CHAPTER 1  Introduction .................................................. 1

CHAPTER 2  Fields Common to All Record Types ....................... 3
  2.1 Database Common: Field Summary .................................. 3
  2.2 Database Common: Field Descriptions .......................... 3

CHAPTER 3  Fields Common to Many Record Types ..................... 9
  3.1 Input Records .......................................................... 9
  3.1.1 Common Fields ..................................................... 9
  3.1.2 Device input ........................................................ 10
  3.1.3 Soft input .......................................................... 10
  3.1.4 Simulation mode .................................................. 11
  3.2 Output Records ....................................................... 11
  3.2.1 Common Fields .................................................... 11
  3.2.2 Soft Output ........................................................ 13
  3.2.3 Output Mode Select ................................................ 14
  3.2.4 Simulation Mode .................................................. 14
  3.2.5 Invalid Alarm Output Action ................................. 15

CHAPTER 4  Ai — Analog Input ............................................ 17
  4.1 Field Summary ....................................................... 17
  4.2 Field Descriptions .................................................. 18
  4.3 Record Support Routines ........................................... 20
  4.4 Record Processing .................................................. 21
  4.5 Device Support ...................................................... 23
  4.5.1 Fields of interest to device support ........................ 23
  4.5.2 Device Support routines ................................. 23
  4.6 Device support for soft records ............................ 24

CHAPTER 5  Ao — Analog Output ......................................... 27
  5.1 Field Summary ....................................................... 27
  5.2 Field Descriptions .................................................. 28
  5.3 Record Support Routines ........................................... 31
  5.4 Record Processing .................................................. 32
  5.5 Device Support ...................................................... 34
  5.5.1 Fields of interest to device support ........................ 34
  5.5.2 Device Support routines ................................. 34
  5.6 Device support for soft records ............................ 35

CHAPTER 6  Bi — Binary Input ........................................... 37
  6.1 Field Summary ....................................................... 37
  6.2 Field Descriptions .................................................. 37
  6.3 Record Support Routines ........................................... 38
  6.4 Record Processing .................................................. 39
  6.5 Device Support ...................................................... 40
  6.5.1 Fields of interest to device support ........................ 40
  6.5.2 Device Support routines ................................. 40
  6.6 Device support for soft records ............................ 41

CHAPTER 7  Bo — Binary Output ......................................... 43
  7.1 Field Summary ....................................................... 43
25.4 Record Processing ........................................ 145
25.5 Device Support ........................................... 145

CHAPTER 26 Stringin — String Input .......................... 147
26.1 Field Summary ........................................... 147
26.2 Field Descriptions ....................................... 147
26.3 Record Support Routines ................................. 147
26.4 Record Processing ....................................... 147
26.5 Device Support .......................................... 148
26.5.1 Fields of interest to device support ................. 148
26.5.2 Device Support routines ............................... 148
26.6 Device support for soft records ......................... 149

CHAPTER 27 Stringout — String Output ....................... 151
27.1 Field Summary ........................................... 151
27.2 Field Descriptions ....................................... 151
27.3 Record Support Routines ................................. 152
27.4 Record Processing ....................................... 152
27.5 Device Support .......................................... 153
27.5.1 Fields of interest to device support ................. 153
27.5.2 Device Support routines ............................... 153
27.6 Device support for soft records ......................... 154

CHAPTER 28 Subroutine ....................................... 155
28.1 Field Summary ........................................... 155
28.2 Field Descriptions ....................................... 156
28.3 Record Support Routines ................................. 158
28.4 Record Processing ....................................... 159
28.5 Example Synchronous Subroutine ....................... 159
28.6 Example Asynchronous Subroutine ....................... 160

CHAPTER 29 Timer ............................................ 163
29.1 Field Summary ........................................... 163
29.2 Field Descriptions ....................................... 164
29.3 Record Support Routines ................................. 165
29.4 Record Processing ....................................... 165
29.5 Device Support .......................................... 165

CHAPTER 30 Waveform ....................................... 167
30.1 Field Summary ........................................... 167
30.2 Field Descriptions ....................................... 167
30.3 Record Support Routines ................................. 168
30.4 Record Processing ....................................... 169
30.5 Device Support .......................................... 169
30.5.1 Fields of interest to device support ................. 169
30.5.2 Device Support routines ............................... 170
30.6 Device support for soft records ......................... 171
30.7 Device support for soft records ......................... 171
This document describes the standard record types supported by the IOC databases. The basic concept of record processing was developed by Bob Dalesio of LANL/GTA. Bob was also the designer and implementer of the initial record types. This software was developed over a period of several years with feedback from LANL/GTA users. Without their ideas EPICS would not exist.

During 1990 and 1991, ANL/APS undertook a major revision of the IOC software with the primary goal being to provide easily extendible record and device support. Marty Kraimer (ANL/APS) was primarily responsible for designing the data structures needed to support extendible record and device support and for making the changes needed to the IOC resident software. Bob Zieman (ANL/APS) designed and implemented the UNIX tools and IOC modules necessary to support the new facilities. Frank Lenkszus (ANL/APS) made extensive changes to the Database Configuration Tool (DCT) necessary to support the new facilities. Janet Anderson (ANL/APS) developed methods to systematically test various features of the IOC software and is presently involved with developing support for new record types.

The current EPICS software development team consists of the following individuals:

LANL/GTA: Roger Cole, Bob Dalesio, Betty Gunther, Jeff Hill, Deb Kerstiens, Andy Kozubal, and Cindy Eaton

ANL/APS: Janet Anderson, Mark Anderson, Ben–chin Cha, Nick Karonis, Jim Kowalkowski, Marty Kraimer, John Winans and Bob Zieman
CHAPTER 1  Introduction

This manual describes all supported 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 several input and output field names that are common to multiple record types and have the same usage wherever they are used. Following the third chapter there is a separate chapter for each record type containing a description of all the fields for that record type except those in database common.

Each chapter contains a field summary table of the form:

FIELD SUMMARY

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
</table>

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?
Interest Interest level for the output of the database test routines.

0 Application developer: Field may change during processing
1 Application developer: Not changed during processing
2 System developer: Major interest
3 System developer: Minor interest
4 No interest: pad field

Initial Initial value when record is created
Access   Is this field accessible via database access?
Modify   Can this field be modified via database access?
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

This chapter contains a description of fields that are common to all records.

