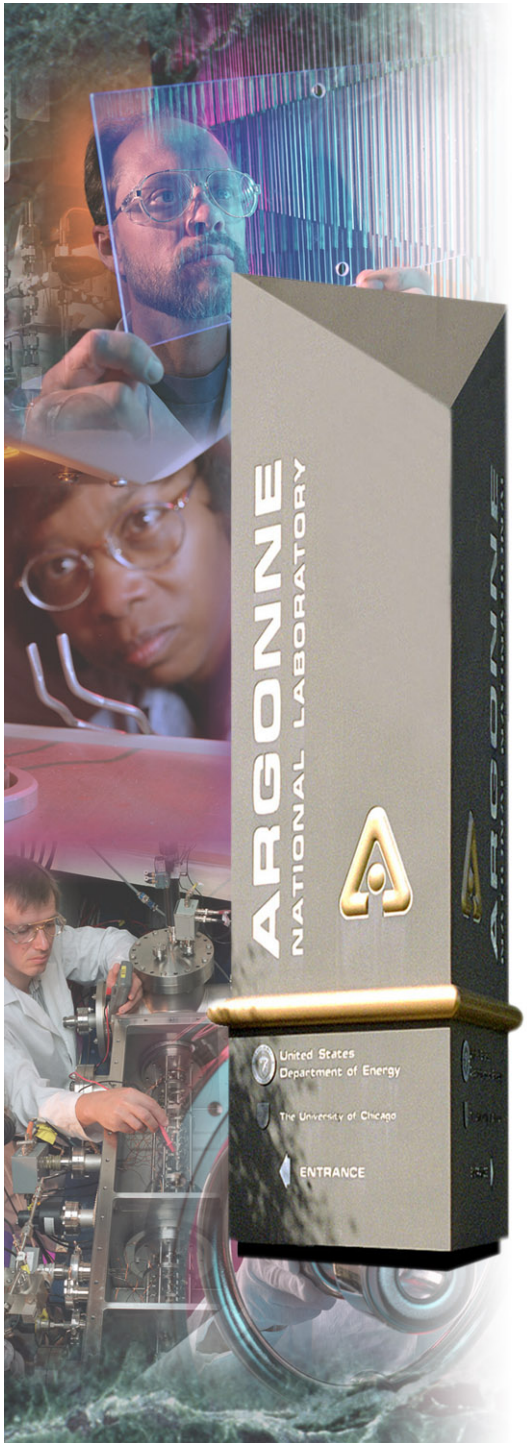


Getting Started with EPICS
Applications / Special Topics

Scans

Tim Mooney
3/15/2005

Argonne National Laboratory

Office of Science
U.S. Department of Energy

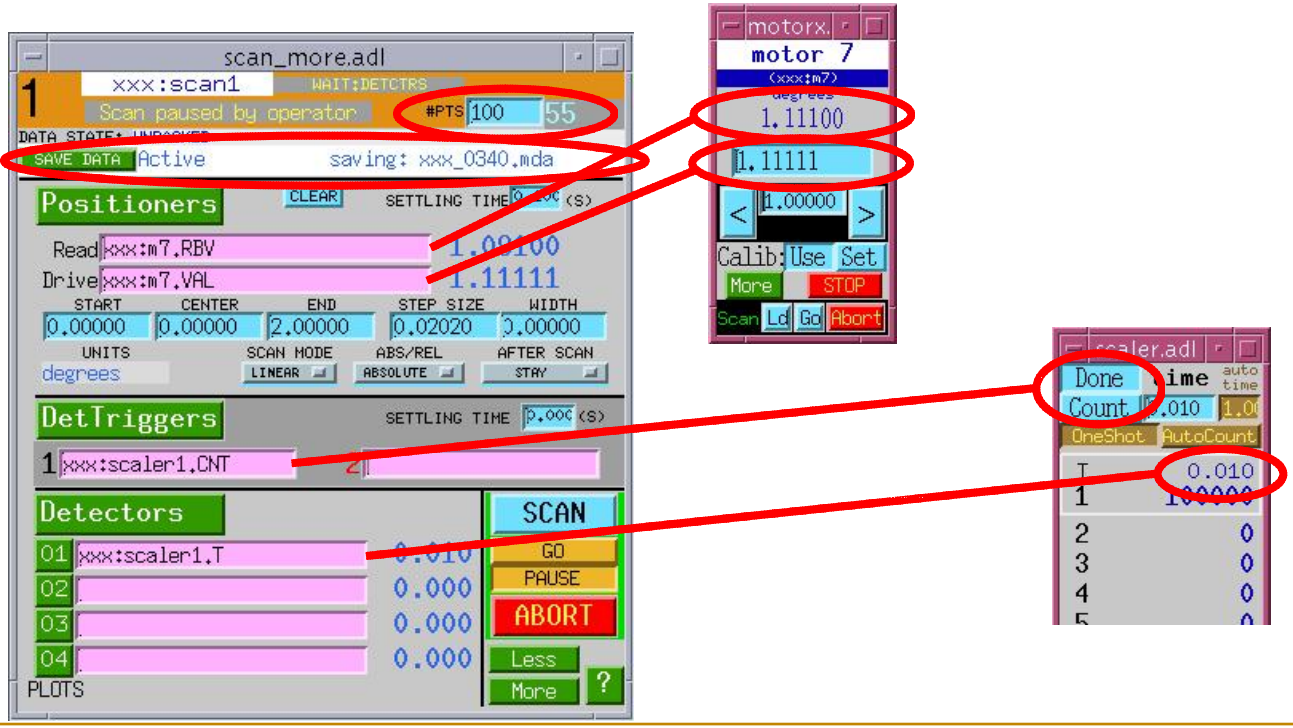
A U.S. Department of Energy
Office of Science Laboratory
Operated by The University of Chicago

The synApps SSCAN module

- **Where is it?**
  - http://www.aps.anl.gov/aod/bcda/synApps/sscan.html
- **What's in it?**
  - Code
    - the sscan record
    - the busy record
    - the recDynLink library
    - the saveData data-storage client
    - the scanparm record
  - EPICS databases
    - scan databases
    - scanParms and alignParms databases
  - MEDM displays
    - scan*.adl
    - scan*_help.adl

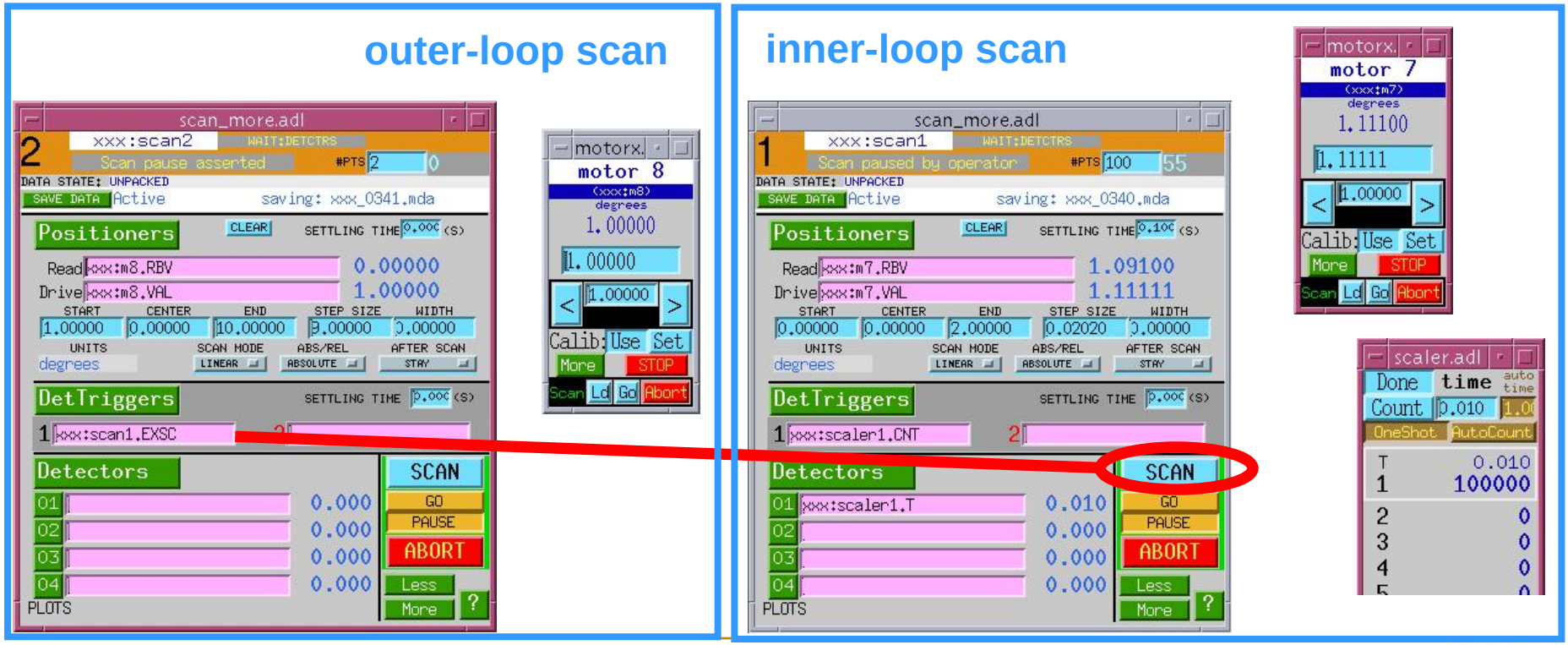
Simple scans

- **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 disk



...Simple scans

- **Multidimensional scan:**
  - Outer-loop scan's *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.
  - No limit to the number of scan dimensions.



