synApps 5.8

Tim Mooney
1/15/2015

Argonne National Laboratory

A U.S. Department of Energy
Office of Science Laboratory
Operated by The University of Chicago
What is synApps?

- A collection of EPICS *modules* for synchrotron-beamline users
  - Web page: 
  - Subversion repository: 
    - https://subversion.xray.aps.anl.gov/synApps

- EPICS modules:
  - alive, autosave, busy, calc, camac, caputRecorder, dac128V, delaygen, dxp, ip, ip330, ipUnidig, love, mca, measComp, modbus, motor, optics, quadEM, softGlue, sscan, std, vac, vme, xxx

- Support directories:
  - documentation, configure, utils
synApps modules

- Modules contain the following kinds of support:
  - Compiled code; libraries, for examples:
    - record and device support
    - SNL (State Notation Language) programs
    - string-calc, array-calc engines
  - EPICS databases and autosave-request files
    - A database is a program written in a high-level language.
    - One or more copies of a database can be run, each with its own private process variables (PV’s).
    - The database designer recommends PV’s to be autosaved by naming them in a .req file.
  - Display files for MEDM, caQtDM, and CSS-BOY
    - The default user interface
  - Documentation
  - Some modules contain iocBoot directories, mostly for testing.
Other EPICS modules included with synApps

- areaDetector (ADCore and ADBinaries) (Mark Rivers)
- Asyn (Mark Rivers, Eric Norum)
- Ipac (Andrew Johnson)
- Seq (Ben Franksen)
- Stream (Dirk Zimoch)
- devlocStats (Stephanie Allison)
- allenBradley (Marty Kraimer)
alive

• Sends operational status of an IOC to a server.
  - Environment variables
  - vxWorks boot parameters
  - Boot time

• Collected information is available at
  [http://bcda.xray.aps.anl.gov/cgi-bin/ioc_alive.cgi](http://bcda.xray.aps.anl.gov/cgi-bin/ioc_alive.cgi)
autosave

• Records latest values of selected EPICS PVs; restores those values when the ioc restarts.
  - Not an archiver; only the latest value is saved
  - Don’t confuse this with saveData, which writes scan data
  - When a list list of PV’s is saved, the entire list is written, even if only one PV has changed.

• Can save/restore any scalar or array-valued PV (synApps 5.1)
  - Array-valued PV must be hosted by the ioc that does the restore operation. (Normally, ioc’s save/restore only their own PV’s.)
  - DBF_MENU, DBF_ENUM PV’s are handled by number.

• Save operation uses channel access for scalars.
• Restore operation uses static database access for scalars.
• Arrays are saved and restored with database access.
Three restore options for save files:

1) before record/device initialization
   - Motor positions must be restored at this time.
   - Arrays cannot be restored at this time. *
   - PV’s that are DBF_NOACCESS before record init (e.g., genSub variable-type fields) cannot be restored at this time. *

2) after record/device initialization
   - to override record-initialization values
   - Link fields cannot be restored at this time. *

3) both before and after record initialization
   - The ‘auto_settings.sav’ file is restored at both times.
   - It’s not an error to attempt to restore a PV at the wrong time.
   - If you restore a motor position at this time, you override the value read from hardware, without writing to hardware.

* Not illegal, just doesn’t work
...autosave

- PV lists can use include files (e.g., `<database_name>.req`), include path.
  - Database developer can supply default include file with database.
  - User can override with custom include file.
- Save triggers:
  - on change of any PV in the list
  - periodically
  - on change of a trigger PV
  - manual
- User can reload save sets.
- Autosave can recover from file-server reboot.
  - Currently, only on vxWorks
- User can choose to save redundant files.
- Autosave reports status via EPICS PV’s.
...autosave

- Sample request file

```plaintext
xxx:my_PV.VAL
xxx:my_array_PV.VAL
file motor_settings.req
P=\$(P), M=m1
...<END><lf>
```

- Sample save file

```plaintext
# save/restore V4.4 Automatically generated...
xxx:my_PV.VAL 1.0
xxx:my_array_PV.VAL @array@ { "0" "0.1" ... "10.2" }
xxx:m1.DIR 0
xxx:m1.DHLM 100
xxx:m1.DLLM -100
...
```
busy

• Allows channel-access clients and asyn drivers to participate in EPICS’ completion reporting

• Records
  - busy – like bo record, but forward link is executed only if VAL==0.
  - Example of use:
Support for **evaluating expressions** entered at run time

**•Records**

- **acalcout** – like **calcout**, but also supports array expressions; user can specify wait-for-completion.