2.1 Database Common: Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>DESC</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SCAN</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>PINI</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>PHAS</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>EVNT</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SDIS</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>DISV</td>
<td>SHORT</td>
<td>Yes</td>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DISA</td>
<td>SHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DTYP</td>
<td>DEVCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>DISP</td>
<td>UCHAR</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PROC</td>
<td>UCHAR</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>MLOK</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLIS</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STAT</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>0</td>
<td>udf</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>SEVR</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>0</td>
<td>invalid</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>NSTA</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>NSEV</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>DISS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td>Option</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSET</td>
<td>SHORT</td>
<td>No</td>
<td>4</td>
<td>compute</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LCNT</td>
<td>UCHAR</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>PACT</td>
<td>UCHAR</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SPVT</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSET</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSET</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPVT</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPRO</td>
<td>UCHAR</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>UDF</td>
<td>UCHAR</td>
<td>No</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PRIO</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FLINK</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

2.2 Database Common: Field Descriptions

NAME  Record Name

An arbitrary 28 character record name supplied by the application devel-
oper. 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.

**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 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)
```

**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.

**DISV** Disable Value

If DISV=DISA, then the record will be disabled, i.e. dbProcess will not process the record.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |
| 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. |
| 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 passive record will be processed whenever a dbPutField is directed to this field. This field is present because often it is not clear which field to write to when the only desire is to force processing. |
| 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)
```

mask is some combination of DBE_ALARM, DBE_VALUE, and DBE_LOG. |
| STAT   | Current Alarm Status |
| SEVR   | Current Alarm Severity |
NSTA New Alarm Status  
NSEV New Alarm Severity

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:

\[
\text{recGblSetSevr}(\text{precord, new\_status, new\_severity})
\]

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.

DISS Disable Alarm Severity

When this record is disabled, it will be put into alarm with this severity and a status of DISABLE_ALARM.

TIME Time

The time when this record was last processed, in standard format.

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.

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.
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.

TPRO  Trace Processing.
If this field is set True, a message is printed each time this record is pro-
cessed and a message is printed for each record processed as a result of
this record being processed.

UDF  VAL Undefined
UDF is initialized to True at IOC initialization. Record and device sup-
port routines which write to the VAL field are responsible for setting UDF
to False.

PRIO  Priority
Scheduling priority for processing I/O Event scanned records and asynch-
ronous record completion tasks.

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

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

3.1 Input Records

3.1.1 Common Fields

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.

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.

3.1.2 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 A hardware error is detected (in this case, it should also raise an alarm condition).
   b The device support routine reads values directly into the VAL field and then sets UDF to False.

3.1.3 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 datatype 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 retrans 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.

### 3.1.4 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.

### 3.2 Output Records

#### 3.2.1 Common Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>Output Link</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTYP</td>
<td>Device Type</td>
</tr>
</tbody>
</table>

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.

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

**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.

### 3.2.2 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.

### 3.2.3 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.

### 3.2.4 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.
3.2.5 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

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.

4.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LINR</td>
<td>CVTCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGUF</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGUL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>AOFF</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ASLO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SMOO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOLO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HIGH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOW</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HHSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HYST</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ADEL</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDEL</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ROFF</td>
<td>LONG</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ESLO</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LALM</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ALST</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>MLST</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>PBRK</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INIT</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LBRK</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>RVAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
4.2 Field Descriptions

**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 not used by record support other than 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
**EGUL**  Engineering Units Low
**ROFF**  Raw Value Offset
**ESLO**  Slope for Linear Conversions

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:

\[ \text{VAL} = (\text{RVAL} + \text{ROFF}) \times \text{ESLO} + \text{EGUL} \]

**EGU**  
Engineering Units  
An ASCII string of up to 16 characters describing the engineering units. This field is not used by record support other than to supply a units description string when `get_units` is called.

**HOPR**  
High Operating Range  
**LOPR**  
Low 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 not used by the record support routines themselves other than to honor calls to `get_graphic_double` or `get_control_double`.

**AOFF**  
Adjustment Offset  
**ASLO**  
Adjustment Slope  
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.

**SMOO**  
Smoothing Factor  
The converted data value is subjected to the following algorithm:

\[ \text{val} = \text{newvalue} \times (1 - \text{smoo}) + \text{oldvalue} \times \text{smoo} \]

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).

**HIHI**  
Hihi Alarm Limit  
**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  
These fields specify the alarm limits and severities.
HYST  Alarm Deadband
ADEL  Archive Deadband
MDEL  Monitor, i.e. value change, 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.

LALM  Value when last monitors for alarm were triggered
ALST  Value when last monitors for archiver were triggered
MLST  Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitor callbacks.

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`.

### 4.3 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. In addition 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

4.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process will no longer be called for this record. Thus error storms will not occur.
2 ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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:

\[ \text{val} = \text{val} \times \text{ASLO} + \text{AOFF} \]

If the conversion algorithm is linear, the raw value is converted via the equation:

\[ \text{val} = \text{val} \times \text{ESLO} + \text{EGUL} \]

If the conversion is via a breakpoint table, the new value is obtained.

The next step is to apply the following smoothing algorithm:

\[ \text{if SMOO equal to 0. or INIT is True, VAL = val} \]
\[ \text{else VAL = val \times (1 – SMOO) + Previous_value \times SMOO} \]

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 Checks 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 Scans forward link if necessary, sets PACT and INIT to False, and returns.
4.5 Device Support

4.5.1 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:

- **PACT, DPVT, UDF**  See Chapter titled Fields Common to All Record Types for description.
- **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,EGUL**  Engineering Units Full and Engineering Units Low. These fields are used to calculate ESLO. Note that these fields correspond to the high and low hardware limits.
- **ESLO,ROFF**  Slope and Raw Offset. 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.
- **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.
- **NSEV,NSTA**  See Chapter titled Fields Common to All Record Types for description.

4.5.2 Device Support routines

Device support consists of the following routines:

- **report(FILE fp, paddr)**
  Not currently used.
- **init()**
  This routine is called once during IOC initialization.
- **init_record(precord)**
  This routine is optional. If provided, it is called by the record support init_record routine.
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 form 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(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 of 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(precord,after)

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

4.6 Device support for soft records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for 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 Soft Input in Fields Common to Many Record Types 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

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.