Scan features

- **0-4 positioners, 0-4 detector triggers, 0-70 detector signals**
  - Positioner and readback values are of type **`double`**
  - Detector values are of type **`float`**
- **Acquisition from scalar and/or array PV's**
  - Array PV's acquire .NPTS elements
- **Number of data points limited only by IOC memory**
  - Standard max. is 2000 (x_i,y_i) points per scan dimension
  - Can increase to ~ EPICS_CA_MAX_ARRAY_BYTES / 8
- **Detector/client wait, data-storage wait**
  - Can wait for multiple data-acquisition clients
  - Only one data-storage client
- **Pause/resume, abort**
  - Data from aborted scans are written to disk
- **Double buffered: writes 1D acquired data after the scan is finished**
  - Can write during next 1D scan

…Scan features

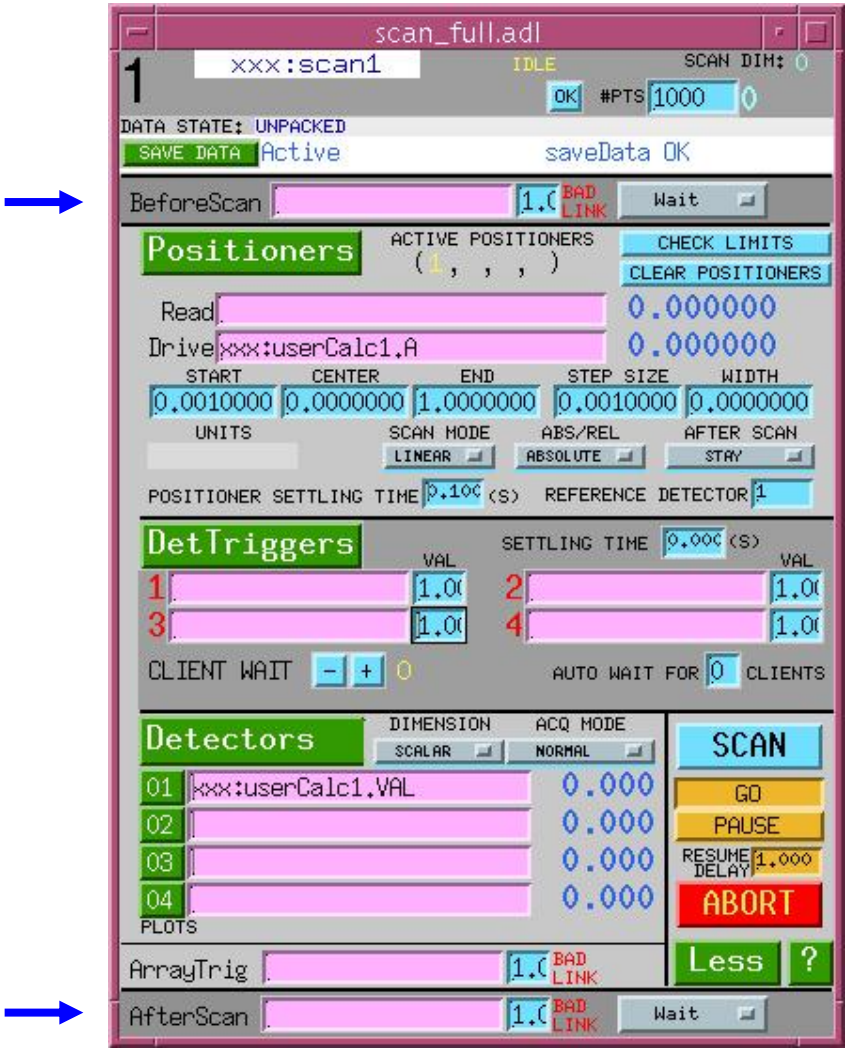
- ***saveData* writes XDR-format (".mda") files to disk.**
  - Files can be read on any type of computer
- **A positioner can have private scan parameters (scanparm record).**
  - Load preset scan parameters with one mouse click
  - Useful for alignment
- **After-scan actions include move to peak, valley, +/-edge.**
  - Can, e.g., track a moving peak through a series of scans
- **scanparm record + after-scan action = automated 1-D alignment.**

Scan implementation

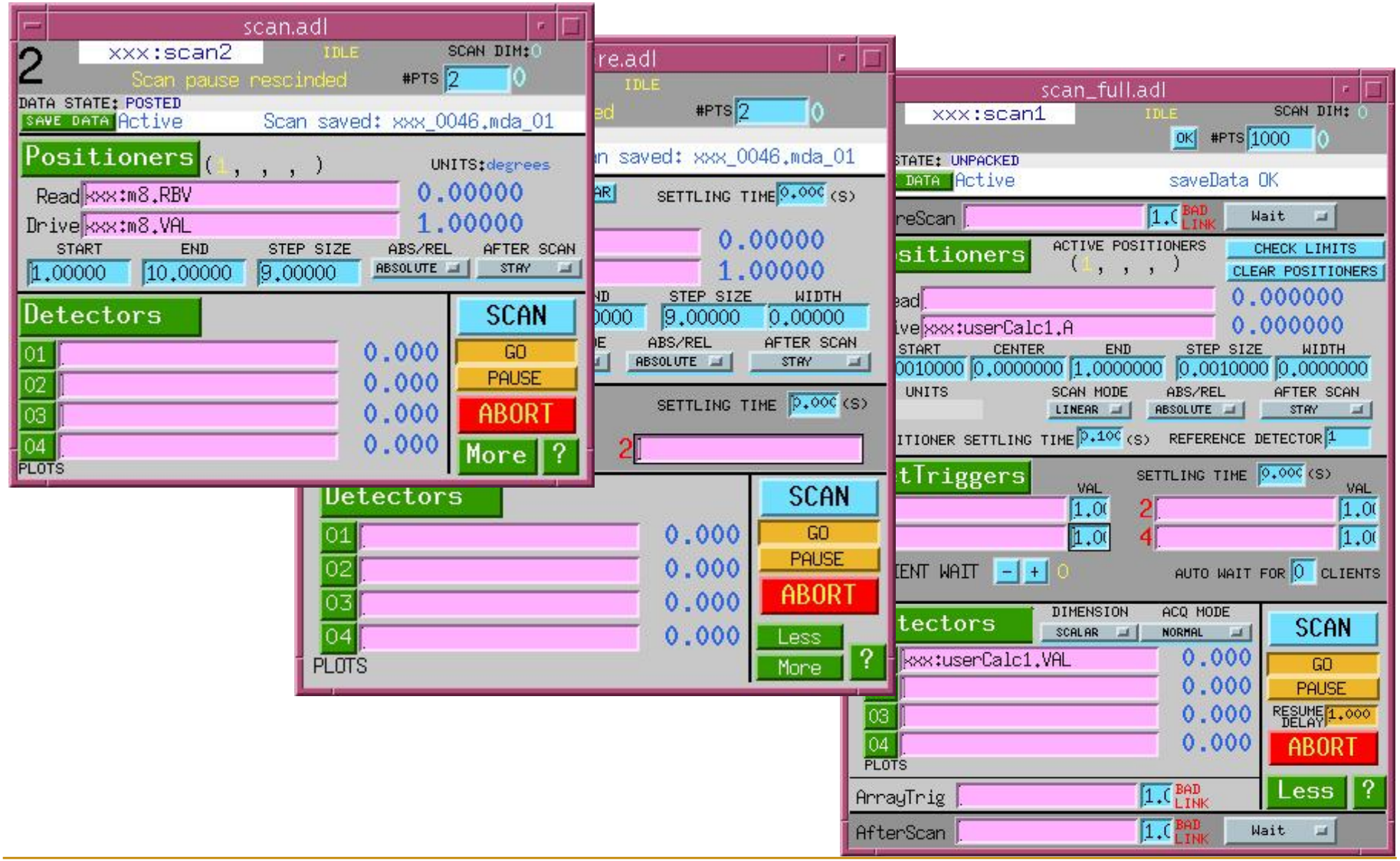
- **The sscan record is a channel-access client**
  - scanned PV's can be hosted by any ioc
  - uses recDynLink library to manage connections with PV's
  - uses ca_put_callback() to set conditions, trigger detectors, and await completion
  - uses ca_get_callback() before acquiring data
- **saveData is a channel-access client**
  - in principle, saveData can monitor sscan records hosted by a different ioc
  - in practice, don't do this if you can avoid it
- **Scan acquisition/storage can run on vxWorks, Linux, or Solaris.**
  - New in synApps 5.1 (EPICS 3.14)
- **The sscan record can be driven by any channel-access client.**
  - manual operation, via MEDM, is one option
  - can simplify user-written scan-control software

Before-scan / after-scan links

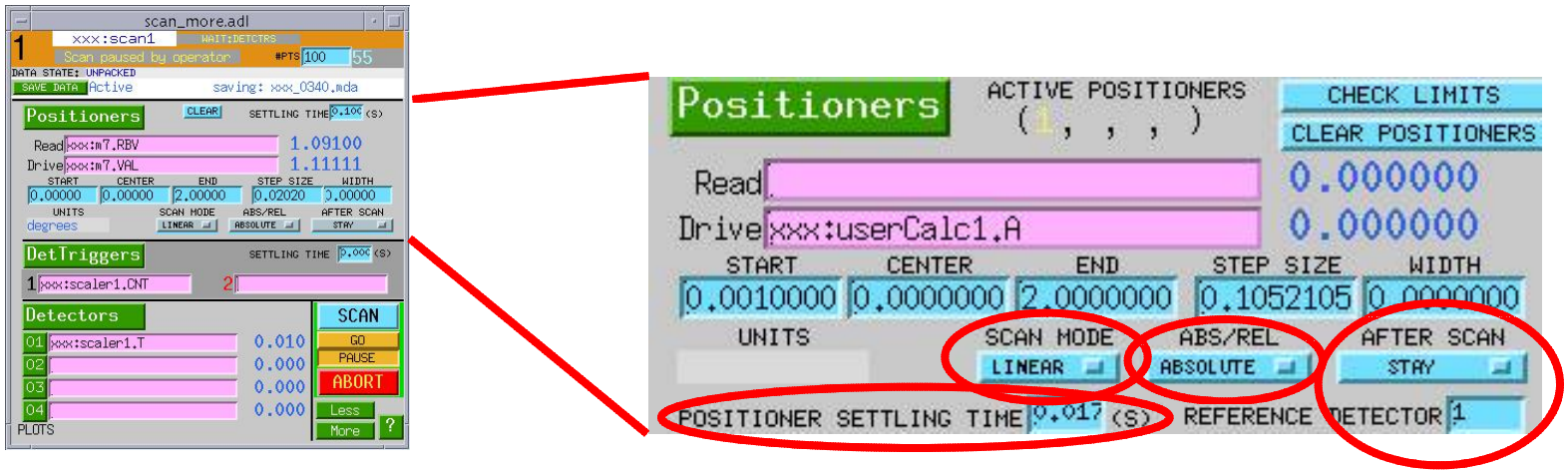
- Can write a constant value to any numeric or menu PV before the scan starts and/or after the scan ends.
- Can wait or not wait for completion of processing started by the write.
- If this sscan record is part of a multidimensional scan, links function on each iteration.
- Outer-loop sscan record can write to these links, and to the values they write.
- These links cannot write to their own sscan record's START, etc. fields



