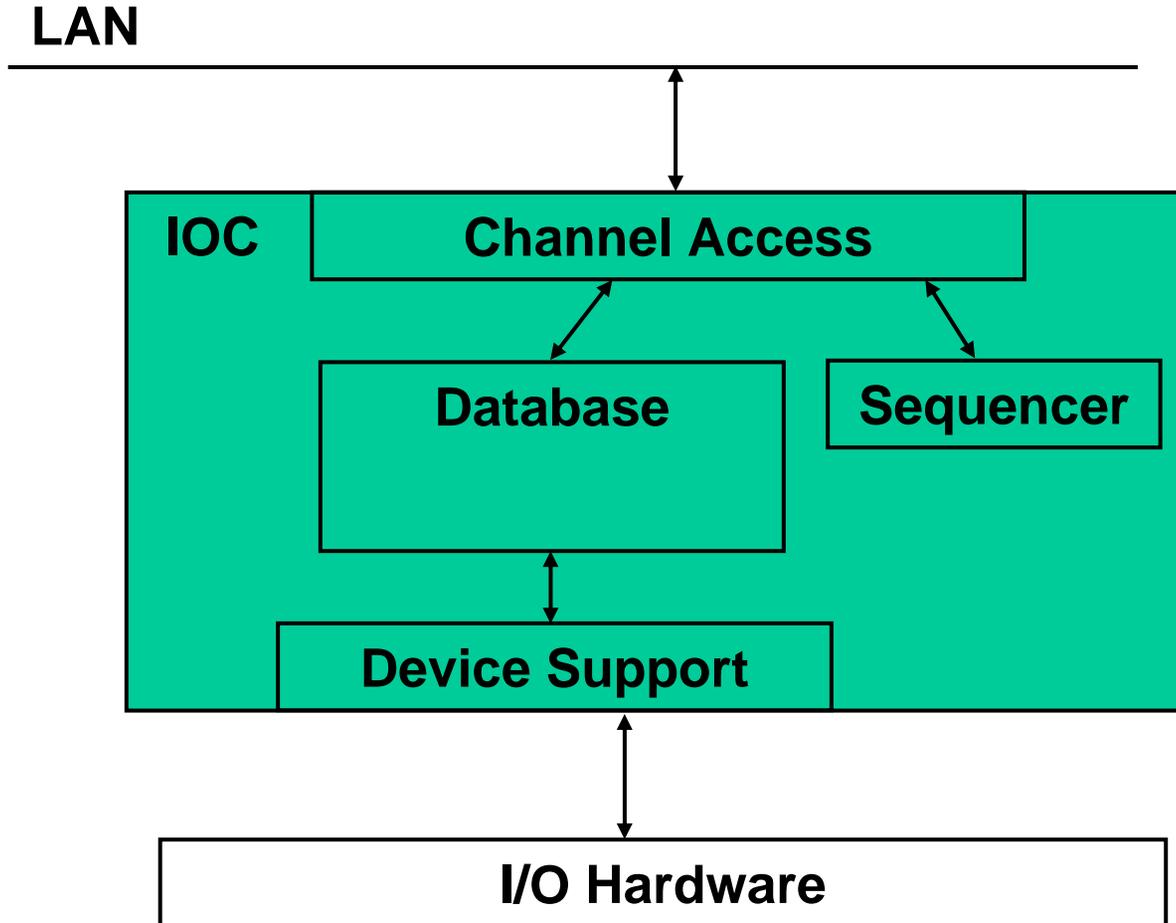


STD Example



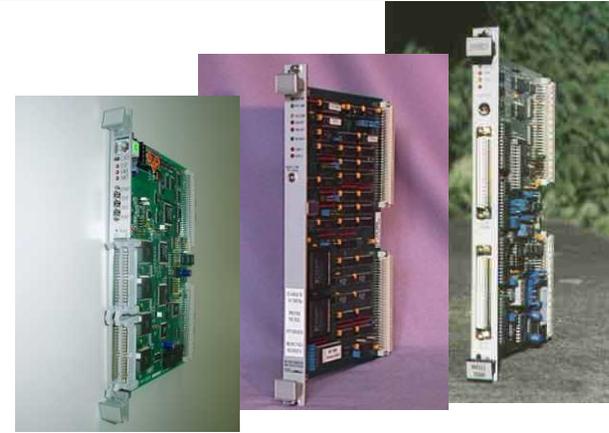
Inside an IOC

The major software components of an IOC (IOC Core)