5.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OVAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OROC</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OIF</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LINR</td>
<td>CVTCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGUF</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGUL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ESLO</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROFF</td>
<td>LONG</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DRVH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>DRVL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOLO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HIGH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LOW</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>HHSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HYST</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ADEL</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDEL</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RVAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ORAW</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>RBV</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ORBV</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>PVAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LALM</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ALST</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
5.2 Field Descriptions

**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, 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.

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.

LINR  Conversion type

NO_CONVERSION, LINEAR and breakpoint table conversion are supported.

EGUF  Engineering Units Full
EGUL  Engineering Units Low
ESLO  Slope for linear conversions
ROFF  Raw value offset

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:

\[ RVAL = \frac{OVAL - EGUL}{ESLO} - ROFF \]

EGU  Engineering Units

ASCII string describing Engineering units. This field is not used by record support other than to supply a units description string when get_units is called.
DRVH  Drive High
DRVL  Drive Low

If these values are defined then VAL will forced to be in the range
DRVL<=VAL<=DRVH

HOPR  High Operating Range
LOPR  Low Operating Range

These fields determine the upper and lower display limits for graphics dis-
plays and the upper and lower control limits for control displays. The
fields are not used by the record support routines themselves other than 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.

HIHI  Hihi Alarm Limit
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

These fields specify the alarm limits and severities.

HYST  Alarm Deadband
ADEL  Archive Deadband
MDEL  Monitor, i.e. value change, Deadband

These parameters specify hysteresis factors for triggering monitor call-
backs, i.e. callbacks specified by calls to caAddEvent or dbAddEvent. A
monitor will not be triggered until VAL changes by more than the speci-
fied amount.

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 trig-
gerred for RVAL.

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**  Value when last monitors for alarm were triggered

**ALST**  Value when last monitors for archiver were triggered

**MLST**  Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitors.

**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

### 5.3 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. In addition, INIT is set True. This causes PBRK, LBRK, and smoothing to be re-initialized. If linear conversion is requested, then VAL is computed from RVAL using the algorithm:
\[ VAL = \frac{(RVAL + ROFF)}{ESLO + EGUL} \]

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. In addition 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

### 5.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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:

\[
RVAL = \frac{(OVAL–EGUL)}{ESLO} – ROFF
\]

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 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 is reduced.

4 Check severity and write the new value. See Invalid Alarm Output Action and Simulation Mode described in Fields Common to Many Record Types 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 Checks 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, sets PACT and INIT False, and returns.
5.5 Device Support

5.5.1 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:

- **PACT, DPVT**: See Chapter titled Fields Common to All Record Types for description.
- **OUT**: This field is used by the device support routines to locate its output.
- **EGUF, EGUL**: These fields are used to calculate ESLO. Note that these fields correspond to the high and low hardware limits.
- **ESLO, ROFF**: Slope and raw value offset for linear conversions from raw to engineering units. The device support routines must calculate this field unless they obtain values already converted to engineering units.
- **RVAL**: Raw data value. This is the value to write to OUT.
- **NSEV, NSTA**: See Chapter titled Fields Common to All Record Types for description.

5.5.2 Device Support routines

Device support consists of the following routines:

**init()**
This routine is called once during IOC initialization.

**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(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(precord)**
This routine must output a new value. It returns one of the following values:
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(precord,after)

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

5.6 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 Soft Output in Fields Common to Many Record Types 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

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.

6.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>ENUM</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>COSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ZNAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ONAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>RVAL</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ORAW</td>
<td>ULONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MASK</td>
<td>ULONG</td>
<td>No</td>
<td>1</td>
<td>compute</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LALM</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MLST</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVAL</td>
<td>USHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

6.2 Field Descriptions

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

6.3 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.

### 6.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process will no longer be called for this record. Thus error storms will not occur.

2. ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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

   \[
   \text{TIME} = \text{tslocaltime}
   \]

   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. Checks 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 Scans forward link, if necessary, sets PACT False, and returns.

6.5 Device Support

6.5.1 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:

- **PACT**, **DPVT**, **UDF**  See Chapter titled Fields Common to All Record Types for description.
- **VAL**  This field is set by a device support routines only if it doesn’t want record support to set it.
- **INP**  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**  The device support routine must give this field a value it it needs to use it.
- **NSEV**, **NSTA**  See Chapter titled Fields Common to All Record Types for description.

6.5.2 Device Support routines

Device support consists of the following routines:

**report(FILE fp, paddr)**

Not currently used.

**init()**

This routine is called once during IOC initialization.

**init_record(precord)**

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

**get_joint_info(int cmd,struct dbCommon *precord,IOSCANPVT *ppvt)**

This routine is called by the ioEventScan system each time the record is added or deleted form 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(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.

### 6.6 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 Soft Input in Fields Common to Many Record Types 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

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.

7.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>ENUM</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIGH</td>
<td>FLOA</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ZNAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ONAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RVAL</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ORAW</td>
<td>ULONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>MASK</td>
<td>ULONG</td>
<td>No</td>
<td>1</td>
<td>compute</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>WDPT</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td>compute</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>COSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RBV</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ORBV</td>
<td>ULONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>MLST</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LALM</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOA</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOV</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

7.2 Field Descriptions

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 (an Input Link)

If DOL is a database or channel access link and OMSL is CLOSED_LOOP, then VAL is read from 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.

**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.

### MLST
Monitor Last

Value when last monitor for value changes was triggered

### LALM
Last Alarmed

Value when last change of state alarm was issued.

### 7.3 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.

7.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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 Invalid Alarm Output Action and Simulation Mode described in Fields Common to Many Record Types 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. Checks 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 Scans forward link if necessary, sets PACT False, and returns.

7.5 Device Support

7.5.1 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:

PACT, DPVT See Chapter titled Fields Common to All Record Types for description.

VAL This field is only of interest to device support routines that do not use MASK and RVAL.

OUT 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.

NSEV,NSTA See Chapter titled Fields Common to All Record Types for description.

7.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, paddr)

Not currently used.

init()

This routine is called once during IOC initialization.

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(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(precord)

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

0           Success.
other       Error.

7.6 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 Soft Output in Fields Common to Many Record Types for details.

Raw Soft Channel

This module is like the previous except that it writes the current value of RVAL.
CHAPTER 8 Calculation

This record calculates an expression.