- **scalcout** – like **calcout**, but also supports string expressions; user can specify wait-for-completion.

- **swait** – like **calcout**, but uses recDynLink (no “PP MS” link attributes), and waits for completion.

- **transform** – like 16 **calcout** records that share a PV data pool

- **sseq** – like **seq**, in base, but can get and put strings; user can specify wait-for-completion.

**•Other code**

- interpolation routines for **aSub** record

- averaging routines for **sub** record

- **sseq**-record editor
...calc

- Databases, display files for run-time programming
  - userCalc, userCalc2out
  - userStringCalc
  - userArrayCalc
  - userTransform
  - userStringSeq
  - userAve
  - interpolation

- Examples of ALL calc expressions can be found in synApps MEDM help displays
camac

• Communication with CAMAC crate/modules

• Records
  - camac – generic BCNAF/data for run-time camac control

• Devices supported
  - VME bus adapter
  - CAMAC crate controller
  - E500 motor controller
  - RTC-018 real-time clock
  - QS-450 quad scaler
caputRecorder

- Support for recording and playing back sequences of channel-access puts.
- User interface for executing python functions

```
import time
import epics

def abort(callback):
    print('abort() called')

def doScan():
    epics.caput('xpp1:madcam', 1, wait=True, timeout=300)
    epics.caput('xpp2:madcam', 2, wait=True, timeout=300)
    epics.caput('xpp3:madcam', 3, wait=True, timeout=300)
    epics.caput('xpp4:madcam', 4, wait=True, timeout=300)
    epics.caput('xpp1:madcam', 1, wait=True, timeout=300)
    epics.caput('xpp2:madcam', 2, wait=True, timeout=300)
    epics.caput('xpp3:madcam', 3, wait=True, timeout=300)
    epics.caput('xpp4:madcam', 4, wait=True, timeout=300)
```

```
# Example usage

caputRecorder = caputRecorder('xF4:caputRec', start=False)
caputRecorder.doScan()  # Record a sequence
```

configure

- Configures all modules in or used by synApps

- RELEASE
  - specifies version number and file path to EPICS base, and to every module synApps/support

- makeReleaseConsistent.pl
  - Edits <module>/configure/RELEASE for every module in synApps/support, to agree with synApps/support/configure/RELEASE
  - “make release” in synApps/support causes this to run.
dac128V

- device support, database, and display files for dac128V IndustryPack module
  - 8-channel, 12-bit DAC
  - Support exists to run a DAC channel manually, or according to an algorithm written at run time, or as a scan positioner, or as part of a PID feedback loop at up to \(~10\) Hz, or as part of a fast PID feedback loop at up to \(~10\) kHz.
delaygen

- device support, database, and display files for delay generators
  - Colby Instruments PDL100A programmable delay line
  - Coherent synchronization delay generator
  - Berkeley Nucleonics 505 pulse/delay generator
documentation

• **TOP-level synApps documentation**
  - What synApps is
  - How to deploy it
  - How to build it
  - How to make a user application from the ‘xxx’ sample module
  - How to fit the user application to a particular set of hardware

• This presentation
dxp

- record, device support, databases, and display files for XIA DXP and Saturn spectroscopy systems
- dxp record for setting DXP parameters
- device support for the mca record
ip

- device support, SNL code, databases, and display files for many message-based devices
  - originally, for devices supported via IndustryPack hardware
  - Note some of this support will inevitably be out of date, or pending access to hardware for testing.

- deviceCmdReply
  - Used to write support at run time for one command/reply message
  - scalcout to format output string
  - asyn record to write/read device
  - scalcout record to parse reply

- devXxStrParm device support
  - New support should use stream/asyn
ip330

- device support, databases, and display files for the IP330 ADC IndustryPack module

- 16/32 channel, 16-bit ADC
  - ip330Scan for periodic, averaged reads of ADC channels
  - ip330Sweep, with the MCA record, for using ip330 as a waveform-digitizer, or to record analog data during a fly scan
  - ip330PID for using the ip330 in a fast-feedback loop
ipUnidig

- device support, databases, and display files for the IPUnidig digital I/O IndustryPack module

- IP-UD-I 24-channel input/output/interrupt module

- DIO316I 48-bit digital I/O module
• **Support for Love controllers**
  - Uses RS485 Octal Serial IndustryPack module, or RS232 module with 232/485 converter
  - Asyn device support for the ao, ai, bo, bi, and mbbi records
  - Asyn Interpose interface for Lovelink
  - Sample application, database, and display files
    - *vxWorks and Linux only*
  - Startup scripts for vxWorks and Linux are provided to configure Ipac, Asyn, and the Interpose interface
mca

