EPICS
Input Output Controller (IOC)
Record Reference Manual

Janet B. Anderson and
Martin R. Kraimer
Argonne National Laboratory
Advanced Photon Source
Issue 1: December 1,1994 - DRAFT
Issue 1a: April 25, 1996 - DRAFT
APS Release 3.12

Table of Contents

Chapter 1: Introduction

1. Overview

This manual describes all supported EPICS record types. The first chapter gives the introduction and describes the field summary table. The second chapter describes the fields in database common, i.e. the fields that are present in every record type. The third chapter describes the input and output field that are common to many record types and have the same usage wherever they are used. Following the third chapter is a separate chapter for each record type containing a description of all the fields for that record type except those in database common.

2. Field Summary Table

Each chapter contains a field summary table of the form:

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP

The meaning of each component of the summary table is as follows:

- **Field**: The field name
- **Type**: The database field type, i.e. `DBF_<type>`
- **DCT**: Is this field definable via the database configuration tool?
- **Initial**: Initial value when record is created
- **Access**: Is this field accessible via database access?

- **Modify**: Can this field be modified via database access?
- **Rec Proc Monitor**: Does the record processing routine trigger monitors by a call to `db_post_event` when this field changes value?
- **PP**: Process passive? Will `dbPutField` call `dbProcessPassive` when this field is processed?

Chapter 2: Fields Common to All Record Types

1. Introduction

This chapter contains a description of fields that are common to all records. These fields are defined in `dbcommon.ascii`.

2. Database Common: Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
NAME	STRING	Yes	0	Yes	No		No
DESC	STRING	Yes	Null	Yes	Yes	No	No
ASG	STRING	Yes	Null	Yes	Yes		No
SCAN	GBLCHOICE	Yes	0	Yes	Yes	No	No
PINI	GBLCHOICE	Yes	0	Yes	Yes	No	No
PHAS	SHORT	Yes	0	Yes	Yes	No	No
EVNT	SHORT	Yes	0	Yes	Yes	No	No
TSE	SHORT	No	0	Yes	Yes		No
TSEL	INLINK	Yes	Null	No	No	N/A	No
DTYP	DEVCHOICE	Yes	0	Yes	No		No
DISV	SHORT	Yes	1	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
DISA	SHORT	No	0	Yes	Yes	No	No
SDIS	INLINK	Yes	0	No	No	N/A	No
MLOK	NOACCESS	No	8	No	No		No
MLIS	NOACCESS	No	12	No	No		No
DISP	UCHAR	No	0	Yes	Yes	Yes	No
PROC	UCHAR	No	0	Yes	Yes	No	Yes
STAT	GBLCHOICE	No	UDF_ALARM	Yes	No	Yes	No
SEVR	GBLCHOICE	No	INVALID_ALARM	Yes	No	Yes	No
NSTA	GBLCHOICE	No	0	Yes	No	No	No
NSEV	GBLCHOICE	No	0	Yes	No	No	No
ACKS	GBLCHOICE	No	0	Yes	No		No
ACKT	GBLCHOICE	No	11	Yes	No		No
DISS	GBLCHOICE	Yes	0	Yes	Yes	No	No
LSET	SHORT	No	0	Yes	No		No
LCNT	UCHAR	No	0	Yes	No		No
PACT	UCHAR	No	0	Yes	No		No
PUTF	UCHAR	No	0	Yes	No		No
RPRO	UCHAR	No	0	Yes	No		No
ASP	NOACCESS	No	4	No	No		No
PPN	NOACCESS	No	4	No	No		No
PPNN	NOACCESS	No	4	No	No		No
SPVT	NOACCESS	No	4	No	No		No
RSET	NOACCESS	No	4	No	No		No
DSET	NOACCESS	No	4	No	No		No
DPVT	NOACCESS	No	4	No	No		No
PRIO	GBLCHOICE	Yes	0	Yes	Yes	No	No
TPRO	UCHAR	No	0	Yes	Yes	No	No
BKPT	NOACCESS	No	1	No	No		No
UDF	UCHAR	No	1	Yes	Yes	No	Yes
TIME	NOACCESS	No	8	Option	No	No	No
FLNK	FWDLINK	Yes	Null	No	No	N/A	No

3. Database Common: Field Descriptions

Name	Summary	Description
NAME	Record Name	An arbitrary 28 character record name supplied by the application developer. This name is the means of identifying a specific record. It must have a unique value across all IOCs attached to the same local area subnet.
DESC	Description	An arbitrary 28 character record description supplied by the application developer.
ASG	Access Security Group	A character string value defining the access security group for this record. If left `NULL`, the record is placed in group `DEFAULT`.
SCAN	Scanning Algorithm	This can be one of the periodic intervals, I/O event, event, or passive.
PINI	Process at Initialization	If this field is set to `TRUE` during database configuration, then the record is processed once at IOC initialization (before the normal scan tasks are started).
PHAS	Scan Phase Number	This field orders the records within a specific SCAN group. This is not meaningful for passive records. All records of a specified phase are processed before those with higher phase number. Whenever possible it is better to use linked passive records to enforce the order of processing rather than phase number.
EVNT	Event Number	Event number for scan type `SCAN_EVENT`. All records with scan type event and the same `EVNT` value will be processed when a call to `post_event` for `EVNT` is made. The call to `post_event` is: `post_event(short event_number)`
TSE	Time Stamp Event	The event number for time stamp. This is only meaningful if the IOC has an associated hardware event receiver. See '`er`' record for details.
TSEL	Time Stamp Event Link	An input link for obtaining the time stamp event number.
DTYP	Device Type	This field specifies the device type for the record. Each record type has its own set of device support routines which are specified in `devSup.ASCII`. If a record type does not have any associated device support, `DTYP` and `DSET` are meaningless.
DISV	Disable Value	If `DISV=DISA`, then the record will be disabled, i.e. `dbProcess` will not process the record.
DISA	Scan Disable Input Link Value	This is the value that is compared with `DISV` to determine if the record is disabled. Its value is obtained via `SDIS` if `SDIS` is a database or channel access link. If `SDIS` is not a database or channel access link, then `DISA` can be set via `dbPutField` or `dbPutLink`.

Name	Summary	Description
SDIS	Scan Disable Input Link	An input link from which to obtain a value for `DISA`. This field is ignored unless it is a database link or a channel access link. If it is a database or a channel access link, `dbProcess` calls `dbGetLink` to obtain a value for `DISA` before deciding to call the processing routine.
MLOK	Monitor Lock	The lock used by the monitor routines when the monitor list is being used. The list is locked whenever monitors are being scheduled, invoked, or when monitors are being added to or removed from the list. This field is accessed only by the `dbEvent` routines.
MLIS	Monitor List	This is the head of the list of monitors connected to this record. Each record support module is responsible for triggering monitors for any fields that change as a result of record processing. Monitors are present if `mlis.count` is greater than zero. The call to trigger monitors is: `db_post_event(precord,&data,mask)` Where "`mask`" is some combination of `DBE_ALARM`, `DBE_VALUE`, and `DBE_LOG`.
DISP	Disable putFields	If this field is set to `TRUE`, then all `dbPutFields` (normally issued by channel access) directed to this record are ignored except to the field `DISP` itself.
PROC	Process Record	A record will be processed whenever a `dbPutField` is directed to this field.
STAT	Current Alarm Status	These four fields are the alarm status and severity fields. `STAT` and `SEVR` are the values seen outside database access. `NSTA` and `NSEV` are the fields the database access, record support, and device support use to set new alarm status and severity values. Whenever any software component discovers an alarm condition, it uses the following macro function: `recGblSetSevr(precord,new_status,new_severity)` This ensures that the current alarm severity is set equal to the highest outstanding alarm. The file `alarm.h` defines all allowed alarm status and severity values.
SEVR	Current Alarm Severity	
NSTA	New Alarm Status	
NSEV	New Alarm Severity	
ACKS	Alarm Acknowledge Severity	Highest severity unacknowledged alarm
ACKT	Alarm Acknowledge Transient	Is it necessary to acknowledge transient alarms?
DISS	Disable Alarm Severity	When this record is disabled, it will be put into alarm with this severity and a status of `DISABLE_ALARM`.
LSET	Lock Set	The lock set to which this record belongs. All records linked in any way via input, output, or forward database links belong to the same lock set. The only exception is that non-process passive input links do not force the linked record to be in the same lock set. The lock sets are determined at IOC initialization time.

Name	Summary	Description
LCNT	Lock Count	The number of times in succession `dbProcess` finds the record active, i.e. `PACT` is `TRUE`. If `dbProcess` finds the record active `MAX_LOCK` (currently set to 10) times in succession, it raises a `SCAN_ALARM`.
PACT	Processing Active	See *Application Developers Guide* for details on usage. `PACT` is `TRUE` while the record is being processed. For asynchronous records `PACT` can be `TRUE` from the time record processing is started until the asynchronous completion occurs. As long as `PACT` is `TRUE`, `dbProcess` will not call the record processing routine.
PUTF	dbPutField Process	Did `dbPutField` cause the current record processing?
RPRO	Reprocess	Reprocess record when current processing completes.
ASP	Access Security Private	
PPN	Address of putNotify	Address of `putNotify` callback.
PPNN	Next Record for putNotify	Next record for `PutNotify`.
SPVT	Scan Private	This field is for private use of the scanning system.
RSET	Address of Record Support Entry Table	See *Application Developers Guide* for details on usage.
DSET	Address of Device Support Entry Table	This address of the device support entry table for this record. The value of this field is determined at IOC initialization time. Record support routines use this field to locate their device support routines.
DPVT	Device Private	This field is for private use of the device support modules.
PRIO	Priority	Scheduling priority for processing I/O Event scanned records and asynchronous record completion tasks.
TPRO	Trace Processing	If this field is set `TRUE`, a message is printed each time this record is processed and a message is printed for each record processed as a result of this record being processed
BKPT	BreakPoint	
UDF	VAL Undefined	`UDF` is initialized to `TRUE` at IOC initialization. Record and device support routines which write to the `VAL` field are responsible for setting `UDF` to `FALSE`.
TIME	Time	The time when this record was last processed, in standard format.
FLNK	Forward Link	This field is a database link. If `FLNK` is specified, processing this record will force a processing of the scan passive forward link record.

Chapter 3: Fields Common to Many Record Types

1. Introduction

This chapter describes input and output fields that are common to multiple record types. These fields have the same meaning whenever they are used.

2. Input Records

Common Fields

Name	Summary	Description
INP	Input Link	This field is used by the device support routines to obtain input. For soft analog records it can be a constant, a database link, or a channel access link.
DTYP	Device Type	`DTYP` specifies the name of the device support module that will input values. Each record type has its own set of device support routines which are specified in `devSup.ascii`. If a record type does not have any associated device support, `DTYP` is meaningless.
RVAL	Raw Value	Whenever possible this field contains the raw data value exactly as it is obtained from the hardware or from the associated device driver.
VAL	Value	This is the record's final value, after any needed conversions have been performed.

Name	Summary	Description
SIMM	Simulation Mode	This field has either the value YES or NO. By setting this field to YES, the record can be switched into simulation mode of operation. While in simulation mode, input will be obtained from SIOL instead of INP.
SIML	Simulation Mode Location	This field can be a constant, a database link, or a channel access link. If SIML is a database or channel access link, then SIMM is read from SIML. If SIML is a constant link then SIMM is initialized with the constant value but can be changed via dbPuts.
SVAL	Simulation Value	This is the record's input value, in engineering units, when the record is switched into simulation mode, i.e. when SIMM is set to YES.
SIOL	Simulation Value Location	This field can be a constant, a database link, or a channel access link. If SIOL is a database or channel access link, then SVAL is read from SIOL. If SIOL is a constant link then SVAL is initialized with the constant value but can be changed via dbPuts.
SIMS	Simulation Mode Alarm Severity	When this record is in simulation mode, it will be put into alarm with this severity and a status of SIMM.

Device Input

A device input routine normally returns one of the following values to its associated record support routine:

- **0**: Success and convert. The input value is in RVAL. The record support module is expected to compute VAL from RVAL.
- **2**: Success, but don't convert. The device support module can specify this value if it does not want any conversions. It might do this for two reasons:
  - A hardware error is detected (in this case, it should also raise an alarm condition).
  - The device support routine reads values directly into the VAL field and then sets UDF to FALSE.

Soft Input

In almost all cases, two special device support modules are provided: Soft and Raw Soft. Both allow INP to be a constant, a database link, or a channel access link. The Soft support module reads input directly into the VAL field and specifies that no conversion of any type should be performed. Thus Soft support allows the record to hold values corresponding to the C data type of the VAL field. Note that for soft input, RVAL is not used. The Raw Soft support module reads input into RVAL and asks that normal conversion to VAL be performed.

The device support read routine normally calls recGblGetLinkValue which performs the following steps:

- If the INP link type is CONSTANT recGblGetLinkValue does nothing and returns with a status of zero.
- If the INP link type is DB_LINK, then dbGetLink is called to obtain a new input value. If dbGetLink returns an error, a LINK_ALARM with a severity of INVALID_ALARM is raised. RecGblGetLinkValue returns the status of dbGetLink.

- If the INP link type is CA_LINK, then dbCaGetLink is called to obtain a new input value. If dbCaGetLink returns an error, a LINK alarm with a severity of INVALID is raised. RecGblGetLinkValue returns the status of dbCaGetLink.

If the return status of recGblGetLinkValue is zero and the INP link type is not CONSTANT, then UDF is set to FALSE. The device support read routine normally returns the status of recGblGetLinkValue.

Simulation Mode

A record can be switched into simulation mode of operation by setting the value of SIMM to YES. During simulation, the record will be put into alarm with a severity of SIMS and a status of SIMM_ALARM. While in simulation mode, input values, in engineering units, will be obtained from SIOL instead of INP. Also, while the record is in simulation mode, there will be no raw value conversion and no calls to device support during record processing.

Normally input records contain a private readValue routine which performs the following steps:

- If PACT is TRUE, the device support read routine is called, status is set to its return code, and readValue returns.
- Call recGblGetLinkValue to get a new value for SIMM if SIML is a DB_LINK or a CA_LINK.
- Check value of SIMM.
- If SIMM is NO, then call the device support read routine, set status to its return code, and return.
- If SIMM is YES, then call recGblGetLinkValue to read the input value from SIOL into SVAL. If success, then set VAL to SVAL and UDF to FALSE and set status to 2 (don't convert) if input record supports conversion. If SIMS is greater than zero, set alarm status to SIMM and severity to SIMS. Set status to the return code from recGblGetLinkValue and return.
- IF SIMM is not YES or NO, a SOFT alarm with a severity of INVALID is raised, and return status is set to -1.

3. Output Records

Common Fields

Name	Summary	Description
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
DTYP	Device Type	DTYP specifies the device support module that will receive values.
VAL	Value	This is the desired value before any conversions to raw output have been performed.

Name	Summary	Description
OVAL	Output Value	OVAL is used to decide when to invoke monitors. Archive and value change monitors are invoked if OVAL is not equal to VAL. If a record type needs to make adjustments, OVAL is used to enforce the maximum rate of change limit before converting the desired value to a raw value.
RVAL	Raw Value	Whenever possible this is the actual value sent to the hardware itself or to the associated device driver.
RBV	Read Back Value	Whenever possible this is the actual read back value obtained from the hardware itself or from the associated device driver.
DOL	Desired Output Location (an Input Link)	DOL can be a constant, a database link, or a channel access link. There is no device support associated with DOL. If DOL is a database or channel access link and OMSL is CLOSED_LOOP, then VAL is obtained from DOL.
OMSL	Output Mode Select	This field has either the value SUPERVISORY or CLOSED_LOOP. DOL is used to determine VAL only if OMSL has the value CLOSED_LOOP. By setting this field the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the VAL field cannot be set via dbPuts.
OIF	Output Full or Incremental (ao record only)	This field, which is only used when input is obtained from DOL, determines if the value obtained from DOL is an increment to add to the current VAL or is the actual VAL desired.
SIMM	Simulation Mode	This field has either the value YES or NO. By setting this field to YES, the record can be switched into simulation mode of operation. While in simulation mode, output will be written to SIOL instead of OUT.
SIML	Simulation Mode Location	This field can be a constant, a database link, or a channel access link. If SIML is a database or channel access link, then SIMM is read from SIML. If SIML is a constant link then SIMM is initialized with the constant value but can be changed via dbPuts.
SIOL	Simulation Value Location	This field can be a constant, a database link, or a channel access link. If SIOL is a database or channel access link, then the output value is written to SIOL. If this link is a constant, the result is no output.
SIMS	Simulation Mode Alarm Severity	When this record is in simulation mode, it will be put into alarm with this severity and a status of SIMM_ALARM.
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. IVOA can be one of the following actions. • Continue normally • Don't drive outputs • Set output equal to IVOV

Name	Summary	Description
IVOV	Invalid Alarm Output Value, In Engineering Units	When new severity has been set to INVALID alarm and IVOA is "Set output equal to IVOV", then, VAL is set to IVOV and converted to RVAL before device support is called.

Soft Output

Normally two soft output device support modules are provided Soft and Raw Soft. Both allow the output link OUT to be a constant, a database link, or a channel access link. It is normally meaningless to use constant output links. The Soft support module writes output from the value associated with OVAL or VAL (if OVAL does not exist). The Raw Soft support module writes the value associated with the RVAL field after conversion has been performed.

The device support write routine normally calls recGblPutLinkValue which performs the following steps:

- If the OUT link type is CONSTANT recGblPutLinkValue does nothing and returns with a status of zero.
- If the OUT link type is DB_LINK, then dbPutLink is called to write the current value. If dbPutLink returns an error, a LINK_ALARM with a severity of INVALID_ALARM is raised. RecGblPutLinkValue returns the status of dbPutLink.
- If the OUT link type is CA_LINK, then dbCaPutLink is called to write the current value. If dbCaPutLink returns an error, a LINK_ALARM with a severity of INVALID_ALARM is raised. RecGblPutLinkValue returns the status of dbCaPutLink.

The device support write routine normally returns the status of recGblPutLinkValue.

Output Mode Select

The fields DOL and OMSL are used to allow the output record to be part of a closed loop control algorithm. OMSL is meaningful only if DOL refers to a database or channel access link. It can have the values SUPERVISORY or CLOSED_LOOP. If the mode is SUPERVISORY, then nothing is done to VAL. If the mode is CLOSED_LOOP and the record type does not contain an OIF field, then each time the record is processed, VAL is set equal to the value obtained from the location referenced by DOL. If the mode is CLOSED_LOOP in record types with an OIF field and OIF is Full, VAL is set equal to the value obtained from the location referenced by DOL; if OIF is Incremental VAL is incremented by the value obtained from DOL.

Simulation Mode

An output record can be switched into simulation mode of operation by setting the value of SIMM to YES. During simulation, the record will be put into alarm with a severity of SIMS and a status of SIMM_ALARM. While in simulation mode, output values, in engineering units, will be written to SIOL instead of OUT. Also, while the record is in simulation mode, there will be no calls to device support during record processing.

Normally output records contain a private writeValue routine which performs the following steps:

- If PACT is TRUE, the device support write routine is called, status is set to its return code, and readValue returns.
- Call recGblGetLinkValue to get a new value for SIMM if SIML is a DB_LINK or a CA_LINK.
- Check value of SIMM.

- If SIMM is NO, then call the device support write routine, set status to its return code, and return.
- If SIMM is YES, then call recGblPutLinkValue to write the output value from VAL or OVAL to SIOL. Set alarm status to SIMM and severity to SIMS, if SIMS is greater than zero. Set status to the return code from recGblPutLinkValue and return.
- If SIMM not one of the above, a SOFT alarm with a severity of INVALID is raised, and return status is set to -1.

Invalid Alarm Output Action

Whenever an output record is put into INVALID alarm severity, IVOA specifies an action to take. The record support process routine for each output record contains code which performs the following steps.

- If new severity is less than INVALID, then call writeValue:
- Else do the following:
  - If IVOA is CONTINUE, then call writeValue.
  - If IVOA is NO_OUTPUT, then do not write output.
  - If IVOA is OUTPUT_IVOV, then set VAL to IVOV, call convert if necessary, and then call writeValue.
  - If IVOA not one of the above, an error message is generated.

Chapter 4: ai - Analog Input

1. Introduction

The normal use for this record type is to obtain an analog value converted to engineering units. Most device support modules obtain values from hardware. Soft device modules are provided to obtain input via database or channel access links or via `dbPutField` or `dbPutLink` requests. The record supports alarm limits, conversion to engineering units, smoothing, and graphics and control limits.

Two soft device support modules are provided. One reads values directly into `VAL`. The other reads values into `RVAL`, which is then converted just like raw values obtained from hardware device support modules. If soft device support with a constant `INP` link is chosen, then the `VAL` field can be modified via `dbPuts`.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	Yes
INP	INLINK	Yes	0	No	No	N/A	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
LINR	CVTCHOICE	Yes	0	Yes	Yes	No	Yes
EGUF	FLOAT	Yes	0	Yes	Yes	No	Yes
EGUL	FLOAT	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
EGU	STRING	Yes	null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
AOFF	FLOAT	Yes	0	Yes	Yes	No	Yes
ASLO	FLOAT	Yes	1	Yes	Yes	No	Yes
SMOO	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
ROFF	LONG	No	0	Yes	Yes	No	Yes
ESLO	DOUBLE	No	1	Yes	No	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No
PBRK	NOACCESS	No	4	No	No		No
INIT	SHORT	No	0	Yes	No	No	No
LBRK	SHORT	No	0	Yes	No	No	No
RVAL	LONG	No	0	Yes	Yes	Yes	Yes
ORAW	LONG	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	DOUBLE	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	Unless `INP` is a constant link and the device support module specifies no conversion, this is the value resulting from the record being processed. If `INP` is a constant, then `VAL` is initialized to the `INP` value but can be changed dynamically via `dbPutField` or `dbPutLink`.
INP	Input Link	This field is used by the device support routines to obtain input. For soft analog records it can be a constant, a database link, or a channel access link.
PREC	Display Precision	Precision with which to display `VAL` and `OVAL`. This field is used by record support to supply a value when `get_precision` is called.
LINR	Conversion Type	No conversion, linear and breakpoint table conversion are supported.
EGUF	Engineering Units Full	These fields are used to perform linear conversions. It is the responsibility of the device support routines to use `EGUF` and `EGUL` to compute `ESLO` and `ROFF`. `EGUF` and `EGUL` must be set by the user to the engineering units corresponding to the high and low ADC limits. For example if the ADC has a range of -10 to +10 Volts, then `EGUF` must be the engineering units value corresponding to 10 volts and `EGUL` to -10 volts. If a linear conversion is specified, `recAi` uses `ESLO`, `ROFF`, and `EGUL` to convert the raw value to engineering units according to the formula: `VAL = (RVAL + ROFF)*ESLO + EGUL`
EGUL	Engineering Units Low	
ROFF	Raw Value Offset	
ESLO	Slope for Linear Conversions	
EGU	Engineering Units	An ASCII string of up to 16 characters describing the engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
AOFF	Adjustment Offset	These fields are adjustment parameters for the raw input values. They are applied to the raw data value returned by the device support routine before any other conversions are performed.
ASLO	Adjustment Slope	
SMOO	Smoothing Factor	The converted data value is subjected to the following algorithm: `val = newvalue * (1 - smoo) + oldvalue * smoo` `SMOO` should have a value between 0 and 1, with 0 meaning no smoothing and 1 meaning ultimate smoothing (in fact, the data value will never change).

Name	Summary	Description
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Hihi Alarm Severity	
HSV	High Alarm Severity	
LSV	Low Alarm Severity	
LLSV	Lolo Alarm Severity	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitors by a call to `db_post_event` or monitor callbacks, i.e. callbacks specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
LALM	Last Alarm Monitor Trigger Value	These fields are used to implement the hysteresis factors for monitor callbacks.
ALST	Last Archiver Monitor Trigger Value	
MLST	Last Value Change Monitor Trigger Value	
INIT	Initialize	This field is used by record support to perform initialization for `LBRK` and for smoothing.
LBRK	Last Breakpoint	Index of last breakpoint interval. `LBRK` is used to perform conversions via breakpoint tables.
PBRK	Address of Breakpoint Table	`PBRK` is used to perform conversions via breakpoint tables.
RVAL	Raw Value	`RVAL` is the raw data value obtained by the device support routine. Unless the device support routine returns value requests that no conversion should be performed, the record support routine converts this value to engineering units.
ORAW	Old Raw Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are triggered for changes in `VAL`.

Name	Summary	Description
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3 Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a device support `read_ai` routine is defined. If either does not exist, an error message is issued and processing is terminated.

`INIT` is then set to `TRUE`.

If device support includes `init_record`, it is called.

process

See next section.

special

The only special processing for analog input records is `SPC_LINCONV`. which is invoked whenever any of the fields `LINR`, `EGUF`, `EGUL` or `ROFF` is changed.

If the device support routine `special_linconv` exists, it is called.

`INIT` is set `TRUE`. This causes `PBRK`, `LBRK`, and smoothing to be reinitialized.

get_value

Fills in the values of the structure `valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_precision

Retrieves `PREC`.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. readValue is called. See Chapter 3 Section "Simulation Mode" on page 11 for details.
3. If PACT has been changed to TRUE, the device support read routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving PACT TRUE.
4. PACT is then set to TRUE, TIME is set to tslocaltime and the return status value of readValue is checked. convert is called only if status is 0. If status is 2, then convert is not called, but status is reset to 0.
5. convert (if necessary): The new raw data value is expected to be in field RVAL. The first step is to set val equal to RVAL + ROFF. The next step is to adjust the raw value via the equation:
   ```
   val = val* ASLO + AOFF
   ```
   If the conversion algorithm is linear, the raw value is converted via the equation:
   ```
   val = val * ESLO + EGUL
   ```
   If the conversion is via a breakpoint table, the new value is obtained.
   The next step is to apply the following smoothing algorithm:
   ```
   if SMOO  equal to 0. or  INIT is True, VAL = val
   else VAL = val * (1 - SMOO) + Previous_value * SMOO
   ```
   Since VAL is now defined, the last step is to set UDF to FALSE.
6. Check alarms: This routine checks to see if the new VAL causes the alarm status and severity to change. If so, NSEV, NSTA and LALM are set. It also honors the alarm hysteresis factor (HYST). Thus the value must change by more than HYST before the alarm status and severity is lowered.
7. Check to see if monitors should be invoked:
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if ADEL and MDEL conditions are met.

- Monitors for RVAL are checked whenever other monitors are invoked.
- NSEV and NSTA are reset to 0.

8. Scan forward link if necessary, set PACT and INIT to FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each analog input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw analog input value whenever read_ai is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2 Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
VAL	Value	This field is used by device support only if it obtains a value already converted to engineering units. See RVAL below.
INP	Input Link	This field is used by the device support routines to locate its input.
EGUF	Engineering Units Full	These fields are used to calculate ESLO. Note that these fields correspond to the high and low hardware limits.
EGUL	Engineering Unit Low	
ESLO	Slope	These fields are used for linear conversions from raw to engineering units. The device support routines must calculate these fields unless they obtain values already in engineering units.
ROFF	Raw Offset	
RVAL	Raw Value	It is the responsibility of the device support routine to give this field a value. If the device support routine obtains a value already in engineering units, it should place the value in VAL and return a value of 2.