### 8.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>CALC</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INPA</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPB</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPC</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPD</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPE</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPF</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPG</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPH</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPJ</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPK</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>B</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>C</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>D</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>E</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>G</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>H</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>I</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>J</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>K</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>L</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LA</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LB</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LC</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LD</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LE</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LF</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LG</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LH</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LI</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LJ</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LK</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LL</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
### 8.2 Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>Value Field</td>
</tr>
<tr>
<td>CALC</td>
<td>Infix expression</td>
</tr>
<tr>
<td>INPA,...,INPL</td>
<td>Input Links. Each may be a constant, a database link, or a channel access link. Any link not defined is ignored.</td>
</tr>
<tr>
<td>A,...,L</td>
<td>Input Values If the corresponding INP field is a constant, this field is initialized with the constant value but can be changed via dbPuts.</td>
</tr>
<tr>
<td>LA,...,LL</td>
<td>Previous Input Values These fields are used to decide when to trigger monitors on A,...,L.</td>
</tr>
<tr>
<td>EGU</td>
<td>Engineering Units</td>
</tr>
</tbody>
</table>
A 16 character ASCII string describing Engineering units. This field is not used by record support other than to supply a units description string when get_units is called.

**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.

**HOPR**
High Operating Range

**LOPR**
Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

**HIHI**
Hihi Alarm Limit

**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

These fields specify the alarm limits and severities.

**HYST**
Alarm Deadband

**ADEL**
Archive Deadband

**MDEL**
Monitor, i.e. value change, 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.

**LALM**
Last Alarmed Value when last monitors for alarm were triggered

**ALST**
Archive Last Value when last monitors for archiver were triggered

**MLST**
Monitor Last Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitors.

**RPCL**
Expression in reverse polish.

### 8.3 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 ac-
cess 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

### 8.4 Record Processing

Routine process implements the following algorithm:

1. Fetches all arguments.
2. Calls 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. Checks 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.

Scans forward link if necessary, sets PACT False, and returns.

8.5 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:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC</td>
<td>Infix expression as entered</td>
</tr>
<tr>
<td>RPCL</td>
<td>Reverse polish expression</td>
</tr>
</tbody>
</table>

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.

8.5.1 Operands

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Use the value specified by input A</td>
</tr>
<tr>
<td>B</td>
<td>Use the value specified by input B</td>
</tr>
<tr>
<td>C</td>
<td>Use the value specified by input C</td>
</tr>
<tr>
<td>D</td>
<td>Use the value specified by input D</td>
</tr>
<tr>
<td>E</td>
<td>Use the value specified by input E</td>
</tr>
<tr>
<td>F</td>
<td>Use the value specified by input F</td>
</tr>
<tr>
<td>G</td>
<td>Use the value specified by input G</td>
</tr>
<tr>
<td>H</td>
<td>Use the value specified by input H</td>
</tr>
<tr>
<td>I</td>
<td>Use the value specified by input I</td>
</tr>
<tr>
<td>J</td>
<td>Use the value specified by input J</td>
</tr>
<tr>
<td>K</td>
<td>Use the value specified by input K</td>
</tr>
<tr>
<td>L</td>
<td>Use the value specified by input L</td>
</tr>
<tr>
<td>RNDM</td>
<td>Random number (unary), random number between 0–1.</td>
</tr>
</tbody>
</table>

8.5.2 Algebraic Operators

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS</td>
<td>Absolute value (unary)</td>
</tr>
<tr>
<td>SQR</td>
<td>Square root (unary)</td>
</tr>
<tr>
<td>MIN</td>
<td>Minimum (binary function)</td>
</tr>
<tr>
<td>MAX</td>
<td>Maximum (binary function)</td>
</tr>
<tr>
<td>CEIL</td>
<td>Ceiling (unary)</td>
</tr>
<tr>
<td>FLOOR</td>
<td>Floor (unary)</td>
</tr>
<tr>
<td>LOG</td>
<td>Log base 10 (unary)</td>
</tr>
<tr>
<td>LOGE</td>
<td>Natural log (unary)</td>
</tr>
<tr>
<td>EXP</td>
<td>Exponential function (unary)</td>
</tr>
<tr>
<td>^</td>
<td>Exponential (binary)</td>
</tr>
</tbody>
</table>
** Exponential (binary)
+  Addition (binary)
−  Subtraction (binary)
*  Multiplication (binary)
/  Division (binary)
%  Modulo (binary)
NOT Negate (unary)

8.5.3 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

8.5.4 Relational Operators

>= Greater than or equal to
>  Greater than
<= Less than or equal to
<  Less than
#  Not equal to
=  Equal to

8.5.5 Logical Operators

&& And
|| Or
!  Not

8.5.6 Bitwise Operators

|  Bitwise or
&  Bitwise and
OR Bitwise or
AND Bitwise and
XOR Bitwise exclusive or
~  One’s complement
<< Left shift
>> Right shift

8.5.7 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.
### 8.5.8 Conditional Expression

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

\[(\text{condition})? \text{True result} : \text{False result}\].

### 8.6 Example Expressions

<table>
<thead>
<tr>
<th>Type</th>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algebraic</td>
<td>A + B</td>
<td>Result is A + B</td>
</tr>
</tbody>
</table>
| 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

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 must 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 subarrays each of length N, and the algorithm applied to each subarray. The result obtained from each subarray is written to the circular buffer referenced by VAL.

9.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RES</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ALG</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>NSAM</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ILIL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IHIL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OFF</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>NUSE</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>BPTR</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPTR</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WPTR</td>
<td>NOACCESS</td>
<td>No</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVB</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>INX</td>
<td>ULONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
9.2 Field Descriptions

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.

NSAM  Number in Sample

Number of elements in VAL.

N  Number

Value of N for AVERAGE and NTO1xxx algorithms.

ILIL,IHIL  Initial low and high interest values. Applies to NTO1xxx applied to INP arrays.

HOPR  High Operating Range

LOPR  Low 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 not used by the record support routines themselves other than to honor calls to getGraphic_double or getControl_double.

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.

EGU  Engineering Units
ASCII string describing Engineering units. This field is not used by record support other than 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.

### 9.3 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.

9.4 Record Processing

Routine process implements the following algorithm:
1  If INP is not a database link just check monitors and the forward link and return.
2  Gets 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, checks 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  Checks 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 Scans forward link if necessary, sets PACT False, and returns.
CHAPTER 10   Event

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.