Device Support

- Device and driver support interface hardware to the database
- Examples of devices....
- VME cards: ADC, DAC, Binary I/O e.t.c.
- Motor controllers
- Oscilloscopes
- PLCs



Device Support

- Usually has to be written for 'new' hardware
- Good news – someone, somewhere has usually written support for your device, or a very similar one before
- See the EPICS web site for available support
- Or ask the EPICS community



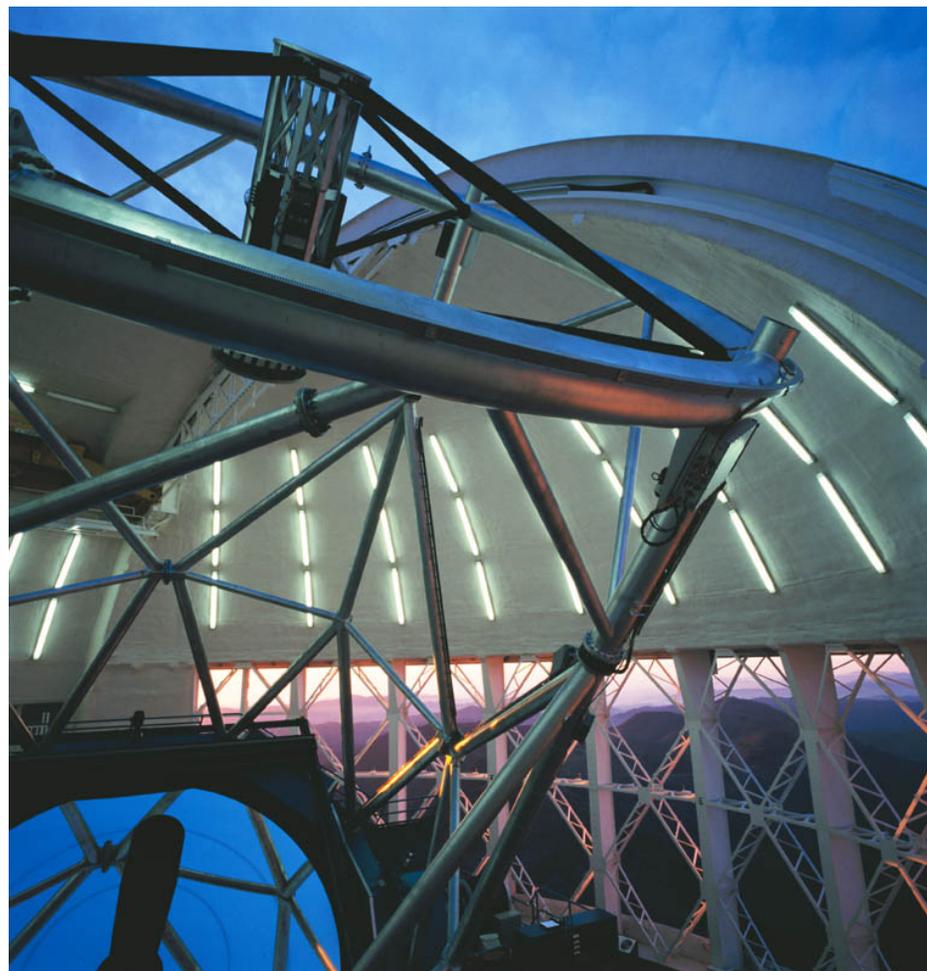
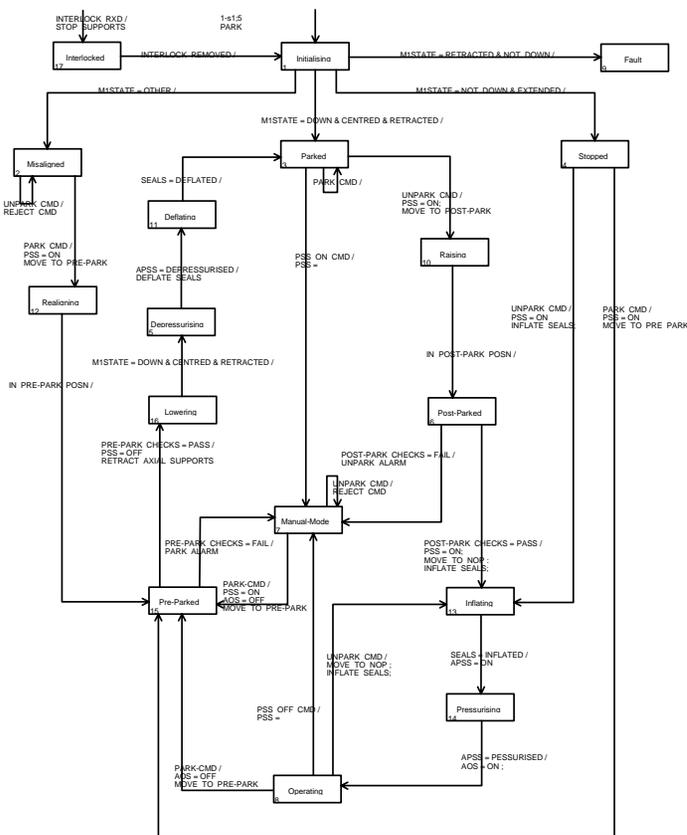
When to use databases