Device Support Routines

Device support consists of the following routines:

report

```
report (FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record (precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info (int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_ai

```
read_ai(precord)
```

This routine must provide a new input value. Asynchronous device support routines will return with `PACT` set to `TRUE`. If `PACT` is `TRUE`, the process routine will just return and not continue processing. When the asynchronous routine completes, it can call process which will again call `read_ai`. Because `PACT` is still `TRUE` `read_ai` knows that this is a request to retrieve the data obtained by the previous call. When finished, `read_ai` should set `PACT` to `FALSE` and return one the following values:

- **0**: Success. A new raw value is placed in `RVAL`. `convert` will be called.
- **2**: Success, but don't call `convert`. This is useful if `read_ai` obtains a value already converted to engineering units or in the event a hardware failure is detected.
- **Other**: Error.

special_linconv

```
special_linconv(precord,after)
```

This routine is called whenever any of the fields `LINR`, `EGUF`, `EGUL` or `ROFF` is modified.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for input records not related to actual hardware devices. The `INP` link type must be either `CONSTANT`, `DB_LINK` or `CA_LINK`.

Soft Channel

This module places a value directly in `VAL`. `read_ai` always returns a value of 2, which means that no conversion will ever be attempted.

If the `INP` link type is constant, then the constant value is stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_ai` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3 Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero, then `read_ai` sets `UDF` to `FALSE`. The status of `recGblGetLinkValue` is returned.

If soft support is chosen, the following fields become meaningless: `LINR`, `EGUF`, `EGUL`, `ESLO`, `ROFF`, `AOFF`, `ASLO`, and `SMOO`. The `read_ai` routine always returns a value of 2 which means don't convert.

Raw Soft Channel

This module is like the previous except that it places its value in `RVAL` and `read_ai` returns a value of 0. Thus the record processing routine will convert the raw value in the normal way.

If raw soft support is chosen, the fields `EGUF` and `EGUL` become meaningless. `ESLO` and `ROFF` always have their default values of 1 and 0.

Chapter 5: ao - Analog Output

1. Introduction

The normal use for this record type is to store values to be sent to Digital to Analog Converters. It can also be used to write values to other records via database or channel access links. The `OUT` field determines how the record is used. The record supports alarm limits, conversion from/to engineering units, and graphics and control limits.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	Yes
OVAL	DOUBLE	No	0	Yes	Yes	Yes	No
OUT	OUTLINK	Yes	0	No	No	N/A	No
OROC	FLOAT	Yes	0	Yes	Yes	No	No
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
OIF	RECCHOICE	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
LINR	CVTCHOICE	Yes	0	Yes	Yes	No	Yes
EGUF	FLOAT	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
EGUL	FLOAT	Yes	0	Yes	Yes	No	Yes
EGU	STRING	Yes	null	Yes	Yes	No	No
ESLO	DOUBLE	No	1	Yes	No	No	No
ROFF	LONG	No	0	Yes	Yes	No	Yes
DRVH	FLOAT	Yes	0	Yes	Yes	No	Yes
DRVL	FLOAT	Yes	0	Yes	Yes	No	Yes
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
AOFF	FLOAT	Yes	0	Yes	Yes	No	Yes
ASLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
RVAL	LONG	No	0	Yes	Yes	Yes	Yes
ORAW	LONG	No	0	Yes	No	No	No
RBV	LONG	No	0	Yes	No	Yes	No
ORBV	LONG	No	0	Yes	No	No	No
PVAL	DOUBLE	No	0	Yes	No	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No
PBRK	NOACCESS	No	4	No	No		No
INIT	SHORT	No	0	Yes	No	No	No
LBRK	SHORT	No	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	DOUBLE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value	This is the desired output value, in engineering units. If DRVH and DRVL are defined, VAL is forced to be within the drive limits. VAL is either obtained from DOL or set via dbPuts.
OVAL	Output Value	This is the desired output value, after adjustments, in engineering units. It is just VAL possibly adjusted by OROC. This is the value used to compute RVAL. OVAL is used to enforce a maximum rate of change limit before converting the desired value to a raw value. If soft device support is selected and OUT is a database or channel access link, this is the value written to the link.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
OROC	Maximum Output Rate of Change	If this is not zero, it specifies the maximum change in value (engineering units) to be sent to OUT each time the record is processed. It is this field that can cause VAL and OVAL to differ.
DOL	Desired Output Location (an Input Link)	If DOL is a database or channel access link and OMSL is CLOSED_LOOP, then VAL is read from DOL. After the check for drive limits, VAL will be set to the value determined by DOL.
OMSL	Output Mode Select	This field has either the value SUPERVISORY or CLOSED_LOOP. DOL is used to determine VAL only if OMSL has the value CLOSED_LOOP. By setting this field the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the VAL field cannot be set via dbPuts.
OIF	Out Full or Incremental	This field is used when input is obtained from DOL, and determines if the value obtained from DOL is an increment to add to the current VAL or is the actual VAL desired.
PREC	Display Precision	Precision with which to display VAL. This field is used by record support to supply a value when get_precision is called.
LINR	Conversion Type	No conversion, linear and breakpoint table conversion are supported.

Name	Summary	Description
EGUF	Engineering Units Full	These fields are used to perform linear conversions. It is the responsibility of the device support routines to use `EGUF` and `EGUL` to compute `ESLO` and `ROFF`. `EGUF` and `EGUL` must be set by the user to the engineering units corresponding to the high and low ADC limits. For example if the DAC has a range of -10 to +10 Volts, then `EGUF` must be the engineering units value corresponding to 10 volts and `EGUL` to -10 volts. If a linear conversion is specified `ESLO`, `ROFF`, and `EGUL` are used to convert the value from/to engineering units using the following formula: `RVAL = (OVAL - EGUL)/ESLO - ROFF`
EGUL	Engineering Units Low	
ESLO	Slope For Linear Conversions	
ROFF	Raw Value Offset	
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
DRVH	Drive High	If these values are defined then `VAL` will forced to be in the range: `DRVL<=VAL<=DRVH`
DRVL	Drive Low	
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`. If these values are defined, they must be in the range: `DRVL<=LOPR<=HOPR<=DRVH.`
LOPR	Low Operating Range	
AOFF	Adjustment Offset	These fields are adjustment parameters for the raw output values. They are applied to the raw output value after conversion from engineering units.
ASLO	Adjustment Slope	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Hihi Alarm Severity	
HSV	High Alarm Severity	
LSV	Low Alarm Severity	
LLSV	Lolo Alarm Severity	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. callbacks specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
RVAL	Raw Data Value	`RVAL` is the value actually sent to the device.
ORAW	Old raw data value	`ORAW` is used to decide if monitors should be triggered for `RVAL`.

Name	Summary	Description
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value.
ORBV	Old read back value	`ORBV` is used to decide if monitors should be triggered for `RBV` at the same time monitors are triggered for changes in `VAL`.
PVAL	Previous Data Value	
LALM	Last Alarm Monitor Trigger Value	These fields are used to implement the hysteresis factors for monitors.
ALST	Last Archiver Monitor Trigger Value	
MLST	Last Value Change Monitor Trigger Value	
INIT	Initialize	This field is used by record support to perform initialization for `LBRK` and for smoothing.
LBRK	Last Breakpoint	Index of last breakpoint interval
PBRK	Breakpoint Pointer	Address of breakpoint table
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information.
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

This routine initializes `SIMM` if `SIML` is a constant or creates a channel access link if `SIML` is `PV_LINK`. If `SIOL` is `PV_LINK` a channel access link is created.

This routine next checks to see that device support is available. If `DOL` is a constant, then `VAL` is initialized with its value and `UDF` is set to `FALSE`.

The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

`INIT` is set `TRUE`. This causes `PBRK`, `LBRK`, and smoothing to be reinitialized

If linear conversion is requested, then `VAL` is computed from `RVAL` using the algorithm:

```
VAL=(RVAL+ROFF)/ESLO+EGUL
```

and `UDF` is set to `FALSE`.

For breakpoint conversion, a call is made to `cvtEngToRawBpt` and `UDF` is then set to `FALSE`. `PVAL` is set to `VAL`.

process

See next section.

special

The only special processing for analog output records is `SPC_LINCONV` which is invoked whenever either of the fields `LINR`, `EGUF`, `EGUL` or `ROFF` is changed

If the device support routine `special_linconv` exists it is called.

`INIT` is set `TRUE`. This causes `PBRK`, `LBRK`, and smoothing to be reinitialized.

get_value

Fills in the values of `struct valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_precision

Retrieves `PREC`.

get_graphic_double

Sets the upper display and lower display limits for the field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double

Sets the upper display and lower control limits for the field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double

Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. Check `PACT`: If `PACT` is `FALSE` call `fetch_values` and convert which perform the following steps:
   - `fetch_values`:
     if `DOL` is `DB_LINK` and `OMSL` is `CLOSED_LOOP` get `value` from `DOL`
     if `OIF` is `INCREMENTAL` then set `value` = `value` + `VAL`
     else `value` = `VAL`
   - `convert`:
     If Drive limits are defined force `value` to be within limits
     Set `VAL` equal to `value`
     Set `UDF` to `FALSE`.
     If `OVAL` is undefined set it equal to `value`
     If `OROC` is defined and not 0 make |`value-OVAL`| <=`OROC`
     Set `OVAL` equal to `value`
     Compute `RVAL` from `OVAL`. using linear or break point table conversion. For linear conversions the algorithm is:
     ```
     RVAL = (OVAL-EGUL)/ESLO -ROFF
     ```
   - For break point table conversion a call is made to `cvtEngToRawBpt`.
3. Check alarms: This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `y` are set. It also honors the alarm hysteresis factor (`HYST`). Thus the value must change by at least `HYST` before the alarm status and severity is reduced.
4. Check severity and write the new value. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for details.
5. If `PACT` has been changed to `TRUE`, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
6. Check to see if monitors should be invoked:
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `ADEL` and `MDEL` conditions are met.
   - Monitors for `RVAL` and for `RBV` are checked whenever other monitors are invoked.
   - `NSEV` and `NSTA` are reset to 0.
7. Scan forward link if necessary, set `PACT` and `INIT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each analog output record must have an associated set of device support routines. The primary responsibility of the device support routines is to output a new value whenever `write_ao` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
OUT	Output Link	This field is used by the device support routines to locate its output.
EGUF	Engineering Units Full	These fields are used to calculate `ESLO`. Note that these fields correspond to the high and low hardware limits.
EGUL	Engineering Unit Low	
ESLO	Slope	These fields are used for linear conversions from raw to engineering units. The device support routines must calculate these fields unless they obtain values already in engineering units.
ROFF	Raw Offset	
RVAL	Raw Value	This is the value to write `OUT`.

Device Support routines

Device support consists of the following routines:

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. It returns a zero for success or a 2 for success, don't convert.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_ao

```
write_ao(precord)
```

This routine must output a new value. Asynchronous device support routines will return with PACT set to TRUE. If PACT is TRUE, the process routine will just return and not continue processing. When the asynchronous routine completes, it can call process which will again call write_ao. When finished, write_ao should set PACT to FALSE and return one the following values:

- **0**: Success.
- **other**: Error.

special_linconv

```
special_linconv(precord,after)
```

This routine is called whenever either of the fields LINR, EGUF, EGUL or ROFF is modified.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for output records not related to actual hardware devices. The OUT link type must be either a CONSTANT, DB_LINK, or CA_LINK.

Soft Channel

This module writes the current value of OVAL.

If the OUT link type is PV_LINK, then dbCaAddInlink is called by init_record. init_record always returns a value of 2, which means that no conversion will ever be attempted.

write_ao calls recGblPutLinkValue to write the current value of VAL. See Chapter 3, Section "Soft Output" on page 13 for details.

Raw Soft Channel

This module is like the previous except that it writes the current value of RVAL.

Chapter 6: bi - Binary Input

1. Introduction

The normal use for this record type is to obtain a binary value, i.e. a value that is 0 or 1. Most device support modules obtain values from hardware and place the value in `RVAL`. For these devices record processing sets `VAL` = (0,1) if `RVAL` is (0, not 0). Devices may optionally read a value directly into `VAL`. Soft device modules are provided to obtain input via database or channel access links or via `dbPutField` or `dbPutLink` requests. Two soft device support modules are provided. One allows `VAL` to be an arbitrary unsigned short integer. The other reads the value into `RVAL` just like normal hardware modules.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	ENUM	No	0	Yes	Yes	Yes	Yes
INP	INLINK	Yes	0	No	No	N/A	No
ZSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
OSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
COSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ZNAM	STRING	Yes	Null	Yes	Yes	No	Yes
ONAM	STRING	Yes	Null	Yes	Yes	No	Yes
RVAL	ULONG	No	0	Yes	Yes	Yes	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
ORAW	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	compute	Yes	No	No	No
LALM	USHORT	No	0	Yes	No	No	No
MLST	USHORT	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	USHORT	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is the value resulting from record processing unless soft device support with a constant `INP` is chosen. If the later is chosen, `VAL`, which is an unsigned short, is given values via `dbPuts`.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
ZSV	Zero Severity	Alarm Severity for state zero.
OSV	One Severity	Alarm Severity for state one.
COSV	Change of State Severity	Alarm Severity for change of state.
ZNAM	Zero Name	ASCII string defining state zero.
ONAM	One Name	ASCII string defining state one.
RVAL	Raw Value	`RVAL` is the value obtained by the device support routine.
ORAW	Old Raw Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are triggered for changes in `VAL`.
MASK	Hardware mask	
LALM	Last Alarmed Value	Value when last change of state alarm was issued.
MLST	Last Monitored Value	Value when last monitor for value changes was triggered.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

process See next section.

get_value Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_enum_str Retrieves ASCII string corresponding to `VAL`.

get_enum_strs Retrieves ASCII strings for `ZNAM` and `ONAM`.

put_enum_str Checks if string matches `ZNAM` or `ONAM`, and if it does, sets `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
4. Convert
```
status=read_bi
PACT = TRUE
TIME = tslocaltime
if status is 0, then set VAL=(0,1) if RVAL is (0, not 0) and UDF = False
if status is 2, set status = 0
```
5. Check alarms: This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV` and `NSTA` and `LALM` are set. Note that if `VAL` is greater than 1, no checking is performed.
6. Check to see if monitors should be invoked:
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `MLST` is not equal to `VAL`.
   - Monitors for `RVAL` are checked whenever other monitors are invoked.
   - `NSEV` and `NSTA` are reset to 0.
7. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw input value whenever `read_bi` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
VAL	Value Field	This field is set by a device support routines only if it doesn't want record support to set it.
INP	Input Link	This field is used by the device support routines to locate its input.
RVAL	Raw Value	It is the responsibility of the device support routine to give this field a value.
MASK	Hardware mask.	The device support routine must give this field a value if it needs to use it.

Device Support routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_bi

```
read_bi(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success. A new raw value is placed in RVAL. The record support module forces VAL to be (0,1) if RVAL is (0, not 0).
- **2**: Success, but don't modify VAL.
- **other**: Error.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for input records not related to actual hardware devices. The INP link type must be either CONSTANT, DB_LINK, or CA_LINK.

Soft Channel

read_bi always returns a value of 2, which means that no conversion is performed.

If the INP link type is constant, then the constant value is stored into VAL by init_record, and UDF is set to FALSE. VAL can be changed via dbPut requests. If the INP link type is PV_LINK, then dbCaAddInlink is called by init_record.

read_bi calls recGblGetLinkValue to read the current value of VAL. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of recGblGetLinkValue is zero, then read_bi sets UDF to FALSE. The status of recGblGetLinkValue is returned.

Raw Soft Channel

This module is like the previous except that values are read into RVAL. read_bi returns a value of 0. Thus the record processing routine will force VAL to be 0 or 1.

Chapter 7: bo - Binary Output

1. Introduction

The normal use for this record type is to store a binary (0 or 1) value to be sent to a Digital Output module. It can also be used to write binary values into other records via database or channel access links.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	ENUM	No	0	Yes	Yes	Yes	Yes
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
OUT	OUTLINK	Yes	0	No	No	N/A	No
DOL	INLINK	Yes	0	No	No	N/A	No
HIGH	FLOAT	Yes	0	Yes	Yes	No	No
ZNAM	STRING	Yes	Null	Yes	Yes	No	Yes
ONAM	STRING	Yes	Null	Yes	Yes	No	Yes
RVAL	ULONG	No	0	Yes	Yes	Yes	Yes
ORAW	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	compute	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
RPVT	NOACCESS	No	0	No	No		No
WDPT	NOACCESS	No	0	No	No		No
ZSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
OSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
COSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
RBV	ULONG	No	0	Yes	No	Yes	No
ORBV	ULONG	No	0	Yes	No	No	No
MLST	USHORT	No	0	Yes	No	No	No
LALM	USHORT	No	0	Yes	No	No	No
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	USHORT	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is the value to be sent to OUT. It is either obtained from DOL or else given a value via dbPuts.
OMSL	Output Mode Select	This field has either the value SUPERVISORY or CLOSED_LOOP. DOL is used to determine VAL only if OMSL has the value CLOSED_LOOP. By setting this field the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the VAL field cannot be set via dbPuts.
DOL	Desired Output Location (Input Link)	If DOL is a database or channel access link and OMSL is CLOSED_LOOP, then VAL is read from DOL.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
HIGH	Seconds to Hold High	If this value is greater than zero, then whenever VAL is set equal to 1, it is reset to zero after HIGH seconds.
ZNAM	Zero Name	ASCII string defining state zero.
ONAM	One Name	ASCII string defining state one.
RVAL	Raw Data Value	RVAL is the value written by the device support routine. If MASK is set by the device support routine, RVAL is computed by record support.
ORAW	Old Raw Data Value	ORAW is used to decide if monitors should be triggered for RVAL at the same time monitors are triggered for changes in VAL.
MASK	Hardware Mask	This value can be set by the device support routine. It is the value sent to the hardware when VAL is not zero.
RPVT	Record Private	
WDPT	Watchdog Pointer	Private field for honoring second to hold HIGH.
ZSV	Zero Severity	Alarm Severity for state zero.
OSV	One Severity	Alarm Severity for state one.
COSV	Change of State Severity	Alarm Severity for change of state.
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value.
ORBV	Old Read Back Value	ORBV is used to decide if monitors should be triggered for RBV at the same time monitors are triggered for changes in VAL.

Name	Summary	Description
MLST	Monitor Last	Value when last monitor for value changes was triggered
LALM	Last Alarmed	Value when last change of state alarm was issued.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information.
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

This routine initializes `SIMM` if `SIML` is a constant or creates a channel access link if `SIML` is `PV_LINK`. If `SIOL` is `PV_LINK` a channel access link is created .

This routine next checks to see that device support is available. The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If `DOL` is a constant, then `VAL` is initialized to 1 if its value is nonzero or initialized to 0 if `DOL` is zero, and `UDF` is set to `FALSE`.

If device support includes `init_record`, it is called. `VAL` is set using `RVAL`, and `UDF` is set to `FALSE`.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_enum_str

Retrieves ASCII string corresponding to `VAL`.

get_enum_strs

Retrieves ASCII strings for `ZNAM` and `ONAM`.

put_enum_str Checks if string matches ZNAM or ONAM, and if it does, sets VAL.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If PACT is FALSE
   - if DOL is DB_LINK and OMSL is CLOSED_LOOP
     - get value from DOL
     - check for link alarm
     - force VAL to be 0 or 1
     - if MASK is defined
       - if VAL is 0 set RVAL = 0
     - else set RVAL = MASK
3. Check alarms: This routine checks to see if the new VAL causes the alarm status and severity to change. If so, NSEV, NSTA and LALM are set.
4. Check severity and write the new value. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for details.
5. If PACT has been changed to TRUE, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving PACT TRUE.
6. Check WAIT. If VAL is 1 and WAIT is greater than 0, process again with a VAL=0 after WAIT seconds.
7. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if MLST is not equal to VAL.
   - Monitors for RVAL and for RBV are checked whenever other monitors are invoked.
   - NSEV and NSTA are reset to 0.
8. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each binary output record must have an associated set of device support routines. The primary responsibility of the device support routines is to write a new value whenever `write_bo` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
VAL	Value Field	This field is only of interest to device support routines that do not use `MASK` and `RVAL`.
OUT	Output Link	This field is used by the device support routines to locate its output.
RVAL	Raw Data Value	If `MASK` is defined then record support sets `RVAL`=(0,`MASK`) if `VAL` is (0, not zero).
MASK	Hardware mask.	The device support module must set this field. Not that if `VAL` is 1, then record processing sets `RVAL` = `MASK`.
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value.

Device Support routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. It should determine `MASK` if it is needed.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_bo

```
write_bo(precord)
```

This routine must output an new value. It returns the following values:

- **0**: Success.
- **other**: Error.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for output records not related to actual hardware devices. The `OUT` link type must be either a `CONSTANT`, `DB_LINK`, or `CA_LINK`.

Soft Channel

This module writes the current value of `VAL`.

If the `OUT` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`. `init_record` always returns a value of 2, which means that no conversion will ever be attempted.

`write_bo` calls `recGblPutLinkValue` to write the current value of `VAL`. See Chapter 3, Section "Soft Output" on page 13 for details.

Raw Soft Channel

This module is like the previous except that it writes the current value of `RVAL`.

Chapter 8: calc - Calculation

1. Introduction

This record calculates an expression.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	No
CALC	STRING	Yes	Null	Yes	Yes	Yes	Yes
INPA	INLINK	Yes	0	No	No	N/A	No
INPB	INLINK	Yes	0	No	No	N/A	No
INPC	INLINK	Yes	0	No	No	N/A	No
INPD	INLINK	Yes	0	No	No	N/A	No
INPE	INLINK	Yes	0	No	No	N/A	No
INPF	INLINK	Yes	0	No	No	N/A	No
INPG	INLINK	Yes	0	No	No	N/A	No
INPH	INLINK	Yes	0	No	No	N/A	No
INPI	INLINK	Yes	0	No	No	N/A	No
INPJ	INLINK	Yes	0	No	No	N/A	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
INPK	INLINK	Yes	0	No	No	N/A	No
INPL	INLINK	Yes	0	No	No	N/A	No
A	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
B	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
C	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
D	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
E	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
F	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
G	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
H	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
I	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
J	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
K	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
L	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
LA	DOUBLE	No	0	Yes	No	No	No
LB	DOUBLE	No	0	Yes	No	No	No
LC	DOUBLE	No	0	Yes	No	No	No
LD	DOUBLE	No	0	Yes	No	No	No
LE	DOUBLE	No	0	Yes	No	No	No
LF	DOUBLE	No	0	Yes	No	No	No
LG	DOUBLE	No	0	Yes	No	No	No
LH	DOUBLE	No	0	Yes	No	No	No
LI	DOUBLE	No	0	Yes	No	No	No
LJ	DOUBLE	No	0	Yes	No	No	No
LK	DOUBLE	No	0	Yes	No	No	No
LL	DOUBLE	No	0	Yes	No	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No
RPCL	NOACCESS	No	0	No	No		No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is calculated, via the `CALC` expression, each time the record is processed.
CALC	Infix Expression	See below for details
INPA,...,INPL	Input Links	Each may be a constant, a database link, or a channel access link. Any link not defined is ignored.
A,...,L	Input Values	If the corresponding `INP` field is a constant, this field is initialized with the constant value but can be changed via `dbPuts`.
LA,...,LL	Previous Input Values	These fields are used to decide when to trigger monitors on A,...,L.
EGU	Engineering Units	A 16 character ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Severity for a Hihi Alarm	
HSV	Severity for a High Alarm	
LSV	Severity for a Low Alarm	
LLSV	Severity for a Lolo Alarm	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	

Name	Summary	Description
LALM	Last Alarmed Value	Values when monitors were last triggered. These fields are used to implement the hysteresis factors for monitors.
ALST	Archive Last Value	
MLST	Monitor Last Value	
RPCL	Expression in reverse polish	

4. Record Support Routines

init_record

For each constant input link, the corresponding value field is initialized with the constant value if the input link is `CONSTANT` or a channel access link is created if the input link is `PV_LINK`.

A routine `postfix` is called to convert the infix expression in `CALC` to reverse polish notation. The result is stored in `RPCL`.

process

See next section.

special

This is called if `CALC` is changed. `special` calls `postfix`.

get_value

Fills in the values of `struct valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_precision

Retrieves `PREC`.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double

Sets the upper control and the lower control limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double

Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine `process` implements the following algorithm:

1. Fetch all arguments.
2. Call routine `calcPerform`, which calculates `VAL` from the `postfix` version of the expression given in `CALC`. If `calcPerform` returns success `UDF` is set to `FALSE`.
3. Check alarms. This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `LALM` are set. It also honors the alarm hysteresis factor (`HYST`). Thus the value must change by at least `HYST` before the alarm status and severity changes.
4. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `ADEL` and `MDEL` conditions are met.
   - Monitors for `A-L` are checked whenever other monitors are invoked.
   - `NSEV` and `NSTA` are reset to 0.
5. Scan forward link if necessary, set `PACT` `FALSE`, and return.

6. Allowed Expressions

The calculation can express algebraic, relational, and logical expressions. The expression is converted to opcode and stored as reverse polish notation in the calculation record. The database fields are as follows:

- **CALC**: infix expression as entered
- **RPCL**: reverse polish expression

The reverse polish calculation is most efficient to evaluate during run-time. The range of expressions supported by the calculation record are separated into operands, algebraic operations, trigonometric, relational operations, logical operations, parenthesis, and the question mark operator.

Operands

- **A**: Use the value specified by input A
- **B**: Use the value specified by input B
- **C**: Use the value specified by input C
- **D**: Use the value specified by input D
- **E**: Use the value specified by input E
- **F**: Use the value specified by input F
- **G**: Use the value specified by input G
- **H**: Use the value specified by input H
- **I**: Use the value specified by input I

- **J**: Use the value specified by input J
- **K**: Use the value specified by input K
- **L**: Use the value specified by input L
- **RNDM**: Random number (unary), random number between 0-1

Algebraic Operators

- **ABS**: Absolute value (unary)
- **SQR**: Square root (unary)
- **MIN**: Minimum (binary function)
- **MAX**: Maximum (binary function)
- **CEIL**: Ceiling (unary)
- **FLOOR**: Floor (unary)
- **LOG**: Log base 10 (unary)
- **LOGE**: Natural log (unary)
- **EXP**: Exponential function (unary)
- **^** : Exponential (binary)
- ****** : Exponential (binary)
- **+** : Addition (binary)
- **-** : Subtraction (binary)
- ***** : Multiplication (binary)
- **/** : Division (binary)
- **%** : Modulo (binary)
- **NOT**: Negate (unary)
- **-** : Subtraction (unary)
- **NINT**: Nearest Integer
- **LN**: Natural Log (synonym for LOGE)
- **SQRT**: Square Root (synonym for SQR)
- **PI**: 3.1415926...
- **D2R**: Conversion from Degrees to Radians (Note: Trig functions assume their arguments are in radians) Degrees = Radians * **D2R**

Trigonometric Operators

- **SIN**: Sine
- **SINH**: Hyperbolic sine
- **ASIN**: Arc sine
- **COS**: Cosine
- **COSH**: Hyperbolic cosine
- **ACOS**: Arc cosine
- **TAN**: Tangent
- **TANH**: Hyperbolic tangent
- **ATAN**: Arc tangent

Relational Operators

- **>=** : Greater than or equal to
- **>** : Greater than
- **<=** : Less than or equal to

- <: Less than
- **#** : Not equal to
- **=** : Equal to

Logical Operators

- **&&** : And
- || : Or
- **!** : Not

Bitwise Operators

- **|** : Bitwise Or
- **&** : Bitwise And
- **OR** : Bitwise Or
- **AND**: Bitwise And
- **XOR**: Bitwise Exclusive Or
- **~** : One's Complement
- **<<** : Left shift
- **>>** : Right shift

Parenthesis and Comma

The open and close parenthesis are supported. Nested parenthesis are supported.

The comma is supported when used to separate the arguments of a binary function.

Conditional Expression

The "C" question mark operator is supported. The format is:

```
(condition)? True result : False result
```

7. Example Expressions

Algebraic

A + B

- Result is A + B

Relational

(A + B) < (C + D)

- Result is 1 if (A+B) < (C+D)
- Result is 0 if (A+B) >= (C+D)

Question Mark

(A+B)<(C+D)?E:F

- Result is E if (A+B) < (C+D)
- Result is F if (A+B) >= (C+D)

(A+B)<(C+D)?E

- Result is E if (A+B) < (C+D)
- Result is unchanged if (A+B) >= (C+D)

Logical

A&B

- Causes the following to occur:
  - Convert A to integer
  - Convert B to integer
  - Bit-wise And A and B
  - Convert result to floating point

Chapter 9: compress - Compression

1. Introduction

The VAL field of this record refers to an array of length NSAM. Unless INP is a database link, the compression algorithm is ignored. If, however, INP is a database link, then this record type supports several algorithms: CIRBUF, AVERAGE, NTO1LOW, NTO1HIGH, and NTO1AVE. Each will be discussed separately.

CIRBUF keeps a circular buffer of length NSAM. Each time the record is processed, it gets the data referenced by INP and puts it into the circular buffer referenced by VAL. Note that when INP refers to a scalar, VAL is just a time ordered circular buffer of values obtained from INP.

If AVERAGE is chosen, then VAL refers to an array of length NSAM that contains an element by element time average of values taken from the array referenced by INP. N successive samples of INP are averaged in order to compute VAL.

If NTO1LOW, NTO1HIGH, or NTO1AVE are chosen, then VAL is a circular buffer of length NSAM. The actual algorithm depends on whether INP references a scalar or an array. If INP refers to a scalar, then N successive time ordered samples of INP are taken. After the Nth sample is obtained a new value, determined by the algorithm (LOW, HIGH, or AVE), is written to the circular buffer referenced by VAL. If INP refers to an array, then each time the record is processed, the array referenced by INP is obtained, divided into sub-arrays each of length N, and the algorithm applied to each sub-array. The result obtained from each subarray is written to the circular buffer referenced by VAL.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	Yes
INP	INLINK	Yes	0	No	No	N/A	No
RES	SHORT	No	0	Yes	Yes	No	No
ALG	RECCHOICE	Yes	0	Yes	No	No	No
NSAM	ULONG	Yes	1	Yes	No	No	No
N	ULONG	Yes	1	Yes	No	No	No
ILIL	FLOAT	Yes	0	Yes	Yes	No	No
IHIL	FLOAT	Yes	0	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
EGU	STRING	Yes	null	Yes	Yes	No	No
OFF	ULONG	No	0	Yes	No	No	No
NUSE	ULONG	No	0	Yes	No	No	No
BPTR	NOACCESS	No	0	No	No		No
SPTR	NOACCESS	No	0	No	No		No
WPTR	NOACCESS	No	0	No	No		No
CVB	DOUBLE	No	0	Yes	No	No	No
INX	ULONG	No	0	Yes	No	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is determined as a result of record processing. It is a double precision array of length `NSAM`.
INP	Input Link	`INP` can be a constant, a database link, or a channel access link. Unless it is a database link, `ALG` is meaningless.
RES	Reset	Setting this field causes the algorithm to start over from the beginning.
ALG	Algorithm	`CIRBUF`, `AVERAGE`, `NTO1LOW`, `NTO1HIGH`, or `NTO1AVE`.

Name	Summary	Description
NSAM	Number in Sample	Number of elements in `VAL`.
N	Number	Value of `N` for `AVERAGE` and `NTO1xxx` algorithms.
ILIL	Initial Low Interest Value	Applies to `NTO1xxx` applied to `INP` arrays.
IHIL	Initial High Interest Value	
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
OFF	Current Offset	
NUSE	Number Used	Number of elements currently stored.
BPTR	Buffer Pointer	Holds array referenced by `VAL`
SPTR	Summing Buffer Pointer	For array averages.
WPTR	Work Buffer Pointer	For `dbGetLinks`.
CVB	Compress Value Buffer	
INX	Current Index of 1,...,N	

4. Record Support Routines

init_record

Space for all necessary arrays is allocated. The addresses are stored in the appropriate fields in the record.

process

See next section.

special

This routine is called when `RSET` is set. It performs a reset.

get_value Fills in the values of struct `valueDes` so that they refer to `VAL`.

cvt_dbaddr This is called by `dbNameToAddr`. It makes the `dbAddr` structure refer to the actual buffer holding the result.

get_array_info Obtains values from the circular buffer referenced by `VAL`.

put_array_info Writes values into the circular buffer referenced by `VAL`.

get_units Retrieves `EGU`.

get_precision Retrieves `PREC`.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

5. Record Processing

Routine `process` implements the following algorithm:

1. If `INP` is not a database link, check monitors and the forward link and return.
2. Get the current data referenced by `INP`.
3. Perform the appropriate algorithm:
   a. `AVERAGE`: Read `N` successive instances of `INP` and perform an element by element average. Until `N` instances have been obtained it just return without checking monitors or the forward link. When `N` instances have been obtained complete the algorithm, store the result in the `VAL` array, check monitors and the forward link, and return.
   b. `CIRBUF`: Write the values obtained from `INP` into the `VAL` array as a circular buffer, check monitors and the forward link, and return.
   c. `NTO1xxx` and `INP` refers to a scalar: Obtain `N` successive values from `INP` and apply the `NTO1xxx` algorithm to these values. Until `N` values are obtained monitors and forward links are not checked. When `N` successive values have been obtained, complete the algorithm, check monitors and the forward link, and return.

d. NTO1xxx and INP refers to an array: The ILIL and IHIL are honored if ILIL<IHIL. The input array is divided into subarrays of length N. The specified NTO1xxx compression algorithm is applied to each subarray and the result stored in the array referenced by VAL. The monitors and forward link are checked.

4. If success, set UDF to FALSE.
5. Check to see if monitors should be invoked:
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

Chapter 10: dfanout

Johnny Tang, Matthew Bickley, and Chip Watson
Continuous Electron Beam Accelerator Facility
Southeastern Universities Research Association

1. Introduction

This record is used to forward data to up to eight other records. It has no associated device support.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	No	Yes
OUTA	OUTLINK	Yes	0	No	No	N/A	No
OUTB	OUTLINK	Yes	0	No	No	N/A	No
OUTC	OUTLINK	Yes	0	No	No	N/A	No
OUTD	OUTLINK	Yes	0	No	No	N/A	No
OUTE	OUTLINK	Yes	0	No	No	N/A	No
OUTF	OUTLINK	Yes	0	No	No	N/A	No
OUTG	OUTLINK	Yes	0	No	No	N/A	No
OUTH	OUTLINK	Yes	0	No	No	N/A	No
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	DOUBLE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is the input value which is used for passing data for all of the output links.
OUTA,..., OUTH	Output Link	On processing of this record, each of these links to which another record is connected will be triggered.
DOL	Desired Output Location (Input Link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`. After the check for drive limits, `VAL` will be set to the value determined by `DOL`.
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`. If these values are defined, they must be in the range: `DRVL<=LOPR<=HOPR<=DRVH.`
LOPR	Low Operating Range	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Hihi Alarm Severity	
HSV	High Alarm Severity	
LSV	Low Alarm Severity	
LLSV	Lolo Alarm Severity	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. callbacks specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	

Name	Summary	Description
LALM	Last Alarm Value	These fields are used to implement the hysteresis factors for monitors.
ALST	Last Archiver Value	
MLST	Last Monitored Value	
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

process

See next section.

5. Record Processing

Routine `process` implements the following algorithm:

1.

Chapter 11: eg - Event Generator

John Winans
Advanced Photon Source
Argonne National Laboratory

1. Introduction

The support for the global event system has been designed to allow application developers to control the APS event generator and receiver boards. This is done by the use of four new record types: `eg`, `egevent`, `er`, `erevent`. These records are customized and are only supported by the device support modules for the APS event generator and receiver boards.

The use of the global event system and its associated records should not be confused with the vanilla EPICS events and the associated `event` records. They are very different.

The Event Generator

The Event Generator is used to generate global event codes and send them out to one or more Event Receivers. A group of interconnected event generators and receivers is referred to as an 'event circuit.' There may be more than one event generator on the same event circuit. And it is possible for a single IOC to be part of multiple event circuits.