10.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>SHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVAL</td>
<td>USHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

10.2 Field Descriptions

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.

10.3 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.

10.4 Record Processing

Routine process implements the following algorithm:

1    ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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 Checks 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 Scans forward link if necessary, sets PACT False, and returns.

10.5 Device Support

10.5.1 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:

PACT, DPVT, UDF  See Chapter titled Fields Common to All Record Types for description.

INP  This field is used by the device support routines to locate its input.

NSEV, NSTA  See Chapter titled Fields Common to All Record Types for description.

PRIO  This value must be used by the device support interrupt handler to set the scheduling priority for processing this record.

10.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, interest)

Not currently used.

init()

This routine is called once during IOC initialization.

init_record(precord)

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

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(precord)

This routine returns the following values:

0 Success.

other Error.

10.6 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 Soft Input in Fields Common to Many Record Types for details.

If the return status of recGblGetLinkValue is zero, then read_event sets UDF to False. The status of recGblGetLinkValue is returned.
CHAPTER 11  Fanout

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

11.1  Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SELM</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SELN</td>
<td>USHORT</td>
<td>No</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SELL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK1</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK2</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK3</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK4</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK5</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNK6</td>
<td>FWDLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

11.2  Field Descriptions

**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.

11.3 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.

11.4 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  Checks 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  Scans forward link if necessary, sets PACT False, and returns.
CHAPTER 12  Histogram

NOTE: This record type is undergoing revision. The field names and functions will change. This record type is used to store frequency counts of a signal into an array of arbitrary length.

## 12.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>See BPTR</td>
<td>No</td>
<td>0</td>
<td>udf</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>NELM</td>
<td>USHORT</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>RES</td>
<td>SHORT</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ULIM</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLIM</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SGNL</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SVL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>udf</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPTR</td>
<td>NOACCESS</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>MCNT</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MDEL</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

## 12.2 Field Descriptions

**VAL**

Value Field.

This field is used to reference the array.

**NELM**

Number of elements in array.

**RES**

Reset Array.

Whenever RES is set to a nonzero value, the array elements counts will be reset to zero. RES is then set to 0.

**ULIM**

Upper Signal Limit.

**LLIM**

Lower 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.

SGNL  Signal Value.

SVL  Signal Value Location, an 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.

MCNT  Monitor Counts

Number of counts since last monitor.

MDEL  Monitor Delta

Monitor count deadband.

12.3 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 support and 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.

12.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process will no longer be called for this record. Thus error storms will not occur.

2. ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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. Checks 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. Scans forward link if necessary, sets PACT and INIT to False, and returns.
CHAPTER 13  Long Input

The normal use for this 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.

13.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOPR</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIGH</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOW</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HHSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HYST</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ADEL</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDEL</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LALM</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ALST</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>MLST</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

13.2 Field Descriptions

**VAL**  Value Field

This is the value resulting from 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 not used by record support other than to supply a units description string when get_units is called.

**HOPR**

High Operating Range

**LOPR**

Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

**HIHI**

Hihi Alarm Limit

**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

These fields specify the alarm limits and severities.

**HYST**

Alarm Deadband

**ADEL**

Archive Deadband

**MDEL**

Monitor, i.e. value change, 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.

**LALM**

Last Alarmed Value when last monitors for alarm were triggered

**ALST**

Archive Last Value when last monitors for archiver were triggered

**MLST**

Monitor Last Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitor callbacks.