MEDM user interface



Positioner options



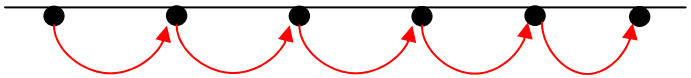
- **SCAN MODE (.P*n*SM - per positioner)**
  - Determines how and to where positioner moves
- **Absolute/Relative (.P*n*AR - per positioner)**
  - Determines how positioner locations are written
- **Positioner delay (.PDLY - affects all positioners)**
  - Delay while positioners are settling, after completing their moves
- **After-scan motion (.PASM - affects all positioners)**
  - Determines what, if anything, is done with positioners when scan is finished

...Positioner options

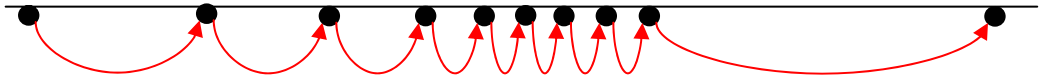
- **SCAN MODE (.P*n*SM - per positioner)**
  - **LINEAR** – Evenly spaced positions are calculated algorithmically
    - You specify positioner locations by setting any three of

START	*CENTER*	*END*	*WIDTH*	*STEP SIZE*	*# POINTS*
.P*n*SP	**.P*n*CP**	**.P*n*EP**	**.P*n*WD**	**.P*n*SI**	**.NPTS**

    - The sscan record reconciles unset parameters

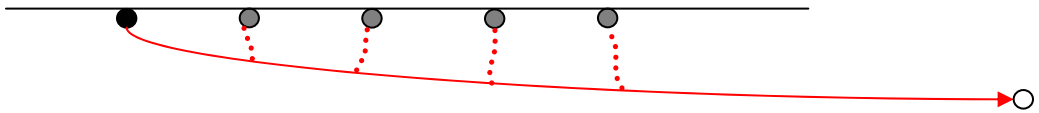


  - **TABLE** – Positioner locations are contained in the **.P*n*PA** array
    - The array must contain at least **.NPTS** values
    - You must arrange for the array to contain the desired positions before starting the scan.
    - The **.P*n*PA** array is never overwritten by the sscan record



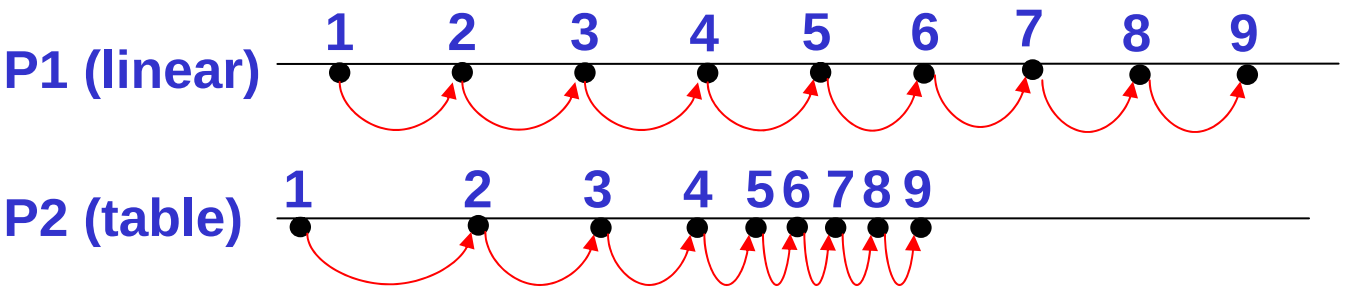
...Positioner options

- **...SCAN MODE (.P*n*SM - per positioner)**
  - **FLY** – data will be acquired *while* positioner moves
    - You specify positions at which data are acquired by setting *START*, *END*, positioner speed, and detector acquisition time.
    - The following algorithm is executed:
      - Positioner sent to *START*; reports completion
      - Detector triggered; reports completion
      - First data point acquired
      - Positioner sent to *END*
      - *NPTS*-1 iterations of
        - Detector triggered; reports completion
        - Data point acquired
    - The timing of data points is controlled by the detector's acquisition time.
    - Fly-mode positioners do not report completion. (The positioner may still be moving after the scan ends.)

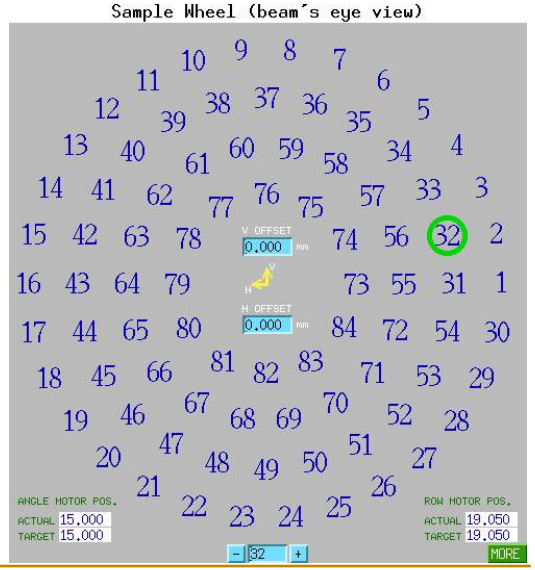


...Positioner options

- **...SCAN MODE (.P*n*SM - per positioner)**
  - OK to mix scan modes:



  - Don't be limited by existing positioner modes
    - A positioner is *anything* you can write to
    - Can specify positions algorithmically, using calcout or transform
      - E.g., sample-wheel
    - Can write to positioner through interpolation table
      - Use a spare positioner readback to get actual positions into the data file



...Positioner options

- **Absolute/Relative (.P*n*AR - per positioner)**
  - If **.P*n*AR** == “ABSOLUTE” (0), positions are sent exactly as given.
  - If **.P*n*AR** == “RELATIVE” (1), positions are added to pre-scan position before being sent to positioner.

- **Settling time (.PDLY - affects all positioners)**
  - If any positioner PV is specified, then after all positioners report completion, the sscan record waits for **.PDLY** seconds before moving to next phase of sscan.
    - Useful for positioners that “ring” after move is completed
    - Useful work-around for positioners that cannot report completion
  - If no positioners, then settling time is ignored.
  - Settling time is adjusted to nearest multiple of system-clock period (typically 1/60Hz).

...Positioner options

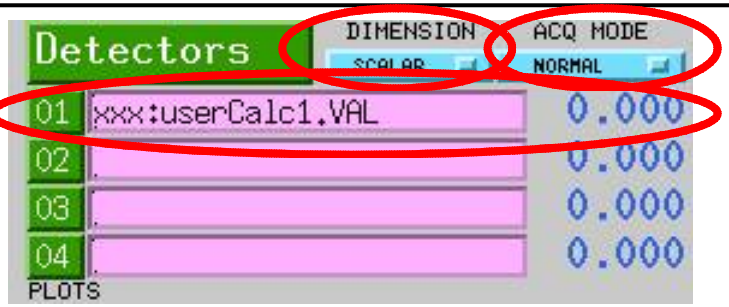
- **After-scan motion (.PASM - affects all positioners)**
  - STAY – positioners are simply left where they ended up
  - START POS – positioners are sent to their *START* positions
  - PRIOR POS – positioners are sent to their pre-scan positions

  - PEAK POS – data from the reference detector (number given by the **.REFD** field, in range [1..70]) is examined. If a peak is found, positioners are sent to where it was acquired.
  - VALLEY POS – similar, but valley instead of peak
  - +EDGE POS – peak of derivative of reference data
  - -EDGE POS – valley of derivative of reference data

Detector triggers

- 0-4 detector triggers (.T*n*PV), intended to start data-acquisition
- Similar to positioners, but value sent (.T*n*CD) is constant
- Triggers execute after all positioners have completed, and after any positioner settling time has elapsed.
- Detector settling time begins after all detector triggers have reported completion.
- If no triggers, then settling time is ignored.