- **Hardware connection**
- **Real time performance – no network latencies**
- **Whenever a database is good enough**

| Advantages | Disadvantages |
|---|-------------------------------------|
| Simplify hardware connection | If you have device support |
| Configuring not programming. | You need to understand database use |
| Database is easily understood by other EPICS developers | |
| Speed - All processing (often) in same machine | |

When to use the sequencer

- For sequencing complex events
- E.g. Parking and unparking a telescope mirror



Photograph courtesy of the Gemini Telescopes project

When to use clients

- To interact with the control system
- Many already exist – MEDM, ALH, Strip Tool, archiver etc.
- For data analysis or visualization
- Supervisory control
- E.g. to manage an accelerator



How fast is EPICS?

- Can be fast or slow, it depends how you use it!
- Use the correct tool for the job; Database, sequencer, custom code (ioc) or custom code (client)
- Ultimately speed depends upon hardware
- Some benchmarks*:

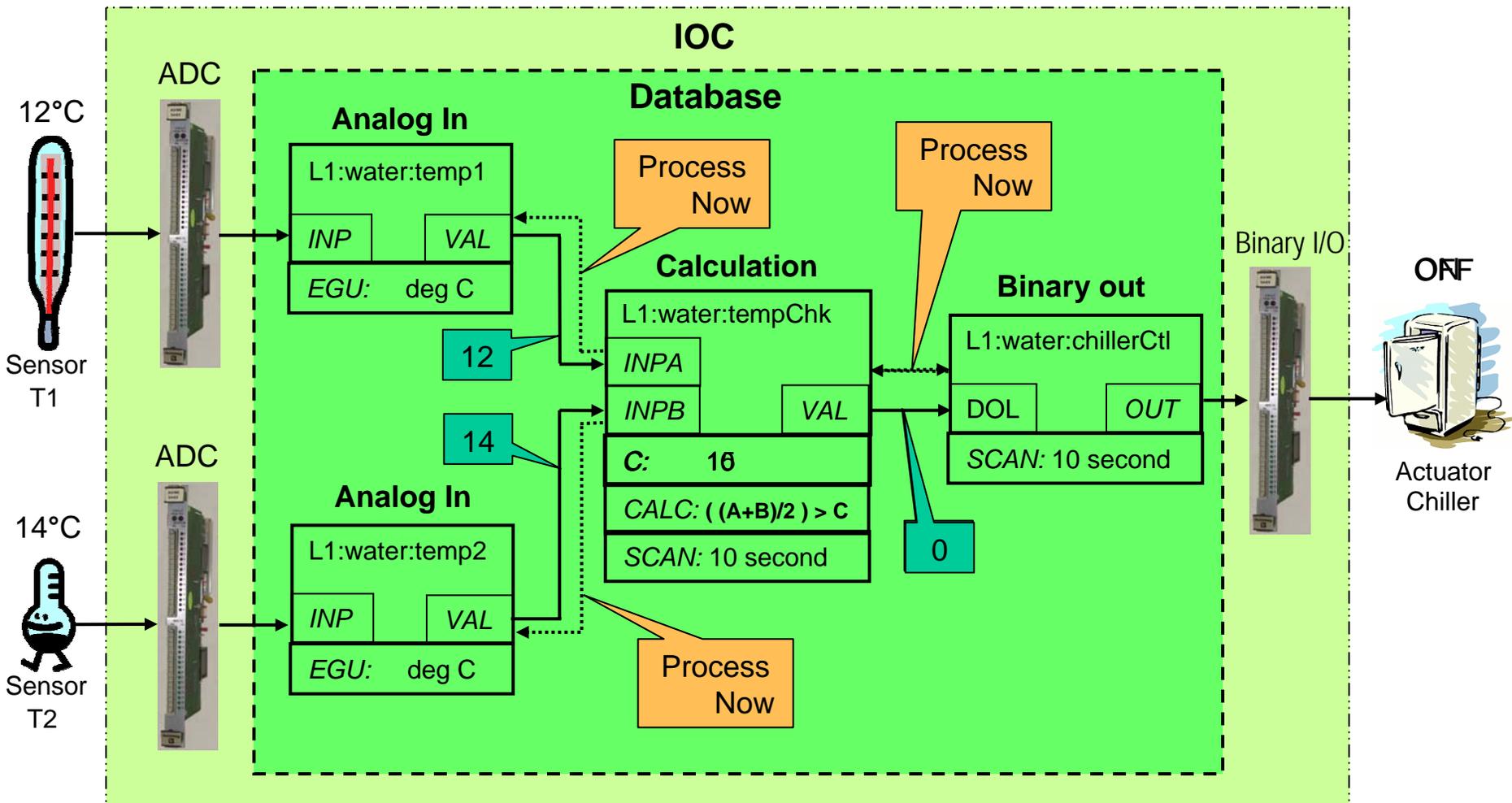
| Machine | OS | CPU | Speed | Rec/sec | %CPU |
|-----------|---------|--------|--------|---------|------|
| MVME167 | vxWorks | 68040 | 33MHz | 6000 | 50 |
| MVME 2306 | vxWorks | PPC604 | 300MHz | 10000 | 10 |
| MVME5100 | vxWorks | PPC750 | 450MHz | 40000** | 10** |
| PC | Linux | PII | 233MHz | 10000 | 27 |
| PC | Linux | P4 | 2.4GHz | 50000 | 9 |

*Benchmark figures courtesy of Steve Hunt (PSI)