## 13.3 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
```

### 13.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process will no longer be called for this record. Thus error storms will not occur.

2. ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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 Checks 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 Scans forward link, if necessary, sets PACT False, and returns.

13.5 Device Support

13.5.1 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:

PACT, DPVT, UDF  See Chapter titled Fields Common to All Record Types for description.

VAL  This field is set by device support routines.

INP  This field is used by the device support routines to locate its input.

NSEV,NSTA  See Chapter titled Fields Common to All Record Types for description.

13.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, paddr)

Not currently used.

init()

This routine is called once during IOC initialization.

init_record(precord)

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

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(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.

13.6 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 Soft Input in Fields Common to Many Record Types 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 14  Long Output

The normal use for this 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.

14.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOLO</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HIGH</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOW</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HHSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HYST</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ADEL</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDEL</td>
<td>LONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LALM</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ALST</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>MLST</td>
<td>LONG</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOA</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOV</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

14.2 Field Descriptions

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 (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.

EGU  Engineering Units

ASCII string describing Engineering units. This field is not used by record support other than to supply a units description string when get_units is called.

HOPR, LOPR  High and Low 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 not used by the record support routines themselves other than 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.

HIHI, HIGH, LOW, LOLO  Hihi, High, Low Alarm Limits

HHSV, HSV, LSV, LLSV  Severity for a Hihi, High, Low Alarm

These fields specify the alarm limits and severities.

HYST  Alarm Deadband

ADEL  Archive Deadband
MDEL Monitor, i.e. value change, 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.

LALM Last Alarmed Value when last monitors for alarm were triggered
ALST Archive Last Value when last monitors for archiver were triggered
MLST Monitor Last Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitors.

14.3 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

14.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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 Soft Input in Fields Common to Many Record Types 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 Invalid Alarm Output Action and Simulation Mode described in Fields Common to Many Record Types 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

14.5 Device Support

14.5.1 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:
PACT, DPVT  See Chapter titled Fields Common to All Record Types for description.

OUT This field is used by the device support routines to locate its output.

NSEV,NSTA  See Chapter titled Fields Common to All Record Types for description.

14.5.2 Device Support routines

Device support consists of the following routines:

`init()`

This routine is called once during IOC initialization.

`init_record(precord)`

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

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

This routine is called by the ioEventScan system each time the record is added or deleted form 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(precord)`

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

0 Success.

other Error.

14.6 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 Soft Output in Fields Common to Many Record Types for details.
CHAPTER 15  Mbbi – MultiBit Binary Input

The normal use for this 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.

15.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>ENUM</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NOBT</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZRVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ONVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TWVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>THVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FRVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SXVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EIVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>NIVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TEVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ELVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FTVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FFVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ZRST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ONST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TWST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>THST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FRST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SXST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EIST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>NIST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TEST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ELST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TTST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### 15.2 Field Descriptions

**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,...,COSV  Zero Severity, One Severity,...

Alarm severity associated with each state.

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.

15.3 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.

15.4 Record Processing

Routine process implements the following algorithm:

1  Checks 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 another process will no longer be called for this record. Thus error storms will not occur.

2  ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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 Checks 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 Scans forward link if necessary, sets PACT False, and returns.

15.5 Device Support

15.5.1 Fields of interest to device support

Each input record must have an associated set of device support routines. The primary re-
sponsibility 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:

PACT, DPVT, UDF  See Chapter titled Fields Common to All Record Types for descrip-
tion.

NOBT  Number of hardware bits accessed. They must be consecutive.

VAL  This field is set by the device support routines if they don’t want record
     support to set it.

INP  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.

NSEV, NSTA See Chapter titled Fields Common to All Record Types for description.

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 This can be set by the device support module at init_record time.

15.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, padr)

Not currently used.

init()

This routine is called once during IOC initialization.

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_joint_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(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.

15.6 Device support for soft records

Two soft device support modules Soft Channel and Raw Soft Channel are provided for multi-bit 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 Soft Input in Fields Common to Many Record Types 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.
The normal use for this 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.

### 16.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>ENUM</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>NOBT</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ZRVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ONVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TWVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>THVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FRVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SXVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EIVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>NIVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TEVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ELVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TVVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TTVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FTVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FFVL</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ZRST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ONST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TWST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>THST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FRST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SXST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>EIST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>NIST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TEST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ELST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TVST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TTST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FTST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>FFST</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
16.2 Field Descriptions

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,...,COSV  Zero Severity, One Severity, ...

Alarm severity associated with each state.

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.

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.

16.3 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.

16.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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 Invalid Alarm Output Action and Simulation Mode described in Fields Common to Many Record Types 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 Checks 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 Scans forward link if necessary, sets PACT False, and returns.

16.5 Device Support

16.5.1 Fields of interest to device support

Each mbbo 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 mbbo input value whenever write_mbbo is called. The device support routines are primarily interested in the following fields:

- PACT, DPVT, UDF See Chapter titled Fields Common to All Record Types for description.
- NOBT Number of hardware bits accessed. They must be consecutive.
- OUT 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.
- NSEV,NSTA See Chapter titled Fields Common to All Record Types for description.
- 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 This can be set by the device support module at init_record time.

16.5.2 Device Support routines

Device support consists of the following routines:
report(FILE fp, paddr)

Not currently used.

init()

This routine is called once during IOC initialization.

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(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(precord)

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

0 Success.

other Error.

16.6 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 Soft Output in Fields Common to Many Record Types for details.
CHAPTER 17  Permissive

This 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.

17.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABL</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>VAL</td>
<td>USHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OVAL</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WFLG</td>
<td>USHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OFLG</td>
<td>USHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

17.2 Field Descriptions

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.

17.3 Record Support Routines

Two record support routines are provided: process, and get_value. Process sets UDF to False, triggers monitors on VAL and WFLG when they change, and scans the forward link if necessary. Get_value fills in struct valueDes so that it refers to VAL.
CHAPTER 18  Pid Control

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

\[ M(n) = KP*(E(n) + KI*SUM_i(E(i)*dT(i)) + KD*(E(n) – E(n–1))/dT(n) + Mr \]

where:

- \( M(n) \) Value of manipulated variable at nth sampling 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:

\[ \text{del}M(n) = KP*( (E(n)–E(n–1)) + E(n)*dT(n)*KI + KD*((E(n) –E(n–1))/dT(n) – (E(n–1)–E(n–2))/dT(n–1))) \]

For this record:

- \( DM \) This is \( \text{del}M(n) \)
- \( VAL \) This is the setpoint
- \( CVAL \) This is current value
- \( ERR \) \( E(n) = VAL – CVAL \)