Detectors



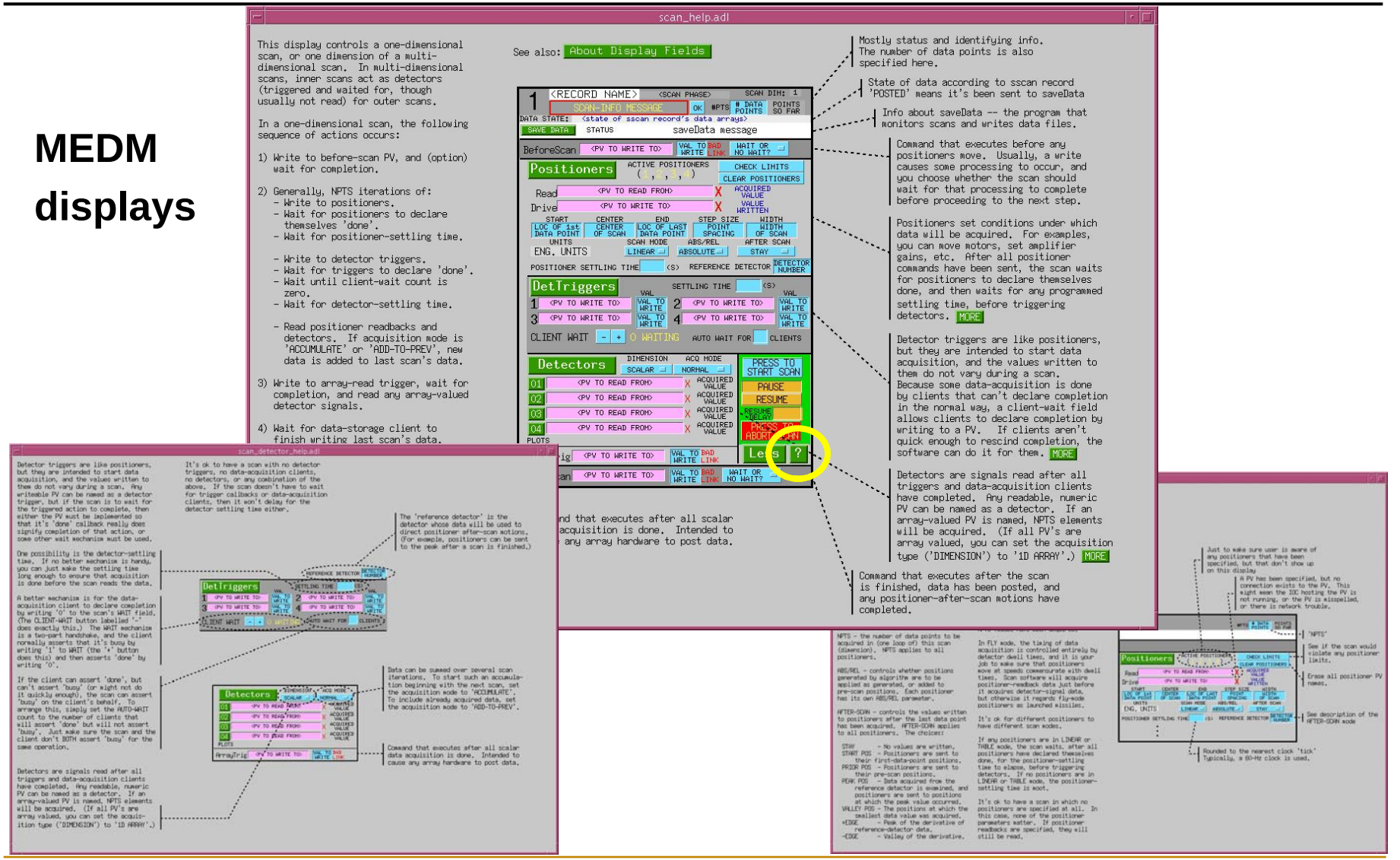
- **PV's to be acquired during scan**
- **0-70 detectors (.D01PV - .D70PV)**
- **Detector options**
  - Acquisition type (**.ACQT**)
    - SCALAR
      - scalar PV's acquired at each positioner location
      - Array PV's (**.NPTS** elements) acquired at end of scan
    - 1D ARRAY
      - use this mode only if ALL detectors are array valued
      - Positioners are only sent to their START positions.
      - In the future, array-valued positioners may be supported.
  - Acquisition mode (**.ACQM**)
    - NORMAL – store values as acquired
    - ACCUMULATE – add detector values, starting with next scan
    - ADD TO PREV – same, but starting with previous scan

Scan controls

- **SCAN**
  - Writing ‘1’ starts this sscan record
  - Writing ‘0’ stops this sscan record. (But with the supplied database, always use the ‘ABORT’ button to stop.)
- **GO/PAUSE**
  - Pause is immediate, Go occurs after delay
- **ABORT**
  - Writes ‘1’ to ‘xxx:allstop.VAL’, which should stop motors
  - Sends “stop” message to *all* sscan records in the supplied database
    - First ‘Abort’ attempt ends scan after outstanding completion callbacks have come in, and data-storage client has released the previous scan’s data arrays.
    - Second ‘Abort’ attempt waits only for data-storage client.
    - Third successive ‘Abort’ attempt kills scan with no regard for consequences.

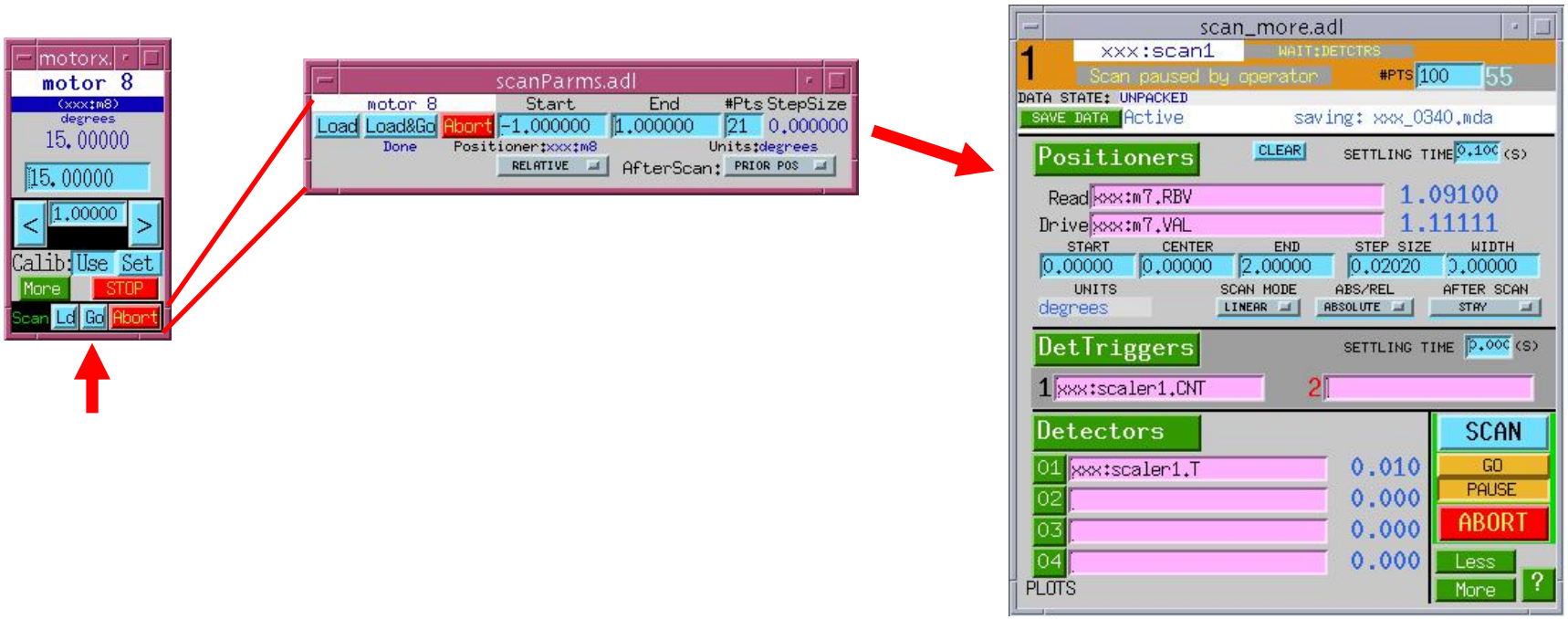
Scan user documentation

MEDM displays



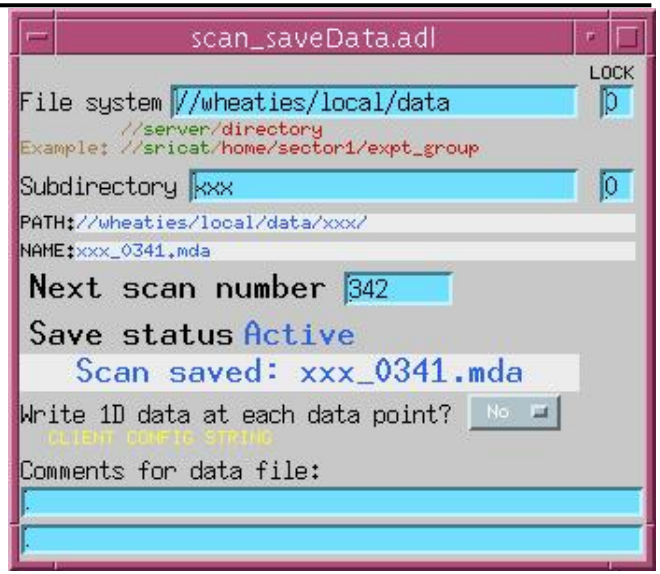
One-click scans

- **The scanparm record executes preprogrammed *linear* scans**
  - Holds scan parameters for a positioner
  - Writes parameters to a particular **sscan** record
  - Optionally executes the **sscan** record
  - Useful for alignment



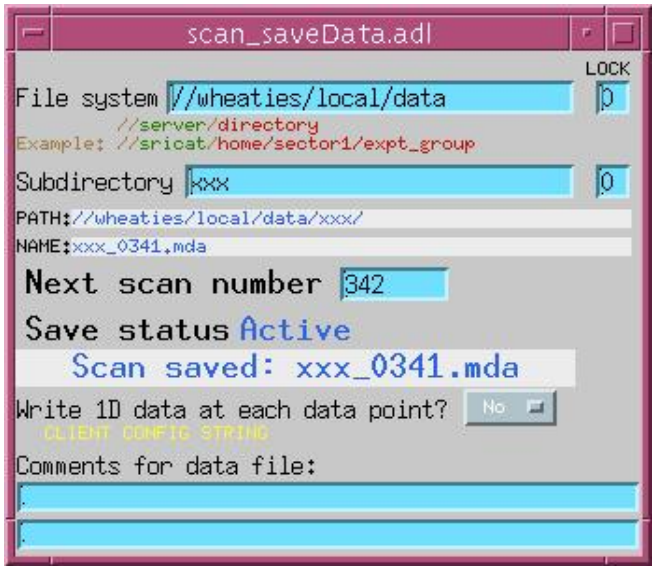
Data storage

- **saveData monitors sscan records and writes their data to numbered files.**
- **Handshake permits pipelined operation.**
- **saveData's boot-time init can specify list of PV's to write with every scan's data**
- **saveData writes "MDA" files**
  - MultiDimensional Archive
  - Binary, cross-platform (XDR) format
  - Format is optimized for run-time access.
  - Format permits file to be closed after each set of writes.
- **Automatic file numbering**
  - e.g., 'xxx_0123.mda', 'xxx_0124.mda'
  - overlap is handled: 'xxx_0123.mda_01'