- Support for **multichannel analyzers, multichannel scalers, and other array-valued detectors**
- mca record
- device support
  - Canberra 556 AIM module (MCA and ICB controller)
  - DSA-2000 Ethernet MCA
  - various Canberra-ICB modules for spectroscopy
  - SIS 3801 (Struck STR7201) and 3820 multichannel scalers
  - (DXP support in dxp module)
  - (IP330 support in ip330 module)
  - (quadEM support in quadEM module)
measComp

- Support for *Measurement Computing USB devices*
  - USB-CTR08 8-channel counter/timer module
  - USB-1608GX-2AO analog I/O module
  - USB-4303 (obsolete) counter/timer module
modbus

- Support for Modbus Protocol
  - Programmable Logic Controllers (PLCs)
  - Other modbus devices, such as temperature controllers
- Uses **asyn** to support various communication links
  - TCP/IP
  - Serial RTU
  - Serial ASCII
Motor record and device support
- stepper and servo motors
- soft-motor support
  - *Put motor “face” on, e.g., a DAC channel*
  - *Drive a hard motor through a nonlinear transform*
- user/dial/raw coordinates
- backlash-takeout algorithm
- pre/post move commands
- many more features
...motor

- Supported motor types
  - Oregon Micro Systems, Inc. (OMS) models; VME8, VME44, VME58, VS4, VX2, MAXv, PC68 and PC78.
  - Newport models MM3000, MM4000/5/6, PM500, ESP300/301/100 and XPSC8.
  - Schneider Electric (formally IMS) models IM483, MDrive and MForce.
  - Advanced Control Systems, Corp. model MCB-4B.
  - McLennan models PM304 and PM600.
  - Physik Instrumente (PI) GmbH & Co. model C-630, C-844, C-848, C-862/863, E-662, E-710 and E-816.
  - MicroMo model MVP 2001 B02.
  - Micos model MoCo dc controller, SMC hydra controller.
  - Delta Tau PMAC2-VME controller.
  - Faulhaber MCDC2805 servo controller.
  - Parker Hannifin, Compumotor Division, 6K Series controllers.
  - New Focus, models; 8750 and 8752.
  - ACS Motion Control, SPiiPlus model.
  - Spectra-Physics, Encoder Mike Controller, Model 18011.
  - Thorlabs, Piezo Controller, Model MDT695.
  - Animatics Corporation SmartMotor.
  - piezosystem jena GmbH EDS data interface module.
  - Kohzu SC-200, SC-400, SC-800 stepper motor controllers.
  - attocube systems AG ANC150 Piezo Step Controller.
  - Aerotech Soloist single-axis servo controller, Ensemble multi-axis digital servo controller.
  - Hytec 8601 Industry Pack based 4-channel Stepper Motor Controller.
options

- **Slits and mirrors**
  - *Four virtual positioners; two real motors*
  - *Automatic sync to motor positions*
  - *Completion reporting*

- **Monochromators**
  - *Nondispersive double-crystal*
    - Geometries: \((Y1, Z2), (Y2, Z2)\)
    - Crystal species: Si, Ge, Diamond, Si (77K)
    - Miller indices, allowed reflections
    - Operational modes:
      - *Use/Set*
      - *Manual/Auto*
    - Managing the vertical beam offset
    - Automatic sync to motor positions
...optics

- **Monochromators**
  - Spherical grating
    - Geometrical variables:
      - 1) Grating line density; radius
      - 2) Tangent-arm length
      - 3) Diffraction order
      - 4) Input/output slit distances
    - Operational modes:
      - Use/Set
      - Manual/Auto
    - Grating-stripe list
    - Manual sync to motor positions
...optics

**...Monochromators**

- *Dispersive double-crystal*
  - Geometries: nested, symmetric
  - Crystal species: Si, Ge, Diamond, Si (77K)
  - Miller indices, allowed reflections
  - Operational modes:
    - *Use/Set*
    - *Manual/Auto*
    - *Theta1 / Theta1&2 / Rock Theta2*
  - Accommodate incident-beam angle shift (“world offset”)
  - Automatic sync to motor positions
...optics

- **Optical table**
  - *Table* record supports a six-degree-of-freedom optical table.
  - Four geometries: SRI, GEOCARS, NEWPORT, and PNC
  - User/client can write either to \((x, y, z, \theta_X, \theta_Y, \theta_Z)\), or to underlying motor records.
  - Table rotates about user-specified point.
  - Table database includes a list of rotation points, selected by menu.
  - Can recover table position from motor positions
  - Partial support for fewer than six degrees of freedom
...optics