EG Records

The `eg` record type is used to select the options of a specific event generator card. In order to properly configure it, you should first be familiar with its operating modes. This is specified in the document "Event System" by Frank Lenksus.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
OUT	OUTLINK	Yes		No	No	No	No
MOD1	RECCHOICE	Yes	0	Yes	Yes	No	Yes
R1SP	DOUBLE	Yes	0	Yes	No	No	Yes
MOD2	RECCHOICE	Yes	0	Yes	Yes	No	Yes
R2SP	DOUBLE	Yes	0	Yes	No	No	Yes
LMD1	RECCHOICE	No	0	Yes	No	No	Yes
LMD2	RECCHOICE	No	0	Yes	No	No	Yes
FIFO	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LFFO	GBLCHOICE	No	0	Yes	No	No	Yes
CLR1	CHAR	No	0	Yes	Yes	No	Yes
CLR2	CHAR	No	0	Yes	Yes	No	Yes
TRG1	CHAR	No	0	Yes	Yes	No	Yes
TRG2	CHAR	No	0	Yes	Yes	No	Yes
ENAB	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LENA	LONG	No	0	Yes	No	No	Yes
TAXI	LONG	No	0	Yes	Yes	Yes	No
LTAX	LONG	No	0	Yes	No	No	No
VME	LONG	No	0	Yes	Yes	No	Yes
ETE0	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET0	LONG	Yes	0	Yes	Yes	No	Yes
LET0	LONG	No	0	Yes	No	No	Yes
ETE1	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET1	LONG	Yes	0	Yes	Yes	No	Yes
LET1	LONG	No	0	Yes	No	No	Yes
ETE2	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET2	LONG	Yes	0	Yes	Yes	No	Yes
LET2	LONG	No	0	Yes	No	No	Yes
ETE3	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET3	LONG	Yes	0	Yes	Yes	No	Yes
LET3	LONG	No	0	Yes	No	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
ETE4	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET4	LONG	Yes	0	Yes	Yes	No	Yes
LET4	LONG	No	0	Yes	No	No	Yes
ETE5	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET5	LONG	Yes	0	Yes	Yes	No	Yes
LET5	LONG	No	0	Yes	No	No	Yes
ETE6	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET6	LONG	Yes	0	Yes	Yes	No	Yes
LET6	LONG	No	0	Yes	No	No	Yes
ETE7	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ET7	LONG	Yes	0	Yes	Yes	No	Yes
LET7	LONG	No	0	Yes	No	No	Yes
VAL	CHAR	No	0	Yes	No	Yes	No

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	Specifies the link number of the event generator board. Only the 'Card' value of the link specification is used.
MOD1	Mode Select for RAM 1	Used to select the operating mode of event sequence `RAM1` or `RAM2`. It is important to know that the configuration of the events in the RAM may not be altered unless it is either '`Off`' or in '`Alternate Mode`'. Should a configuration attempt be made when the RAM is not in one of these modes, it will be deferred until the RAM mode is changed to either '`Off`' or '`Alternate`'. When `MOD1` is set to alternate from any other mode, `MOD2` will also be set to alternate. If `MOD1` is changed from alternate to any other mode, `MOD2` will be set to off.
MOD2	Mode Select for RAM 2	
R1SP	RAM 1 Speed	Event clock 1 rate in Hz. This must be set to the clock rate of the signal source on the Event `CLK` 1 input. It is only used to calculate the 'desired position' value of events that are placed into the sequence RAM. These events are specified by the use of '`egevent`' records. If all '`egevent`' record types that use the generator being configured will be using 'Clock Ticks' as their 'Delay Units' the value placed into `R1SP` is not used and may be left as zero.
R2SP	RAM 2 Speed	
LMD1	Last Operating Mode 1	
LMD2	Last Operating Mode 2	
FIFO	FIFO Enable	Used to enable or disable the input-fifo on the generator board. The fifo is used to allow more than one event generator to exist on the same fiber-optic line.
LFFO	Last FIFO Enable	
CLR1	Clear RAM 1	Performing a 'put' operation on this field causes sequence `RAM1` or `RAM2` to be cleared. The use of this field is undefined (and will cause great problems) if there are any '`egevent`' records configured in the database. This is provided for testing purposes.
CLR2	Clear RAM 2	
TRG1	Manual Trigger RAM 1	If a 'put' operation is performed on this field, a one-time trigger on sequence `RAM1` or `RAM2` will be initiated. The result would be that same as if there were a hardware trigger applied to the 'Event `TRG1`' input on the card.
TRG2	Manual Trigger RAM 2	
ENAB	Master Enable	Master card enable. No events are generated unless the card is enabled. In general, there should never be any reason to disable an event generator. This is provided for testing purposes.
LENA	Last Master Enable	

Name	Summary	Description
TAXI	Taxi Violation Flag	This is set to a non-zero value when there has been a taxi violation. It simply reflects that state of the violation signal on the taxi receiver module. Taxi violations can not occur when `FIFO` is set to '`Off`.'
LTAX	Last Taxi Violation	
VME	Manual Event Generation via VME Access	Used to send out a one-shot event code. A put to this field will cause the event to be sent. It appears to be zero if it is ever read. This can be 'OUT-LINKed' to by an other record in order to generate an arbitrary event when it is processed.
ETEn	Event Trigger Enable	(n=0-7) These are the enables for the trigger event inputs on the card. They must be set to '`On`' in order to send the trigger event codes.
ETn	Event Trigger	(n=0-7) These are used to program the event codes that correspond to the trigger event inputs on the card.
LETn	Last Trigger Event	(n=0-7)
VAL	Value	Not Used.

4. Record Processing

It is intended that `eg` records be set to passive processing only. They are not altered by the device support code in response to being processed. Their purpose is only to specify the operating modes of the event generator and, as such, are processed if and when a value in it is altered by a `put` operation. To start things going, however, they should have their '`process at init`' flag set to `YES`.

As an observation, it is not advisable to alter the `R1SP` or `R2SP` fields. You may, but then all related `egevent` records must be processed again, in order to recalculate their desired position values. The `MOD1` and `MOD2` fields can also cause some nasty side effects if they are altered between any two non-off modes. In general, the operating mode should be set to `OFF` and then to some other mode if it is desired to switch between two modes.

5. Device Support

The device support module for the event generator may be used by `eg` and `egevent` record types.

In order to configure the event generator device support, a call must be made to set the address for each of the event generator cards present in the IOC. This configuration call is as follows:

```
EgConfigure(<card number>, <Base address in A16>)
```

The <card number> field may be 0-4 and is used to specify which card is to be configured. This is the card number that is referenced in the eg and egevent records when building the database. The <Base address in A16> field is a 16-bit number that represents the address of the card in the A16 memory space.

Database records that specify card numbers that are not configured will generate 'bad field' errors when they are initialized by iocInit. And will then be ignored by the event generator device support if ever processed.

6. Event Records

The regular EPICS event records may be used when it is desired to cause a record to process upon a given event code by the global event system. This is accomplished by configuring an event record (the regular EPICS event record) and selecting the APS event receiver for the DTYP field. The card and signal are then used to select the event receiver card and event number, respectively. Any time the event number specified is received by the event receiver, that event record will be processed.

This is the only relationship between the vanilla EPICS event codes (and their associated records) and the global APS event system.

7. Event System Observations

This section describes a few observations that would otherwise be left to experimentation for the user figure out. Much of the annoyances described here have ben left in the system because there are simple work arounds, or they represent situations that should never be encountered on a running system.

Event Generator Sequence RAM Modes

It is intended that the modes of the eg never be altered. It was considered that the MOD1 and MOD2 fields be made SPC_NOMOD fields. But, during the system testing of the event hardware itself, it became useful to be able to make adjustments to the operating mode. Thus the ability to change the mode was put into the record and device support. However, exactly what is done when the mode is changed is probably not useful to the database designer.

First of all, remember that the sequence RAMs can not be updated unless they are either in ALT mode or OFF. This is due to the hardware constraints. In order to alter a sequence RAM that is not set to ALT or OFF, the RAM must be changed to one of those modes, altered, and then reset back to the desired mode. (No it is not reasonable to do this automatically.) Should the mode be carelessly altered, the EG card will have the mode updated, but the sequence RAM(s) will not be updated again until the mode is set to ALT or OFF. (In an actual application program, it is not reasonable to think that the operating modes of the sequence RAMs will be changed.)

Unless you have a strong need to use more than one sequence RAM at the same time, it is strongly recommended that the ALT mode be used. This is so that you may alter the event positions on the fly when debugging.

EGevent Records

The delay selected in an `egevent` record is done by specifying the desired period of delay. The period is converted to click ticks by the use of the `R1SP` and `R2SP` values specified in the `eg` record. Since only one event code may be in any single sequence `RAM` position at any given time, any collisions are resolved at `RAM` load time by scanning for the 'next' unused position. Thus it is possible that the same database end up loading the `RAM`s in two different images depending on the order that the records get processed (that earlier records get higher priority.) If this causes problems, it is recommended that the units of delay be specified in 'clock ticks' and that the same delay values not be used in multiple records.

The use of 'clock ticks' as the delay units specification will eliminate the rounding caused by the conversion from alternate units into clock ticks.

ER Interrupts

It should be obvious that the event system is capable of generating VME interrupts at a rate that far exceeds the CPUs ability to process them. Much care should be put into the design of the databases that control the event system so that this does not happen. It has been observed that when such a problem does occur, vxWorks dies and prints "Work queue overflow" on its console.

8. Example Database for Global Time Synchronization

This section describes a few example database records that are used to set up the event system. The records shown here are those used by the global synchronous timing system. This provides a good example of event generation from database records as well as from trigger inputs. It also includes a heartbeat generator that is required by the event system itself.

Timing System Overview

The example timing system uses a free running 1000 hertz clock. This clock is input on the `TRG0` line of the EG card on the master timing IOC. Each time the `TRG0` input is pulsed, an Increment Time Stamp (`0x7C`) event should be sent out so that the ER cards can update their notion of the time.

Additionally, the timing system has to take care of high order counter truncation and slave resynchronization. This is handled by the use of the Reset Time Stamp Counters (`0x7D`) event (the processing of the time stamp information is described in more detail in the document on the global timing system.)

Event Generator Database Records

The records related to the event generator card are used to initialize and generate events. The `eg` record used on the timing system master is:

```
record(eg, "$(prefix)_eg")
    {
        field(DESC, "")
        field(ASG, "")
        field(SCAN, "Passive")
        field(PINI, "YES")
        field(PHAS, "0")
        field(EVNT, "0")
        field(TSE, "0")
        field(TSEL, "0")
        field(DTYP, "APS event generator G")
        field(DISV, "1")
        field(SDIS, "0")
        field(DISS, "NO_ALARM")
```

```
        field(PRIO, “LOW”)
        field(FLNK, “0”)
        field(OUT, “#C0 S0 @”)
        field(MOD1, “Off”)
        field(R1SP, “0”)
        field(MOD2, “Off”)
        field(R2SP, “0”)
        field(FIFO, “NO”)
        field(ENAB, “YES”)
        field(ETE0, “YES”)
        field(ET0, “0x7c”)
        field(ETE1, “NO”)
        field(ET1, “0x0”)
        field(ETE2, “NO”)
        field(ET2, “0x0”)
        field(ETE3, “NO”)
        field(ET3, “0x0”)
        field(ETE4, “NO”)
        field(ET4, “0x0”)
        field(ETE5, “NO”)
        field(ET5, “0x0”)
        field(ETE6, “NO”)
        field(ET6, “0x0”)
        field(ETE7, “NO”)
        field(ET7, “0x0”)
    }
```

The important items of note are that `ENAB` is `YES`, `ETE0` is `YES`, `ET0` is set to `0x7C`, and that the record be set to process at init time. The scan should always be set to passive since it only makes sense to process the record when the field values change.

In order to take care of the heart beat (`0x7A`) and time stamp reset/resync (`0x7D`) events, `longout` records are used that have their output links pointed to the `VME` field on the above `eg` record. When they are processed, the `VAL` field of the `longout` record is sent out on the event system.

```
record(longout, “$(prefix)_hbeat”)
    {
        field(DESC, “”)
        field(ASG, “”)
        field(SCAN, “1 second”)
        field(PINI, “NO”)
        field(PHAS, “0”)
        field(EVNT, “0”)
        field(TSE, “0”)
        field(TSEL, “0”)
        field(DTYP, “Soft Channel”)
        field(DISV, “1”)
        field(SDIS, “0”)
        field(DISS, “NO_ALARM”)
        field(PRIO, “LOW”)
        field(FLNK, “0”)
        field(OUT, “$(prefix)_eg.VME  PP NMS”)
        field(DOL, “122”)
        field(OMSL, “supervisory”)
        field(EGU, “rocks”)
        field(HOPR, “0”)
        field(LOPR, “0”)
        field(HIHI, “0”)
        field(LOLO, “0”)
        field(HIGH, “0”)
        field(LOW, “0”)
        field(HHSV, “NO_ALARM”)
        field(LLSV, “NO_ALARM”)
        field(HSV, “NO_ALARM”)
        field(LSV, “NO_ALARM”)
```

```
        field(HYST, “0”)
        field(ADEL, “0”)
        field(MDEL, “0”)
        field(SIOL, “0”)
        field(SIML, “0”)
        field(SIMS, “NO_ALARM”)
        field(IVOA, “Continue normally”)
        field(IVOV, “0”)
    }

record(longout, “$(prefix)_resync”)
    {
        field(DESC, “”)
        field(ASG, “”)
        field(SCAN, “10 second”)
        field(PINI, “NO”)
        field(PHAS, “0”)
        field(EVNT, “0”)
        field(TSE, “0”)
        field(TSEL, “0”)
        field(DTYP, “Soft Channel”)
        field(DISV, “1”)
        field(SDIS, “0”)
        field(DISS, “NO_ALARM”)
        field(PRIO, “LOW”)
        field(FLNK, “0”)
        field(OUT, “$(prefix)_eg.VME  PP MS”)
        field(DOL, “125”)
        field(OMSL, “supervisory”)
        field(EGU, “rocks”)
        field(HOPR, “0”)
        field(LOPR, “0”)
        field(HIHI, “0”)
        field(LOLO, “0”)
        field(HIGH, “0”)
        field(LOW, “0”)
        field(HHSV, “NO_ALARM”)
        field(LLSV, “NO_ALARM”)
        field(HSV, “NO_ALARM”)
        field(LSV, “NO_ALARM”)
        field(HYST, “0”)
        field(ADEL, “0”)
        field(MDEL, “0”)
        field(SIOL, “0”)
        field(SIML, “0”)
        field(SIMS, “NO_ALARM”)
        field(IVOA, “Continue normally”)
        field(IVOV, “0”)
    }
```

There should be nothing interesting about the `longout` records described above. The only important thing is that they properly point to the `VME` field of the `eg` record.

Event Receiver Database Records

The records used in the event receiver database are used to initialize the event receiver card. The `er` record used in the receiver database is:

```
record(er, “$(prefix)_ER”)
    {
        field(DESC, “”)
        field(ASG, “”)
        field(SCAN, “Passive”)
        field(PINI, “YES”)
        field(PHAS, “0”)
        field(EVNT, “0”)
        field(TSE, “0”)
        field(TSEL, “0”)
```

```
        field(DTYP, “APS event receiver”)
        field(DISV, “1”)
        field(SDIS, “0”)
        field(DISS, “NO_ALARM”)
        field(PRIO, “LOW”)
        field(FLNK, “0”)
        field(OUT, “#C0 S0 @”)
        field(ENAB, “YES”)
        field(TRG0, “Disabled”)
        field(TRG1, “Disabled”)
        field(TRG2, “Disabled”)
        field(TRG3, “Disabled”)
        field(TRG4, “Disabled”)
        field(TRG5, “Disabled”)
        field(TRG6, “Disabled”)
        field(OTP0, “Disabled”)
        field(OTP1, “Disabled”)
        field(OTP2, “Disabled”)
        field(OTP3, “Disabled”)
        field(OTP4, “Disabled”)
        field(OTP5, “Disabled”)
        field(OTP6, “Disabled”)
        field(OTP7, “Disabled”)
        field(OTP8, “Disabled”)
        field(OTP9, “Disabled”)
        field(OTPA, “Disabled”)
        field(OTPB, “Disabled”)
        field(OTPC, “Disabled”)
        field(OTPD, “Disabled”)
        field(OTL0, “Disabled”)
        field(OTL1, “Disabled”)
        field(OTL2, “Disabled”)
        field(OTL3, “Disabled”)
        field(OTL4, “Disabled”)
        field(OTL5, “Disabled”)
        field(OTL6, “Disabled”)
        field(DG0E, “Disabled”)
        field(DG0D, “0”)
        field(DG0W, “0”)
        field(DG1E, “Disabled”)
        field(DG1D, “0”)
        field(DG1W, “0”)
        field(DG2E, “Disabled”)
        field(DG2D, “0”)
        field(DG2W, “0”)
        field(DG3E, “Disabled”)
        field(DG3D, “0”)
        field(DG3W, “0”)
    }
```

Much like the `eg` record, the only interesting to note is that this record is passive and processed at init time.

Now, in order to cause an IRQ to occur when the reset/resync time stamp event is received, we use the following `erevent` record:

```
record(erevent, “$(prefix)_erevent7d”)
    {
        field(DESC, “”)
        field(ASG, “”)
        field(SCAN, “Passive”)
        field(PINI, “YES”)
        field(PHAS, “0”)
        field(EVNT, “0”)
        field(TSE, “0”)
        field(TSEL, “0”)
        field(DTYP, “APS event receiver”)
```

```
        field(DISV, “1”)
        field(SDIS, “0”)
        field(DISS, “NO_ALARM”)
        field(PRIO, “LOW”)
        field(FLNK, “0”)
        field(OUT, “#C0 S0 @”)
        field(ENAB, “Enabled”)
        field(ENM, “0x7d”)
        field(OUT0, “Disabled”)
        field(OUT1, “Disabled”)
        field(OUT2, “Disabled”)
        field(OUT3, “Disabled”)
        field(OUT4, “Disabled”)
        field(OUT5, “Disabled”)
        field(OUT6, “Disabled”)
        field(OUT7, “Disabled”)
        field(OUT8, “Disabled”)
        field(OUT9, “Disabled”)
        field(OUTA, “Disabled”)
        field(OUTB, “Disabled”)
        field(OUTC, “Disabled”)
        field(OUTD, “Disabled”)
        field(VME, “Enabled”)
    }
```

Interesting points here are that the output link field points to the same ER card as the above er record. The event number specified in the `ENM` field is the reset/resync time stamp event, and we can see that the `VME` field is set to `ENABLED`. This does nothing more than to tell the ER card that we want an IRQ on event number `0x7D`. Note that we could also have turned on any of the output pulse/level outputs as well.

We need not include a record to enable anything on the increment time stamp or heart beat events as they are handled by the ER card automatically.

Exactly what happens when the IRQ arrives for event `0x7D` is described in detail in the global timing documentation. Suffice it to say that the timing system registers a callback with the event receiver driver that gets called upon receipt of the event.

Should you desire to process a database record upon the receipt of an event (in this case event number `0x7D`) you may use a regular EPICS `event` record and set it up like this:

```
record(event, “$(prefix)_event”)
    {
        field(DESC, “”)
        field(ASG, “”)
        field(SCAN, “Passive”)
        field(PINI, “NO”)
        field(PHAS, “0”)
        field(EVNT, “0”)
        field(TSE, “0”)
        field(TSEL, “0”)
        field(DTYP, “APS event receiver”)
        field(DISV, “1”)
        field(SDIS, “0”)
        field(DISS, “NO_ALARM”)
        field(PRIO, “LOW”)
        field(FLNK, “$(prefix)_calc1.PROC PP MS”)
        field(INP, “#C0 S125 @”)
        field(SIOL, “0”)
        field(SIML, “0”)
        field(SIMS, “NO_ALARM”)
    }
```

Interesting tidbits here are that the record's INP link is set to the ER card, the signal number is set to the event number of interest, and that the forward link field be set to the record you wish to process upon receipt of the event code. Remember also that the VME interrupt must be enabled for the desired event code (in this case, 125 (0x7D)) by the use of an erevent record type for the same event number, that has the VME field set to ENABLED.

9. Event System Observations

This section describes those items that might otherwise be overlooked by the overwhelming detail of the record support fields. Here we provide a simple overview of the ways events can be generated by the EG card and what can be done with them by the ER card.

Event Generator Input Sources

50 ohm Trigger Inputs

The Event Generator hardware can generate event codes from 50 ohm input sources. The event codes generated are configured in the ET0-ETn fields and enabled by the ETE0-ETEn fields in the event generator record. The trigger inputs are edge sensitive and generate the event code placed in the corresponding ETn field of the EG record.

Software (EG records)

It is possible to generate any event code at any time by writing it to the VME field of the eg record. The VME field has a 'write-only' kind-of operation. Reading it will always return the value zero and not cause any events to be transmitted.

Sequence RAMs

The sequence RAMs are programmed by the use of the egevent records. Each record represents a single event code that is placed into a sequence RAM. The record describes the event code number and its position in the RAM (in terms of time offset from trigger.) If the RAM is enabled in the eg record, and a trigger is present (either 50 ohm input or by writing a value to the eg records TRGn field) the sequence RAM will be cycled thru and the present events will be sent out.

Event Receiver Outputs

The event receiver has many output configurations available. This section provides an overview of each one.

One Time Output Pulse

Any given event can generate a one-time one microsecond output pule if the output pulse enable is set for the desired signal in the er record and the related trigger bits are set in the mapping RAM via an erevent record.

Programmable Delay and Width Pulse

Any given event that is received can cause a pulse to be generated after a specified delay, and last for a specified width. The delay and width values are specified for the desired signal by the er record's DGnD and DGnW fields. It has to be enabled by the use of the DGnE field as well and the corresponding bits have to be set in the mapping RAM by the use of erevent records.

Level Outputs

Any pair of events may be used to toggle an output signal by enabling it in the `er` record and by setting the corresponding bits in the mapping `RAM` by use of a `erevent` records.

Special One Time Output Pulse

These outputs are designed such that if the event code has its high order bit set, the seven low order bits are presented on these output lines as 1 microsecond pulses. The idea here is that you can have up to seven pulses generated simultaneously. This mode is NOT configurable, but can be enabled on a per-bit basis.

VME Interrupt and Record Processing

The `er` board is capable of generating a VME interrupt upon receipt of an event code. This is enabled via an `erevent` record that has the `VME` field enabled. When this is done, an regular EPICS `event` record can be processed when the IRQs are received. The `event` record to be processed has to have its scan rate set to "`I/O Intr`" mode. The card and signal fields in the `event` record's link are used to specify the ER card number and the event number that is to cause the record to be processed.

Chapter 12: egevent - Event Generator Event

John Winans
Advanced Photon Source
Argonne National Laboratory

1. Introduction

The support for the global event system has been designed to allow an application developer to control the APS event generator and receiver boards. This is done by the use of four new record types: `eg`, `egevent`, `er`, `erevent`. These records are customized and are only supported by the device support modules for the APS event generator and receiver boards.

For more detailed information on the APS event generator and receiver records refer to Chapter 11 on page 69.

The Event Generator

The Event Generator is used to generate global event codes and send them out to one or more Event Receivers. A group of interconnected event generators and receivers is referred to as an 'event circuit.' There may be more than one event generator on the same event circuit. And it is possible for a single IOC to be part of multiple event circuits.

EGEVENT Records

The `egevent` record is used in conjunction with an `eg` record in order to specify a single event that is to be placed into a sequence RAM. The event code and its time displacement from the trigger are specified in this record.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
OUT	OUTLINK	Yes	0	No	No	No	No
ENM	LONG	Yes	0	Yes	Yes	No	Yes
LEVT	LONG	No	0	Yes	No	No	Yes
RAM	RECCHOICE	Yes	0	Yes	Yes	No	Yes
LRAM	RECCHOICE	No	0	Yes	No	No	Yes
DELY	FLOAT	Yes	0	Yes	Yes	No	Yes
ADLY	FLOAT	No	0	Yes	No	No	No
DPOS	LONG	No	0	Yes	No	No	No
APOS	LONG	No	0	Yes	No	No	No
LDLY	FLOAT	No	0	Yes	No	No	No
UNIT	RECCHOICE	Yes	0	Yes	Yes	No	Yes
VAL	CHAR	No	0	Yes	No	Yes	No
ELN	NOACCESS	No	12	No	No	No	No
SELF	NOACCESS	No	4	No	No	No	No

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	Used to specify what event generator link that this event is related to. Only the Card number is used
ENM	Event Number	The event number that is to be placed into the sequence `RAM`.
LEVT	Last Event Number	
RAM	Sequence RAM Specifier	Which `RAM` the event is to be placed into. (Ignored when the generator is in ‘Alternate’ mode.)
LRAM	Last RAM	
DELY	Desired Delay	The desired time delay between the trigger that starts the `RAM` sequence and when this event should be sent. This field must be expressed in the units selected in the `UNIT` field described below.

Name	Summary	Description
ADLY	Actual Delay	This is a read-only field that is set to the actual delay value after accounting for rounding caused by the clock resolution as well as collisions that can occur if more than one event is placed into the same sequence RAM location.
DPOS	Desired Position	This is a read-only field that represents desired position in the sequence RAM that the event should be placed. It is expressed in clock ticks.
APOS	Actual Position	This is a read-only field that represents the actual position in the sequence RAM that the event is placed. It is expressed in clock ticks.
LDLY	Last Desired Delay	
UNIT	Delay Specifier Units	The time units used to express the delay value in the DELY and ADLY fields.
VAL	Value Field	Not used.
ELN	List Node	
SELF	Self Pointer	

4. Record Processing

It is intended that egevent records be set to passive processing only. They are not altered by the device support code in response to being processed. Their purpose is only to specify the desired position and code of an event in a sequence RAM. The read-only fields will be updated as necessary when ever the sequence RAM is reloaded. To start things going, however, they should have their 'process at init' flag set to YES.

Sequence RAMs are reloaded when ever any of the egevent records related to it has its DELY, ENM or UNIT values changed. It is not advisable to alter the UNIT field unless the associated sequence RAM mode is set to 'Off'.

5. Device Support

The device support module for the event generator may be used by eg and egevent record types.

In order to configure the event generator device support, a call must be made to set the address for each of the event generator cards present in the IOC. This configuration call is as follows:

```
EgConfigure(<card number>, <Base address in A16>)
```

The <card number> field may be 0-4 and is used to specify which card is to be configured. This is the card number that is referenced in the eg and egevent records when building the database. The <Base address in A16> field is a 16-bit number that represents the address of the card in the A16 memory space.

Database records that specify card numbers that are not configured will generate 'bad field' errors when they are initialized by `iocInit`. And will then be ignored by the event generator device support if ever processed.

Chapter 13: er - Event Receiver

John Winans
Advanced Photon Source
Argonne National Laboratory

1. Introduction

The support for the global event system has been designed to allow an application developer to control the APS event generator and receiver boards. This is done by the use of four new record types: `eg`, `egevent`, `er`, `erevent`. These records are customized and are only supported by the device support modules for the APS event generator and receiver boards.

For more detailed information on the APS event generator and receiver records refer to Chapter 11 on page 69.

ER Records

The `er` record type is used to select the options of a specific event receiver card. In order to properly configure it, you should first be familiar with its operating modes. This is specified in the document "Event System" by Frank Lenksus.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	CHAR	No	0	Yes	No	Yes	No
OUT	OUTLINK	Yes	0	No	No	No	No
ENAB	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TAXI	LONG	No	0	Yes	No	Yes	No
LTAX	LONG	No	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
TRG0	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG1	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG2	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG3	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG4	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG5	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TRG6	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP0	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP1	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP2	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP3	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP4	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP5	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP6	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP7	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP8	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTP9	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTPA	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTPB	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTPC	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTPD	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL0	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL1	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL2	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL3	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL4	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL5	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OTL6	RECCHOICE	Yes	0	Yes	Yes	No	Yes
DGCM	LONG	No	0	Yes	No	No	No
DG0E	RECCHOICE	Yes	0	Yes	Yes	No	Yes
DG0D	USHORT	Yes	0	Yes	Yes	No	Yes
DG0W	USHORT	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
DG1E	RECCHOICE	Yes	0	Yes	Yes	No	Yes
DG1D	USHORT	Yes	0	Yes	Yes	No	Yes
DG1W	USHORT	Yes	0	Yes	Yes	No	Yes
DG2E	RECCHOICE	Yes	0	Yes	Yes	No	Yes
DG2D	USHORT	Yes	0	Yes	Yes	No	Yes
DG2W	USHORT	Yes	0	Yes	Yes	No	Yes
DG3E	RECCHOICE	Yes	0	Yes	Yes	No	Yes
DG3D	USHORT	Yes	0	Yes	Yes	No	Yes
DG3W	USHORT	Yes	0	Yes	Yes	No	Yes

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	Used to specify which event receiver board is represented by this record.
ENAB	Master Enable	Master card enable. No events will be received if the card is disabled.
TAXI	Taxi Violation Flag	Set to a non-zero value if there is a taxi violation on the event receiver board.
LTAX	Last Taxi Violation	
TRGn	Trigger Enable	(n=0-6) Trigger event enables. Setting these allows the corresponding bit to be set on the event receiver.
OTPn	One Time Pulse Enable	(n=0-D) Setting these allows the corresponding bit to be set in the event receiver.
OTLn	Output Level Enable	(n=0-6) Output level enables. Setting these allows the corresponding bit to be set in the event receiver.
DGCM	Delay Generator Change Mask	
DGnE	Delay Generator Enable	(n=1-3) Programmable pulse delay enables.
DGnD	Delay Generator Delay Value	(n=1-3) Delay value used for the programmable pulse delay outputs. These values must be expressed in 10-mHz clock periods and has no other selectable resolution.
DGnW	Delay generator Width Value	(n=1-3) Width of the programmable pulse. These values must be expressed in 10-mHz clock periods.
VAL	Value Field	Not used.

4. Record Processing

It is intended that `er` records be set to passive processing only. They are not altered by the device support code in response to being processed. Their purpose is only to specify the operating modes of the event receiver and, as such, are processed if and when a value in it is altered by a put operation. To start things going, however, they should have their '`process at init`' flag set to `YES`.

5. Device Support

The device support for the event receiver may be used by `er`, `erevent` and `event` record types.

In order to configure the event receiver device support, a call must be made to set the address for each of the event receiver cards present in the IOC. This configuration call is as follows:

```
ErConfigure(<card>, <A16 board address>, <IRQ Vector>, <IRQ Level>)
```

Where `<card>` is the card to be configured, `<A16 board address>` is the 16-bit address of the board in A16 space, `<IRQ Vector>` is the vector number to use when generating VME interrupts, and `<IRQ Level>` is the VME backplane.

Chapter 14: erevent - Event Receiver Event

John Winans
Advanced Photon Source
Argonne National Laboratory

1. Introduction

The support for the global event system has been designed to allow an application developer to control the APS event generator and receiver boards. This is done by the use of four new record types: `eg`, `egevent`, `er`, `erevent`. These records are customized and are only supported by the device support modules for the APS event generator and receiver boards.

For more detailed information on the APS event generator and receiver records refer to Chapter 11 on page 69.

EREVENT Records

The `erevent` records are used to specify what bits are to be set in the event receiver mapping `RAM`. The use of these bits depend on the which outputs are enabled on the event receiver card (specified in the `er` record.) Additionally, this record type is used to select the VME interrupt and time-latch option.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	CHAR	No	0	Yes	No	Yes	No
OUT	OUTLINK	Yes	0	No	No	No	No
ENAB	RECCHOICE	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
ENM	LONG	Yes	0	Yes	Yes	No	Yes
LENM	LONG	No	0	Yes	No	No	Yes
LOUT	LONG	No	0	Yes	No	No	Yes
OUT0	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT1	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT2	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT3	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT4	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT5	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT6	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT7	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT8	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUT9	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUTA	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUTB	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUTC	RECCHOICE	Yes	0	Yes	Yes	No	Yes
OUTD	RECCHOICE	Yes	0	Yes	Yes	No	Yes
VME	RECCHOICE	Yes	0	Yes	Yes	No	Yes

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	Specifies the card containing the mapping RAM to be programmed.
ENAB	Event Enable	Enables the operation of this record. If this is set to '`Disabled`', then the values in the record are ignored.
ENM	Event Number	The event number to be described. This indicates the position in the mapping RAM that is to be programmed.
LENM	Last Event Number	
LOUT	Last Output Enable Mask	

Name	Summary	Description
OUTn	Output Enable	(n=0-D) May be set to 'Enable' or 'Disable' in order to set or clear the corresponding bit in the mapping RAM. The meanings of these bits depend on what outputs are enabled on the event receiver board (selected in the related er record) and are described in the document "Event System" by Frank Lenksus.
VME	VME Interrupt Enable	May be set to 'Enable' or 'Disable' in order to select a VME interrupt upon the occurrence of event ENM. The time is automatically latched when the VME interrupt is generated.
VAL	Value Field	Not used.