...Data storage

- **Location of data files**
  - ‘File system’ + ‘subdirectory’
  - vxWorks:
    - File system is NFS-mount point
    - ‘//<hostname>’ is required
  - Linux, Solaris:
    - saveData doesn’t mount the file system (system administrator does this)
    - ‘//<hostname>’, if present, is ignored
- **Cannot write to ‘File system’ or ‘subdirectory’ while a scan is in progress. (See ‘LOCK’ PV.)**
- **Don’t delete or rename the directory saveData is writing to.**
- **Comment PV’s saved only if they are named in saveData.req**



saveData.req init file

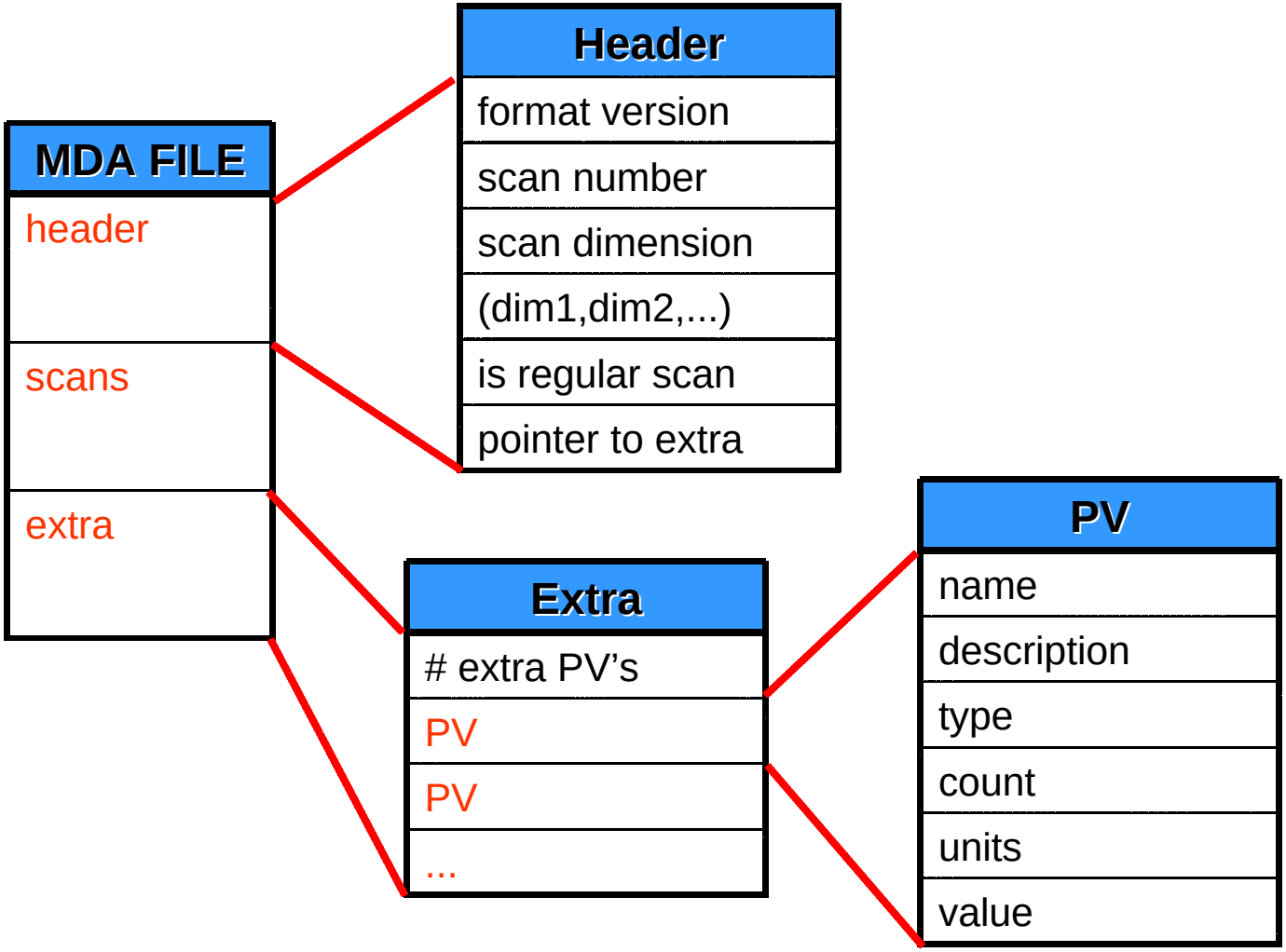
```
[prefix]
$(P)

[status]
$(P)saveData_status
...

[scanRecord]
$(P)scanH
$(P)scan1
$(P)scan2
$(P)scan3
$(P)scan4

[extraPV]
#<PV name> <description>
$(P)scaler1.TP  “scaler preset (s)”
$(P)scaler1.NM1 “scaler chan 1 desc”
...
```

Section head

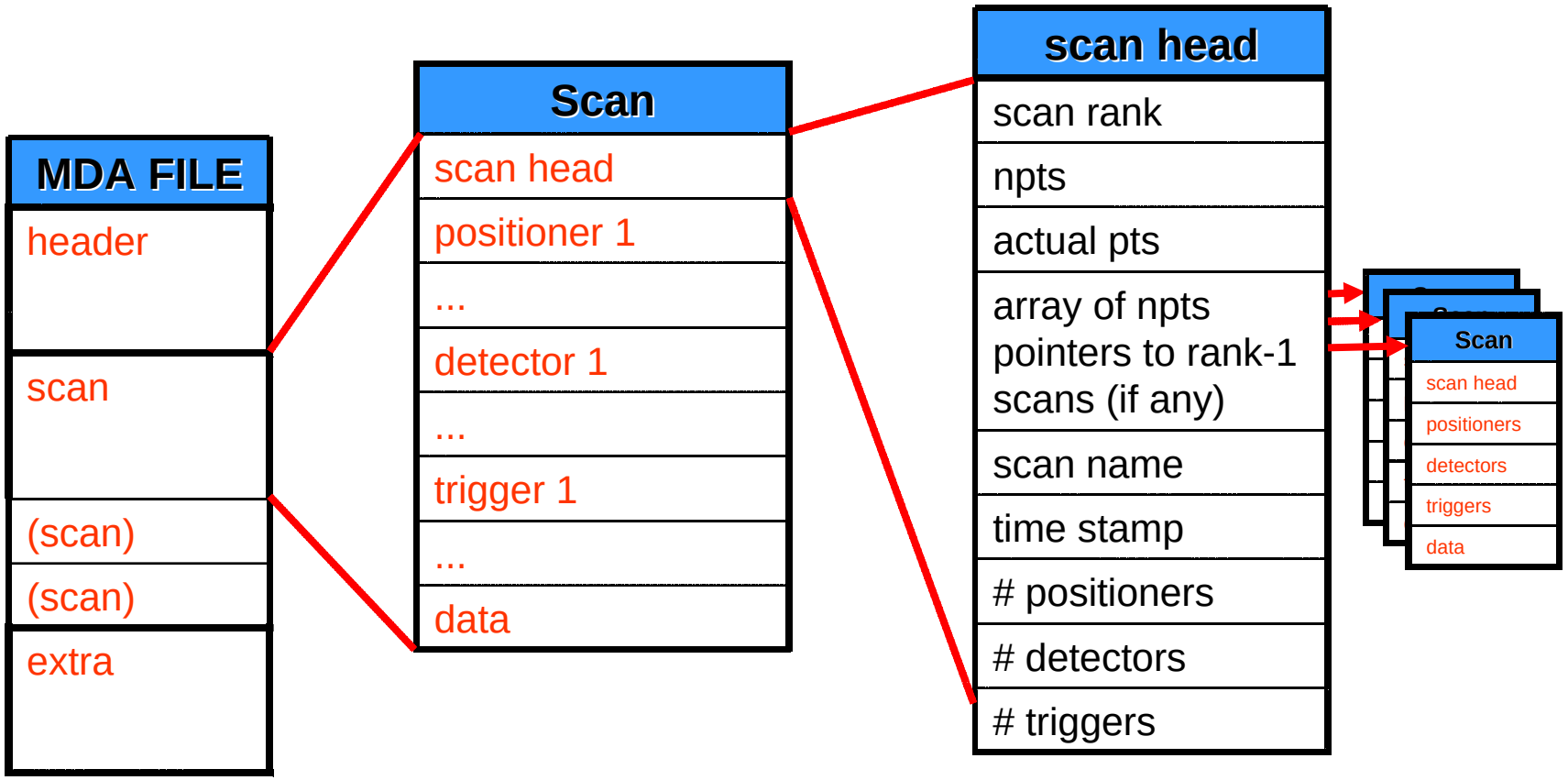
List of sscan records to monitor:

List of PV’s to be saved with every scan (Normally, this is the only section you modify.)