18.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>FLOAT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>CVL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STPL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDT</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>KP</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>KI</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>KD</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOLO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIGH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
18.2 Field Descriptions

**VAL**  
Setpoint Value, in engineering units.  
This is the value that the control algorithm attempts to achieve.

**CVL**  
Controlled Value Location (an 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 (an 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 not used by record support other than 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 not used by record support other than to supply a units description string when get_units is called.

HOPR  High Operating Range
LOPR  Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

HIHI  Hihi Alarm Limit
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

These fields specify the alarm limits and severities.
HYST  Alarm Deadband
ADEL  Archive Deadband
MDEL  Monitor, i.e. value change, 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.

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  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 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 contribution repeats the proportional contribution.

CT  Clocks Ticks  Clock ticks when previous process occurred.

DT  Time difference in seconds between processing steps.

DERR    Delta Error. Change in error since last time step.
LALM    Value when last monitors for alarm were triggered
ALST    Value when last monitors for archiver were triggered
MLST    Value when last monitors for value changes were triggered.

These fields are used to implement the hysteresis factors for monitors.

18.3 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

18.4 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 Checks 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 Scans forward link if necessary, sets PACT False, and returns.
CHAPTER 19  Pulse Counter

The normal use for this record type is to record counts.

19.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GTYP</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HGV</td>
<td>SHORT</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SGL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SGV</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OSGV</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSIZ</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNTE</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CNTS</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>4.3e+9</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CMD</td>
<td>RECCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SCMD</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

19.2 Field Descriptions

**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.
<table>
<thead>
<tr>
<th><strong>SGL</strong></th>
<th>Soft Gate Location (an Input Link)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If SGL is a database link and GTYP is software, then SGV will be set to the value read from SGL.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SGV</strong></th>
<th>Soft Gate Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This can be inactive or active. This will enable and disable counting if GTYP is software.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>OSGV</strong></th>
<th>Old Soft Gate Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This is the previous value of SGV.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>CSIZ</strong></th>
<th>Counter size. 16 bit or 32 bit counter.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>CNTE</strong></th>
<th>Count Edge</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This can be Rising Edge or Falling Edge. This field forces counting on rising or falling edge of source signal.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>CNTS</strong></th>
<th>Count Source</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>HOPR</strong></th>
<th>High Operating Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LOPR</strong></td>
<td>Low Operating Range</td>
</tr>
<tr>
<td></td>
<td>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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>CMD</strong></th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>Read the current value of the counter.</td>
</tr>
<tr>
<td>Clear</td>
<td>Clear the counter. Note that the counter is also stopped. The Start command must be issued to restart the counter.</td>
</tr>
<tr>
<td>Start</td>
<td>Start counting.</td>
</tr>
<tr>
<td>Stop</td>
<td>Stop counting.</td>
</tr>
<tr>
<td>Setup</td>
<td>Setup the counter. Counting will not begin until the Start command is issued.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>SCMD</strong></th>
<th>Save Command</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This is the saved value of CMD.</td>
</tr>
</tbody>
</table>
19.3 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.

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.

g_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.

19.4 Record Processing

The routine process implements the following algorithm:

1. Checks 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 another process 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 command 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

19.5 Device Support

19.5.1 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:

- **CSIZ**: This will determine to a 16 bit or 32 bit count is to be used. With 32 bit, two counter are used.

- **CMD**: The device support routine is responsible for processing the commands READ, CLEAR, START, STOP, and SETUP.

- **GTYP,IGV**: If GTYP is internal, device support is responsible for using IGV to determine gating control.

- **CNTE**: This field is used by the device support routines to force counting on leading or falling edge of signal.

- **CNTS**: Device support must use CNTS to set count source during setup.

19.5.2 Device Support routines

Device support consists of the following routines:

- **report()**: This routine is optional. If provided, it prints a report of all device modules.

- **init()**: This routine is called once during IOC initialization.

- **init_record(precord)**: This routine is optional. If provided, it is called by the record support init_record routine.

- **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(precord)**: This routine issues commands to the output device. It returns the following values:
0  Success.

other  Error.
CHAPTER 20  Pulse Delay

The normal use for this record type is to generate output pulses.

20.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNIT</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DLY</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WIDE</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ODLY</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OWID</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTYP</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CEDG</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ECS</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ECR</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>VAL</td>
<td>SHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PFLD</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LLOW</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TTYP</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HTS</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>STL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STV</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>GATE</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>GLNK</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

20.2 Field Descriptions

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 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.

HOPR  High Operating Range
LOPR  Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

PREC  Display Precision

Precision with which to display DLY. This field is not used by record support other than 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.

20.3 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.
20.4 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 another process 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. Scans forward link if necessary, sets PACT False, and returns.

20.5 Device Support

20.5.1 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:
OUT  This field is used by the device support routines to locate its output.

WIDE  Device support must use WIDE for pulse width.

DLY  Device support must use DLY for the delay after trigger edge until beginning of pulse.

LLOW  Device support must use to determine logic low level.

UNIT  All values that refer to time measure will be in this time unit.

VAL  This field will contain a 1 if a trigger occured since the last time the record was processed if the device supports it.

PFLD  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  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  This field wil be used to set the hardware trigger source if the device supports it.

STV  This field will be used for software to trigger an output delayed pulse if the device supports it.

CEDG  This field is used by the device support routines to force clock timing on leading or falling edge of signal.

CTYP,ECS  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.

**20.5.2 Device Support routines**

Device support consists of the following routines:

**report()**

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

**init()**

This routine is called once during IOC initialization.
**init_record(precord)**

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

**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(precord)**

This routine issues commands to the output device.
CHAPTER 21  Pulse Train

The normal use for this record type is to generate an output pulse train.

21.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNIT</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PER</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DCY</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>OPER</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ODCY</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>GTYP</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HGV</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SGL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SGV</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OSGV</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>VAL</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>CTYP</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CEDG</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ECS</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ECR</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LLOW</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

21.2 Field Descriptions

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 (Seconds, milliseconds, microseconds, nanoseconds, picoseconds).

PER  Period, in UNITs

Pulse train period.

DCY  Duty Cycle, percent

Percent of time that signal is high.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPER</td>
<td>Old Period, in UNITS Value when last monitors for period were triggered.</td>
</tr>
<tr>
<td>ODCY</td>
<td>Old Duty Cycle, percent Value when last monitors for duty cycle were triggered.</td>
</tr>
<tr>
<td>GTYP</td>
<td>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.</td>
</tr>
<tr>
<td>HGV</td>
<td>Hardware Gate Value This field is device dependant.</td>
</tr>
<tr>
<td>SGL</td>
<td>Soft Gate Location (an Input Link)</td>
</tr>
<tr>
<td>SGV</td>
<td>Soft Gate Value This can be inactive (no gating) or active.</td>
</tr>
<tr>
<td>OSGV</td>
<td>Old Soft Gate Value This is the previous value of SGV.</td>
</tr>
<tr>
<td>VAL</td>
<td>Value This field is not used.</td>
</tr>
<tr>
<td>CTYP</td>
<td>Clock Type. This can be internal or external.</td>
</tr>
<tr>
<td>CEDG</td>
<td>Clock Signal Edge This can be Rising Edge or Falling Edge. This field forces counting on rising or falling edge of source signal.</td>
</tr>
</tbody>
</table>
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
LOPR Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

PREC Display Precision

Precision with which to display DLY. This field is not used by record support other than to supply a value when get_precision is called.

LLOW Low Logic Level

   Logic Low=0
   Logic Low=1

21.3 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.

21.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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. Sets UDF to False.

5. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

21.5 Device Support

21.5.1 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:

- **UNIT** This field will be used to identify the time units used for time fields.
- **OUT** This field is used by the device support routines to locate its output.
- **PER** Device support must use PER for pulse period.
- **DCY** Device support must use DCY for the percent of time the signal is high.
- **LLOW** Device support must use to determine logic low level.
- **CEDG** This field is used by the device support routines to force counting on leading or falling edge of signal.