**Extrapolated from performance figures provided by L.Hoff, BNL

- Database design and periodic scanning effect *apparent* system speed

Apparent performance

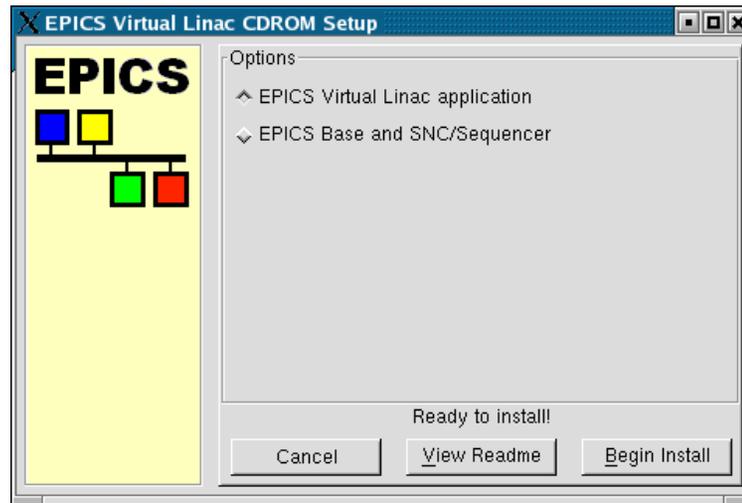


The EPICS web site

- **The central site for EPICS information**
- **Documentation**
- **CA Clients**
- **Device support**
- **Tech-talk**
- <http://www.aps.anl.gov/epics>

Installing the virtual LINAC

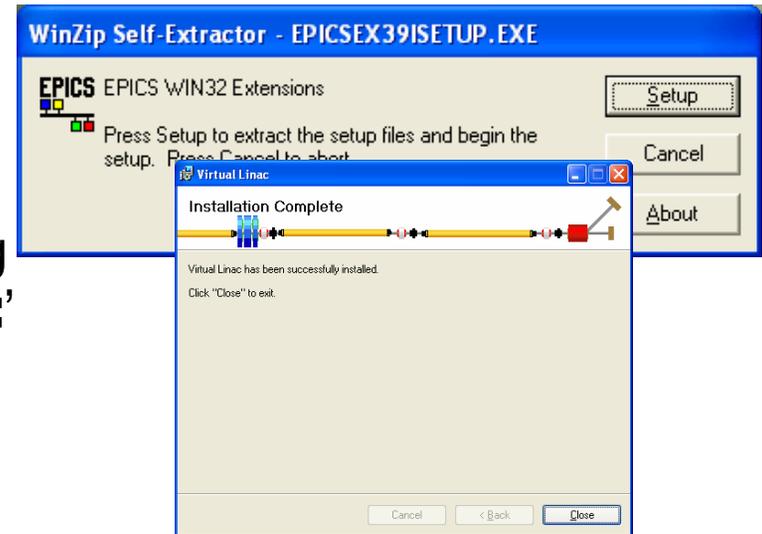
- **Linux, Solaris, Mac OSX**
 - Insert the CD
 - Mount the CD (if it's not automounted)
 - Run the setup.sh script from the CD



- 'cd' to your install directory
- Run 'start.sh' to start both medm and the virtual LINAC
- Or start things individually using the scripts provided

Installing the virtual LINAC - Windows

- **Note: You will need Exceed V7.0 or later installed**
 - Insert the CD
 - If autorun is enabled you will see a screen with instructions, if not open 'WIN32/README.HTM' (on the CD) in your browser
 - Install the Extensions by running 'WIN32/EPICSEX39ISETUP.EXE' from the CD
 - Install the Virtual LINAC by executing 'WIN32/VIRTUALLINACSETUP.EXE' from the CD
 - Program icons will appear on the desktop and start menu
 - Run 'MEDM Virtual Linac' to start the MEDM screen
 - Run 'start Virtual Linac' to start the virtual LINAC IOC



If You Don't Have the CD

- The CD image and individual OS versions can be obtained from
<http://www.aps.anl.gov/epics/download/examples/index.php>
- Remember, the CD image is an *image* file. You may need to use a command such as “*Create CD from image file*” on your Windows CD creation program

Review

- **Input Output Controllers are a fundamental part of an EPICS control system**
- **The database is the primary means of telling an IOC what to do**
- **An EPICS database is composed of records configured to perform an application**
- **Channel Access is a means for other computers to communicate with record fields**
- **Sequencer programs can be used to sequence complex operations**
- **Device support software allows records to interact with hardware inputs and outputs**
- **EPICS is fast and efficient but can appear slow if used without consideration**

Acknowledgements

- **Andrew Johnson (APS-Controls)**
- **Bob Dalesio (LANL)**
- **Deb Kerstiens (LANL)**
- **Rozelle Wright (LANL)**