4. Record Processing

It is intended that erevent records be set to passive processing only. They are not altered by the device support code in response to being processed. Their purpose is only to specify the desired actions to be performed upon receipt of a specific event code. To start things going, however, they should have their 'process at init' flag set to YES.

5. Device Support

The device support for the event receiver may be used by er, erevent and event record types.

In order to configure the event receiver device support, a call must be made to set the address for each of the event receiver cards present in the IOC. This configuration call is as follows:

```
ErConfigure(<card>, <A16 board address>, <IRQ Vector>, <IRQ Level>)
```

Where <card> is the card to be configured, <A16 board address> is the 16-bit address of the board in A16 space, <IRQ Vector> is the vector number to use when generating VME interrupts, and <IRQ Level> is the VME backplane.

Chapter 15: Event

1. Introduction

The normal use for this record type is to post an event and/or process a forward link. Device support for this record can provide a hardware interrupt handler routine for I/O Event scanned records.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	SHORT	Yes	0	Yes	Yes	Yes	No
INP	INLINK	Yes	0	No	No	N/A	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	USHORT	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	Event number to post.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

If device support includes `init_record`, it is called.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
2. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
3. If `VAL` > 0, post event number `VAL`.
4. Check to see if monitors should be invoked. Alarm monitors are invoked if the alarm status or severity has changed `NSEV` and `NSTA` are reset to 0.

5. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each record must have an associated set of device support routines. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
INP	Input Link	This field is used by the device support routines to locate its input.
PRIO	Priority	This value must be used by the device support interrupt handler to set the scheduling priority for processing this record.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, interest)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_event

```
read_event(precord)
```

This routine returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

A soft device support module is provided. The `INP` link type must be either `CONSTANT`, `DB_LINK`, or `CA_LINK`.

If the `INP` link type is `CONSTANT`, then the constant value is stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_event` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero, then `read_event` sets `UDF` to `FALSE`. The status of `recGblGetLinkValue` is returned.

Chapter 16: Fanout

1. Introduction

This record is used to trigger the processing of up to six other records. It has no associated device support.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	No	Yes
SELM	RECCHOICE	Yes	0	Yes	Yes	No	No
SELN	USHORT	No	1	Yes	Yes	No	No
SELL	INLINK	Yes	0	No	No	N/A	No
LNK1	FWDLINK	Yes	0	No	No	N/A	No
LNK2	FWDLINK	Yes	0	No	No	N/A	No
LNK3	FWDLINK	Yes	0	No	No	N/A	No
LNK4	FWDLINK	Yes	0	No	No	N/A	No
LNK5	FWDLINK	Yes	0	No	No	N/A	No
LNK6	FWDLINK	Yes	0	No	No	N/A	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field exits only because every record type must have a `VAL` field so that `dNameToAddr` succeeds when a field name is not specified.
SELM	Select Mechanism:	`SELECT_ALL`: Select all links `SELECTED`: Use `SELN` as index (1 to 6) `MASK`: Use `SELN` as a mask to select an arbitrary combination of links.
SELN	Link Selection	If `SELM=SELECT_ALL` then this field is not used. If `SELM=SELECTED` then this is the index (1 to 6) of the link to select. If `SELM=MASK` then this is the mask (in decimal) used to determine the selected links. For example, if `SELN`=1, then `LNK1` will be processed. If `SELN`=3 then `LNK1` and `LNK2` will be processed. If `SELN`=63 then all links `LNK1`, `LNK2`, ... `LNK6` will be processed.
SELL	Link Selection Location	`SELN` is read from `SELL`. `SELL` can be a constant, a database link, or a channel access link.
LNK1,...,LNK6	Link Selection Forward Links	Link selection forward links are always processed in numeric order. That is `LNK1` is always processed before `LNK2`, `LNK2` before `LNK3`, etc.

4. Record Support Routines

init_record

This routine initializes `SELN` with the value of `SELL`, if `SELL` type is `CONSTANT` link, or creates a channel access link if `SELL` type is `PV_LINK`.

process

See next section.

5. Record Processing

Routine `process` implements the following algorithm:

1. `PACT` is set to `TRUE`.
2. The link selection `SELN` is fetched.
3. Depending on the selection mechanism, the link selection forward links are processed. and `UDF` is set to `FALSE`.
4. Check to see if monitors should be invoked.
    - Alarm monitors are invoked if the alarm status or severity has changed.

- NSEV and NSTA are reset to 0.

5. Scan forward link if necessary, set PACT FALSE, and return.

Chapter 17: Histogram

1. Introduction

The `histogram` record type is used to store frequency counts of a signal into an array of arbitrary length.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	See BPTR	No	0	Yes	No	No	No
NELM	USHORT	Yes	1	Yes	No	No	No
CSTA	SHORT	No	1	Yes	No		No
CMD	RECCHOICE	No	0	Yes	Yes		No
ULIM	DOUBLE	Yes	0	Yes	Yes	No	No
LLIM	DOUBLE	Yes	0	Yes	Yes	No	No
WDTH	DOUBLE	No	0	Yes	No		No
SGNL	DOUBLE	No	0	Yes	Yes	Yes	No
SVL	INLINK	Yes	0	No	No	N/A	No
BPTR	NOACCESS	No	0	No	No		No
WDOG	NOACCESS	No	0	No	No		No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
MCNT	SHORT	No	0	Yes	No		No
MDEL	SHORT	Yes	0	Yes	Yes	No	No
SDEL	FLOAT	Yes	0	Yes	No		No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	DOUBLE	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is used to reference the array.
NELM	Number of elements in array	
CSTA	Collections Status	
CMD	Collections Control	
ULIM	Upper Signal Limit	These fields determine the range of signal values to be used. This range is subdivided into `NELM` equal intervals. The histogram array elements contain frequency counts of `SGNL` values for these intervals. Values of the signal outside these limits are not used by the record support routines. Whenever `ULIM` or `LLIM` are changed, the array elements counts will be reset to zero.
LLIM	Lower Signal Limit	
WDTH	Element Width	
SGNL	Signal Value	
SVL	Signal Value Location (input link)	This field can be a constant, a database link, or a channel access link. If `SVL` is a database or channel access link, then `SGNL` is read from `SVL`. If `SVL` is a constant link then `SGNL` is initialized with the constant value but can be changed via `dbPuts`.
BPTR	Buffer Pointer	Address of unsigned long array of frequency values.
WDOG	Watchdog Callback	
MCNT	Monitor Counts	Number of counts since last monitor.
MDEL	Monitor Delta	Monitor count deadband.

Name	Summary	Description
SDEL	Monitor Seconds Dband	
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

Using `NELM`, space for the unsigned long array is allocated and the width `WDTH` of the array is calculated

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device supportand a device support read routine are available. If device support includes `init_record`, it is called.

process

See next section.

special

Special is invoked whenever the fields `CMD`, `SGNL`, `ULIM`, or `LLIM` are changed. If `SGNL` is changed, `add_count` is called.

If `ULIM` or `LLIM` are changed, `WDTH` is recalculated and `clear_histogram` is called.

If `CMD` is less or equal to 1, `clear_histogram` is called and `CMD` is reset to 0. If `CMD` is 2, `CSTA` is set to `TRUE` and `CMD` is reset to 0. If `CMD` is 3, `CSTA` is set to `FALSE` and `CMD` is reset to 0.

`clear_histogram` zeros out the histogram array. `add_count` increments the frequency in the histogram array.

get_value

Fills in the values of struct `valueDes` so that they refer to the array.

cvt_dbaddr

This is called by `dbNameToAddr`. It makes the `dbAddr` structure refer to the actual buffer holding the array.

get_array_info Obtains values from the array referenced by `VAL`.

put_array_info Writes values into the array referenced by `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving `PACT` `TRUE`.
4. Add count to histogram array.
5. Check to see if monitors should be invoked. Alarm monitors are invoked if the alarm status or severity has changed. Archive and value change monitors are invoked if `MDEL` conditions are met. `NSEV` and `NSTA` are reset to 0.
6. Scan forward link if necessary, set `PACT` and `INIT` to `FALSE`, and return.

Chapter 18: longin - Long Input

1. Introduction

The normal use for the `longin` record type is to input an integer value of up to 32 bits. Soft device modules are provided to obtain input via database or channel access links or via `dbPutField` or `dbPutLink` requests.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	Yes	Yes
INP	INLINK	Yes	0	No	No	N/A	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	LONG	Yes	0	Yes	Yes	No	No
LOPR	LONG	Yes	0	Yes	Yes	No	No
HIHI	LONG	Yes	0	Yes	Yes	No	Yes
LOLO	LONG	Yes	0	Yes	Yes	No	Yes
HIGH	LONG	Yes	0	Yes	Yes	No	Yes
LOW	LONG	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	LONG	Yes	0	Yes	Yes	No	No
ADEL	LONG	Yes	0	Yes	Yes	No	No
MDEL	LONG	Yes	0	Yes	Yes	No	No
LALM	LONG	No	0	Yes	No	No	No
ALST	LONG	No	0	Yes	No	No	No
MLST	LONG	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	LONG	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is the value resulting form record processing. If `INP` is a constant, then `VAL` is initialized to the `INP` value but it can be changed dynamically via `dbPutField` or `dbPutLink`.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	

Name	Summary	Description
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Severity for a Hihi Alarm	
HSV	Severity for a High Alarm	
LSV	Severity for a Low Alarm	
LLSV	Severity for a Lolo Alarm	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. callbacks specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
LALM	Last Alarmed Value	Value when last monitors for alarm/archiver/value changes were triggered. These fields are used to implement the hysteresis factors for monitor callbacks.
ALST	Archive Last Value	
MLST	Monitor Last Value	
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

process See next section.

get_value Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_units Retrieves `EGU`.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
4. Check alarms. This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `LALM` are set. It also honors the alarm hysteresis factor (`HYST`). Thus the value must change by more than `HYST` before the alarm status and severity is lowered.
5. Check to see if monitors should be invoked.

- Alarm monitors are invoked if the alarm status or severity has changed.
- Archive and value change monitors are invoked if `ADEL` and `MDEL` conditions are met.
- `NSEV` and `NSTA` are reset to 0.

6. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each long input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new input value whenever `read_longin` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
VAL	Value Field	This field is set by device support routines.
INP	Input Link	This field is used by the device support routines to locate its input.

Device Support routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_longin

```
read_longin(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success. A new value is placed in `VAL`.
- **Other**: Error.

7. Device Support For Soft Records

This module places a value directly in `VAL`.

If the `INP` link type is constant, then the constant value is stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_longin` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero then `read_longin` sets `UDF` to `FALSE`. `read_longin` returns the status of `recGblGetLinkValue`.

Chapter 19: longout - Long Output

1. Introduction

The normal use for the `longout` record type is to store integer values of up to 31 bits. It can also be used to write values to other records via database or channel access links. The `OUT` field determines how the record is used. The record supports alarm limits and graphics and control limits.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	Yes	Yes
OUT	OUTLINK	Yes	0	No	No	N/A	No
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	LONG	Yes	0	Yes	Yes	No	No
LOPR	LONG	Yes	0	Yes	Yes	No	No
HIHI	LONG	Yes	0	Yes	Yes	No	Yes
LOLO	LONG	Yes	0	Yes	Yes	No	Yes
HIGH	LONG	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LOW	LONG	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	LONG	Yes	0	Yes	Yes	No	No
ADEL	LONG	Yes	0	Yes	Yes	No	No
MDEL	LONG	Yes	0	Yes	Yes	No	No
LALM	LONG	No	0	Yes	No	No	No
ALST	LONG	No	0	Yes	No	No	No
MLST	LONG	No	0	Yes	No	No	No
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	LONG	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value	This is the desired output value, in engineering units. If `DRVH` and `DRVL` are defined, `VAL` is forced to be within the drive limits. `VAL` is either obtained from `DOL` or set via `dbPuts`.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
DOL	Desired Output Location (input link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`. After the check for drive limits `VAL` will be set to the value determined by `DOL`.

Name	Summary	Description
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`. If these values are defined, they must be in the range `DRVL<=LOPR<=HOPR<=DRVH.`
LOPR	Low Operating Range	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Hihi Alarm Severity	
HSV	High Alarm Severity	
LSV	Low Alarm Severity	
LLSV	Lolo Alarm Severity	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. callbacks specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
LALM	Last Alarmed Value	Value when last monitors for alarm/archiver/value change were triggered. These fields are used to implement the hysteresis factors for monitors.
ALST	Archive Last Value	
MLST	Monitor Last Value	

Name	Summary	Description
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3 Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3 Section "Invalid Alarm Output Action" on page 14 for more information.
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

This routine initializes `SIMM` if `SIML` is a constant or creates a channel access link if `SIML` is `PV_LINK`. If `SIOL` is `PV_LINK` a channel access link is created.

This routine next checks to see that device support is available. The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If `DOL` is a constant, then `VAL` is initialized to its value and `UDF` is set to `FALSE`. If `DOL` type is a `PV_LINK` then `dbCaAddInlink` is called to create a channel access link.

If device support includes `init_record`, it is called.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If PACT is FALSE and OMSL is CLOSED_LOOP recGblGetLinkValue is called to read the current value of VAL. See Chapter 3 Section "Soft Input" on page 10 for details. If the return status of recGblGetLinkValue is zero then UDF is set to FALSE.
3. Check alarms. This routine checks to see if the new VAL causes the alarm status and severity to change. If so, NSEV, NSTA and LALM are set. It also honors the alarm hysteresis factor (HYST). Thus the value must change by more than HYST before the alarm status and severity is lowered.
4. Check severity and write the new value. See Chapter 3 Section "Simulation Mode" on page 13 and Chapter 3 Section "Invalid Alarm Output Action" on page 14 for details.
5. If PACT has been changed to TRUE, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving PACT TRUE.
6. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if ADEL and MDEL contitions are met.
   - NSEV and NSTA are reset to 0.
7. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each long output record must have an associated set of device support routines. The primary responsibility of the device support routines is to output a new value whenever `write_longout` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2 Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
OUT	Output Link	This field is used by the device support routines to locate its output.

Device Support Routines

Device support consists of the following routines:

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_longout

```
write_longout(precord)
```

This routine must output a new value. It returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

This module writes the current value of `VAL`.

If the `OUT` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`write_longout` calls `recGblPutLinkValue` to write the current value of `VAL`. See Chapter 3 Section "Soft Output" on page 13 for details.

Chapter 20: mbbi - MultiBit Binary Input

1. Introduction

The normal use for the `mbbi` record type is to obtain a binary value that represents one of up to 16 states. Most device support modules obtain values from hardware and place the value in `RVAL`. For these devices record processing uses `RVAL` to determine the current state (`VAL` is given a value between 0 and 15). Devices may optionally read a value directly into `VAL`. Soft device modules are provided to obtain input via database or channel access links or via `dbPutField` or `dbPutLink` requests. Two soft device support modules are provided. One allows `VAL` to be an arbitrary unsigned short integer. The other reads the value into `RVAL` just like normal hardware modules.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	ENUM	No	0	Yes	Yes	Yes	Yes
NOBT	SHORT	Yes	0	Yes	No		No
INP	INLINK	Yes	0	No	No	N/A	No
ZRVL	ULONG	Yes	0	Yes	Yes	No	Yes
ONVL	ULONG	Yes	0	Yes	Yes	No	Yes
TWVL	ULONG	Yes	0	Yes	Yes	No	Yes
THVL	ULONG	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
FRVL	ULONG	Yes	0	Yes	Yes	No	Yes
FVVL	ULONG	Yes	0	Yes	Yes	No	Yes
SXVL	ULONG	Yes	0	Yes	Yes	No	Yes
SVVL	ULONG	Yes	0	Yes	Yes	No	Yes
EIVL	ULONG	Yes	0	Yes	Yes	No	Yes
NIVL	ULONG	Yes	0	Yes	Yes	No	Yes
TEVL	ULONG	Yes	0	Yes	Yes	No	Yes
ELVL	ULONG	Yes	0	Yes	Yes	No	Yes
TVVL	ULONG	Yes	0	Yes	Yes	No	Yes
TTVL	ULONG	Yes	0	Yes	Yes	No	Yes
FTVL	ULONG	Yes	0	Yes	Yes	No	Yes
FFVL	ULONG	Yes	0	Yes	Yes	No	Yes
ZRST	STRING	Yes	Null	Yes	Yes	No	Yes
ONST	STRING	Yes	Null	Yes	Yes	No	Yes
TWST	STRING	Yes	Null	Yes	Yes	No	Yes
THST	STRING	Yes	Null	Yes	Yes	No	Yes
FRST	STRING	Yes	Null	Yes	Yes	No	Yes
FVST	STRING	Yes	Null	Yes	Yes	No	Yes
SXST	STRING	Yes	Null	Yes	Yes	No	Yes
SVST	STRING	Yes	Null	Yes	Yes	No	Yes
EIST	STRING	Yes	Null	Yes	Yes	No	Yes
NIST	STRING	Yes	Null	Yes	Yes	No	Yes
TEST	STRING	Yes	Null	Yes	Yes	No	Yes
ELST	STRING	Yes	Null	Yes	Yes	No	Yes
TVST	STRING	Yes	Null	Yes	Yes	No	Yes
TTST	STRING	Yes	Null	Yes	Yes	No	Yes
FTST	STRING	Yes	Null	Yes	Yes	No	Yes
FFST	STRING	Yes	Null	Yes	Yes	No	Yes
ZRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ONSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TWSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
THSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
FRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SXSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
EISV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
NISV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TESV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ELSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TTSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FTSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FFSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
UNSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
COSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
RVAL	ULONG	No	0	Yes	Yes	Yes	Yes
ORAW	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	0	Yes	No	No	No
MLST	USHORT	No	0	Yes	No	No	No
LALM	USHORT	No	0	Yes	No	No	No
SDEF	SHORT	No	0	Yes	No	No	No
SHFT	USHORT	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	USHORT	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	Unless `INP` is a constant link, this is the value resulting from the record being processed. If `INP` is a constant, then `VAL` is initialized to the `INP` value but can be changed dynamically via `dbPutField` or `dbPutLink`. It normally is the index (0 to 15) of the current state.
NOBT	Number of Bits	Number of bits set in hardware mask.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
ZRVL,...,FFVL	Zero Value, One Value ...	Masks for hardware value associated with each state.
ZRST,...,FFST	Zero String, One String ...	Strings associated with each state.
ZRSV,...,FFSV	Zero Severity, One Severity,...	Alarm severity associated with each state.
UNSV	Unknown State Severity	
COSV	Change of State Severity	
RVAL	Raw Data Value	`RVAL` is the value obtained by the device support routine. Unless the device support routine specifies no conversion, `VAL` is determined as follows: A temporary variable `rval` is set equal to `RVAL`. It is then shifted right `SHFT` bits. After shifting, the result should match one of the values `ZRVL,...,FFVL`.
ORAW	Old Raw Data Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are triggered for changes in `VAL`.
MASK	Mask	Mask used by device support routine to read hardware register. Record support sets low order `NOBT` bits. Device support can shift this value.
SHFT	Shift	Number of bits to shift values obtained from `RVAL`.
LALM	Last Alarmed	Value when last change of state alarm was issued.
MLST	Monitor Last	Value when last monitor for value changes was triggered
SDEF	States Defined?	Record support uses this field to save time if no states are defined

Name	Summary	Description
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

Clears `MASK` and then sets the `NOBT` low order bits.

If device support includes `init_record`, it is called.

`init_common` is then called to determine if any states are defined. If states are defined, `SDEF` is set to `TRUE`.

process

See next section.

special

Calls `init_common` to compute `SDEF` when any of the fields `ZRVL`, ... `FFVL` change value.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_enum_str

Retrieves ASCII string corresponding to `VAL`.

get_enum_strs

Retrieves ASCII strings for `ZRST`,...`FFST`.

put_enum_str

Checks if string matches `ZRST`,...`FFST` and if it does, sets `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
4. Convert.
   - `status=read_mbbi`
   - `PACT` = `TRUE`
   - `TIME` = `tsLocalTime`
   - If `status` is 0, then determine `VAL`
     - Set `rval` = `RVAL`
     - Shift `rval` right `SHFT` bits
     - If at least one state value is defined
       - Set `UDF` to `TRUE`
       - If `RVAL` is `ZRVL`,...,`FFVL` then set
       - `VAL` equals index of state
       - `UDF` set to `FALSE`
       - Else set `VAL` = undefined
     - Else set `VAL` = `RVAL`
       - Set `UDF` to `FALSE`
     - If `status` is 1, return(0)
   - If `status` is 2, set `status` = 0
5. Check alarms. This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `LALM` are set.
6. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `MLST` is not equal to `VAL`.
   - Monitors for `RVAL` are checked whenever other monitors are invoked.
   - `NSEV` and `NSTA` are reset to 0.
7. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw input value whenever `read_mbbi` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
NOBT	Number of Bits	Number of hardware bits accessed. They must be consecutive.
VAL	Value Field	This field is set by the device support routines if they don't want record support to set it.
INP	Input Link	This field is used by the device support routines to locate its input.
RVAL	Raw Data Value	It is the responsibility of the device support routine to give this field a value.
MASK	Mask	This is a mask used to read the hardware. Record support sets the low order `NOBT` bits. The device support routine can shift the bits. The device support routine should perform the shift in in `init_record`.
SHFT	Shift	This can be set by the device support module at `init_record` time.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. If it uses `MASK`, it should shift it as necessary and also give `SHFT` a value.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_mbbi

```
read_mbbi(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success. A new raw value is placed in `RVAL`. The record support module determines `VAL` from `RVAL`, `SHFT`, and `ZEVL` ... `FFVL`.
- **2**: Success, but don't modify `VAL`.
- **Other**: Error.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for multibit binary input records not related to actual hardware devices. The `INP` link type must be either `CONSTANT`, `DB_LINK`, or `CA_LINK`.

Soft Channel

`read_mbbi` always returns a value of 2, which means that no conversion is performed.

If the `INP` link type is constant, then the constant value is stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. `VAL` can be changed via `dbPut` requests. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_mbbi` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero, then `read_mbbi` sets `UDF` to `FALSE`. The status of `recGblGetLinkValue` is returned.

Raw Soft Channel

This module is like the previous except that values are read into `RVAL`, `VAL` is computed from `RVAL`, and `read_mbbi` returns a value of 0. Thus the record processing routine will determine `VAL` in the normal way.

Chapter 21: mbbo - MultiBit Binary Output

1. Introduction

The normal use for the `mbbo` record type is to send a binary value (representing one of up to 16 states) to a Digital Output module. It can also be used to write to other records via database or channel access links.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	ENUM	No	0	Yes	Yes	Yes	Yes
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
NOBT	SHORT	Yes	0	Yes	No		No
OUT	OUTLINK	Yes	0	No	No	N/A	No
ZRVL	ULONG	Yes	0	Yes	Yes	No	Yes
ONVL	ULONG	Yes	0	Yes	Yes	No	Yes
TWVL	ULONG	Yes	0	Yes	Yes	No	Yes
THVL	ULONG	Yes	0	Yes	Yes	No	Yes
FRVL	ULONG	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
FVVL	ULONG	Yes	0	Yes	Yes	No	Yes
SXVL	ULONG	Yes	0	Yes	Yes	No	Yes
SVVL	ULONG	Yes	0	Yes	Yes	No	Yes
EIVL	ULONG	Yes	0	Yes	Yes	No	Yes
NIVL	ULONG	Yes	0	Yes	Yes	No	Yes
TEVL	ULONG	Yes	0	Yes	Yes	No	Yes
ELVL	ULONG	Yes	0	Yes	Yes	No	Yes
TVVL	ULONG	Yes	0	Yes	Yes	No	Yes
TTVL	ULONG	Yes	0	Yes	Yes	No	Yes
FTVL	ULONG	Yes	0	Yes	Yes	No	Yes
FFVL	ULONG	Yes	0	Yes	Yes	No	Yes
ZRST	STRING	Yes	Null	Yes	Yes	No	Yes
ONST	STRING	Yes	Null	Yes	Yes	No	Yes
TWST	STRING	Yes	Null	Yes	Yes	No	Yes
THST	STRING	Yes	Null	Yes	Yes	No	Yes
FRST	STRING	Yes	Null	Yes	Yes	No	Yes
FVST	STRING	Yes	Null	Yes	Yes	No	Yes
SXST	STRING	Yes	Null	Yes	Yes	No	Yes
SVST	STRING	Yes	Null	Yes	Yes	No	Yes
EIST	STRING	Yes	Null	Yes	Yes	No	Yes
NIST	STRING	Yes	Null	Yes	Yes	No	Yes
TEST	STRING	Yes	Null	Yes	Yes	No	Yes
ELST	STRING	Yes	Null	Yes	Yes	No	Yes
TVST	STRING	Yes	Null	Yes	Yes	No	Yes
TTST	STRING	Yes	Null	Yes	Yes	No	Yes
FTST	STRING	Yes	Null	Yes	Yes	No	Yes
FFST	STRING	Yes	Null	Yes	Yes	No	Yes
ZRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ONSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TWSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
THSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
FVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SXSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
EISV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
NISV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TESV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ELSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TVSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
TTSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FTSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
FFSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
UNSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
COSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
RVAL	ULONG	No	0	Yes	Yes	Yes	Yes
ORAW	ULONG	No	0	Yes	No	No	No
RBV	ULONG	No	0	Yes	No	Yes	No
ORBV	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	0	Yes	No	No	No
MLST	USHORT	No	0	Yes	No	No	No
LALM	USHORT	No	0	Yes	No	No	No
SDEF	SHORT	No	0	Yes	No	No	No
SHFT	USHORT	No	0	Yes	No	No	No
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	USHORT	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is the index of the state value to be sent to `OUT`.
DOL	Desired Output Location (an Input Link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`.
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field, the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
NOBT	Number of Bits	Number of bits in hardware mask.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
ZRVL,...,FFVL	Zero Value, One Value, ...	Masks for hardware value associated with each state.
ZRST,...,FFST	Zero State, One State, ...	Strings associated with each state.
ZRSV,...,FFSV	Zero Severity, One Severity, ...	Alarm severity associated with each state.
UNSV	Unknown State Severity	
COSV	Change of State Severity	
RVAL	Raw Data Value	`RVAL` is the value to be written to the hardware device. It is determined by the record support module using `VAL` as the index of the values stored in `ZRVL`,...`FFVL`. The value is also shifted left `SHFT` bits.
ORAW	Old Raw Data Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are generated for changes in `VAL`.
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value.
ORAW	Old Read Back Value	`ORBV` is used to decide if monitors should be triggered for `RBV` at the same time monitors are triggered for changes in `VAL`.
MASK	Mask	Mask used by device support routine to read hardware register. Record support sets low order `NOBT` bits. Device support can shift this value.

Name	Summary	Description
MLST	Monitor Last	Value when last monitor for value changes was triggered.
LALM	Last Alarmed	Value when last change of state alarm was issued.
SDEF	States Defined?	Record support uses this field to save time if no states are defined.
SHFT	Shift	Number of bits to shift values obtained from ZRVL,...,FFVL.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

This routine initializes SIMM if SIML is a constant or creates a channel access link if SIML is PV_LINK. If SIOL is PV_LINK a channel access link is created.

This routine next checks to see that device support is available. The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If DOL is a constant, then VAL is initialized to its value and UDF is set to FALSE.

MASK is cleared and then the NOBT low order bits are set.

If device support includes init_record, it is called.

init_common is then called to determine if any states are defined. If states are defined, SDEF is set to TRUE.

If device support returns success, VAL is then set from RVAL and UDF is set to FALSE.

process

See next section.

special

Computes SDEF when any of the fields ZRVL,...FFVL change value.

get_value Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_enum_str Retrieves ASCII string corresponding to `VAL`.

get_enum_strs Retrieves ASCII strings for `ZRST`,...`FFST`.

put_enum_str Checks if string matches `ZRST`,...`FFST` and if it does, sets `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If `PACT` is `FALSE`
   - If `DOL` is `DB_LINK` and `OMSL` is `CLOSED_LOOP`
     - Get value from `DOL`
     - Set `UDF` to `FALSE`
     - Check for link alarm
   - If any state values are defined
     - If `VAL` > 15, then raise alarm and go to 4
     - Else using `VAL` as index set `RVAL` = one of `ZRVL`,...`FFVL`
   - Else set `RVAL` = `VAL`
   - Shift `RVAL` left `SHFT` bits
3. Convert
   - If `PACT` is `FALSE`, compute `RVAL`
     - If `VAL` is 0,...,15, set `RVAL` from `ZRVL`,...,`FFVL`
     - If `VAL` out of range, set `RVAL` = undefined
   - Status=`write_mbbo`
4. Check alarms. This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `LALM` are set.
5. Check severity and write the new value. See Chapter 3, Section "Simulation Mode" on page 13 and Chapter 3, Section "Invalid Alarm Output Action" on page 14 for details.
6. If `PACT` has been changed to `TRUE`, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving `PACT` `TRUE`.
7. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `MLST` is not equal to `VAL`.
   - Monitors for `RVAL` and `RBV` are checked whenever other monitors are invoked.
   - `NSEV` and `NSTA` are reset to 0.

8. Scan forward link if necessary, set `PACT` `FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each `mbbo` record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw `mbbo` value whenever `write_mbbo` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
NOBT	Number of Bits	Number of hardware bits accessed. They must be consecutive.
OUT	Output Link	This field is used by the device support routines to locate its output.
RVAL	Raw data value.	This is the value to be written to `OUT`.
RBV	Read Back Value	It is the responsibility of the device support modules to set this field.
MASK	Mask	This is a mask used to read the hardware. Record support sets the low order `NOBT` bits. The device support routine can shift the bits. The device support routine should perform the shift in in `init_record`.
SHFT	Shift	This can be set by the device support module at `init_record` time.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. If `MASK` is used, it should be shifted if necessary and `SHFT` given a value.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_mbbo

```
write_mbbo(precord)
```

This routine must output a new value. It returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

This module writes the current value of `VAL`.

If the `OUT` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`write_mbbo` calls `recGblPutLinkValue` to write the current value of `VAL`. See Chapter 3, Section "Soft Output" on page 13 for details.

Chapter 22: mbbiDirect - MultiBit Binary Input Direct

Johnny Tang, Matthew Bickley, and Chip Watson
Continuous Electron Beam Accelerator Facility
Southeastern Universities Research Association

1. Introduction

The `mbbiDirect` record retrieves a sixteen bit hardware value and converts it to an array of sixteen unsigned characters, each representing a bit of the word. These fields (`B0-B15`) are set to one if a bit is set, and zero if not. This record's operation is similar to that of an `mbbi`, and it has many fields in common with it.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	USHORT	No	0	Yes	No	Yes	Yes
NOBT	SHORT	Yes	0	Yes	No		No
INP	INLINK	Yes	0	No	No	N/A	No
RVAL	ULONG	No	0	Yes	Yes	Yes	Yes
ORAW	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	0	Yes	No	No	No
MLST	USHORT	No	0	Yes	No	No	No
LALM	USHORT	No	0	Yes	No	No	No
SDEF	SHORT	No	0	Yes	No	No	No
SHFT	USHORT	No	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
SIOL	INLINK	Yes	0	No	No	No	No
SVAL	USHORT	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	No	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
B0	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B1	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B2	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B3	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B4	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B5	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B6	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B7	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B8	UCHAR	Yes	0	Yes	Yes	Yes	Yes
B9	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BA	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BB	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BC	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BD	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BE	UCHAR	Yes	0	Yes	Yes	Yes	Yes
BF	UCHAR	Yes	0	Yes	Yes	Yes	Yes

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	Unless `INP` is a constant link, this is the value resulting from the record being processed. If `INP` is a constant, then `VAL` is initialized to the `INP` value but can be changed dynamically via `dbPutField` or `dbPutLink`.
NOBT	Number of Bits	Number of bits set in hardware mask.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
B0, ..., BF	Bit 0 Value, Bit 1 Value ...	Each represents a bit of the word.
RVAL	Raw Data Value	`RVAL` is the value obtained by the device support routine. Unless the device support routine specifies no conversion, `VAL` is determined as follows: A temporary variable `rval` is set equal to `RVAL`.
ORAW	Old Raw Data Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are triggered for changes in `VAL`.
MASK	Mask	Mask used by device support routine to read hardware register. Record support sets low order `NOBT` bits. Device support can shift this value.
SHFT	Shift	Number of bits to shift values obtained from `RVAL`.
LALM	Last Alarmed	Value when last change of state alarm was issued.
MLST	Monitor Last	Value when last monitor for value changes was triggered.
SDEF	States Defined	
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

This routine initializes SIMM with the value of SIML if SIML type is CONSTANT link or creates a channel access link if SIML type is PV_LINK. SVAL is likewise initialized if SIOL is CONSTANT or PV_LINK.