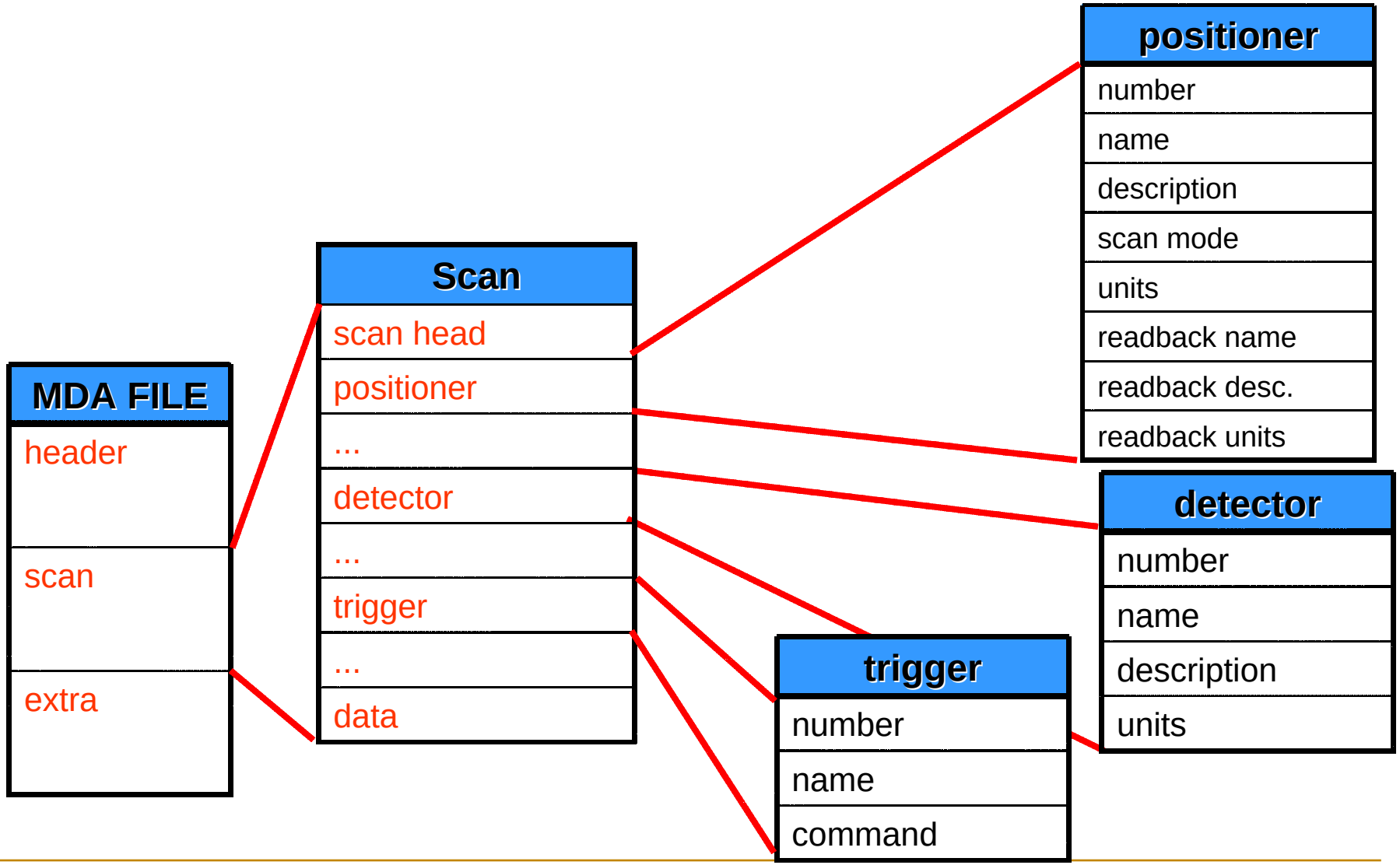
MDA file format



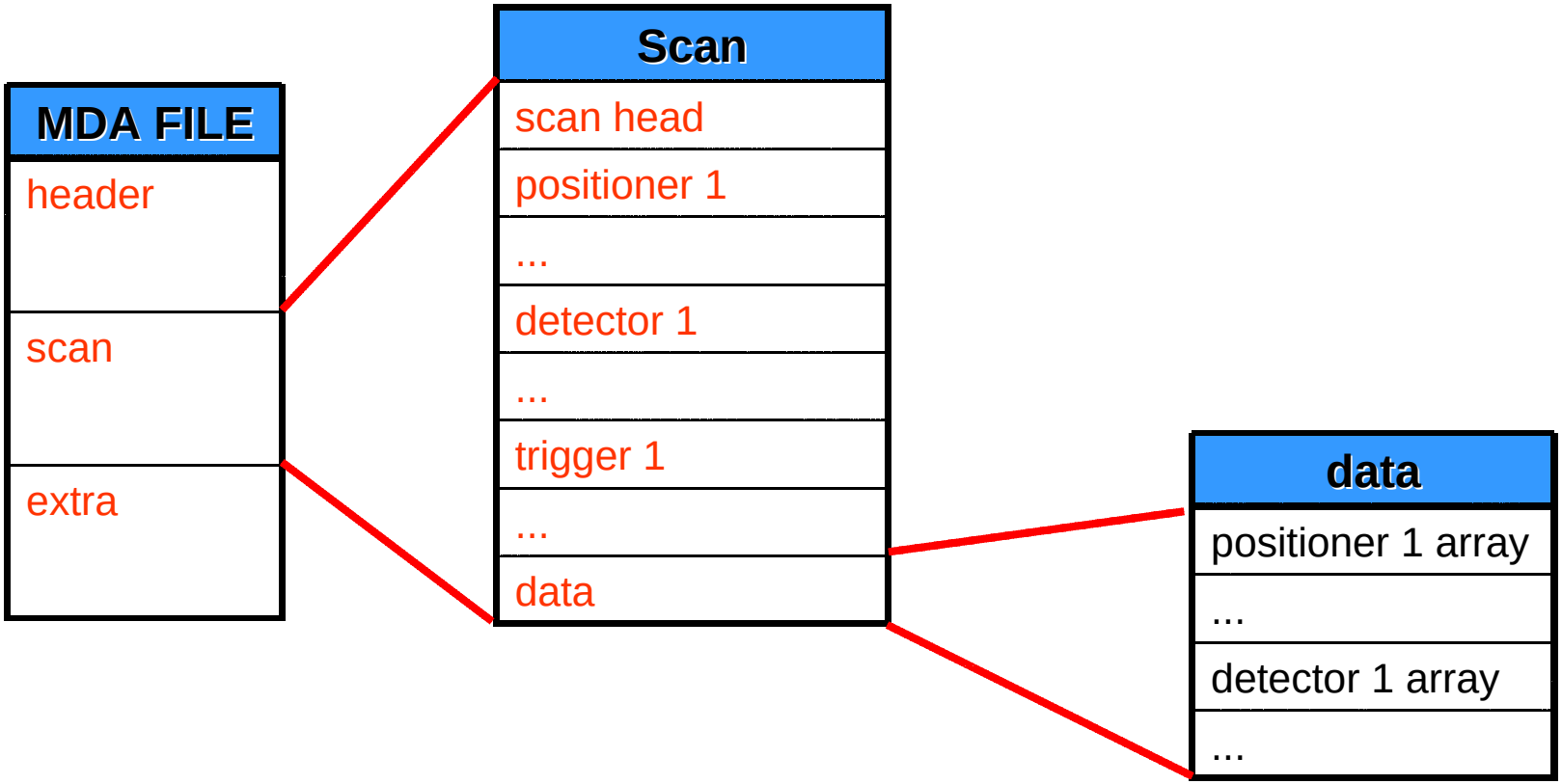
...MDA file format



...MDA file format



...MDA file format



Other data-acquisition-related software

- **Data-visualization tools for use with synApps**
  - Run-time look at scan data
  - Offline tools for data-file manipulation
  - Supports 1-3 dimensional data
  - Distributed independently of ioc software
  - See lecture "***Data Visualization.***"

- **CCD data-acquisition tools**
  1) CCD module (see lecture "***Detectors and Feedback****")*
  2) Portable CA Server based CCD support, and related software
     - *http://www.aps.anl.gov/aod/bcda/dataAcq/index.php*
  - Both of these solutions allow an EPICS CA client to drive data acquisition.
  - Both support ca_put_callback(), as needed by the **sscan** record.

Completion reporting

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

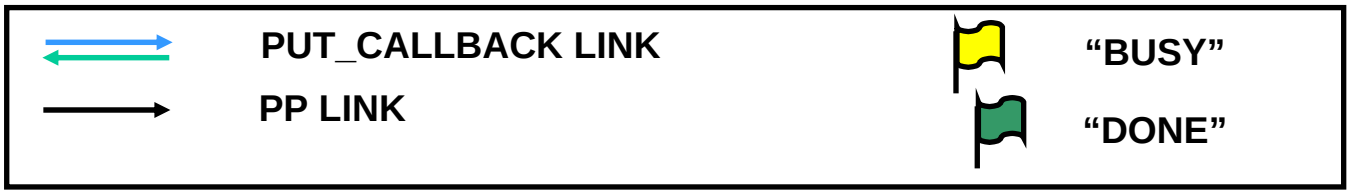
- **Database operations spanning more than one ioc:**
  - Use records with put_callback links to span iocs:
    - **calcout** with asynchronous device support
    - **sscan**, **swait** (i.e., a synApps “userCalc”)
    - **sseq** or **sCalcout** (with .WAIT* = “Wait”)

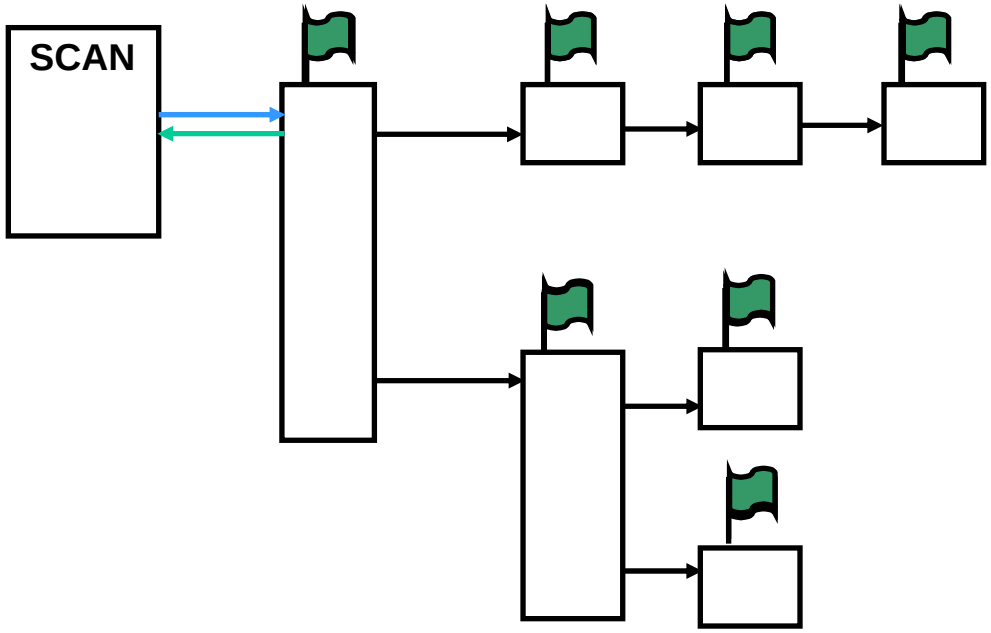
- **Cases in which a CA client performs part of the operation:**
  1) Database sets a **busy** record via PP or put_callback link.
  2) CA client clears the **busy** record when operation is done.