- Optical table
  - Geometries
    - SRI
    - GeoCARS
    - Newport
    - PNC
  
  - Calibration-sync
    - Use/Set – changes to [X, Y, ..]. move table / change calibration
    - Zero – redefine current [X, Y, …] as zero
    - Sync – update [X, Y, …] from motors, honoring calibration
    - Init – clear calibration and sync to motors

- Table record sets motor speeds so that motors start/stop together.
quadEM

• Support for **four-input electrometers**.
  - APS quad electrometer
softGlue

- Support for FPGA-based digital electronics.
**sscan**

- **Support for user-programmable data-acquisition**
  - *sscan* record executes scans
  - *saveData* saved data to a file
  - *recDynLink* implements *sscanRecord* links

- **A one-dimensional scan:**
  - Do NPTS times:
    - Set conditions  
      e.g., move motors; wait for completion
    - Trigger detectors  
      e.g., start scaler; wait for completion
    - Acquire data  
      read detector signals; store in arrays
  - Write data to NFS file

- **Multidimensional scan:**
  - Same as a 1-D, but detector trigger executes inner-loop scan.
  - *saveData* monitors a set of *sscan* records, determines scan dimension when scan starts, and writes data as it is acquired.
...sscan

- scan features:
  - Three 1-D scan types: constant-step-size, table-driven, fly
  - Unlimited number of data points, scan dimensions
  - 0-4 positioners, 0-4 detector triggers, 0-70 detector signals
  - Acquisition from scalar and 1-D-array-valued PV's
  - Detector/client wait, data-storage wait
  - Pause/resume, abort
  - Double buffered: can write 1-D acquired data during next 1-D scan
  - `saveData` writes self-describing XDR-format (".mda") files to NFS-mounted disk (vxWorks only, at present).
  - A positioner can have private scan parameters (scanparm record).
  - After-scan actions include move to peak, valley, and edge.
  - scanparm record + after-scan action = automated 1-D alignment, so you can easily implement an “Align” button.
...sscan

- The sscan record
  - performs 1-D scan
  - before-scan link – optional completion callback
  - positioner: any writable, numeric, scalar PV (menus, enums are ok)
  - detector trigger: any writable, numeric, scalar PV
  - detector signal: any readable, numeric, scalar or 1D array PV
  - array detectors: exactly <scanRecord>.NPTS elements are acquired
  - array trigger: callback indicates array data are ready to read
  - after-scan link – optional completion callback
  - pause/resume
  - abort (<scanRecord>.EXSC -> 0) wait for callbacks, cleanup
  - kill (two aborts in a row) abandon callbacks
  - handshake with multiple display / data-acquisition clients
  - handshake with data-storage client
Other data-acquisition-related software

- Data-visualization tools for use with the sscan record
  - scanSee (*Ben-chin Cha, John Hammonds*)
  - dview, sview (*Dohn Arms*)
  - utils/mdautils (*Dohn Arms*)
  - utils/mdaExplorer (*Tim Mooney*)
  - utils/mdaPythonUtils (*Tim Mooney*)
std

- Epid record
  - Extended PID record
- Scaler record
  - Controls a set of counters with a common clock, gate, and trigger
- Soft-motor database
  - Run-time programmable soft-motor/transform/hard-motor database
  - Quick solution for driving a motor through a nonlinear transform
- 4-step database
  - Up to four steps of (set condition; read data) with an end calculation
  - Originally developed for dichroism experiments
- PID control
- Femto current-amplifier support
- genTweak, ramp_tweak
- selector
- pvHistory
- Alarm clock, countdown timer
**utils**

- **changePrefix**
  - Change EPICS prefix throughout a copy of the xxx module
- **changePrefixLoc**
  - Change EPICS prefix for all files in an ioc directory
- **copyAdl**
  - Find MEDM-display files; copy to specified directory.
- **mdaExplorer, mdaPythonUtils, mdautils-src**
  - Display, manipulate scan-data (MDA) files
- **snapDb**
  - “Freeze” a collection of userCalcs into a custom database, display
- **Subversion utilities:**
  - logModuleFromTag, releaseNotesFromTag, makeTar
• Support for vacuum measurement and control
  - vs, digital records
  - Granville-Phillips GP307, GP350
  - Televac MM200, CC10
  - Digitel 500/1500
  - Gamma MPC, MPCe, LPC, SPC
### VME