This routine next checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

Clears MASK and then sets the NOBT low order bits.

If device support includes init_record, it is called.

refresh_bits is then called to refresh all the bit fields based on a hardware value.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to VAL.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. readValue is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If PACT has been changed to TRUE, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving PACT TRUE.
4. Convert.
   - status=read_mbbiDirect
   - PACT = TRUE
   - TIME = tsLocalTime
   - If status is 0, then determine VAL
     - Set rval = RVAL
     - Shift rval right SHFT bits
     - Set VAL = RVAL
     - If status is 1, return(0)
   - If status is 2, set status = 0
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if MLST is not equal to VAL.
   - Monitors for RVAL are checked whenever other monitors are invoked.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw input value whenever `read_mbbiDirect` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
NOBT	Number of Bits	Number of hardware bits accessed. They must be consecutive.
VAL	Value Field	This field is set by the device support routines if they don't want record support to set it.
INP	Input Link	This field is used by the device support routines to locate its input.
RVAL	Raw Data Value	It is the responsibility of the device support routine to give this field a value.
MASK	Mask	This is a mask used to read the hardware. Record support sets the low order `NOBT` bits. The device support routine can shift the bits. The device support routine should perform the shift in init_record.
SHFT	Shift	This can be set by the device support module at `init_record` time.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. If it uses `MASK`, it should shift it as necessary and also give `SHFT` a value.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_mbbiDirect

```
read_mbbiDirect(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success. A new raw value is placed in `RVAL`. The record support module determines `VAL` from `RVAL` and `SHFT`.
- **2**: Success, but don't modify `VAL`.
- **Other**: Error.

7. Device Support For Soft Records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for multibit binary input direct records not related to actual hardware devices. The `INP` link type must be either `CONSTANT`, `DB_LINK`, or `CA_LINK`.

Soft Channel

`read_mbbiDirect` always returns a value of 2, which means that no conversion is performed.

If the `INP` link type is constant, then the constant value is stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. `VAL` can be changed via `dbPut` requests. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_mbbiDirect` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero, then `read_mbbi` sets `UDF` to `FALSE`. The status of `recGblGetLinkValue` is returned.

Raw Soft Channel

This module is like the previous except that values are read into `RVAL`, `VAL` is computed from `RVAL`, and `read_mbbiDirect` returns a value of 0. Thus the record processing routine will determine `VAL` in the normal way.

Chapter 23: mbboDirect - MultiBit Binary Output Direct

Johnny Tang, Matthew Bickley, and Chip Watson
Continuous Electron Beam Accelerator Facility
Southeastern Universities Research Association

1. Introduction

The `mbboDirect` record performs the opposite function to that of the `mbbiDirect` record. It accumulates bits (in the fields `B0 - BF`) as unsigned characters, and converts them to a word which is then written out to hardware. If a bit field is non-zero, it is interpreted as a binary one. On the other hand, if it is zero, it is interpreted as a binary zero.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	USHORT	No	0	Yes	No	Yes	Yes
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
NOBT	SHORT	Yes	0	Yes	No		No
OUT	OUTLINK	Yes	0	No	No	N/A	No
B0	UCHAR	No	0	Yes	Yes	No	Yes
B1	UCHAR	No	0	Yes	Yes	No	Yes
B2	UCHAR	No	0	Yes	Yes	No	Yes
B3	UCHAR	No	0	Yes	Yes	No	Yes
B4	UCHAR	No	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
B5	UCHAR	No	0	Yes	Yes	No	Yes
B6	UCHAR	No	0	Yes	Yes	No	Yes
B7	UCHAR	No	0	Yes	Yes	No	Yes
B8	UCHAR	No	0	Yes	Yes	No	Yes
B9	UCHAR	No	0	Yes	Yes	No	Yes
BA	UCHAR	No	0	Yes	Yes	No	Yes
BB	UCHAR	No	0	Yes	Yes	No	Yes
BC	UCHAR	No	0	Yes	Yes	No	Yes
BD	UCHAR	No	0	Yes	Yes	No	Yes
BE	UCHAR	No	0	Yes	Yes	No	Yes
BF	UCHAR	No	0	Yes	Yes	No	Yes
RVAL	ULONG	No	0	Yes	No	Yes	Yes
ORAW	ULONG	No	0	Yes	No	No	No
RBV	ULONG	No	0	Yes	No	Yes	No
ORBV	ULONG	No	0	Yes	No	No	No
MASK	ULONG	No	0	Yes	No	No	No
MLST	ULONG	No	0	Yes	No	No	No
LALM	ULONG	No	0	Yes	No	No	No
SHFT	ULONG	No	0	Yes	No	No	No
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	USHORT	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is the value to be sent to `OUT`.
DOL	Desired Output Location (Input Link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`.
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field, the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
NOBT	Number of Bits	Number of bits in hardware mask.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
B0,...,BF	Bit 0 Value, Bit 1 Value, ...	Each represents a bit of the word to be output.
RVAL	Raw Data Value	`RVAL` is the value to be written to the hardware device. It is determined by the record support module using `VAL` as the bit values stored in `B0`,...`BF`. The value is also shifted left `SHFT` bits.
ORAW	Old Raw Data Value	`ORAW` is used to decide if monitors should be triggered for `RVAL` at the same time monitors are generated for changes in `VAL`.
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value
ORBV	Old Read Back Value	`ORBV` is used to decide if monitors should be triggered for `RBV` at the same time monitors are triggered for changes in `VAL`.
MASK	Mask	Mask used by device support routine to read hardware register. Record support sets low order `NOBT` bits. Device support can shift this value.
MLST	Monitor Last	Value when last monitor for value changes was triggered.
LALM	Last Alarmed	Value when last change of state alarm was issued.
SHFT	Shift	Number of bits to shift values obtained from `VAL`.

Name	Summary	Description
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

init_record

This routine initializes `SIMM` if `SIML` is a constant or creates a channel access link if `SIML` is `PV_LINK`. If `SIOL` is `PV_LINK` a channel access link is created.

This routine next checks to see that device support is available.The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If `DOL` is a constant, then `VAL` is initialized to its value and `UDF` is set to `FALSE`.

`MASK` is cleared and then the `NOBT` low order bits are set.

If device support includes `init_record`, it is called.

`init_common` is then called to determine if any states are defined. If states are defined, `SDEF` is set to `TRUE`.

If device support returns success, `VAL` is then set from `RVAL` and `UDF` is set to `FALSE`.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This

ensures that processes will no longer be called for this record. Thus error storms will not occur.

2. If PACT is FALSE
   - If DOL is DB_LINK and OMSL is CLOSED_LOOP
     - Get value from DOL
     - Set PACT to FALSE
3. Convert
   - If PACT is FALSE, compute RVAL
     - Set RVAL = VAL
     - Shift RVAL left SHFT bits
   - Status=write_mbboDirect
4. If PACT has been changed to TRUE, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving PACT TRUE.
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if MLST is not equal to VAL.
   - Monitors for RVAL and RBV are checked whenever other monitors are invoked.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each mbboDirect record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new raw mbbo value whenever write_mbboDirect is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
NOBT	Number of Bits	Number of hardware bits accessed. They must be consecutive.
OUT	Output Link	This field is used by the device support routines to locate its output.
RVAL	Raw data value	This is the value to be written to OUT.
RBV	Read Back Value	It is the responsibility of the device support modules to set this field.

Name	Summary	Description
MASK	Mask	This is a mask used to read the hardware. Record support sets the low order `NOBT` bits. The device support routine can shift the bits. The device support routine should perform the shift in in `init_record`.
SHFT	Shift	This can be set by the device support module at `init_record` time.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine. If `MASK` is used, it should be shifted if necessary and `SHFT` given a value.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_mbboDirect

```
write_mbboDirect(precord)
```

This routine must output a new value. It returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

This module writes the current value of `VAL`.

If the `OUT` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`write_mbboDirect` calls `recGblPutLinkValue` to write the current value of `VAL`. See Chapter 3, Section "Soft Output" on page 13 for details.

Chapter 24: Permissive

1. Introduction

The `permissive` record is for communication between a server and a client. An example is a sequence program client and an operator interface server. Two fields are used `VAL` and `WFLG`. The method of use is as follows:

1. Initially both `VAL` and `WFLG` are 0, which means `OFF`.
2. When the server is ready to accept a request, it sets `WFLG` equal to 1, which means `ON`.
3. The client monitors `WFLG`. Until it turns `ON`, the client must not change `VAL`.
4. When the client wants to notify the server it turns `VAL ON`.
5. The server notices that `VAL` is `ON`. He sets both `WFLG` and `VAL OFF`. Performs whatever action is associated with this permissive (a private matter server and client), and when ready to accept a new request sets `WFLG ON`.

By using multiple permissive records a sequence program can communicate its current state to a client.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LABL	STRING	Yes	Null	Yes	Yes	No	Yes
VAL	USHORT	No	0	Yes	Yes	Yes	Yes
OVAL	USHORT	No	0	Yes	No		No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
WFLG	USHORT	No	0	Yes	Yes	Yes	Yes
OFLG	USHORT	No	0	Yes	No		No

3. Field Descriptions

Name	Summary	Description
LABL	Label	A descriptive string.
VAL	Value	Client sets this field when it wants service from server. Only the client should set this field. The server clears it.
OVAL	Old Value	Used to decide if monitors should be triggered. Value change monitors are invoked if `OVAL` is not equal to `VAL`.
WFLG	Watchdog Flag	Server sets this field when it is ready to accept a request. Only the server should modify this field.
OFLG	Old Flag Value	Used to decide if monitors should be triggered.

4. Record Support Routines

Two record support routines are provided: `process`, and `get_value`.

process

`process` sets `UDF` to `FALSE`, triggers monitors on `VAL` and `WFLG` when they change, and scans the forward link if necessary.

get_value

`get_value` fills in struct `valueDes` so that it refers to `VAL`.

Chapter 25: pid - PID Control

1. Introduction

The `pid` record type provides a Proportional, Integral, and Derivative (PID) control algorithm. A discrete form of the PID algorithm is:

$$M_n = KP * (E_n + KI * SUMi\ (E_i * dT_i)) + KD * (E_n - E_{n-1})/dT_n + Mr$$

Where:

- M_n: Value of manipulated variable at nth sampliing instant
- KP, KI, KD: Proportional, Integral, and Derivative gains
- E_n: Error at nth sampling instant
- SUM_i: Sum from i=0 to i=n
- dT_n: Time difference between n-1 and n
- Mr: Midrange adjustment

Taking the first difference yields:

$$delM(n) = KP * ((E_n - E_{n-1}) + E_n * dT_n * KI + KD * ((E_n - E_{n-1})/dT_n - (E_{n-1} - E_{n-2})/dT_{n-1}))$$

For this record:

- DM: This is delM(n)
- VAL: This is the setpoint
- CVAL: This is current value
- ERR: E_n = VAL - CVAL

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	FLOAT	No	0	Yes	Yes	Yes	Yes
CVL	INLINK	Yes	0	No	No	N/A	No
STPL	INLINK	Yes	0	No	No	N/A	No
SMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
MDT	FLOAT	Yes	0	Yes	Yes	No	No
KP	FLOAT	Yes	0	Yes	Yes	No	No
KI	FLOAT	Yes	0	Yes	Yes	No	No
KD	FLOAT	Yes	0	Yes	Yes	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	No
LOLO	FLOAT	Yes	0	Yes	Yes	No	No
HIGH	FLOAT	Yes	0	Yes	Yes	No	No
LOW	FLOAT	Yes	0	Yes	Yes	No	No
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	No
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	No
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	No
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	No
HYST	FLOAT	Yes	0	Yes	Yes	No	No
ADEL	FLOAT	Yes	0	Yes	Yes	No	No
MDEL	FLOAT	Yes	0	Yes	Yes	No	No
ODEL	FLOAT	Yes	0	Yes	Yes	No	No
CVAL	FLOAT	No	0	Yes	Yes	No	No
DM	FLOAT	No	0	Yes	No	Yes	No
ODM	FLOAT	No	0	Yes	No	Yes	No
P	FLOAT	No	0	Yes	No	Yes	No
I	FLOAT	No	0	Yes	No	Yes	No
D	FLOAT	No	0	Yes	No	Yes	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
CT	ULONG	No	0	Yes	No	Yes	No
DT	FLOAT	No	0	Yes	No	Yes	No
ERR	FLOAT	No	0	Yes	No	Yes	No
DERR	FLOAT	No	0	Yes	No	Yes	No
LALM	FLOAT	No	0	Yes	No	No	No
ALST	FLOAT	No	0	Yes	No	No	No
MLST	FLOAT	No	0	Yes	No	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Setpoint Value, in engineering units	This is the value that the control algorithm attempts to achieve.
CVL	Controlled Value Location (input link)	This is a link specifying the location of the controlled variable. This must be a database link. Each time the record is processed the current value referenced by `CVL` is read into `CVAL`.
STPL	Setpoint Location (input link)	If `STPL` is a database or channel access link and `SMSL` is `CLOSED_LOOP`, then `VAL` is read from `STPL`. `STPL` and `SMSL` act just like `DOL` and `OMSL`.
SMSL	Setpoint Mode Select	This is `SUPERVISORY` or `CLOSED_LOOP`. `VAL` is obtained from `STPL` only when this is `CLOSED_LOOP`. By setting this field, the record can be switched between supervisory and closed loop. Note that closed loop is useful for cascaded control records.
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
MDT	Minimum Delta Time, in seconds	Minimum time difference between processing in seconds. If this is zero, the minimum time is one clock tick.
KP	Proportional Gain	
KI	Integral Gain, in repeats per minute	The number of times per minute that the integral contribution repeats the proportional contribution.
KD	Derivative Gain, in minutes per repeat	The number of minutes until the derivative contribution repeats the proportional contribution.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	

Name	Summary	Description
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Severity for a Hihi Alarm	
HSV	Severity for a High Alarm	
LSV	Severity for a Low Alarm	
LLSV	Severity for a Lolo Alarm	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
ODEL	Output Delta	This parameter specifies a hysteresis factor for triggering monitor callbacks for `DM`, `P`, `I`, `D`, `CT`, `DT`, `ERR`, and `DERR`. It refers to the change in `DM`. Whenever monitors are triggered for `DM`, monitors for the other fields are also triggered.
CVAL	Value of Controlled Variable, in engineering units	This value is obtained from `CVL` each time the record is processed.
DM	Change in Manipulated Variable	This is the value computed by the `pid` algorithm. It is an increment to be added to the controller output. Note that in most cases this will be read via the `DOL` field of an analog output record. The analog output record will be configured with `OIF` set to incremental.
ODM	Old DM	`ODM` is used to decide if monitors should be triggered for `DM`.
P	Proportional contribution	Proportional contribution to `DM`, in engineering units.
I	Integral contribution to DM, in repeats per minute	The number of times per minute that the integral contribution repeats the proportional contribution.
D	Derivative contribution to DM, in minutes	The number of minutes until the derivative contribtuion repeats the proportional contribution.
CT	Clocks Ticks	Clock ticks when previous process occurred.

Name	Summary	Description
DT	Time difference	Time difference in seconds between processing steps.
ERR	Error	Current error (VAL - CVAL).
DERR	Delta Error	Change in error since last time step.
LALM	Value when last monitors for alarm were triggered	These fields are used to implement the hysteresis factors for monitors.
ALST	Value when last monitors for archiver were triggered	
MLST	Value when last monitors for value changes were triggered	

4. Record Support Routines

init_record

This routine initializes VAL with the value of STPL and sets UDF to FALSE if STPL type is CONSTANT link or creates a channel access link if STPL type is PV_LINK.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to VAL.

get_units

Retrieves EGU.

get_precision

Retrieves PREC.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is P, I, D, CVAL, VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is P, I, D, CVAL, VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine process implements the following algorithm:

1. If CVL is not a database link, a major alarm is declared and the algorithm completes.
2. The current value of CVAL is obtained from CVL.
3. If STPL is a database or channel access link and SMSL is CLOSED_LOOP, then VAL is obtained from STPL and UDF is set to FALSE.
4. The time difference since the last time step is calculated. If it is less than MDT or if no ticks have occurred since the last time the algorithm was executed, process just completes without raising any alarms, checking monitors, or scanning the forward link.
5. The new values of P, I, D, OUT, CT, DT, ERR, and DERR are computed.
6. Check alarms. This routine checks to see if the new VAL causes the alarm status and severity to change. If so, NSEV, NSTA and LALM are set. It also honors the alarm hysteresis factor (HYST). Thus the value must change by more than HYST before the alarm status and severity is lowered.
7. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if ADEL and MDEL conditions are met.
   - Value change monitors on DM are invoked if ODEL conditions are met. If monitors are triggered from DM, they are also triggered for P, I, D, CT, DT, ERR, and DERR.
   - NSEV and NSTA are reset to 0.
8. Scan forward link if necessary, set PACT FALSE, and return.

Chapter 26: pulseCounter

1. Introduction

The normal use for the pulseCounter record type is to record counts.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	ULONG	No	0	Yes	Yes	Yes	No
OUT	OUTLINK	Yes	0	No	No	N/A	No
GTYP	RECCHOICE	Yes	0	Yes	Yes	No	No
HGV	SHORT	Yes	0	Yes	Yes	No	No
SGL	INLINK	Yes	0	No	No	N/A	No
SGV	RECCHOICE	Yes	0	Yes	Yes	No	No
OSGV	SHORT	No	0	Yes	No		No
CSIZ	RECCHOICE	Yes	1	Yes	Yes		No
CNTE	RECCHOICE	Yes	0	Yes	Yes	No	No
CNTS	SHORT	Yes	0	Yes	Yes	No	No
HOPR	FLOAT	Yes	4.3e+9	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
CMD	RECCHOICE	No	0	Yes	Yes	Yes	Yes
SCMD	USHORT	No	0	Yes	No		No
CPTR	ULONG	No	0	Yes	No		No

3. Field Descriptions

Name	Summary	Description
VAL	Counter Value	The read command places the current value of the counter into the `VAL` field.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
GTYP	Gate Type	This can be Hardware or software. If `GTYP` is hardware, then `HGV` determines gating control. If `GTYP` is software, the `SGV` determines gating control.
HGV	Hardware Gate Value	If `GTYP` is hardware, then this field is device dependent.
SGL	Soft Gate Location (Input Link)	If `SGL` is a database link and `GTYP` is software, then `SGV` will be set to the value read from `SGL`.
SGV	Soft Gate Value	This can be inactive or active. This will enable and disable counting if `GTYP` is software.
OSGV	Old Soft Gate Value	This is the previous value of `SGV`.
CSIZ	Counter size	16 bit or 32 bit counter.
CNTE	Count Edge	This can be Rising Edge or Falling Edge. This field forces counting on rising or falling edge of source signal.
CNTS	Count Source	
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support routines to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
CMD	Command	**Read**: Read the current value of the counter. **Clear**: Clear the counter. Note theat the counter is also stopped. The Start command must be issued to restart the counter. **Start**: Start counting. **Stop**: Stop counting. **Setup**: Setup the counter. Counting will not begin until the Start command is issued.
SCMD	Save Command	This is the saved value of `CMD`.
CPTR	Callback	

4. Record Support Routines

init_record

This routine next checks to see that device support is available. If it does not exist, an error message is issued and processing is terminated.

If SGL is a constant and GTYP is software, then SGV is initialized with its value. If SGL type is PV_LINK a channel access link is created.

Device support is then checked to see if cmd_pc is defined.

If device support includes init_record, it is called.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to VAL.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is VAL the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double

Sets the upper control and the lower control limits for a field. If the field is VAL the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

5. Record Processing

The routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If SGL is DB_LINK and GTYP is Software, get SGV from SGL. If SGV has changed, save the CMD value, call the command routine with START if SGV =0 or with STOP if SGV is 1, reset the command to the saved value, and set alarms if return status not zero. If the device is not done (PACT TRUE), then issue a callback request for this record to process and return
3. If CMD is not READ, call command routine and set CMD to READ. If the device is not done (PACT TRUE), then issue a callback request for this record to process again and return.
4. Call command routine. If device support set PACT to TRUE, then return.
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors on CMD are invoked if values have changed.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each record must have an associated set of device support routines. The primary responsibility of the device support routines is to issue commands to the output device. The device support routines are primarily interested in the following fields:

Name	Summary	Description
CSIZ	Counter size	This will determine to a 16 bit or 32 bit count is to be used. With 32 bit, two counters are used.
CMD	Command	The device support routine is responsible for processing the commands `READ`, `CLEAR`, `START`, `STOP`, and `SETUP`.
GTYP,IGV	Gate Type	If `GTYP` is internal, device support is responsible for using `IGV` to determine gating control.
CNTE	Count Edge	This field is used by the device support routines to force counting on leading or falling edge of signal.
CNTS	Count Source	Device support must use `CNTS` to set count source during setup.

Device Support Routines

Device support consists of the following routines:

report

```
report()
```

This routine is optional. If provided, it prints a report of all device modules.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

cmd_pc

```
cmd_pc(precord)
```

This routine issues commands to the output device. It returns the following values:

- **0**: Success.
- **Other**: Error.

Chapter 27: pulseDelay

1. Introduction

The normal use for the `pulseDelay` record type is to generate output pulses.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
OUT	OUTLINK	Yes	0	No	No	N/A	No
UNIT	RECCHOICE	Yes	0	Yes	Yes	No	No
DLY	DOUBLE	Yes	0	Yes	Yes	Yes	Yes
WIDE	DOUBLE	Yes	0	Yes	Yes	Yes	Yes
ODLY	DOUBLE	No	0	Yes	No		No
OWID	DOUBLE	No	0	Yes	No		No
CTYP	RECCHOICE	Yes	0	Yes	Yes	No	No
CEDG	RECCHOICE	Yes	0	Yes	Yes	No	No
ECS	SHORT	Yes	0	Yes	Yes	No	No
ECR	DOUBLE	Yes	0	Yes	Yes	No	No
VAL	RECCHOICE	No	0	Yes	Yes	Yes	No
PFLD	USHORT	No	0	Yes	No		No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LLOW	RECCHOICE	Yes	0	Yes	Yes	No	No
TTYP	RECCHOICE	Yes	0	Yes	Yes	No	No
HTS	ENUM	Yes	0	Yes	Yes	No	Yes
STL	INLINK	Yes	0	No	No	N/A	No
STV	RECCHOICE	Yes	0	Yes	Yes	No	Yes
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
GATE	RECCHOICE	Yes	1	Yes	Yes	No	Yes
GLNK	INLINK	Yes	0	Yes	No	N/A	No

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
UNIT	Time units	Time units of delay and width. (Seconds, Milliseconds, Microseconds, Nanoseconds, Picoseconds).
DLY	Pulse Delay, in UNITS of time	Delay after trigger edge until beginning of pulse.
WIDE	Pulse Width, in UNITS of time	Width of pulse generated.
ODLY	Old Delay	Value when last monitors for delay were triggered.
OWID	Old Width	Value when last monitors for width were triggered.
CTYP	Clock Type	Hardware/Software. If software selected, then clock automatically determined by software. If hardware selected, then clock determined by `ECS` and `ECR`.
CEDG	Clock Signal Edge	This can be Rising Edge or Falling Edge. This field forces clock timing on rising or falling edge of source signal.
ECS	External Clock Source	If `CTYP` is internal, this field is ignored. If `CTYP` is external, then this field is device dependent.
ECR	External Clock Rate, in Hz	Clock rate for external clock source.
VAL	Value	This field is will contain value 1 if a trigger was detected since the last time the record was processed and a 0 otherwise.
PFLD	Processing Field	This field is set to indicate if which of the following fields changed since last processed: `DLY`, `WIDE`, `STV`, `GATE`, or `HTS`.
LLOW	Low Logic Level	0: Logic Low=0 1: Logic Low=1
TTYP	Trigger Type. (Hardware/ Software)	This field indicates where the pulse trigger will come from. Hardware indicates `HTS` will be used, software will use `STL`, `STV`.
HTS	Hardware Trigger source	The source of the delayed pulse trigger.
STL	Soft Trigger Location (Input link)	This value for `STV` will be read from here if this is set.
STV	Soft Trigger Value	This can be enabled or disable. This will trigger a delayed pulse if `TTYP` set to software and device allows it.

Name	Summary	Description
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
PREC	Display Precision	Precision with which to display `DLY`. This field is used by record support to supply a value when `get_precision` is called.
GATE	Gate for enable/ disable of pulse generation	This field can be used to enable and disable the pulses.
GLNK	Gate Location	This field is used to determine where to get the value for `GATE`.

4. Record Support Routines

init_record

This routine first checks that device support is available. Device support is then checked to see if `write_pd` is defined.

Next this routine initializes `STV` with the value of `STL` if `STL` type is `CONSTANT` link or creates a channel access link if `STL` type is `PV_LINK`.

`GATE` is likewise initialized if `GLNK` is `CONSTANT` or `PV_LINK`.

If device support includes `init_record`, it is called.

process

See next section.

special

Sets the `PFLD` field to indicate if write to `DLY`, `STV`, `GATE` or `HTS` field caused processing of the record.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_precision

Retrieves `PREC`.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is `VAL`, `DLY`, `ODLY`, `WIDE` or `OWID` the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is VAL, DLY, ODLY, WIDE or OWID the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. The values for STV and GATE are then fetched.
3. Call write_pd routine.
4. PFLD is reset to zero.
5. If device support set PACT to TRUE, then return.
6. Set UDF to FALSE.
7. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors on DLY and WIDE are invoked if values have changed.
   - NSEV and NSTA are reset to 0.
8. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each record must have an associated set of device support routines. The primary responsibility of the device support routines is to issue commands to the output device. The device support routines are primarily interested in the following fields:

Name	Summary	Description
OUT	Output Link	This field is used by the device support routines to locate its output.
WIDE	Pulse Width	Device support must use WIDE for pulse width
DLY	Pulse Delay	Device support must use DLY for the delay after trigger edge until beginning of pulse.
LLOW	Low Logic Level	Device support must use to determine logic low level.
UNIT	Time Units	All values that refer to time measure will be in this time unit.

Name	Summary	Description
VAL	Value	This field will contain a 1 if a trigger occured since the last time the record was processed if the device supports it.
PFLD	Processing Field	This field is used by some devices to indicate if the record was scanned to adjust certain fields such as delay or trigger source. If the device has a destructive read, then changes to these types of fields will only could writes to the device instead of a read and a write.
TTYP	Trigger Type	This field is used by the device support routines to force triggering on leading or falling edge of signal if the specified device supports it.
HTS	Hardware Trigger Source	This field wil be used to set the hardware trigger source if the device supports it.
STV	Soft Trigger Soure	This field will be used for software to trigger an output delayed pulse if the device supports it.
CEDG	Clock Signal Edge	This field is used by the device support routines to force clock timing on leading or falling edge of signal.
CTYP	Clock Type	If `CTYP` is external, device support is responsible for using `ECR` for the clock rate and if `CTYP` is internal, `ECS` is the clock source.
ECS	External Clock Source	
ECR	External Clock Rate	

Device Support Routines

Device support consists of the following routines:

report

```
report()
```

This routine is optional. If provided, it prints a report of all device modules.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_pd

```
write_pd(precord)
```

This routine issues commands to the output device.

Chapter 28: pulseTrain

1. Introduction

The normal use for the `pulseTrain` record type is to generate an output pulse train.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
OUT	OUTLINK	Yes	0	No	No	N/A	No
UNIT	RECCHOICE	Yes	0	Yes	Yes	No	No
PER	DOUBLE	Yes	0	Yes	Yes	Yes	No
DCY	DOUBLE	Yes	0	Yes	Yes	Yes	No
OPER	DOUBLE	No	0	Yes	No		Yes
ODCY	DOUBLE	No	0	Yes	No		Yes
GTYP	RECCHOICE	Yes	0	Yes	Yes	No	No
HGV	SHORT	Yes	0	Yes	Yes	No	No
SGL	INLINK	Yes	0	No	No	N/A	No
SGV	RECCHOICE	Yes	0	Yes	Yes	No	No
OSGV	SHORT	No	0	Yes	No		No
VAL	SHORT	No	0	Yes	Yes	Yes	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
CTYP	RECCHOICE	Yes	0	Yes	Yes	No	No
CEDG	RECCHOICE	Yes	0	Yes	Yes	No	No
ECS	SHORT	Yes	0	Yes	Yes	No	No
ECR	DOUBLE	Yes	0	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
LLOW	RECCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
UNIT	Units of time	Units of time (Seconds, milliseconds, microseconds, nanoseconds, picoseconds).
PER	Period, in UNITs	Pulse train period.
DCY	Duty Cycle, percent	Percent of time that signal is high.
OPER	Old Period, in UNITS	Value when last monitors for period were triggered.
ODCY	Old Duty Cycle, percent	Value when last monitors for duty cycle were triggered.
GTYP	Gate Type	This can be hardware or software. If `GTYP` is hardware, then `HGV` determines gating control. If `GTYP` is software, the `SGV` determines gating control.
HGV	Hardware Gate Value	This field is device dependant.
SGL	Soft Gate Location (Input Link)	If `SGL` is a database link and `GTYP` is software, then `SGV` will be set to the value read from `SGL`.
SGV	Soft Gate Value	This can be inactive (no gating) or active.
OSGV	Old Soft Gate Value	This is the previous value of `SGV`.
VAL	Value	This field is not used.
CTYP	Clock Type	This can be internal or external.
CEDG	Clock Signal Edge	This can be Rising Edge or Falling Edge. This field forces counting on rising or falling edge of source signal.
ECS	External Clock Source	If `CTYP` is internal, this field is ignored. If `CTYP` is external, then this field is device dependent.
ECR	External Clock Rate, in Hz	Clock rate for external clock source.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
PREC	Display Precision	Precision with which to display `DLY`. This field is used by record support to supply a value when `get_precision` is called.
LLOW	Low Logic Level	Logic Low=0 Logic Low=1

4. Record Support Routines

init_record

This routine first checks that device support is available. If SGL is a constant then HGV is initialized with its value or a channel access link is created if SGL type is PV_LINK.

Device support is then checked to see if write_pt is defined.

If device support includes init_record, it is called.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to VAL.

get_precision

Retrieves PREC.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is VAL, PER, or OPER the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double

Sets the upper control and the lower control limits for a field. If the field is VAL or PER the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If SGL is DB_LINK and GTYP is Software, get SGV from SGL. If SGV has changed, save the duty cycle DCY value, call the write_pt routine with duty cycle =0, reset the duty cycle to the saved value, and set alarms if return status not zero. Then set the old soft gate value OSGV to SGV.
3. Call write_pt routine. If device support set PACT to TRUE, then return.
4. Set UDF to FALSE.
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors on PER and DCY are invoked if values have changed.
   - NSEV and NSTA are reset to 0.

6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each record must have an associated set of device support routines. The primary responsibility of the device support routines is to issue commands to the output device. The device support routines are primarily interested in the following fields:

Name	Summary	Description
UNIT	Units of time	This field will be used to identify the time units used for time fields.
OUT	Output Link	This field is used by the device support routines to locate its output.
PER	Period, in UNITs	Device support must use PER for pulse period.
DCY	Duty Cycle, percent	Device support must use DCY for the percent of time the signal is high.
LLOW	Low Logic Level	Device support must use to determine logic low level.
CEDG	Clock Signal Edge	This field is used by the device support routines to force counting on leading or falling edge of signal.
GTYP	Gate Type	Device support is responsible for using IGV to determine gating control if GTYP is internal, or SGV if GTYP is external.
IGV		
SGV	Soft Gate Value	
CTYP	Clock Type	If CTYP is external, device support is responsible for using ECR for the clock rate and if CTYP is internal, ECS is the clock source.
ECS	External Clock Source	
ECR	External Clock Rate, in Hz	

Device Support Routines

Device support consists of the following routines:

report