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

...Completion reporting

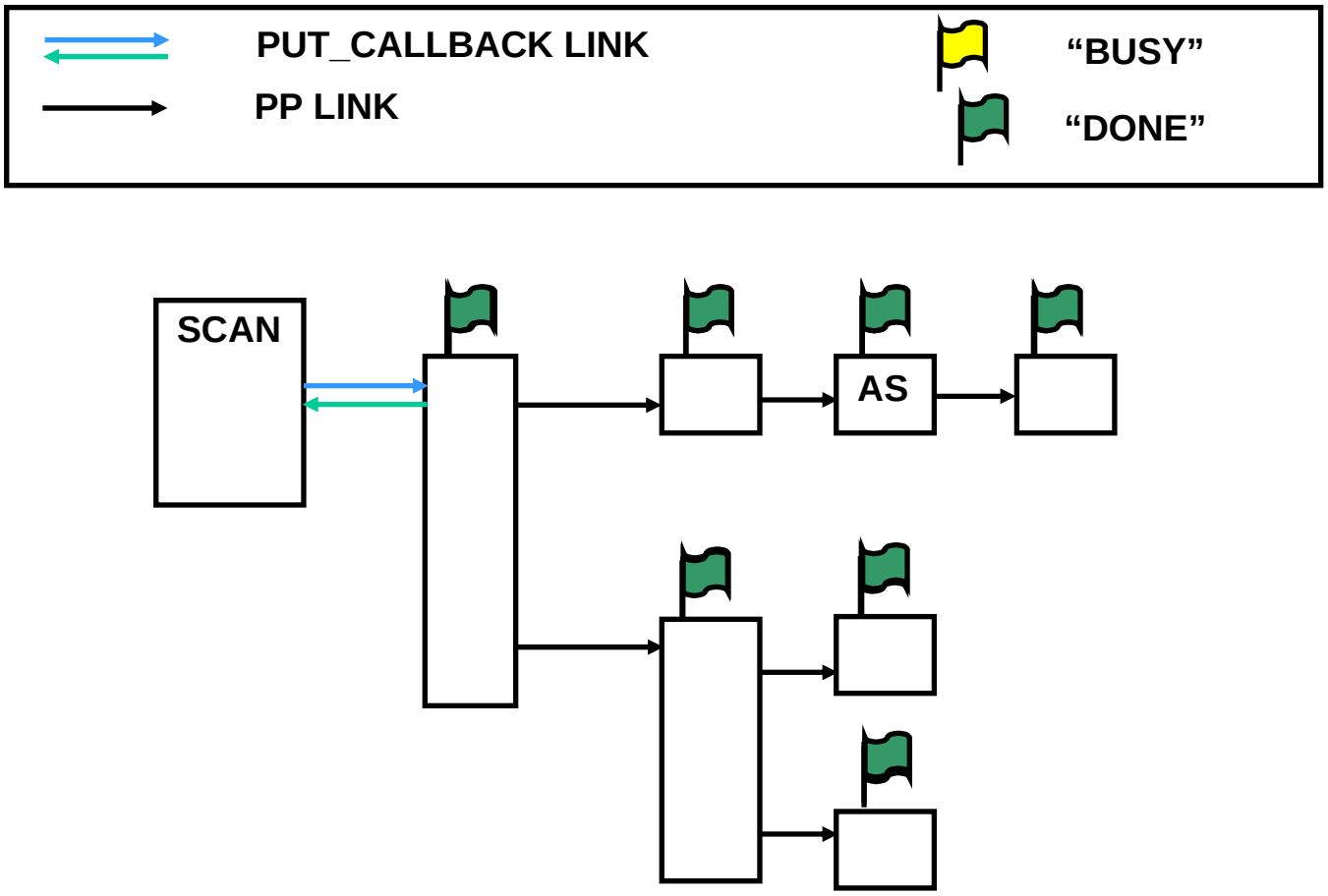
- **Use only PP links and forward links in execution chain.**