- **VME record**
  - Provides run-time access to VME bus
  - Great for testing hardware
  - Run-time programmed control of an unsupported VME board

- **Device support for VME hardware**
  - Joerger scaler
  - APS bunch-clock generator
  - APS machine-status interface
  - Heidenhain encoder interpolator
  - Generic A32 VME interface
  - HP Laser interferometer
  - VMI4116 16-bit DAC
  - Acromag 9440 16-bit digital input
• Prototype user directory
  - Builds everything in synApps into a load module
  - Contains command files to load/configure ~everything in synApps
  - Contains sample top-level display files
  - Contains sample script to set environment variables and start up the sample user interface
  - Contains table of recommended address/interrupt configuration for selected VME and IndustryPack hardware

• Two ways to use this module
  1) Make copies; run changePrefix; build; customize; run a beamline
     - *this is the recommended use*
     - *detailed instructions in support/documentation*
  2) Reference/grab bag
For developers: features of synApps

- **extended-processing records**
  - records that are neither synchronous nor asynchronous, as these terms are described in the *EPICS Application Developer’s Guide*

- **completion reporting**
  - All databases behave correctly when written to by `ca_put_callback()`.

- **recDynLink links**
  - Similar to standard EPICS links, but no “PP NMS” attributes

- **GUI standards**
  - Default colors for menus, PV values, links, etc.

- **coordinated motions**
  - Many of the databases in synApps (especially in ‘optics’) involve coordinated motion of several motors.

- **initialization of complex databases**
  - Some common EPICS initialization problems are handled in various synApps databases.
Coordinated motions

• Simple cases: database (transform records)
  - Slits, mirrors, spherical-grating monochromator

• More complicated cases: SNL code
  - Multiple-crystal monochromators

• Very complicated cases: custom record
  - Optical table, scan

• Criteria a useful coordination should meet:
  - Report completion to ca_put_callback()
  - Share control of base positioners with CA clients
  - Recover state from the states of base positioners
Completion reporting

- **Simple prescription for databases contained within a single IOC:**
  - Use only PP links and forward links in execution chain.

- **Database operations spanning more than one IOC:**
  - Use records with put_callback links to span IOCs:
    - `calcout` with asynchronous device support
    - `sscan`, `swait`
    - `sseq` or `sCalcout` (*with .WAIT* = “Wait”)

- **Cases in which a CA client performs part of the operation:**
  1) Database sets a `busy` record via PP or put_callback link.
  2) CA client clears the `busy` record when operation is done.

- **Cases in which part of the operation is driven by a CP link:**
  - Not different from above; a CP link is a CA client.
Initialization of complex databases

- **Initial values: .VAL vs. .DOL**
  - Most records allow .VAL field to be set in the database.
  - Note that .DOL cannot be used for constant strings.

- **Save-restore and interaction with record/device initialization**
  1) save-restore pass 0
  2) record/device initialization → *device support can use pass-0 value*
  3) save-restore pass 1 → *pass-1 overrides record/device-init value*

- **.PINI (Process at INIt) uses and limitations**
  - This is the normal mechanism for database initialization.
  - What if you need a value from some other .PINI-initialized record, and that record hasn’t processed yet?
  - Note .PHAS is not considered in .PINI processing.
Initialization of complex databases

- **Contending with link alarms**
  - If you have an input link to a record with .UDF=1, you get a link alarm.
  - .UDF=1 until a record processes. (In 3.14.1+, database can specify .UDF)
  - The transform record can abort execution on a link alarm (or not).

- **Initialization problems with CP links**
  - You have a CP link to a field that is a calculation result.
  - If the calc result is the same as the field’s initial value, you’ll have the right value, but you won’t know that you have the right value, and you won’t know for how long to wait to be sure.
  - The transform record always posts its initial calculation result.

- **Programmatically initializing link fields**
  - Link field must be written with a CA link (because lock-set recalc).
  - .PINI processing occurs before CA is running (EPICS 3.13.5+).
  - Can’t use .PINI; Drive init from a scan task; set init record to “Passive” when init is done.
synApps in use at APS

• Deployed/released on APSshare
  - New minor releases (e.g., what would be synApps 5.7.x, if minor releases were numbered) are done by adding new versions of modules to the synApps support directory.
  - Beamline iocs select module versions with their RELEASE files.
    - There are 256 versions of synApps 5.7 installed on APSshare.
    - There are ~130,000 versions of synApps 5.6.
  - A new major release is done when it becomes impractical to upgrade the previous release - for example, because:
    - a new version of EPICS base is needed, or
    - a new non-backward-compatible version of asyn, seq, etc. is needed.