```
report()
```

This routine is optional. If provided, it prints a report of all device modules.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_pt

```
write_pt(precord)
```

This routine issues commands to the output device. It returns the following values:

- **0**: Success.
- **Other**: Error.

Chapter 29: scan

Ned D. Arnold
Advanced Photon Source
Argonne National Laboratory

1. Introduction

The basic function of a scan record is to move "positioners" through a series of steps and record "detector" data at each of the positions (the whole sequence is referred to as a "scan"). Once the scan parameters are properly initialized, the scan record coordinates the entire scan and notifies any interested "clients" when the scan is complete. The data is stored in arrays within the record rather than collected point by point by an external application program. This allows for much faster scans than those coordinated from an external application program on a point to point basis.

A single scan record supports a one dimensional scan. It is also possible to link scan records together to define multi-dimensional scans in a quite complex configuration.

Each scan record can control up to four "positioners" and acquire data from up to four process variables (typically detector data or measured positions of devices) during a scan. Two additional output variables can be defined to trigger other process variables (usually "detectors") between the positioning phase and the data acquisition phase. These outputs will be referred to as "detector triggers".

Although the typical use of a scan record is to move "positioners" and record "detector" data at each position, it can also be used for other applications. Any controllable device can be scanned through incremental values while recording data from any other process variables. For example, one of the "positioner" PVs could be used to vary the gain of a detector or the speed of a motor during a scan. Another example would be to use the scan record to vary several power supplies and record the beam position at each value of the supplies. In this context, the scan record becomes a general purpose "Vary w, x, y, z and record a, b, c, d" record. Therefore, throughout this document the word "positioner" and "controller" will be used synonymously. When referring to the data being recorded at each point, the word "detector" will be used.

All of the process variable names used to identify controllers, detectors, and detector triggers are specified using “reassignable links”. This allows a scan to be configured on the fly. Scan parameters, including the names of controllers and detectors, can be saved and restored using the BURT.

NOTE: In this version, the PVs used in the “reassignable links” fields must reside in the same IOC.

A Simple Single Dimensional Scan

The simplest database configuration for using a `scan` record is shown in Figure 1. A thorough understanding of the operation of this configuration will allow more complex scans to be developed easily.

In Figure 1, when the scan is initiated, the `scan` record commands several positioners (transform record and motor record) to move to their starting positions. The WAIT_1 record detects when all movement is complete and forces the SCAN record to process again. The SCAN record realizes that the positioning is complete and thus triggers the Detectors. The WAIT_2 record detects when data is valid and forces the SCAN record to process yet again. The SCAN record will then read the Detector Data and command the positioners to their next step. This will continue until the SCAN record has completed the appropriate number of steps. At the end of the scan, the SCAN record contains data arrays for each of the “detectors”, as well as arrays that contain the positions to which each controller was commanded at each point. A simple x-y plot using this data will provide the detector data vs. position results.

Figure 1: Typical Database Support for a SCAN

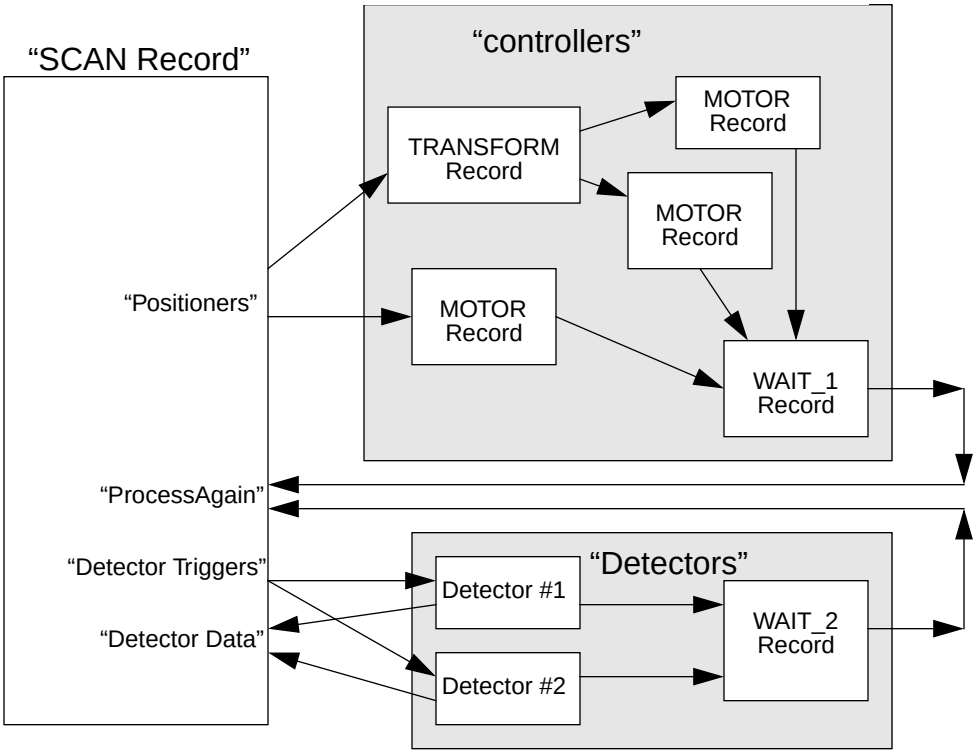
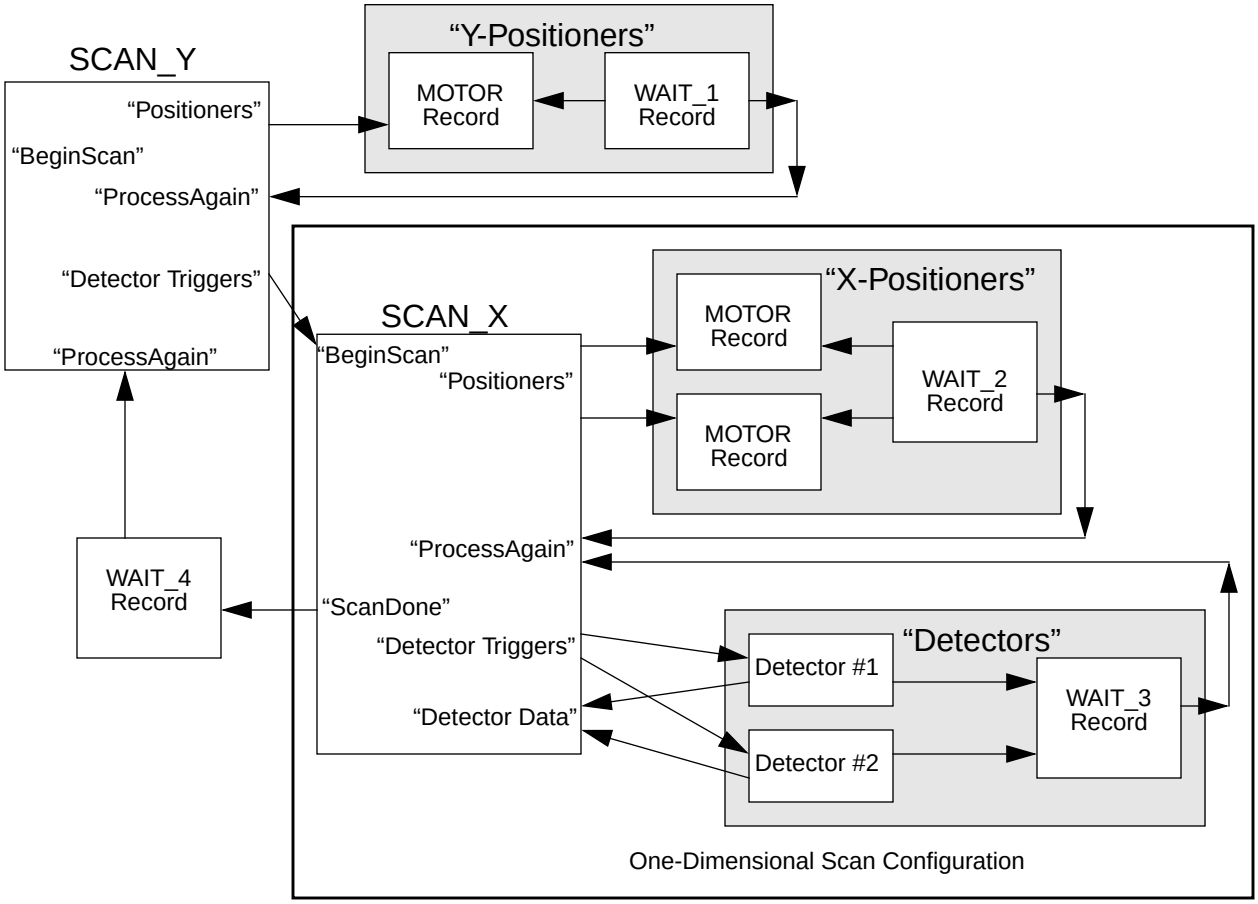


Figure 2: A Two Dimensional Scan Implementation



Two Dimensional Scanning

Figure 2 illustrates using two scan records to accomplish a two dimensional scan. The SCAN_X record controls the positioners for the X axis, the SCAN_Y record controls the Y positioner.

To initiate a scan, the SCAN_Y record is commanded to begin. It commands its "positioners" to the specified starting point. The WAIT_1 record detects when all motors are stopped and forces the SCAN_Y record to process again. The SCAN_Y record will now write to its "Detector Trigger", which in this case begins a scan of the SCAN_X record. The SCAN_X record will now go through its entire programmed scan, acquiring data from the detectors at each point.

When the SCAN_X record is complete, the WAIT_4 record will force the processing of SCAN_Y, which will increment the position of the y-controller and initiate the x scan once again.

This approach to configuring a two dimensional scan is very flexible. Note that to "test" the x scan, one could write to the "BeginScan" field of SCAN_X which would perform an entire x scan. Although the SCAN_Y record would get processed after the x scan was complete (via the WAIT_4 record), nothing would happen because it was not in the middle of a scan. In addition, any of the motors can be moved individually when a scan is not in process without

any unexpected behavior (detectors would not be triggered unless the operator did it deliberately). One could even build a three dimensional scan by adding an additional scan record that initiates the y-scan after positioning a z-controller.

Defining a Scan

Several options are available to control the execution of a scan. All parameters must be properly configured prior to initiating the scan. This section describes the options that are available while the next section itemizes all the parameters required.

Positioners

Each scan record may control up to four "positioners" that are commanded to a new "desired position" prior to collecting data at each point. The positioners are defined by typing in an ASCII string that represents the process variable name of the controller.

There are three modes for determining the desired value for the positioner. If a positioner is specified as "`linear`", its desired value is determined by using parameters such as `start_position`, `step_increment`, `number_of_steps`, and `end_position`. This mode will be discussed in more detail in the next paragraph. If a positioner is specified as "`lookup`", its next position is found in an array that has been loaded into the record prior to initiating a scan. If the positioner is specified as "`on-the-fly`", it is commanded to the `end_position` after the first data point is collected and not changed again for the duration of the scan.

A linear scan involving a single positioner is fully defined by three parameters, e.g., the start position, the step size, and the number of data points. A scan involving `N` controllers is defined by merely `2N+1` parameters, since the number of data points must be the same for all controllers. For the convenience of interactive users, and to support channel-access clients that define scans differently, the `scan` record provides for six redundant scan-definition parameters (for the first controller only): `START`, `END`, `CENTER`, `WIDTH`, `STEP`, and `NPTS`. The record calculates values for unspecified parameters so that the set is always self consistent, and it imposes an upper limit (`MAXPTS`) on `NPTS`. `WIDTH` may be negative.

There is no unique prescription for removing inconsistencies among redundant parameters, and no hardcoded set of preferences among parameters is likely to please everyone. Therefore, the `scan` record allows the user to "freeze" parameters, so that they will not be changed in the record's internal attempts to ensure self-consistency of the parameter set. Frozen parameters can be changed by the user and by any other client. It is the user's responsibility to ensure that frozen parameters do not prevent freely specifying unfrozen parameters. For example, if both `STEP` and `NPTS` are frozen, changes to `WIDTH` will be rejected. Similarly, if both `START` and `CENTER` are frozen, changes to `END` and `WIDTH` will have no effect. By default, `START`, `STEP`, and `NPTS` are frozen. When the record cannot adjust the parameters to be consistent, a flag is raised (`ALRT`) and a message reported (`SMSG`).

Although this approach may seem to present the user with an overwhelming number of choices, it should be noted that "by default" the user only has to enter `START`, `STEP`, and `NPTS` to fully define the scan of a controller. The "user interface" (usually `medm` or another CA client) need only present the user with these fields. However, by changing the "freeze flags" from the default and presenting the user with different fields to fill in, the scan can be defined in a completely flexible way. The result is that a simple scan can be defined easily, but advanced users are not limited in flexibility.

For those controllers defined as "`linear`", each desired position is placed in the "desired position" array that is used for lookup mode. Therefore, after the scan is complete, this array will always contain the sequence of desired positions to which the controller was commanded.

These values will have been provided by the user for "`lookup`" mode and calculated by the `scan` record for "`linear`" mode. This array will contain no useful data if "`on-the-fly`" mode was used.

Position Verification Readback Process Variable and Delta

For each positioner, the user may specify a process variable that corresponds to the actual (or measured) position of the motor. If this readback field is configured, the `scan` record will confirm after each movement that the actual position is within a specified delta to the desired position. If the delta is exceeded, the scan will abort and the record will go into an alarm state. A text field within the record (`SMSG`) will inform the operator of the error condition.

Detector Trigger Process Variables and Desired Command

If valid process variable names are entered into the "Detector Trigger" fields, the `scan` record will write the specified "Desired Command" (a floating point number) to that process variable between the positioning phase and the data acquisition phase.

If neither Detector Trigger contains a valid PV, the `scan` record will skip this step and acquire the data immediately. Note that specifying a "Detector Trigger" requires the `scan` record to be processed an additional time each point.

Data to be Acquired

Each `scan` record can acquire data from four process variables at each point in the scan. This data will most commonly be from a detector or from a position readback (which would record the actual motor positions at each point and could then be compared to the desired position array).

Reading SCAN Results

The scan results will most frequently be read as arrays of positions and arrays of detector data where each detector data element corresponds to the position element.

For single dimension scans, the scan is complete when the ExecuteScan flag is set back to zero by the record processing routine. The application program can then read the position arrays and the data arrays (or have a monitor set on them so the record will "post_monitors" when complete).

For two dimension scans similar to Figure 2, the application program should read the arrays from the SCAN_X record after the completion of each x scan and correlate it to the current y controller information. This will allow the application program to display data after each x scan. The `scan` record will buffer the data for only one x scan, so the application must read the arrays before the next x scan is completed. If the scan is fast enough that this may prevent a problem, the application program can contribute to the `wait` record algorithm such that the y - controllers are not moved until the application program has completely read the previous SCAN_X data.

On slow scans, the application program may want to see the scan is process on a point by point basis. Therefore, the scan record will "`post_monitors`" on fields that it updates each point, but it will not post monitors faster than 20 times per second. If a scan is proceeding at a rate less than 20 points per second, every point will be posted. If a scan is proceeding at 100 steps per second, scalar values will be posted every 5th point (approx). In either case, the array data will contain every point at the completion of the scan. It is not recommended that the application program use the point to point data except for keeping the operator aware of the progress of the scan.

Checking Scan Parameters Against Limits

Prior to beginning an actual scan, the record can be commanded to check the scan parameters to ensure that all positioner requests are within reasonable limits. The record will do a "dry run" by calculating every positioner value (or looking it up in the table) and comparing it with the "drive limits" associated with that positioner's Process Variable (drive limits are an attribute of most process variables). If any step would exceed the drive limits, the operator is notified via the `SMSG` field.

2. Configurable Parameters

A brief summary of the configurable parameters within a `scan` record is presented below. For a more detailed explanation, see Field Description section.

- Record Initialization
  - Maximum Number of Steps in any scan (used to allocate memory)
- SCAN Definition Parameters
  - Number of steps for This Scan
  - SCAN Record SCAN Mechanism (Normal Choices for any record)
- Positioner Definition (Four positioners)
  - PV Name
  - PV Name Valid Flag (monitor only)
  - "Lookup/Linear/On-the-Fly" Mode Selection
  - Start Position and Freeze Flag
  - Width and Freeze Flag
  - Center and Freeze Flag
  - Step Increment and Freeze Flag
  - End Position and Freeze Flag
  - Desired Position Array
  - Position Verification Readback PV Name
  - Position Verification Readback PV Name Valid Flag (monitor only)
  - Position Verification Delta
  - Current Position (monitor only)
- Detector Definition (Four Detectors)
  - PV Name
  - PV Name Valid Flag (monitor only)
  - Current value (monitor only)
  - Detector Data Array (monitor only)
- Detector Trigger (if desired)
  - PV Name
  - Command to write

3. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VERS	FLOAT	No	1.0	Yes	No	No	No
VAL	DOUBLE	No	0	Yes	Yes	No	No
SMSG	STRING	No	Null	Yes	Yes	Yes	No
CMND	ENUM	No	0	Yes	Yes	Yes	No
ALRT	UCHAR	No	0	Yes	No	Yes	No
RPVT	NOACCESS	No	Null	No	No	No	No
MPTS	SHORT	Yes	100	Yes	No	No	No
EXSC	SHORT	No	0	Yes	Yes	Yes	No
PXSC	UCHAR	No	0	Yes	No	No	No
NPTS	SHORT	Yes	100	Yes	Yes	Yes	No
FPTS	RECCHOICE	Yes	1	Yes	Yes	No	No
CPT	SHORT	No	0	Yes	No	Yes*	No
PCPT	SHORT	No	0	Yes	No	No	No
TOLP	ULONG	No	0	Yes	No	No	No
P1PV	STRING	Yes	Null	Yes	Yes	No	No
P2PV	STRING	Yes	Null	Yes	Yes	No	No
P3PV	STRING	Yes	Null	Yes	Yes	No	No
P4PV	STRING	Yes	Null	Yes	Yes	No	No
R1PV	STRING	Yes	Null	Yes	Yes	No	No
R2PV	STRING	Yes	Null	Yes	Yes	No	No
R3PV	STRING	Yes	Null	Yes	Yes	No	No
R4PV	STRING	Yes	Null	Yes	Yes	No	No
T1PV	STRING	Yes	Null	Yes	Yes	No	No
T2PV	STRING	Yes	Null	Yes	Yes	No	No
D1PV	STRING	Yes	Null	Yes	Yes	No	No
D2PV	STRING	Yes	Null	Yes	Yes	No	No
D3PV	STRING	Yes	Null	Yes	Yes	No	No
D4PV	STRING	Yes	Null	Yes	Yes	No	No
P1DB	NOACCESS	No	Null	No	No	No	No
P2DB	NOACCESS	No	Null	No	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
P3DB	NOACCESS	No	Null	No	No	No	No
P4DB	NOACCESS	No	Null	No	No	No	No
R1DB	NOACCESS	No	Null	No	No	No	No
R2DB	NOACCESS	No	Null	No	No	No	No
R3DB	NOACCESS	No	Null	No	No	No	No
R4DB	NOACCESS	No	Null	No	No	No	No
T1DB	NOACCESS	No	Null	No	No	No	No
T2DB	NOACCESS	No	Null	No	No	No	No
D1DB	NOACCESS	No	Null	No	No	No	No
D2DB	NOACCESS	No	Null	No	No	No	No
D3DB	NOACCESS	No	Null	No	No	No	No
D4DB	NOACCESS	No	Null	No	No	No	No
P1NV	LONG	No	0	Yes	Yes	Yes	No
P2NV	LONG	No	0	Yes	Yes	Yes	No
P3NV	LONG	No	0	Yes	Yes	Yes	No
P4NV	LONG	No	0	Yes	Yes	Yes	No
R1NV	LONG	No	0	Yes	Yes	Yes	No
R2NV	LONG	No	0	Yes	Yes	Yes	No
R3NV	LONG	No	0	Yes	Yes	Yes	No
R4NV	LONG	No	0	Yes	Yes	Yes	No
T1NV	LONG	No	0	Yes	Yes	Yes	No
T2NV	LONG	No	0	Yes	Yes	Yes	No
D1NV	LONG	No	0	Yes	Yes	Yes	No
D2NV	LONG	No	0	Yes	Yes	Yes	No
D3NV	LONG	No	0	Yes	Yes	Yes	No
D4NV	LONG	No	0	Yes	Yes	Yes	No
P1SM	RECCHOICE	Yes	0	Yes	Yes	No	No
P1SP	FLOAT	Yes	0	Yes	Yes	Yes	No
P1FS	RECCHOICE	Yes	0	Yes	Yes	No	No
P1SI	FLOAT	Yes	0	Yes	Yes	Yes	No
P1FI	RECCHOICE	Yes	0	Yes	Yes	No	No
P1CP	FLOAT	Yes	0	Yes	Yes	Yes	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
P1FC	RECCHOICE	Yes	0	Yes	Yes	No	No
P1EP	FLOAT	Yes	0	Yes	Yes	Yes	No
P1FE	RECCHOICE	Yes	0	Yes	Yes	No	No
P1WD	FLOAT	Yes	0	Yes	Yes	Yes	No
P1FW	RECCHOICE	Yes	0	Yes	Yes	No	No
P1DV	FLOAT	No	0	Yes	No	Yes*	No
P1LV	FLOAT	No	0	Yes	No	No	No
R1CV	FLOAT	No	0	Yes	No	Yes*	No
R1LV	FLOAT	No	0	Yes	No	No	No
R1DL	FLOAT	Yes	0	Yes	Yes	No	No
P1EU	STRING	Yes	16	Yes	Yes	No	No
P1HR	FLOAT	Yes	0	Yes	Yes	No	No
P1LR	FLOAT	Yes	0	Yes	Yes	No	No
P1PR	SHORT	Yes	0	Yes	Yes	No	No
P1PA	FLOAT ARRAY	No	Null	Yes	Yes	Yes	No
P2SM	RECCHOICE	Yes	0	Yes	Yes	No	No
P2SP	FLOAT	Yes	0	Yes	Yes	Yes	No
P2FS	RECCHOICE	Yes	0	Yes	Yes	No	No
P2SI	FLOAT	Yes	0	Yes	Yes	Yes	No
P2FI	RECCHOICE	Yes	0	Yes	Yes	No	No
P2CP	FLOAT	Yes	0	Yes	Yes	Yes	No
P2FC	RECCHOICE	Yes	0	Yes	Yes	No	No
P2EP	FLOAT	Yes	0	Yes	Yes	Yes	No
P2FE	RECCHOICE	Yes	0	Yes	Yes	No	No
P2WD	FLOAT	Yes	0	Yes	Yes	Yes	No
P2FW	RECCHOICE	Yes	0	Yes	Yes	No	No
P2DV	FLOAT	No	0	Yes	No	Yes*	No
P2LV	FLOAT	No	0	Yes	No	No	No
R2CV	FLOAT	No	0	Yes	No	Yes*	No
R2LV	FLOAT	No	0	Yes	No	No	No
R2DL	FLOAT	Yes	0	Yes	Yes	No	No
P2EU	STRING	Yes	16	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
P2HR	FLOAT	Yes	0	Yes	Yes	No	No
P2LR	FLOAT	Yes	0	Yes	Yes	No	No
P2PR	SHORT	Yes	0	Yes	Yes	No	No
P2PA	FLOAT ARRAY	No	Null	Yes	Yes	Yes	No
P3SM	RECCHOICE	Yes	0	Yes	Yes	No	No
P3SP	FLOAT	Yes	0	Yes	Yes	Yes	No
P3FS	RECCHOICE	Yes	0	Yes	Yes	No	No
P3SI	FLOAT	Yes	0	Yes	Yes	Yes	No
P3FI	RECCHOICE	Yes	0	Yes	Yes	No	No
P3CP	FLOAT	Yes	0	Yes	Yes	Yes	No
P3FC	RECCHOICE	Yes	0	Yes	Yes	No	No
P3EP	FLOAT	Yes	0	Yes	Yes	Yes	No
P3FE	RECCHOICE	Yes	0	Yes	Yes	No	No
P3WD	FLOAT	Yes	0	Yes	Yes	Yes	No
P3FW	RECCHOICE	Yes	0	Yes	Yes	No	No
P3DV	FLOAT	No	0	Yes	No	Yes*	No
P3LV	FLOAT	No	0	Yes	No	No	No
R3CV	FLOAT	No	0	Yes	No	Yes*	No
R3LV	FLOAT	No	0	Yes	No	No	No
R3DL	FLOAT	Yes	0	Yes	Yes	No	No
P3EU	STRING	Yes	16	Yes	Yes	No	No
P3HR	FLOAT	Yes	0	Yes	Yes	No	No
P3LR	FLOAT	Yes	0	Yes	Yes	No	No
P3PR	SHORT	Yes	0	Yes	Yes	No	No
P3PA	FLOAT ARRAY	No	Null	Yes	Yes	Yes	No
P4SM	RECCHOICE	Yes	0	Yes	Yes	No	No
P4SP	FLOAT	Yes	0	Yes	Yes	Yes	No
P4FS	RECCHOICE	Yes	0	Yes	Yes	No	No
P4SI	FLOAT	Yes	0	Yes	Yes	Yes	No
P4FI	RECCHOICE	Yes	0	Yes	Yes	No	No
P4CP	FLOAT	Yes	0	Yes	Yes	Yes	No
P4FC	RECCHOICE	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
P4EP	FLOAT	Yes	0	Yes	Yes	Yes	No
P4FE	RECCHOICE	Yes	0	Yes	Yes	No	No
P4WD	FLOAT	Yes	0	Yes	Yes	Yes	No
P4FW	RECCHOICE	Yes	0	Yes	Yes	No	No
P4DV	FLOAT	No	0	Yes	No	Yes*	No
P4LV	FLOAT	No	0	Yes	No	No	No
R4CV	FLOAT	No	0	Yes	No	Yes*	No
R4LV	FLOAT	No	0	Yes	No	No	No
R4DL	FLOAT	Yes	0	Yes	Yes	No	No
P4EU	STRING	Yes	16	Yes	Yes	No	No
P4HR	FLOAT	Yes	0	Yes	Yes	No	No
P4LR	FLOAT	Yes	0	Yes	Yes	No	No
P4PR	SHORT	Yes	0	Yes	Yes	No	No
P4PA	FLOAT ARRAY	No	Null	Yes	Yes	Yes	No
D1CV	FLOAT	No	0	Yes	No	Yes*	No
D1LV	FLOAT	No	0	Yes	No	No	No
D1EU	STRING	Yes	16	Yes	Yes	No	No
D1HR	FLOAT	Yes	0	Yes	Yes	No	No
D1LR	FLOAT	Yes	0	Yes	Yes	No	No
D1PR	SHORT	Yes	0	Yes	Yes	No	No
D1DA	FLOAT ARRAY	No	Null	Yes	No	Yes	No
D2CV	FLOAT	No	0	Yes	No	Yes*	No
D2LV	FLOAT	No	0	Yes	No	No	No
D2EU	STRING	Yes	16	Yes	Yes	No	No
D2HR	FLOAT	Yes	0	Yes	Yes	No	No
D2LR	FLOAT	Yes	0	Yes	Yes	No	No
D2PR	SHORT	Yes	0	Yes	Yes	No	No
D2DA	FLOAT ARRAY	No	Null	Yes	No	Yes	No
D3CV	FLOAT	No	0	Yes	No	Yes*	No
D3LV	FLOAT	No	0	Yes	No	No	No
D3EU	STRING	Yes	16	Yes	Yes	No	No
D3HR	FLOAT	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
D3LR	FLOAT	Yes	0	Yes	Yes	No	No
D3PR	SHORT	Yes	0	Yes	Yes	No	No
D3DA	FLOAT ARRAY	No	Null	Yes	No	Yes	No
D4CV	FLOAT	No	0	Yes	No	Yes*	No
D4LV	FLOAT	No	0	Yes	No	No	No
D4EU	STRING	Yes	16	Yes	Yes	No	No
D4HR	FLOAT	Yes	0	Yes	Yes	No	No
D4LR	FLOAT	Yes	0	Yes	Yes	No	No
D4PR	SHORT	Yes	0	Yes	Yes	No	No
D4DA	FLOAT ARRAY	No	Null	Yes	No	Yes	No
T1CD	FLOAT	Yes	0	Yes	Yes	No	No
T2CD	FLOAT	Yes	0	Yes	Yes	No	No

* Monitors on these fields are not posted any faster than 20 Hz. Some changes may not be posted! Do not rely on these fields for step-to-step info.

4. Field Descriptions

This section describes the fields that will be of interest to a typical application developer. All array field names end with the character “A”. It is hoped that this convention will make it easier to remember field name abbreviations.

Name	Summary	Description
VERS	Code Version	Reflects the version of `scan` record processing routines.
VAL	Value Field	Not Used.
SMSG	State Message	This field contains a message from the record alerting the operator to any error conditions. It can be cleared by writing a ‘0’ to the `CMND` field.
CMND	Command Field	This field is used to send commands to the record. The following commands are currently defined: 0 - Clear the State Message Field 1 - Execute a “dry run” scan and check the desired positions against the control limits of the positioner(s).
ALRT	Alert Field	This field is set to one when an alert condition exists in the record (e.g Scan parameters too constrained, defined scan exceeds limits, scan aborted, etc). The cause of the alert will be indicated in the State Message Field (`SMSG`).

Name	Summary	Description
RPVT	Record Private	Pointer to a structure that maintains information about the `scan` record.
MPTS	Maximum Number of Points	This field is used to specify the maximum number of points that will be used for any scan defined for this record. This value is used to allocate memory for all the array oriented fields, so the value will significantly affect memory usage.
EXSC	Execute Scan	Writing a '1' to this field will initiate a scan. Writing a '0' will abort the scan. The record will reset this field to '0' when the scan is finished.
PXSC	Previous Execute Scan	Status of the `EXSC` flag the last time the record was processed.
NPTS	Number of Points	This entry defines the number of points for the scan. If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent. This value is constrained to be less than or equal to `MPTS`.
FPTS	Freeze Flag	
CPT	Current Point	This field contains the current point of an active scan. The posting of events on this field is throttled to 20 Hz, so for fast scans not every new value will be posted.
PCPT	Previous Current Point	This field contains the value of `CPT` that was last posted (not every change is posted, see `CPT` desctiption).
TOLP	Time of Last Posting	Tick count of last time monitors were posted on the "throttled" fields (e.g. `CPT`)
PnPV	Positioner n Process Variable	(n=1-4): These fields contain the Process Variable names of the controllers that will be commanded during the scan.
PnNV	Positioner n Name Valid	(n=1-4): These flags indicate if the ASCII string entered in `PnPV` was found to be an existing Process Variable.
PnDV	Positioner n Desired Value	(n=1-4): These fields contain the desired value for each postioner for the current point in the scan. The posting of events on these fields is throttled to 20 Hz, so for fast scans not every "desired value" will be posted.
PnLV	Positioner n Last Value	(n=1-4): Last value 'posted' for `PnDv`.
RnPV	Readback n Process Variable	(n=1-4): If specified, the `scan` record will read this "actual position" of the controller to confirm it has reached its "desired position". If this value is within a specified deadband of the desired position, the scan will continue.
RnNV	Readback n Name Valid	(n=1-4): These flags indicate if the ASCII string entered in `RnPV` was found to be an existing Process Variable.
RnCV	Readback n Current Value	(n=1-4) : These fields contain the current value for each postioner readback for the current point in the scan. The posting of events on these fields is throttled to 20 Hz, so for fast scans not every new value will be posted.

Name	Summary	Description
RnLV	Readback n Last Value	(n=1-4): Last value 'posted' for `RnCV`.
RnDL	Readback n Delta	(n=1-4) : If `RnPV` is specified, the scan record will confirm that the Read Back value is within this dead band, or the scan will be halted.
PnSM	Positioner n Step Mode	(n=1-4) : Indicates if the desired positions for a controller are to be calculated in a linear fashion (incremented by step size), found in a "desired position array", or given a final position command at the beginning of the scan (`on-the-fly`).
PnPA	Positioner n Position Array	(n=1-4): This array contains the step positions for a controller if it uses the "`lookup`" mode (see `PnSM`). This array is filled in by the record during a scan if the controller was in "`linear`" mode. If the number of steps was less than `MPTS`, the remainder of the array is filled with the data from the last step.
PnSP	Positioner n Start Position	(n=1-4): For controllers in the "`linear`" or "`on-the-fly`" mode, this field specifies the start position for the controller. If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent.
PnFS	Positioner n Freeze Flag	
PnSI	Positioner n Step Increment	(n=1-4): For controllers in the "`linear`" mode, this field specifies the step increment. If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent.
PnFI	Positioner n Freeze Flag	
PnEP	End Position	(n=1-4): For controllers in the "`linear`" mode, this field contains the last position to which the controller will be commanded. If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent.
PnFE	Positioner n Freeze Flag	
PnCP	Positioner n Center Position	(n=1-4): For controllers in the "`linear`" mode, this field may be used to define the center position of a scan. If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent.
PnFC	Positioner n Freeze Flag	
PnWD	Positioner n Width	(n=1-4): For controllers in the "`linear`" mode, this field may be used to define the width of the scan (the distance from the start position to the finish position). If the freeze flag is set, the record will not change this value in its attempt to keep the scan definition parameters consistent.
PnFW	Positioner n Freeze Flag	
PnLR	Positioner n Low Range	(n=1-4): These are user configurable fields to describe low and high ranges for the positioner data (used by some channel access clients).
PnHR	Positioner n High Range	

Name	Summary	Description
PnPR	Positioner n Precision	(n=1-4) : This is a user configurable fields to describe the display precision for the positioner data (used by some channel access clients).
PnEU	Positioner n Engineering Units	(n=1-4) : This is a user configurable fields to describe the units for the positioner data (used by some channel access clients).
TnPV	Detector Trigger n Process Variable	(n=1-2): These fields contain the Process Variable names of the “Detector Triggers” that are written to between the positioning phase and the data acquisition phase of the record.
TnNV	Trigger n Name Valid	(n=1-2): These flags indicate if the ASCII string entered in `TnPV` was found to be an existing Process Variable.
TnCD	Trigger n Command	(n=1-2): This is the data that is written to the Detector Trigger PV’s.
DnPV	Data n Process Variable	(n=1-4): These fields contain the Process Variable names of the data that will be recorded at each point within the scan.
DnNV	Data n Name Valid	(n=1-4): These flags indicate if the ascii string entered in `DnPV` was found to be an existing Process Variable.
DnCV	Detector n Current Value	(n=1-4): These fields contain the current value for each detector for the current point in the scan. The posting of events on these fields is throttled to 20 Hz, so for fast scans not every new value will be posted.
DnLV	Detector n Last Value	(n-1-4): Last value ‘posted’ for `DnCV`.
DnDA	Detector n Data Array	(n=1-4): This array (length = `MPTS`) contains the detector data for each point in the scan. If the number of steps was less than `MPTS`, the remainder of the array is filled with the data from the last step.
DnLR	Detector n Low Range	(n=1-4): These are user configurable fields to describe low and high ranges for the detector data (used by some channel access clients).
DnHR	Detector n High Range	
DnPR	Detector n Precision	(n=1-4): This is a user configurable fields to describe the display precision for the detector data (used by some channel access clients).
DnEU	Detector n Engineering Units	(n=1-4): This is a user configurable fields to describe the units for the detector data (used by some channel access clients).
PnDB	Positioner n dbAddr	(n=1-4): Pointer to the `dbAddr` structure of the PV entered in `PnPv`. If PV is not found, the value is `NULL`.
RnDB	Readback n dbAddr	(n=1-4): Pointer to the `dbAddr` structure of the PV entered in `RnPv`. If PV is not found, the value is `NULL`.

Name	Summary	Description
TnDB	Trigger n dbAddr	(n=1-2): Pointer to the `dbAddr` structure of the PV entered in `TnPv`. If PV is not found, the value is `NULL`.
DnDB	Detector n dbAddr	(n=1-4): Pointer to the `dbAddr` structure of the PV entered in `DnPv`. If PV is not found, the value is `NULL`.

5. Record Support Routines

6. Record Processing

7. Device Support

8. Device Support For Soft Records

Chapter 30: sel - Select

1. Introduction

The `sel` record computes a value based on input obtained from up to 12 inputs. The selection algorithm can be one of the following: Specified, Highest, Lowest, Median. Each input can be a constant, a database link, or a channel access link.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	No	Yes	No
SELM	RECCHOICE	Yes	0	Yes	Yes	No	No
SELN	USHORT	No	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
NVL	INLINK	Yes	0	No	No	N/A	No
INPA	INLINK	Yes	0	No	No	N/A	No
INPB	INLINK	Yes	0	No	No	N/A	No
INPC	INLINK	Yes	0	No	No	N/A	No
INPD	INLINK	Yes	0	No	No	N/A	No
INPE	INLINK	Yes	0	No	No	N/A	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
INPF	INLINK	Yes	0	No	No	N/A	No
INPG	INLINK	Yes	0	No	No	N/A	No
INPH	INLINK	Yes	0	No	No	N/A	No
INPI	INLINK	Yes	0	No	No	N/A	No
INPJ	INLINK	Yes	0	No	No	N/A	No
INPK	INLINK	Yes	0	No	No	N/A	No
INPL	INLINK	Yes	0	No	No	N/A	No
A	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
B	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
C	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
D	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
E	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
F	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
G	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
H	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
I	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
J	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
K	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
L	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
LA	DOUBLE	No	0	Yes	No	No	No
LB	DOUBLE	No	0	Yes	No	No	No
LC	DOUBLE	No	0	Yes	No	No	No
LD	DOUBLE	No	0	Yes	No	No	No
LE	DOUBLE	No	0	Yes	No	No	No
LF	DOUBLE	No	0	Yes	No	No	No
LG	DOUBLE	No	0	Yes	No	No	No
LH	DOUBLE	No	0	Yes	No	No	No
LI	DOUBLE	No	0	Yes	No	No	No
LJ	DOUBLE	No	0	Yes	No	No	No
LK	DOUBLE	No	0	Yes	No	No	No
LL	DOUBLE	No	0	Yes	No	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is the computed value, determined as a result of record processing.
SELM	Select Mechanism	`SELECTED`: Use `SELN` as index (0 to 15) `SELECT_HIGH`: Select highest `SELECT_LOW`: Select lowest `SELECT_MEDIAN`: Select median value.
SELN	Select Number	Index of selected input. If `SELM=SELECTED`, then this is the index (0 to 15) of the input to select.
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
NVL	Index Value Location, an input link	IF `NVL` is a constant, `SELN` is set to the constant value. If `NVL` is a database or channel access link then `SELN` is read from `NVL`.
INPA,...,INPL	Input A, Input B, ...	The input links. Each may be a constant, a database link, or a channel access link. Any link not defined is ignored. NOTE: In order to implement `SELM` it is necessary to recognize missing values. The value `1e30` was selected to represent MISSING. If a link is a constant with value 0 (zero, the default) then at record initialization time the corresponding A, ... L field is set equal to MISSING.
A,...,L	A, B, ...	The input values If the corresponding `INP` field is a constant, this field is initialized with the constant value but can be changed via `dbPuts`.
LA,...,LL	Last A, Last B, ...	Previous input values These fields are used to decide when to trigger monitors on A,...,L.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	

Name	Summary	Description
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	Severity for a Hihi Alarm	
HSV	Severity for a High Alarm	
LSV	Severity for a Low Alarm	
LLSV	Severity for a Lolo Alarm	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
LALM	Last Alarmed Value	These fields are used to implement the hysteresis factors for monitors.
ALST	Archive Last Value	
MLST	Monitor Last Value	

4. Record Support Routines

init_record

IF `NVL` is a constant, `SELN` is set to its value. If `NVL` is a `PV_LINK` a channel access link is created.

For each constant input link, the corresponding value field is initialized with the constant value (or `1e30` if the constant has the value 0).

For each input link that is of type `PV_LINK`, a channel access link is created.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_precision Retrieves PREC.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is VAL, HIHI, HIGH, LOW, or LOLO, the limits are set to HOPR and LOPR, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine process implements the following algorithm:

1. If NVL is a database or channel access link, SELN is obtained from NVL. Fetch all values if database or channel access links. If SELM is SELECTED, then only the selected link is fetched.
2. Implement the appropriate selection algorithm. For SELECT_HIGH, SELECT_LOW, and SELECT_MEDIAN, input fields are ignored if they are undefined. If success, UDF is set to FALSE.
3. Check alarms. This routine checks to see if the new VAL causes the alarm status and severity to change. If so, NSEV, NSTA and LALM are set. It also honors the alarm hysteresis factor (HYST). Thus the value must change by more than HYST before the alarm status and severity is lowered.
4. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if ADEL and MDEL conditions are met
   - Monitors for A-L are checked whenever other monitors are invoked
   - NSEV and NSTA are reset to 0.
5. Scan forward link if necessary, set PACT FALSE, and return.

Chapter 31: seq - Sequence

1. Introduction

The `seq` record is used to trigger the processing of up to ten other records. It has no associated device support. It is similar to the fanout record except it will fetch an input value and write an output value instead of simply processing a collection of forward links.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	No	Yes
SELM	RECCHOICE	Yes	0	Yes	Yes	No	No
SELN	USHORT	No	1	Yes	Yes	No	No
SELL	INLINK	Yes	0	No	No	N/A	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
DLY1	DOUBLE	Yes	0	Yes	Yes	No	No
DOL1	INLINK	Yes	0	No	No	N/A	No
DO1	DOUBLE	No	0	Yes	Yes	No	No
LNK1	OUTLINK	Yes	0	No	No	N/A	No
DLY2	DOUBLE	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
DOL2	INLINK	Yes	0	No	No	N/A	No
DO2	DOUBLE	No	0	Yes	Yes	No	No
LNK2	OUTLINK	Yes	0	No	No	N/A	No
DLY3	DOUBLE	Yes	0	Yes	Yes	No	No
DOL3	INLINK	Yes	0	No	No	N/A	No
DO3	DOUBLE	No	0	Yes	Yes	No	No
LNK3	OUTLINK	Yes	0	No	No	N/A	No
DLY4	DOUBLE	Yes	0	Yes	Yes	No	No
DOL4	INLINK	Yes	0	No	No	N/A	No
DO4	DOUBLE	No	0	Yes	Yes	No	No
LNK4	OUTLINK	Yes	0	No	No	N/A	No
DLY5	DOUBLE	Yes	0	Yes	Yes	No	No
DOL5	INLINK	Yes	0	No	No	N/A	No
DO5	DOUBLE	No	0	Yes	Yes	No	No
LNK5	OUTLINK	Yes	0	No	No	N/A	No
DLY6	DOUBLE	Yes	0	Yes	Yes	No	No
DOL6	INLINK	Yes	0	No	No	N/A	No
DO6	DOUBLE	No	0	Yes	Yes	No	No
LNK6	OUTLINK	Yes	0	No	No	N/A	No
DLY7	DOUBLE	Yes	0	Yes	Yes	No	No
DOL7	INLINK	Yes	0	No	No	N/A	No
DO7	DOUBLE	No	0	Yes	Yes	No	No
LNK7	OUTLINK	Yes	0	No	No	N/A	No
DLY8	DOUBLE	Yes	0	Yes	Yes	No	No
DOL8	INLINK	Yes	0	No	No	N/A	No
DO8	DOUBLE	No	0	Yes	Yes	No	No
LNK8	OUTLINK	Yes	0	No	No	N/A	No
DLY9	DOUBLE	Yes	0	Yes	Yes	No	No
DOL9	INLINK	Yes	0	No	No	N/A	No
DO9	DOUBLE	No	0	Yes	Yes	No	No
LNK9	OUTLINK	Yes	0	No	No	N/A	No
DLYA	DOUBLE	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
DOLA	INLINK	Yes	0	No	No	N/A	No
DOA	DOUBLE	No	0	Yes	Yes	No	No
LNKA	OUTLINK	Yes	0	No	No	N/A	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	Not used.
SELM	Select Mechanism:	`SELECT_ALL`: Select all links `SELECTED`: Use `SELN` as index (1 to 6) `MASK`: Use `SELN` as a mask to select an arbitrary combination of links.
SELN	Link Selection	If `SELM=SELECT_ALL` then this field is not used. If `SELM=SELECTED` then this is the index (1 to 6) of the link to select. If `SELM=MASK` then this is the mask (in decimal) used to determine the selected links. For example, if `SELN`=1, then `LNK1` will be processed. If `SELN`=3 then `LNK1` and `LNK2` will be processed. If `SELN`=63 then links `LNK1`, `LNK2`, ... `LNK6` will be processed.
SELL	Link Selection Location	`SELN` is read from `SELL`. `SELL` can be a constant, a database link, or a channel access link.
PREC	Display Precision	Precision with which to display `DLY1-DLYA` and `DO1-DOA` fields. This field is used by record support to supply a value when `get_precision` is called.
DLY1-DLYA	Delay time	This represents the delay time (in seconds) to wait before processing the input and output link pair (ie. `DOLn` and `LNKn`.)
DOL1-DOLA	Input link selection	`DO` is read from `DOL`. `DOL` can be a constant, database link or channel access link. If it is a constant, it is only copied to the `DO` field once at initialization time. Otherwise, it is re-fetched each time the record is processed.
DO1-DOA	Desired output value	This field holds the desired output value that will be placed in the output location indicated by the `LNK` field.
LNK1-LNKA	Output link field	`DO` is written to `LNK`. `LNK` can be a database link or a channel access link.

4. Record Support Routines

The only record support routine is `process`.

First, `PACT` is set to `TRUE`, and the link selection is fetched. Depending on the selection mechanism, the link selection output links are processed in order from `LNK1` to `LNKA`. When `LNKn` is processed, the corresponding `DLYn` value is used to generate a delay via watchdog timer.

After `DLYn` seconds have expired, the input value is fetched from `DOn` (if `DOLn` is constant) or `DOLn` (if `DOLn` is a database link or channel access link) and written to `LNKn`.

When all links are completed, an asynchronous completion call back to `dbProcess` is made (see the *Application Developer's Guide* for more information on asynchronous processing.)

Then `UDF` is set to `FALSE`.

Monitors are checked.

The forward link is scanned, `PACT` is set `FALSE`, and the `process` routine returns.

For the delay mechanism to operate properly, the record is processed asynchronously. The only time the record will not be processed asynchronously is when there are no non-`NULL` output links selected (ie. when it has nothing to do.) The processing of the links is done via callback tasks at the priority set in the `PRIO` field in `dbCommon` (see the *Application Developer's Guide* for more information on callback tasks.)

Chapter 32: State

1. Introduction

The `state` record is used to store an arbitrary ASCII string.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	STRING	Yes	Null	Yes	Yes	Yes	Yes
OVAL	STRING	No	Null	Yes	No		

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	An arbitrary string value
OVAL	Old Value	Used to decide when to invoke monitors.

4. Record Support Routines

Two record support routines are provided:

process `process` triggers monitors on `VAL` when it changes and scans the forward link if necessary.

get_value `get_value` fills in struct `valueDes` so that it refers to `VAL`.

Chapter 33: Stepper Motor

1. Introduction

The `steppermotor` record type is used to control stepper motors.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	FLOAT	No	0	Yes	Yes	Yes	Yes
OUT	OUTLINK	Yes	0	No	No	N/A	No
RDBL	INLINK	Yes	0	No	No	N/A	No
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
ACCL	FLOAT	Yes	0	Yes	Yes	No	No
VELO	FLOAT	Yes	0	Yes	Yes	No	No
DIST	FLOAT	Yes	0	Yes	Yes	No	No
IVAL	FLOAT	Yes	0	Yes	Yes	No	No
MODE	RECCHOICE	Yes	0	Yes	Yes	No	No
CMOD	RECCHOICE	Yes	0	Yes	Yes	No	No
IALG	RECCHOICE	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
MRES	USHORT	Yes	0	Yes	Yes	No	No
ERES	USHORT	Yes	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
DRVH	FLOAT	Yes	0	Yes	Yes	No	Yes
DRVL	FLOAT	Yes	0	Yes	Yes	No	Yes
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
ADEL	FLOAT	Yes	0	Yes	Yes	No	No
MDEL	FLOAT	Yes	0	Yes	Yes	No	No
RDBD	FLOAT	Yes	0	Yes	Yes	No	No
RTRY	SHORT	Yes	0	Yes	Yes	No	No
STHM	SHORT	No	0	Yes	Yes	No	Yes
STOP	SHORT	No	0	Yes	Yes	No	Yes
DMOV	SHORT	No	0	Yes	Yes	No	No
RVAL	LONG	No	0	Yes	Yes	Yes	Yes
RBV	FLOAT	No	0	Yes	Yes	Yes	No
RRBV	LONG	No	0	Yes	Yes	Yes	No
ALST	FLOAT	No	0	Yes	No	No	No
MLST	FLOAT	No	0	Yes	No	No	No
INIT	SHORT	No	0	Yes	Yes	No	Yes
MCW	SHORT	No	0	Yes	Yes	No	Yes
MCCW	SHORT	No	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
CW	SHORT	No	0	Yes	Yes	No	Yes
CCW	SHORT	No	0	Yes	Yes	No	Yes
DIR	SHORT	No	0	Yes	Yes	No	Yes
MOVN	SHORT	No	0	Yes	Yes	No	No
CVEL	SHORT	No	0	Yes	Yes	No	No
RCNT	SHORT	No	0	Yes	Yes	No	No
POSM	SHORT	No	0	Yes	Yes	No	No
LVAL	FLOAT	No	0	Yes	No	No	No
EPOS	FLOAT	No	0	Yes	Yes	No	No
MPOS	FLOAT	No	0	Yes	Yes	No	No
MISS	FLOAT	No	0	Yes	Yes	No	No
LVEL	FLOAT	No	0	Yes	Yes	No	No
LACC	FLOAT	No	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value	This is the desired output value, in engineering units. If `DRVH` and `DRVL` are defined, `VAL` is forced to be within the drive limits. `VAL` is either obtained from `DOL` or set via `dbPuts`.
OUT	Output Link	This field is used by the device support routines to locate the stepper motor.
RDBL	Read Back Location (Input link)	This link is used to obtain the read back value when a physical read back is attached to the device being driven from the stepper motor.
DOL	Desired Output Location (Input Link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`. After the check for drive limits, `VAL` will be set to the value determined by `DOL`.
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field, the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
ACCL	Acceleration	Number of seconds to reach `VELO` velocity.

Name	Summary	Description
VELO	Velocity	Rotations per second.
DIST	Distance	Distance moved by each pulse of the stepper motor.
IVAL	Initial Value	
MODE	Mode	Velocity or Position.
CMOD	Current Mode	Velocity or Position.
IALG	Initialization Algorithm	None, Move to positive limit, Move to negative limit.
MRES	Motor Pulses per Revolution	
ERES	Encoder Pulses per Revolution	
PREC	Display Precision	Precision with which to display. This field is used by record support to supply a value when `get_precision` is called.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
DRVH	Drive High	If these values are defined, then `VAL` will forced to be in the range `DRVL<=VAL<=DRVH`
DRVL	Drive Low	
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
HHSV	HiHi Alarm Severity	These fields specify the alarm severities.
HSV	High Alarm Severity	
LSV	Low Alarm Severity	
LLSV	LoLo Alarm Severity	
HLSV	Hardware Limit Violation Severity	
ADEL	Archive Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
MDEL	Monitor, i.e. value change, Deadband	

Name	Summary	Description
RDBD	Retry Deadband	
RTRY	Number Of Retries Before Failure	
STHM	Set Home	Setting this field to 1 via a `dbPut` is a command to set home to the current position of the stepper motor. This field will automatically be reset to 0 after the command is accepted.
STOP	Stop	Setting this field to 1 will cause the motor to stop if it is moving. This field will automatically be reset to 0 after the command is accepted.
DMOV	Done Moving to Value	
RVAL	Raw Data Value	`RVAL` is the value actually sent to the device.
RBV	Read Back Value	This is the actual read back value obtained from the hardware itself or from the associated device driver. It is the responsibility of the device support routine to give this field a value.
RRBV	Raw Read Back Value	Raw read back value obtained from the encoder.
ALST	Archive Last Value	Value when last monitors for archiver were triggered These fields are used to implement the hysteresis factors for monitors.
MLST	Monitor Last Value	
INIT	Initialize	
MCW	Motor Clockwise Limit Switch Value	
MCCW	Motor Counter Clockwise Limit Switch Value	
CW		Is motor clockwise limit switch `TRUE`?
CCW		Is motor counter clockwise limit switch True?
DIR	Current Direction	
MOVN		Is motor moving?
CVEL		Has Constant velocity been attained?
RCNT	Current Retry Count	
LVAL	Last Value	
POSM	Positive Motion	
EPOS	Encoder Read Back Position	
MPOS	Motor Position	

Name	Summary	Description
MISS	First Attempt Error	
LVEL	Last Velocity Set	
LACC	Last Acceleration Set	

4. Record Support Routines

init_record

This routine checks to see that device support is available. The routine next checks to see if the device support `sm_command` routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

If `DOL` is a constant, then `VAL` is initialized with its value and `UDF` is set to `FALSE`. If `DOL` is a `PV_LINK` then a channel access link is created.

`init_sm` is then called.

init_record

Not written yet.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_units

Retrieves `EGU`.

get_precision

Retrieves `PREC`.

get_graphic_double

Sets the upper display and lower display limits for a field. If the field is `VAL`, `LVAL`, `MPOS`, `RBV`, `EPOS`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double

Sets the upper control and the lower control limits for a field. If the field is `VAL`, `LVAL`, `MPOS`, `RBV`, `EPOS`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Not yet written

6. Device Support

At the present time, device support is intimately connected to record support. The compumotor 1830 and the OMS 6 axis controllers are supported.

Chapter 34: stringin - String Input

1. Introduction

The `stringin` record is used to input an arbitrary ASCII string.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	STRING	Yes	Null	Yes	Yes	Yes	Yes
OVAL	STRING	No	Null	Yes	No		No
INP	INLINK	Yes	0	No	No	N/A	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SVAL	STRING	No	Null	Yes	Yes	No	Yes
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value	An arbitrary ASCII string of 40 characters. It is either obtained from `INP` or else given a value via `dbPuts`.
OVAL	Output Value	Old ASCII string. Used to decide when to invoke monitors. If `VAL` differs from `OVAL`, monitors will be invoked.
INP	Input Link	This field is used by the device support routines to obtain input. For soft records, it can be a constant, a database link, or a channel access link.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

Three record support routines are provided: `init_record`, `process`, and `get_value`.

init_record

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `SVAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a record support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

process

See next section.

get_value

Fills in the values of struct `valueDes` so that they refer to `VAL`.

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. `readValue` is called. See Chapter 3, Section "Simulation Mode" on page 11 for details.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed reading a new input value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
4. `TIME` is set to `tslocaltime`
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `OVAL` is not equal to `VAL`.
   - `NSEV` and `NSTA` are reset to 0.
6. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each `stringin` input record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new ASCII string value whenever `read_stringin` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
VAL	Value	This field is set by the device support routines.
INP	Input Link.	This field is used by the device support routines to locate its input.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_stringin