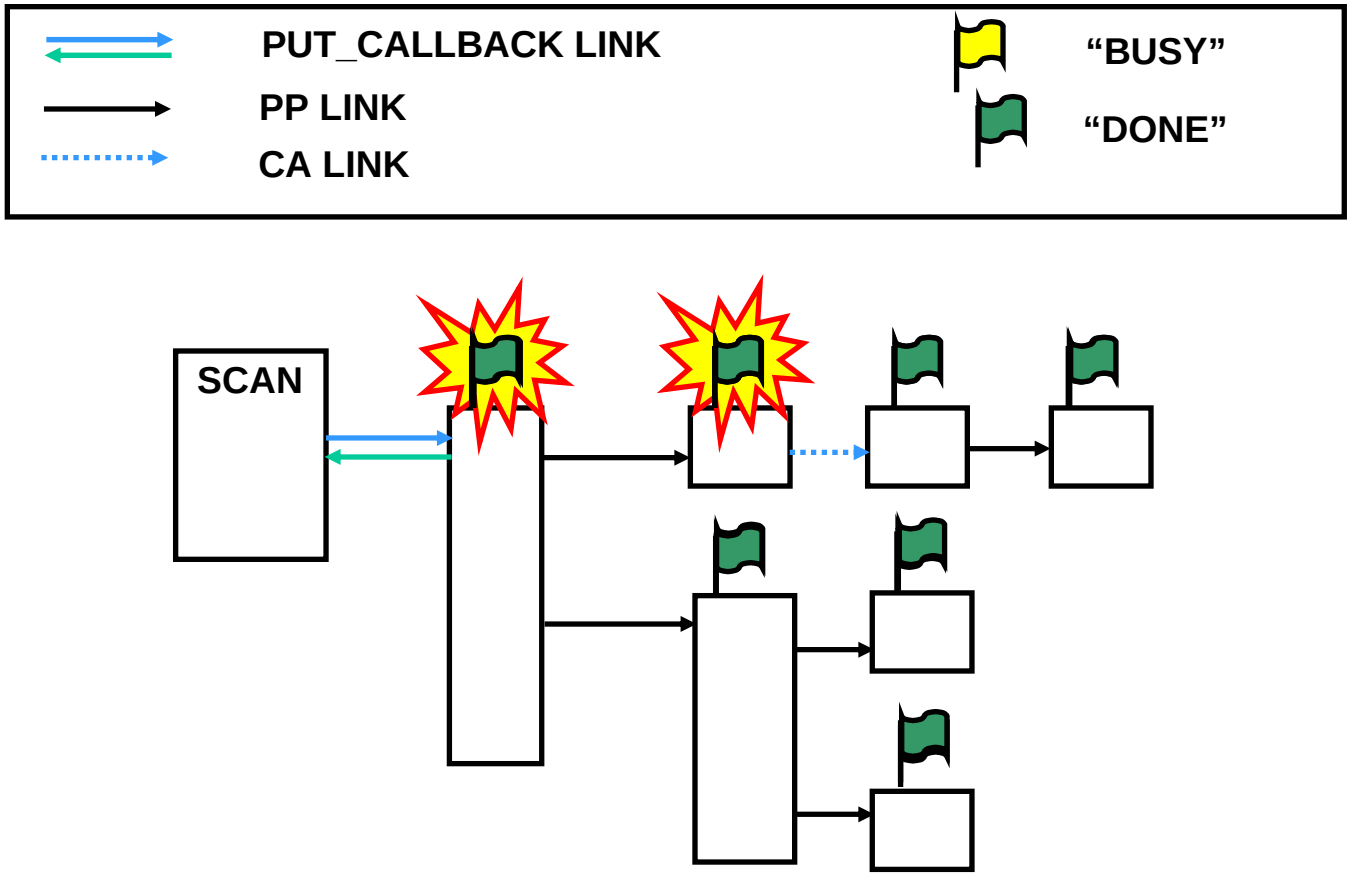
…Completion reporting

- **Same as before, but with an *asynchronous* record**



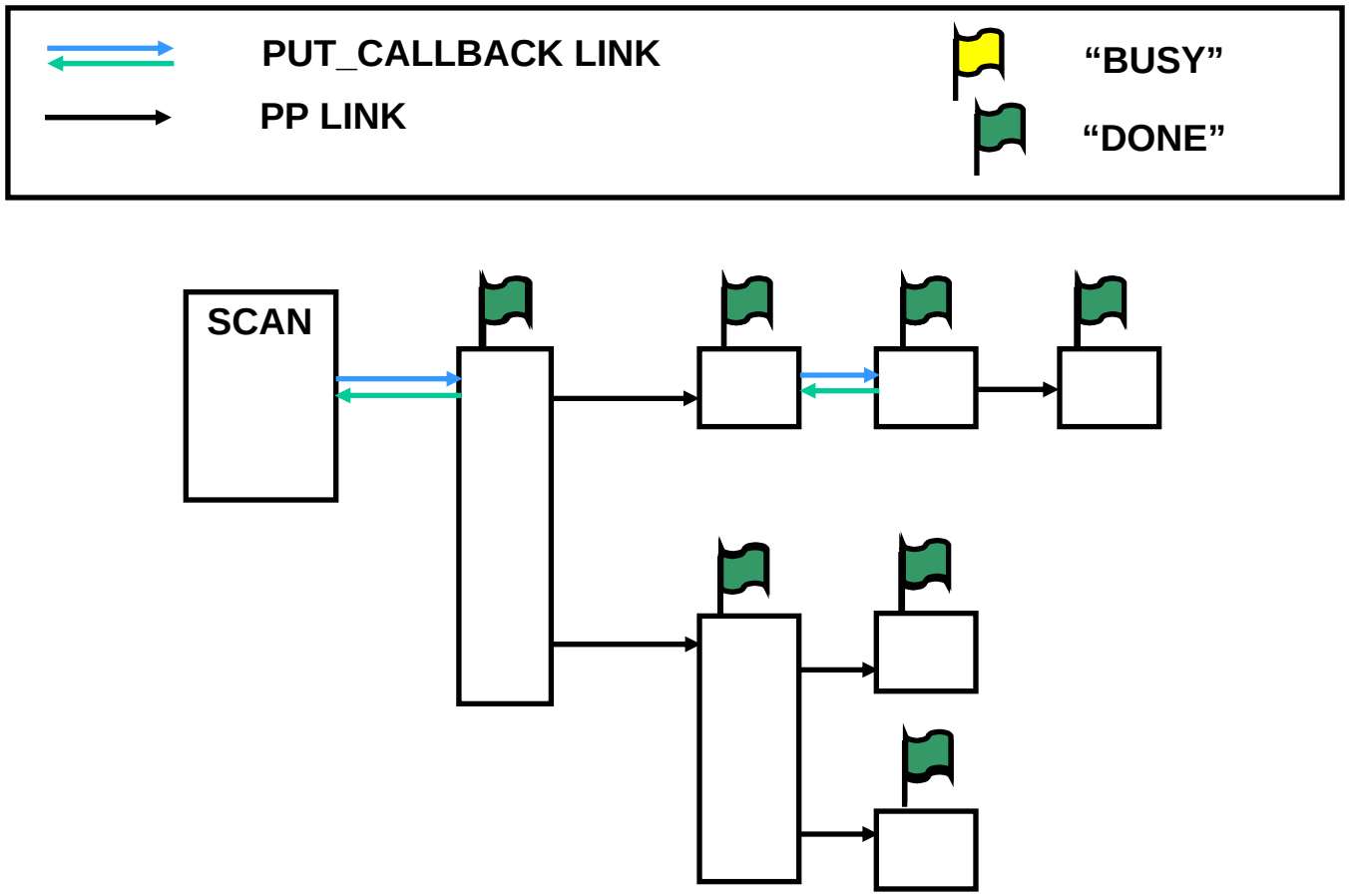
…Completion reporting

- **Premature “DONE” report, because CA-link execution is not traced**



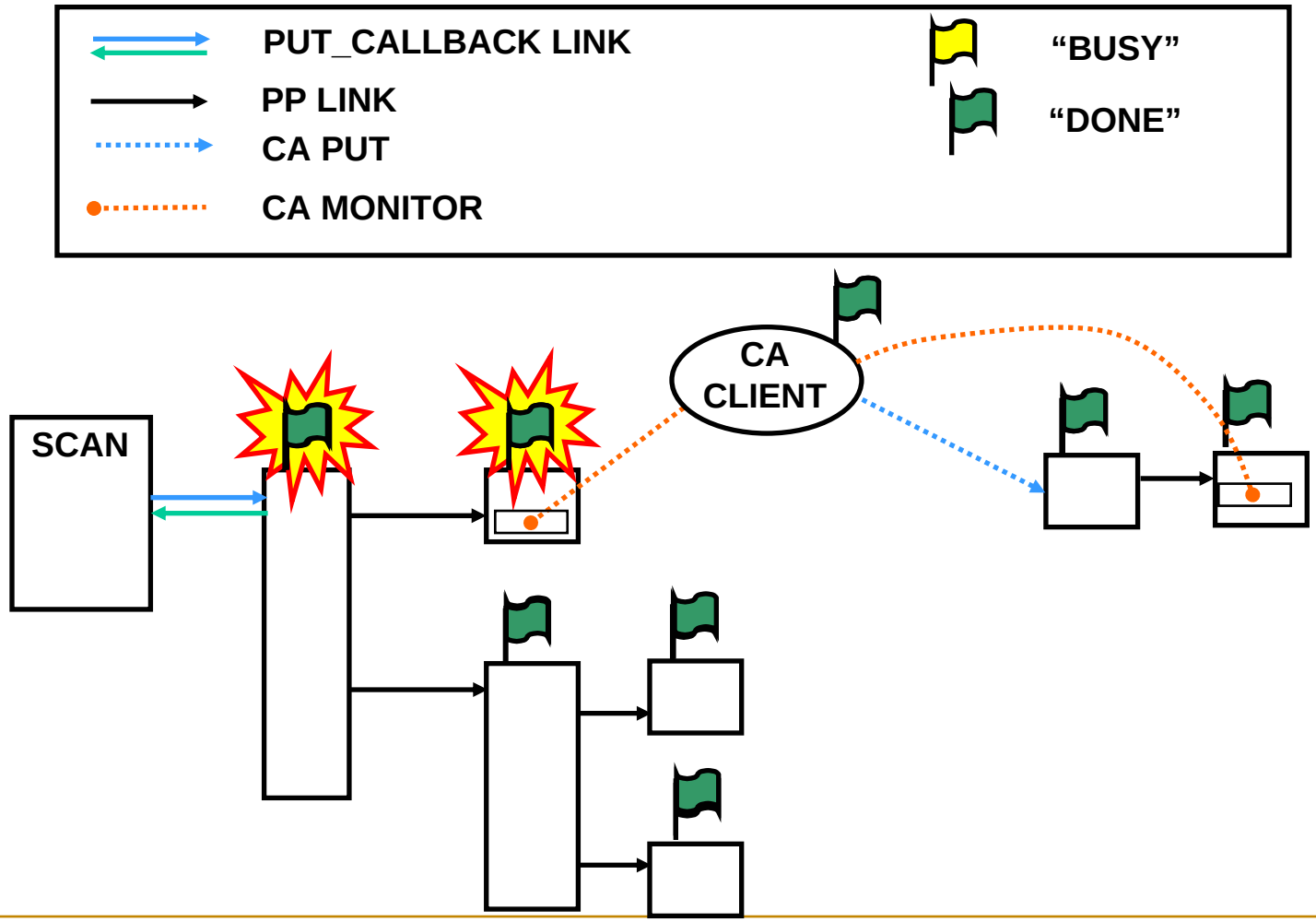
…Completion reporting

- **Premature-DONE problem fixed with a PUT_CALLBACK link**



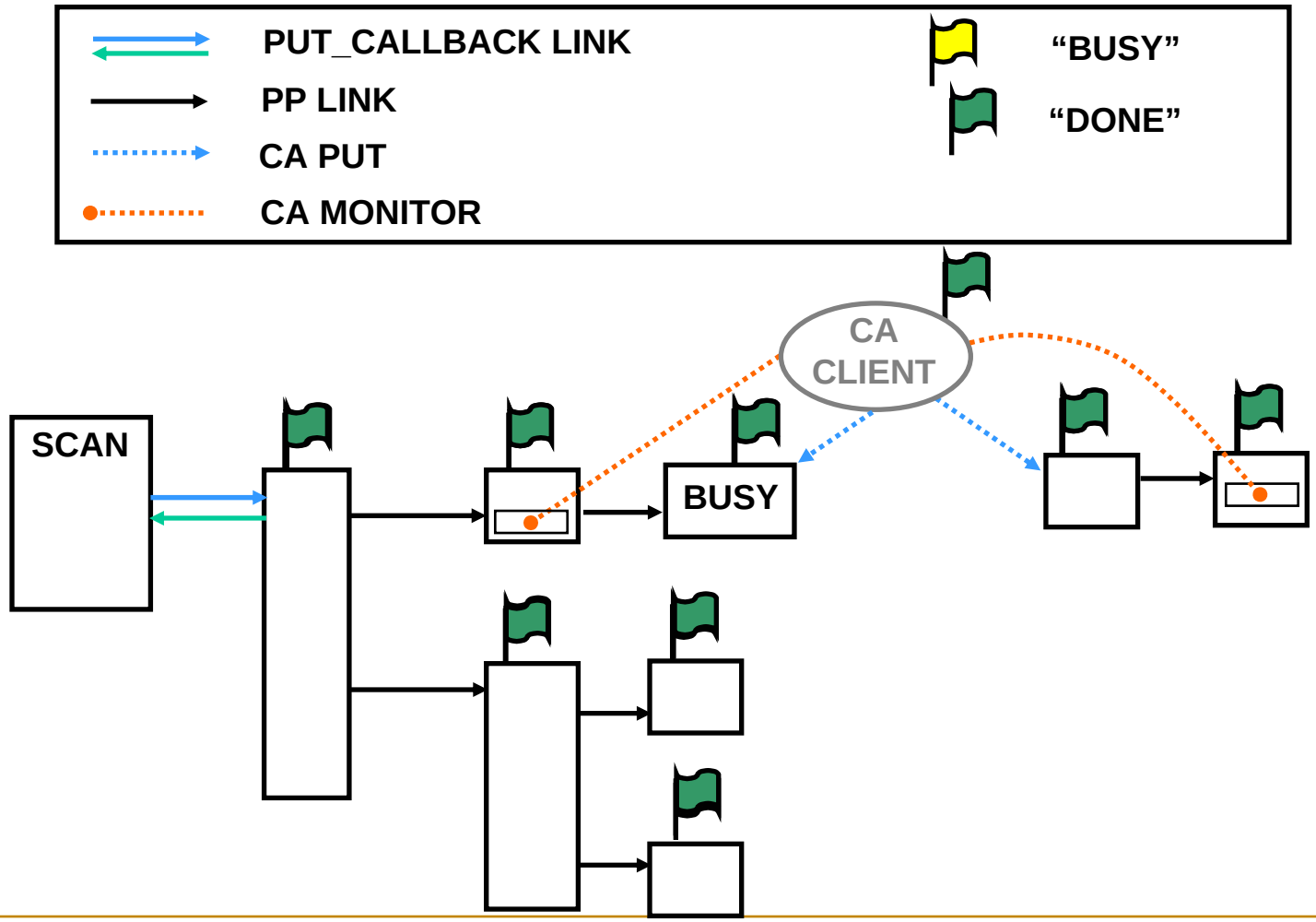
…Completion reporting

- **Premature “DONE” because CA-client processing is not traced**



…Completion reporting

- Premature “DONE” problem fixed with a ‘BUSY’ record



…Completion reporting