```
read_stringin(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success. A new ASCII string is stored into `VAL`.
- **Other**: Error.

7. Device Support For Soft Records

This module places a value directly in `VAL`.

If the `INP` link type is constant, the double constant, if non-zero, is converted to a string and stored into `VAL` by `init_record`, and `UDF` is set to `FALSE`. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_stringin` calls `recGblGetLinkValue` to read the current value of `VAL`. See Chapter 3, Section "Soft Input" on page 10 for details.

If the return status of `recGblGetLinkValue` is zero, then `read_stringin` sets `UDF` to `FALSE`. The status of `recGblGetLinkValue` is returned.

Chapter 35: stringout - String Output

1. Introduction

The `stringout` record is used to output an arbitrary ASCII string.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	STRING	Yes	Null	Yes	Yes	Yes	Yes
OVAL	STRING	No	Null	Yes	No		No
DOL	INLINK	Yes	0	No	No	N/A	No
OMSL	GBLCHOICE	Yes	0	Yes	Yes	No	No
OUT	OUTLINK	Yes	0	No	No	N/A	No
SIOL	OUTLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOA	GBLCHOICE	Yes	0	Yes	Yes	No	No
IVOV	STRING	Yes	Null	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value	An arbitrary ASCII string of 40 characters. This is the field to be sent to `OUT`. It is either obtained from `DOL` or else given a value via `dbPuts`.
OVAL	Old Value	Used to decide when to invoke monitors. If `VAL` differs from `OVAL`, then monitors will be invoked.
DOL	Desired Output Location (Input Link)	If `DOL` is a database or channel access link and `OMSL` is `CLOSED_LOOP`, then `VAL` is read from `DOL`.
OMSL	Output Mode Select	This field has either the value `SUPERVISORY` or `CLOSED_LOOP`. `DOL` is used to determine `VAL` only if `OMSL` has the value `CLOSED_LOOP`. By setting this field, the record can be switched between supervisory and closed loop mode of operation. While in closed loop mode, the `VAL` field cannot be set via `dbPuts`.
OUT	Output Link	This field is used by the device support routines to decide where to send output. For soft records, it can be a constant, a database link, or a channel access link. If the link is a constant, the result is no output.
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	
IVOA	Invalid Alarm Output Action	Whenever the record is put into INVALID alarm severity IVOA specifies an action. See Chapter 3, Section "Invalid Alarm Output Action" on page 14 for more information
IVOV	Invalid Alarm Output Value	

4. Record Support Routines

Three record support routines are provided: `init_record`, `process`, and `get_value`.

init_record

This routine initializes `SIMM` if `SIML` is a constant or creates a channel access link if `SIML` is `PV_LINK`. If `SIOL` is `PV_LINK` a channel access link is created.

This routine next checks to see that device support is available. The routine next checks to see if the device support write routine is defined. If either device support or the device support write routine does not exist, an error message is issued and processing is terminated.

If DOL is a constant, then the type double constant, if non-zero, is converted to a string and stored into VAL and UDF is set to FALSE. If DOL type is a PV_LINK then dbCaAddInlink is called to create a channel access link.

If device support includes init_record, it is called.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to VAL.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. If PACT is FALSE and OMSL is CLOSED_LOOP, recGblGetLinkValue is called to read the current value of VAL. See Chapter 3, Section "Soft Input" on page 10 for details. If the return status of recGblGetLinkValue is zero then UDF is set to FALSE.
3. Check severity and write the new value. See Chapter 3, Section "Simulation Mode" on page 13 and Chapter 3, Section "Invalid Alarm Output Action" on page 14 for details.
4. If PACT has been changed to TRUE, the device support write output routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving PACT TRUE.
5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if OVAL is not equal to VAL.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

Each `stringout` output record must have an associated set of device support routines. The primary responsibility of the device support routines is to write a new value whenever `write_stringout` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
VAL	Value	This is the field written by the device support routines.
OUT	Output Link	This field is used by the device support routines to locate its output.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

write_stringout

```
write_stringout(precord)
```

This routine must output a new value. It returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

This module writes the current value of `VAL`.

If the `OUT` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`write_so` calls `recGblPutLinkValue` to write the current value of `VAL`. See Chapter 3, Section "Soft Output" on page 13 for details.

Chapter 36: subArray

Who is this from?

1. Introduction

The normal use for the `subArray` record type is to obtain sub-arrays from waveform records. Setting either the `NELM` or `INDX` fields causes the record to be processed with the new value, so that applications in which the length and position of a subarray of a waveform record are dynamically varied can be implemented using standard EPICS operator interface tools. The first element of the sub-array, that at location `INDX` in the referenced waveform record, can be displayed as a scalar, or the entire subarray (of length `NELM`) can be displayed in the same way as a waveform record. If there are fewer than `NELM` elements in the referenced waveform after the `INDX`, only that number of elements actually available are returned, and `NORD` is set to reflect this. This record type does not support writing new values into waveform records.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	(See FTVL)	No	0	Yes	Yes	Yes	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
FTVL	GBLCHOICE	Yes	0	Yes	No	No	
INP	INLINK	Yes	0	No		N/A	
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
MALM	ULONG	Yes	1	Yes	No	No	
NELM	ULONG	Yes	1	Yes	Yes	No	Yes
INDX	ULONG	Yes	0	Yes	Yes	No	Yes
NORD	LONG	No	0	Yes	No	No	
BPTR	NOACCESS	No					

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is used to reference the sub-array.
PREC	Display Precision	Precision with which to display `VAL`. This field is not used by record support other than to supply a value when `get_precision` is called.
FTVL	Field Type of Value	This is `DBF_STRING`, ... ,`DBF_ENUM`.
INP	Input Link	This field is used by the device support routines to obtain input. It can be a database link, or a channel access link.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
MALM	Maximum Number Of Elements In Sub-array	Generally this should be set to the `NELM` of the waveform record being pointed to.
NELM	Number Of Elements In Sub-array	
INDX	Index Into Referenced Array	Index of (offset into) waveform record being referenced; used as first element of sub-array.
NORD	Number of Elements Read	Number of elements that were read of the desired subarray. This could be less than `NELM` depending on `INDX` and the `NELM` of the referenced waveform record.
BPTR	Buffer Pointer	Holds address of sub-array.

4. Record Support Routines

init_record

Using MALM and FTVL, space for the array is allocated. The array address is stored in BPTR. This routine checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated. If device support includes init_record, it is called.

process

See next section.

get_value

Fills in the values of struct valueDes so that they refer to the sub-array.

cvt_dbaddr

This is called by dbNameToAddr. It makes the dbAddr structure refer to the actual buffer holding the result.

get_array_info

Retrieves NELM.

put_array_info

Sets NORD.

get_units

Retrieves EGU.

get_prec

Retrieves PREC.

5. Record Processing

Routine process implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the PACT field still set to TRUE. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. Sanity check NELM and INDX. If NELM is greater than MALM it is set to MALM. If INDX is greater than MALM it is set to MALM-1.
3. Call device support read routine. This routine is expected to place the desired sub-array at the beginning of the buffer and set NORD to the number of elements of the sub-array that were read.
4. If PACT has been changed to TRUE, the device support read routine has started but has not completed writing the new value. In this case, the processing routine merely returns,

leaving PACT TRUE. Otherwise, process sets PACT TRUE at this time. This asynchronous processing logic is not currently used but has been left in place.

5. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are always invoked.
   - NSEV and NSTA are reset to 0.
6. Scan forward link if necessary, set PACT FALSE, and return.

6. Device Support

Fields Of Interest To Device Support

The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2 Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
UDF	VAL Undefined	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
INP	Input Link	This field is used by the device support routines to locate its input.
FTVL	Field Type of Value	This is DBF_STRING, ... ,DBF_ENUM. The device support routine should check that this is correctly defined.
MALM	Maximum Number Of Elements In Sub-array	Number of elements that will fit in the array the record allocates. Device support must never return more elements than this
NELM	Number Sub-array Elements	Number of elements in desired sub-array.
INDX	Index Into Referenced Array	Index of beginning of desired sub-array in source array.
BPTR	Buffer Pointer	Address of array device support must copy the source array into.
NORD	Number Of Elements Read	Device support must set this value when it completes.

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

Not currently used.

init_record

```
init_record(precord)
```

This routine is called by the record support `init_record` routine.

read_sa

```
read_sa(precord)
```

Enough of the source waveform is read into `BPTR`, from the beginning of the source, to include the requested sub-array. The sub-array is then copied to the beginning of the buffer. `NORD` is set to indicate how many elements of the sub-array were acquired.

7. Device Support For Soft Records

Only the device support module Soft Channel is provided. The `INP` link type must be either `DB_LINK` or `CA_LINK`.

Soft Channel

`INP` is expected to point to a waveform record.

Chapter 37: sub - Subroutine

1. Introduction

The sub record provides a subroutine escape mechanism.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	Yes
INAM	STRING	Yes	Null	Yes	No		No
SNAM	STRING	Yes	Null	Yes	No		No
SADR	NOACCESS	No	0	No	No		No
STYP	SHORT	No	0	Yes	No	No	No
INPA	INLINK	Yes	0	No	No	N/A	No
INPB	INLINK	Yes	0	No	No	N/A	No
INPC	INLINK	Yes	0	No	No	N/A	No
INPD	INLINK	Yes	0	No	No	N/A	No
INPE	INLINK	Yes	0	No	No	N/A	No
INPF	INLINK	Yes	0	No	No	N/A	No
INPG	INLINK	Yes	0	No	No	N/A	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
INPH	INLINK	Yes	0	No	No	N/A	No
INPI	INLINK	Yes	0	No	No	N/A	No
INPJ	INLINK	Yes	0	No	No	N/A	No
INPK	INLINK	Yes	0	No	No	N/A	No
INPL	INLINK	Yes	0	No	No	N/A	No
A	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
B	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
C	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
D	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
E	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
F	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
G	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
H	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
I	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
J	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
K	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
L	DOUBLE	No	0	Yes	Yes/No	Yes	Yes
LA	DOUBLE	No	0	Yes	No	No	No
LB	DOUBLE	No	0	Yes	No	No	No
LC	DOUBLE	No	0	Yes	No	No	No
LD	DOUBLE	No	0	Yes	No	No	No
LE	DOUBLE	No	0	Yes	No	No	No
LF	DOUBLE	No	0	Yes	No	No	No
LG	DOUBLE	No	0	Yes	No	No	No
LH	DOUBLE	No	0	Yes	No	No	No
LI	DOUBLE	No	0	Yes	No	No	No
LJ	DOUBLE	No	0	Yes	No	No	No
LK	DOUBLE	No	0	Yes	No	No	No
LL	DOUBLE	No	0	Yes	No	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
BRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is determined by the subroutine as a result of record processing.
INAM	Initialization Name	This is the name of the initialization entry. It is called once at record initialization time.
SNAM	Subroutine Name	This the the name of the processing routine. It is called by the the record processing routine.
SADR	Subroutine Address	Filled in by record processing.
STYP	Subroutine Symbol Type	Filled in by record processing.
INPA,...,INPL	Input Link A, Input Link B, ...	The input links. Each may be a constant, a database link, or a channel access link. Any link not defined is ignored.
A,...,L	A, B, ...	The input values If the corresponding `INP` field is a constant, this field is initialized with the constant value but can be changed via `dbPuts`.
LA,...,LL	Last A, Last B, ...	Previous input values These fields are used to decide when to trigger monitors on A,...,L.
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	

Name	Summary	Description
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities.
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
LOLO	Lolo Alarm Limit	
BRSV	Severity for a subroutine return value less than 0.	
HHSV	Severity for a Hihi Alarm	
HSV	Severity for a High Alarm	
LSV	Severity for a Low Alarm	
LLSV	Severity for a Lolo Alarm	
HYST	Alarm Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
ADEL	Archive Deadband	
MDEL	Monitor, i.e. value change, Deadband	
LALM	Last Alarm Monitor Trigger Value	These fields are used to implement the hysteresis factors for monitors.
ALST	Last Archiver Monitor Trigger Value	
MLST	Last Value Change Monitor Trigger Value	

4. Record Support Routines

init_record

For each constant input link, the corresponding value field is initialized with the constant value. For each input link that is of type `PV_LINK`, a channel access link is created.

If an initialization subroutine is defined, it is located and called.

The processing subroutine is located and its address and type stored in `SADR` and `STYP`.

process

See next section.

get_value Fills in the values of struct `valueDes` so that they refer to `VAL`.

get_units Retrieves `EGU`.

get_precision Retrieves `PREC`.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is `VAL`, `HIHI`, `HIGH`, `LOW`, or `LOLO`, the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_alarm_double Sets the following values:

```
upper_alarm_limit = HIHI
upper_warning_limit = HIGH
lower_warning_limit = LOW
lower_alarm_limit = LOLO
```

5. Record Processing

Routine `process` implements the following algorithm:

1. If `PACT` is `FALSE` then fetch all arguments.
2. Call the subroutine and check return value.
   - Call `subroutine`
   - Set `PACT TRUE`
   - If return value is 1, return
3. Check alarms. This routine checks to see if the new `VAL` causes the alarm status and severity to change. If so, `NSEV`, `NSTA` and `LALM` are set. It also honors the alarm hysteresis factor (`HYST`). Thus the value must change by more than `HYST` before the alarm status and severity is lowered.
4. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are invoked if `ADEL` and `MDEL` conditions are met.
   - Monitors for `A-L` are are invoked if value has changed.
   - `NSEV` and `NSTA` are reset to 0.
5. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Example Synchronous Subroutine

This is an example that merely increments VAL each time process is called.

```
#include        <vxWorks.h>
#include        <types.h>
#include        <stdioLib.h>

#include        <dbDefs.h>
#include        <subRecord.h>
#include        <dbCommon.h>
#include        <recSup.h>


long subInit(psub)
    struct subRecord     *psub;
{
    printf("subInit was called\n");
    return(0);
}

long subProcess(psub)
    struct subRecord     *psub;
{

    psub->val++;
    return(0);
}
```

7. Example Asynchronous Subroutine

This example shows an asynchronous subroutine. It uses (actually misuses) fields A and B. Field A is taken as the number of seconds until asynchronous completion. Field B is a flag to decide if messages should be printed. Lets assume A>0 and B=1. The following sequence of actions will occur:

1. subProcess is called with pact FALSE. It performs the following steps.
    a. Computes, from A, the number of ticks until asynchronous completion should occur.
    b. Prints a message stating that it is requesting an asynchronous callback.
    c. Calls the vxWorks watchdog start routine.
    d. Sets pact TRUE and returns a value of 0. This tells record support to complete without checking alarms, monitors, or the forward link.
2. When the time expires, the system wide callback task calls myCallback. myCallback locks the record, calls process, and unlocks the record.
3. Process again calls subProcess, but now pact is TRUE. Thus the following is done:
    a. VAL is incremented.
    b. A completion message is printed.
    c. subProcess returns 0. The record processing routine will complete record processing.

```
#include   <vxWorks.h>
#include   <types.h>
#include   <stdioLib.h>
#include   <wdLib.h>
#include   <callback.h>
#include   <dbDefs.h>
#include   <dbAccess.h>
```

```
#include   <subRecord.h>
/* control block for callback*/
struct callback {
    CALLBACK   callback;
    struct dbCommon *precord;
    WDOG_ID wd_id;
};
void myCallback(pcallback)
    struct callback *pcallback;
{
    struct dbCommon *precord=pcallback->precord;
    struct rset  *prset=(struct rset *)(precord->rset);
    dbScanLock(precord);
    (*prset->process)(precord);
    dbScanUnlock(precord);
}
long subInit(psub)
    struct subRecord  *psub;
{
    struct callback *pcallback;
    pcallback = (struct callback *)(calloc(1,sizeof(struct callback)));
    psub->dpvt = (void *)pcallback;
    callbackSetCallback(myCallback,pcallback);
    pcallback->precord = (struct dbCommon *)psub;
    pcallback->wd_id = wdCreate();
    printf(”subInit was called\n”);
    return(0);
}
long subProcess(psub)
    struct subRecord  *psub;
{
    struct callback *pcallback=(struct callback *)(psub->dpvt);
    int  wait_time;
    /* sub.inp must be a CONSTANT*/
    if(psub->pact) {
        psub->val++;
        if(psub->b)
            printf(”%s subProcess Completed\n”,psub->name);
            return(0);
    } else {
        wait_time = (long)(psub->a * vxTicksPerSecond);
        if(wait_time<=0){
            if (psub->b)
                printf(”%s subProcess synchronous processing\n”,psub->name);
                psub->pact = TRUE;
                return(0);
        }
        if (psub->b){
            callbackSetPriority(psub->prio,pcallback);
            printf(”%s Starting asynchronous processing\n”,psub->name);
            wdStart(pcallback->wd_id,wait_time,callbackRequest,(int)pcallback);
            return(1);
        }
    }
    return(0);
}
```

Chapter 38: Timer

1. Introduction

The function of the `timer` record has been replaced by the `pulseCounter`, `pulseDelay`, `pulseTrain`, and `Event` records. The `Timer` record type is included for backward compatibility.

This record type interacts with timer modules.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
TORG	INLINK	Yes	0	No	No	N/A	No
OUT	OUTLINK	Yes	0	No	No	N/A	No
VAL	SHORT	No	0	Yes	Yes	Yes	Yes
TSRC	RECCHOICE	Yes	0	Yes	Yes	No	No
PTST	RECCHOICE	Yes	0	Yes	Yes	No	Yes
TEVT	SHORT	Yes	0	Yes	Yes	No	Yes
PREC	SHORT	Yes	0	Yes	Yes	No	No
TIMU	RECCHOICE	Yes	0	Yes	Yes	No	No
MAIN	GBLCHOICE	Yes	1	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
RDT1	FLOAT	No	0	Yes	Yes	No	No
RDW1	FLOAT	No	0	Yes	Yes	No	No
PDLY	FLOAT	Yes	0	Yes	Yes	No	No
DUT1	FLOAT	Yes	0	Yes	Yes	No	Yes
OPW1	FLOAT	Yes	0	Yes	Yes	No	Yes
DUT2	FLOAT	Yes	0	Yes	Yes	No	Yes
OPW2	FLOAT	Yes	0	Yes	Yes	No	Yes
DUT3	FLOAT	Yes	0	Yes	Yes	No	Yes
OPW3	FLOAT	Yes	0	Yes	Yes	No	Yes
DUT4	FLOAT	Yes	0	Yes	Yes	No	Yes
OPW4	FLOAT	Yes	0	Yes	Yes	No	Yes
DUT5	FLOAT	Yes	0	Yes	Yes	No	Yes
OPW5	FLOAT	Yes	0	Yes	Yes	No	Yes
T1DL	DOUBLE	No	0	Yes	Yes	No	No
T1WD	DOUBLE	No	0	Yes	Yes	Yes	No
T2DL	DOUBLE	No	0	Yes	Yes	No	No
T2WD	DOUBLE	No	0	Yes	Yes	No	No
T3DL	DOUBLE	No	0	Yes	Yes	No	No
T3WD	DOUBLE	No	0	Yes	Yes	No	No
T4DL	DOUBLE	No	0	Yes	Yes	No	No
T4WD	DOUBLE	No	0	Yes	Yes	No	No
T5DL	DOUBLE	No	0	Yes	Yes	No	No
T5WD	DOUBLE	No	0	Yes	Yes	No	No
T1TD	FLOAT	No	0	Yes	Yes	Yes	No
T1LD	FLOAT	No	0	Yes	Yes	Yes	No
T2TD	FLOAT	No	0	Yes	Yes	No	No
T2LD	FLOAT	No	0	Yes	Yes	No	No
T3TD	FLOAT	No	0	Yes	Yes	No	No
T3LD	FLOAT	No	0	Yes	Yes	No	No
T4TD	FLOAT	No	0	Yes	Yes	No	No
T4LD	FLOAT	No	0	Yes	Yes	No	No
T5TD	FLOAT	No	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
T5LD	FLOAT	No	0	Yes	Yes	No	No
TRDL	FLOAT	No	0	Yes	Yes	No	No
TDIS	SHORT	No	0	Yes	Yes	No	Yes

3. Field Descriptions

Name	Summary	Description
TORG	Trigger Delay Origin (input link)	This is a link specifying the location of the trigger delay value. This must be a constant, a database link, or a channel access link. If `TORG` is a database link, then `TRDL` is read from `TORG`.
OUT	Output Link	This field is used by the device support routines to decide where to send output.
VAL	Value Field	This field is used to force record processing.
TSRC	Clock Source	External or internal
PTST	Pre-Trigger State	Low or high
TEVT		Event Number To Be Posted On Trigger
PREC	Display Precision	
TIMU	Timer Units	Milli, micro, nano. pico seconds
MAIN	Maintain on Reboot	
RDT1	Reboot Delay of 1	
RPW1	Reboot Width of 1	
PDLY	Delay Source to Input	
DUTn	Delay Width For Trigger n	(n=1-5): In timer units.
OPWn	Output Pulse Width For Trigger n	(n=1-5): In timer units.
TnDL	Delay Width For Trigger n	(n=1-5): In seconds.
TnWD	Pulse Width Of Trigger n	(n=1-5): In seconds.
TnTD	Trailing Delay Of Trigger n	(n=1-5): (TiLD+OPWi)
TnLD	Leading Delay Of Trigger n	(n=1-5): (DUTi+TRDL)
TRDL	Trigger Delay	Obtained from trigger delay origin `TORG`.
TDIS	Timing Pulse Disable	

4. Record Support Routines

init_record

process See next section.

get_value Fills in the values of struct `valueDes` so that they refer to the array.

5. Record Processing

This section not yet written.

6. Device Support

Currently device support is intimately combined with record support.

Chapter 39: Wait

Ned D. Arnold
Advanced Photon Source
Argonne National Laboratory

1. Introduction

This chapterdescribes the capabilities and use of the `wait` record. The `wait` record is derived fom the standard `calc` record with the following additional features: "Reassignable" PV links, an Output Link, a Desired Output input link, an output event number to post, and several options as to when it will execute the output link and event posting. The `wait` record also has the capability to "process" as a result of an input changing (via CA monitors).

The `wait` record is a powerful record type that can be used to do "conditional" processing within the database. Its name is derived from the original requirement that initiated its development, i.e. "I want to *wait* until all the motors have stopped and then trigger the detector". The sections below describe the capabilities of the record.

"Reassignable" PV Links

Like the `calc` record, the `wait` record has 12 input links for fetching variables used in the calculation. Unlike the `calc` record, these input links can be modified during run time. The record contains ASCII fields in which a new `Process_Variable.Field` name may be written. The record will use the new link the next time the record is processed.

A consequence of reassignable links is that one cannot force the processing of an input record prior to retrieving the data (i.e. there is no `.PP` flag). This should be considered when designing a database using the `wait` record.

In this initial version, the "reassignable" PV links do not support channel access connections external to the IOC. Until this feature is added, the specified Process Variables must reside on the same IOC.

Output Links and Output Events

The `wait` record has two kinds of "outputs". The first is an output link to which data will be written when appropriate (see next section). The data to be written can be the result of the calculation (use `VAL`), fetched from another link (use `DOL` and specify a "Desired Output Location Name" - `DOLN`) or a user entered constant (use `DOL`, leave `DOLN` blank, enter constant in `DOLD`).

As a result of the "reassignable links", there is no application specific control over whether the record specified as the output link will be processed when the data is "put" (i.e. there is no `.PP` flag associated with the link.) The processing of the destination record will depend on its scan mechanism and ASCII record definition file Process Passive values.

The other "output" of a `wait` record is an event. If a non-zero value is entered into the `OEVT` (Output Event) field, the record will "post an event" (using the entered number as the event number) whenever the output link is executed. This is a way of initiating *several* other records to process as a result of a calculation.

Output Link Execution Options

The outputs of the `wait` record are not necessarily executed every time the record processes. This allows "downstream" processing of records to be done conditionally. The "output execution" options are decsribed below.

- **Every Time**: Outputs are executed every time the record processes.
- On Change: Outputs are executed if the result of the calculation (`VAL`) is different than the previous time the record was processed.
- **When Zero**: Outputs are executed if the result of the calculation is zero.
- **When Non-zero**: Outputs are executed if the result of the calculation is not zero.
- **Transition To Zero**: Outputs are executed if the result of the calculation is zero and the previous value was not zero.
- **Transition To Non-zero**: Outputs are executed if the result of the calculation is not zero and the previous value was zero.

Process Record on Input Change

In addition to the standard scan mechanisms avaliable to all records (periodic, passive, event, etc), the `wait` record can be specified to process as a result of one of i's input values changing (using channel access monitors). This offers immediate response to an input change (rather than waiting for the next periodic scan) while minimizing record processing that is not required. The scan mechanism choice of I/O Intr will enable this feature.

For this release, an additional module, `caMonitor.o`, must be compiled and loaded with the `wait` record support to provide this feature.

A word of caution is in order. Because of the event driven nature of this feature, it is quite easy to configure a database that results in an infinite loop that uses all available CPU time. If the `wait` record is set to process as a result of a channel changing and the processing of the `wait` record causes the channel to change again, an infinite loop will result. The symptom will be a loss of all channel access connections (lower priority tasks) even though the shell responds normally. Using the vxWorks utility "`spy`" will confirm the predicament by showing 0% free CPU time.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
INIT	SHORT	No	0	Yes	No	No	No
CBST	NOACCESS	No	Null	No	No	No	No
INAN	STRING	Yes	Null	Yes	Yes	No	No
INBN	STRING	Yes	Null	Yes	Yes	No	No
INCN	STRING	Yes	Null	Yes	Yes	No	No
INDN	STRING	Yes	Null	Yes	Yes	No	No
INEN	STRING	Yes	Null	Yes	Yes	No	No
INFN	STRING	Yes	Null	Yes	Yes	No	No
INGN	STRING	Yes	Null	Yes	Yes	No	No
INHN	STRING	Yes	Null	Yes	Yes	No	No
ININ	STRING	Yes	Null	Yes	Yes	No	No
INJN	STRING	Yes	Null	Yes	Yes	No	No
INKN	STRING	Yes	Null	Yes	Yes	No	No
INLN	STRING	Yes	Null	Yes	Yes	No	No
INAA	NOACCESS	No	Null	No	No	No	No
INBA	NOACCESS	No	Null	No	No	No	No
INCA	NOACCESS	No	Null	No	No	No	No
INDA	NOACCESS	No	Null	No	No	No	No
INEA	NOACCESS	No	Null	No	No	No	No
INFA	NOACCESS	No	Null	No	No	No	No
INGA	NOACCESS	No	Null	No	No	No	No
INHA	NOACCESS	No	Null	No	No	No	No
INIA	NOACCESS	No	Null	No	No	No	No
INJA	NOACCESS	No	Null	No	No	No	No
INKA	NOACCESS	No	Null	No	No	No	No
INLA	NOACCESS	No	Null	No	No	No	No
INAV	LONG	No	0	Yes	Yes	Yes	No
INBV	LONG	No	0	Yes	Yes	Yes	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
INCV	LONG	No	0	Yes	Yes	Yes	No
INDV	LONG	No	0	Yes	Yes	Yes	No
INEV	LONG	No	0	Yes	Yes	Yes	No
INFV	LONG	No	0	Yes	Yes	Yes	No
INGV	LONG	No	0	Yes	Yes	Yes	No
INHV	LONG	No	0	Yes	Yes	Yes	No
INIV	LONG	No	0	Yes	Yes	Yes	No
INJV	LONG	No	0	Yes	Yes	Yes	No
INKV	LONG	No	0	Yes	Yes	Yes	No
INLV	LONG	No	0	Yes	Yes	Yes	No
A	DOUBLE	No	0	Yes	Yes	Yes	Yes
B	DOUBLE	No	0	Yes	Yes	Yes	Yes
C	DOUBLE	No	0	Yes	Yes	Yes	Yes
D	DOUBLE	No	0	Yes	Yes	Yes	Yes
E	DOUBLE	No	0	Yes	Yes	Yes	Yes
F	DOUBLE	No	0	Yes	Yes	Yes	Yes
G	DOUBLE	No	0	Yes	Yes	Yes	Yes
H	DOUBLE	No	0	Yes	Yes	Yes	Yes
I	DOUBLE	No	0	Yes	Yes	Yes	Yes
J	DOUBLE	No	0	Yes	Yes	Yes	Yes
K	DOUBLE	No	0	Yes	Yes	Yes	Yes
L	DOUBLE	No	0	Yes	Yes	Yes	Yes
LA	DOUBLE	No	0	Yes	Yes	No	No
LB	DOUBLE	No	0	Yes	Yes	No	No
LC	DOUBLE	No	0	Yes	Yes	No	No
LD	DOUBLE	No	0	Yes	Yes	No	No
LE	DOUBLE	No	0	Yes	Yes	No	No
LF	DOUBLE	No	0	Yes	Yes	No	No
LG	DOUBLE	No	0	Yes	Yes	No	No
LH	DOUBLE	No	0	Yes	Yes	No	No
LI	DOUBLE	No	0	Yes	Yes	No	No
LJ	DOUBLE	No	0	Yes	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LK	DOUBLE	No	0	Yes	Yes	No	No
LL	DOUBLE	No	0	Yes	Yes	No	No
CALC	STRING	Yes	36	Yes	Yes	No	No
RPCL	NOACCESS	No	184	No	No	No	No
CLCV	LONG	No	0	Yes	Yes	Yes	No
VAL	DOUBLE	No	0	Yes	Yes	Yes	No
OVAL	DOUBLE	No	0	Yes	Yes	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
OOPT	RECCHOICE	Yes	0	Yes	Yes	No	No
OUTN	STRING	Yes	Null	Yes	Yes	No	No
OUTA	NOACCESS	No	Null	No	No	No	No
OUTV	LONG	No	0	Yes	Yes	Yes	No
DOPT	RECCHOICE	Yes	0	Yes	Yes	No	No
DOLN	STRING	Yes	Null	Yes	Yes	No	No
DOLA	NOACCESS	No	Null	No	No	No	No
DOLV	LONG	No	0	Yes	Yes	Yes	No
DOLD	DOUBLE	Yes	0	Yes	Yes	Yes	No
OEVT	USHORT	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No
SIOL	INLINK	No	Null	Yes	No	No	No
SVAL	DOUBLE	No	0	Yes	Yes	No	No
SIML	INLINK	No	Null	Yes	No	No	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

This section describes the fields that will be of interest to a typical application developer.

Name	Summary	Description
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
CBST	Callback Structure	Pointer to a record private structure.
INnN	Input n Process Variable Name	(n=A thru L): These fields contain the Process Variable names of the inputs that will be used for the calculation.
INnV	Input n Valid	(n=A thru L): These flags indicate if the ASCII string entered in `INnN` was found to be a valid (existing) Process Variable.
A,...,L	Input Values	If the corresponding `INP` field is a constant, this field is initialized with the constant value but can be changed via `dbPuts`.
LA,...,LL	Previous Input Values	These fields are used to decide when to trigger monitors on A,...,L.
CALC	Calculation String	Expression to be calculated when the record is processed. Identical to the `CALC` field in the `calc` record.
RPCL	Reverse Polish Calc String	String used for interpreting `CALC` string.
CLCV	Calculation String Valid	This flag is set when the `CALC` string is valid.
VAL	Value Field	This field is calculated, via the `CALC` expression, each time the record is processed.
OVAL	Old Value	
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
OOPT	Output Option	This menu specifies when to execute the output link and post the output event. Choices are “Every Time”, “On Change”, “When Zero”, “When Non-zero”, “Transition To Zero”, “Transition To Non-zero”
OUTN	Output Link Process Variable Name	This field contains the Process Variable name of the output link.
OUTA	Output Address	Pointer to the `dbAddr` structure of the PV in `OUTN`.
OUTV	Output Link Valid	This flag indicates if the ASCII string entered in `OUTN` was found to be a valid (existing) Process Variable.

Name	Summary	Description
DOLN	Desired Output Location Process Variable Name	This field contains the name of the Process Variable from which the desired output data will be retrieved (if `DOPT` so indicates).
DOLA	Desired Output Location Address	Pointer to the `dbAddr` Structure of the PV in `DOLN`.
DOLV	Desired Output Location Valid	This flag indicates if the ASCII string entered in `DOLN` was found to be a valid (existing) Process Variable.
DOLD	Desired Output Location Data	This field contains the data fetched from the PV specified in `DOLN` (if valid) or a user entered value (if `DOLN` is not valid).
DOPT	Data Option	This menu specifies whether to use the `DOLD` field or the `VAL` field as output data.
OEVT	Output Event	If non-zero, the specified event number will be “posted” when the output link is executed.
ADEL	Archive Deadband	These parameters specify hysteresis factors for triggering monitor callbacks, i.e. monitors specified by calls to `caAddEvent` or `dbAddEvent`. A monitor will not be triggered until `VAL` changes by more than the specified amount.
MDEL	Monitor, i.e. value change, Deadband	
ALST	Archive Last Value	These fields are used to implement the hysteresis factors for monitors.
MLST	Monitor Last Value	
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 11 for more information.
SIML	Simulation Mode Location	
SVAL	Simulation Value	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

5. Record Processing

6. Device Support

7. Device Support For Soft Records

Chapter 40: Waveform

1. Introduction

The `waveform` record type stores arrays as data. The array can contain any of the supported data types.

2. Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	See FTVL	No	0	Yes	Yes	Yes	Yes
RARM	SHORT	Yes	0	Yes	Yes	No	Yes
PREC	SHORT	Yes	0	Yes	Yes	No	No
INP	INLINK	Yes	0	No	No	N/A	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
NELM	ULONG	Yes	1	Yes	No	No	No
FTVL	GBLCHOICE	Yes	0	Yes	No	No	No
BPTR	NOACCESS	No	0	No	No		No
NORD	ULONG	No	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
BUSY	SHORT	No	0	Yes	No	No	No
SIOL	INLINK	Yes	0	No	No	N/A	No
SIML	INLINK	Yes	0	No	No	N/A	No
SIMM	GBLCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

3. Field Descriptions

Name	Summary	Description
VAL	Value Field	This is used to reference the array.
RARM	Rearm	When set to 1, the device will be rearmed.
PREC	Display Precision	Precision with which to display `VAL`. This field is used by record support to supply a value when `get_precision` is called.
INP	Input Link	This field is used by the device support routines to obtain input.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when `get_units` is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphics displays and the upper and lower control limits for control displays. The fields are used by record support to honor calls to `get_graphic_double` or `get_control_double`.
LOPR	Low Operating Range	
NELM	Number of Elements, in array	
FTVL	Field Type of Value	This is `DBF_STRING`, ... , `DBF_ENUM`.
BPTR	Buffer Pointer	Holds address of array.
NORD	Number of Elements Read	
BUSY	Busy	Is device busy?
SIMM	Simulation Mode	Simulation mode process variables. Refer to Chapter 3, Section "Simulation Mode" on page 13 for more information.
SIML	Simulation Mode Location	
SIOL	Simulation Value Location	
SIMS	Simulation Mode Alarm Severity	

4. Record Support Routines

init_record

Using `NELM` and `FTVL` space for the array is allocated. The array address is stored in the record.

This routine initializes `SIMM` with the value of `SIML` if `SIML` type is `CONSTANT` link or creates a channel access link if `SIML` type is `PV_LINK`. `VAL` is likewise initialized if `SIOL` is `CONSTANT` or `PV_LINK`.

This routine next checks to see that device support is available and a device support read routine is defined. If either does not exist, an error message is issued and processing is terminated.

If device support includes `init_record`, it is called.

process See next section.

get_value Fills in the values of struct `valueDes` so that they refer to the array.

cvt_dbaddr This is called by `dbNameToAddr`. It makes the `dbAddr` structure refer to the actual buffer holding the result.

get_array_info Obtains values from the array referenced by `VAL`.

put_array_info Writes values into the array referenced by `VAL`.

get_units Retrieves `EGU`.

get_prec Retrieves `PREC`.

get_graphic_double Sets the upper display and lower display limits for a field. If the field is `VAL` the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_control_double Sets the upper control and the lower control limits for a field. If the field is `VAL` the limits are set to `HOPR` and `LOPR`, else if the field has upper and lower limits defined they will be used, else the upper and lower maximum values for the field type will be used.

get_graphic_double Sets the following values:

```
upper_disp_limit = HOPR
lower_disp_limit = LOPR
```

get_control_double Sets the following values

```
upper_ctrl_limit = HOPR
lower_ctrl_limit = LOPR
```

5. Record Processing

Routine `process` implements the following algorithm:

1. Check to see that the appropriate device support module exists. If it doesn't, an error message is issued and processing is terminated with the `PACT` field still set to `TRUE`. This ensures that processes will no longer be called for this record. Thus error storms will not occur.
2. Call device support read routine.
3. If `PACT` has been changed to `TRUE`, the device support read routine has started but has not completed writing the new value. In this case, the processing routine merely returns, leaving `PACT TRUE`.
4. Check to see if monitors should be invoked.
   - Alarm monitors are invoked if the alarm status or severity has changed.
   - Archive and value change monitors are always invoked.
   - `NSEV` and `NSTA` are reset to 0.
5. Scan forward link if necessary, set `PACT FALSE`, and return.

6. Device Support

Fields Of Interest To Device Support

Each `waveform` record must have an associated set of device support routines. The primary responsibility of the device support routines is to obtain a new array value whenever `read_wf` is called. The device support routines are primarily interested in the following fields:

Name	Summary	Description
PACT	Processing Active	See Chapter 2, Section "Database Common: Field Descriptions" on page 4 for descriptions.
DPVT	Device Private	
NSEV	New Alarm Severity	
NSTA	New Alarm Status	
INP	Input Link	This field is used by the device support routines to locate its input.
RATE	Sampling Rate	Some device support modules may find this useful.
PTSS	Pretrigger Samples	Some device support modules may find this useful.
NELM	Number Of Elements In Array	
FTVL	Field Type Of Value	This is `DBF_STRING`, ... , `DBF_ENUM`. The device support routine should check that this is correctly defined.
RARM	Rearm	When set to 1, the device will be rearmed. The device support routine should reset it to 0 when done.
BPTR	Holds Address Of Array	

Name	Summary	Description
NORD	Number Of Elements Read	Device support must set this value when it completes.
BUSY	Is device busy?	

Device Support Routines

Device support consists of the following routines:

report

```
report(FILE fp, paddr)
```

Not currently used.

init

```
init()
```

This routine is called once during IOC initialization.

init_record

```
init_record(precord)
```

This routine is optional. If provided, it is called by the record support `init_record` routine.

get_ioint_info

```
get_ioint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)
```

This routine is called by the `ioEventScan` system each time the record is added or deleted from an I/O event scan list. `cmd` has the value (0,1) if the record is being (added to, deleted from) an I/O event list. It must be provided for any device type that can use the `ioEvent` scanner.

read_wf

```
read_wf(precord)
```

This routine must provide a new input value. It returns the following values:

- **0**: Success.
- **Other**: Error.

7. Device Support For Soft Records

If `INP` is a constant link, then `read_wf` does nothing. In this case, the record can be used to hold arrays written via `dbPuts`. If `INP` is a database or channel access link, the new array value is read from the link. `NORD` is set.

This module places a value directly in `VAL`.

If the `INP` link type is constant, then `NORD` is set to zero. If the `INP` link type is `PV_LINK`, then `dbCaAddInlink` is called by `init_record`.

`read_wf` calls `recGblGetLinkValue` which performs the following steps:

- If the `INP` link type is `CONSTANT` `recGblGetLinkValue` does nothing.

- If the `INP` link type is `DB_LINK`, then `dbGetLink` is called to obtain a new input value. If `dbGetLink` returns an error, a `LINK_ALARM` with a severity of `INVALID_ALARM` is raised.
- If the `INP` link type is `CA_LINK`, then `dbCaGetLink` is called to obtain a new input value. If `dbCaGetLink` returns an error, a `LINK_ALARM` with a severity of `INVALID_ALARM` is raised.

`NORD` is set to the number of values returned and `read_wf` returns.