GTYP,IGV
SGV       Device support is responsible for using IGV to determine gating control if
          GTYP is internal, or SGV if GTYP is external.

CTYP,ECS
ECR       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.

21.5.2 Device Support routines

Device support consists of the following routines:

report()

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

init()

This routine is called once during IOC initialization.

init_record(precord)

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

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
form 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 scan-
ner.

write_pt(precord)

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

0           Success.

other        Error.
CHAPTER 22  Select

This 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.

22.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SELM</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SELN</td>
<td>USHORT</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>NVL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPA</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPB</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPC</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPD</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPE</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPF</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPG</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPH</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPI</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPJ</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPK</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>B</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>C</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>D</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>E</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>G</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>H</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>I</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>J</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>K</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>L</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LA</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LB</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LC</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LD</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LE</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LF</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LG</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LH</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
22.2 Field Descriptions

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 not used by record support other than 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.

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 not used by record support other than to supply a units description string when get_units is called.

HOPR High Operating Range
LOPR Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

HIHI Hihi Alarm Limit
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

These fields specify the alarm limits and severities.

HYST Alarm Deadband
ADEL Archive Deadband
MDEL Monitor, i.e. value change, 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.

LALM Last Alarmed. Value when last monitors for alarm were triggered
ALST Archive Last. Value when last monitors for archiver were triggered
MLST Monitor Last. Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitors.

22.3 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. 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:

\[
\text{upper\_alarm\_limit} = \text{hihi} \\
\text{upper\_warning\_limit} = \text{high}
\]
lower_warning_limit = low
lower_alarm_limit = lolo

22.4 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. Implements 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.
CHAPTER 23 Sequence

This 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.

23.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SELM</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SELN</td>
<td>USHORT</td>
<td>No</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SELL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DLY1</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DO1</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK1</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY2</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DO2</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK2</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY3</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL3</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO3</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK3</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY4</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL4</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO4</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK4</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY5</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL5</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO5</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK5</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY6</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL6</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO6</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK6</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY7</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL7</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO7</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK7</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLY8</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL8</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DO8</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK8</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 23.2 Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Required</th>
<th>Width</th>
<th>Readable</th>
<th>Value</th>
<th>Update</th>
<th>Foreground</th>
<th>Background</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>Value</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DLY9</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOL9</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DO9</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNK9</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DLYA</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOLA</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DOA</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LNKA</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

#### 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 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 not used by record support other than 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.

23.3 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 PRI0 field in dbCommon (see the Application Developer’s Guide for more information on callback tasks.)
CHAPTER 24  State

This record is used to store an arbitrary ASCII string.

24.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OVAL</td>
<td>STRING</td>
<td>No</td>
<td>3</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

24.2 Field Descriptions

VAL  Value Field

An arbitrary string value

OVAL  Old Value

Used to decide when to invoke monitors.

24.3 Record Support Routines

Two record support routines are provided: process, and get_value. Process triggers monitors on VAL when it changes and scans the forward link if necessary. Get_value fills in struct valueDes so that it refers to VAL.
CHAPTER 25  Stepper Motor

This record type is used to control stepper motors. This record may undergo revision.

### 25.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>FLOAT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RDBL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ACCL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>VELO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DIST</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVAL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MODE</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CMOD</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IALG</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MRES</td>
<td>USHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ERES</td>
<td>USHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DRVH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DRVL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HIHI</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOLO</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HIGH</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LOW</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HHSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>HLSV</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ADEL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MDEL</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RDBD</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RTRY</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>STHM</td>
<td>SHORT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>STOP</td>
<td>SHORT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DMOV</td>
<td>SHORT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RVAL</td>
<td>LONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RBV</td>
<td>FLOAT</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>RRBV</td>
<td>LONG</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
25.2 Field Descriptions

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 (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.

ACCL  Acceleration.

Number of seconds to reach VELO velocity.

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 not used by record support other than to supply a value when get_precision is called.

EGU  Engineering Units

ASCII string describing Engineering units. This field is not used by re-
cord support other than to supply a units description string when get_units is called.

**DRVH**  Drive High  
**DRVL**  Drive Low  

If these values are defined, then VAL will be forced to be in the range \( DRVL \leq VAL \leq DRVH \)

**HOPR**  High Operating Range  
**LOPR**  Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

**HIHI**  Hihi Alarm Limit  
**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  
**HLSV**  Severity for a hardware limit violation  

These fields specify the alarm limits and severities.

**ADEL**  Archive Deadband  
**MDEL**  Monitor, i.e. value change, Deadband  

These parameters specify hysteresis factors for triggering monitor call-backs, 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.

**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.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STOP</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>DMOV</td>
<td>Done Moving to Value</td>
</tr>
<tr>
<td>RVAL</td>
<td>Raw Data Value</td>
</tr>
<tr>
<td></td>
<td>RVAL is the value actually sent to the device.</td>
</tr>
<tr>
<td>RBV</td>
<td>Read Back Value</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>RRBV</td>
<td>Raw Read Back Value.</td>
</tr>
<tr>
<td></td>
<td>Raw read back value obtained from the encoder.</td>
</tr>
<tr>
<td>ALST</td>
<td>Archive Last. Value when last monitors for archiver were triggered</td>
</tr>
<tr>
<td>MLST</td>
<td>Monitor Last. Value when last monitors for value changes were triggered</td>
</tr>
<tr>
<td></td>
<td>These fields are used to implement the hysteresis factors for monitors.</td>
</tr>
<tr>
<td>INIT</td>
<td>Initialize.</td>
</tr>
<tr>
<td>MCW</td>
<td>Motor clockwise limit switch value.</td>
</tr>
<tr>
<td>MCCW</td>
<td>Motor counter clockwise limit switch value.</td>
</tr>
<tr>
<td>CW</td>
<td>Is motor clockwise limit switch True?</td>
</tr>
<tr>
<td>CCW</td>
<td>Is motor counter clockwise limit switch True?</td>
</tr>
<tr>
<td>DIR</td>
<td>Current direction.</td>
</tr>
<tr>
<td>MOVN</td>
<td>Is motor moving?</td>
</tr>
<tr>
<td>CVEL</td>
<td>Has Constant velocity been attained?</td>
</tr>
<tr>
<td>RCNT</td>
<td>Current retry count.</td>
</tr>
</tbody>
</table>
LVAL  Last value.

POSM  Positive motion.

EPOS  Encoder read back position

MPOS  Motor position

MISS  First attempt error

LVEL  Last velocity set

LACC  Last acceleration set

DPVT  Device private

25.3 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

25.4 Record Processing

Not yet written

25.5 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 26 Stringin – String Input

This record is used to input an arbitrary ASCII string.

26.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OVAL</td>
<td>STRING</td>
<td>No</td>
<td>3</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVAL</td>
<td>STRING</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

26.2 Field Descriptions

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.

26.3 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.

26.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process will no longer be called for this record. Thus error storms will not occur.

2. ReadValue is called. See Simulation Mode described in Fields Common to Many Record Types 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

26.5 Device Support

26.5.1 Fields of interest to device support

Each string in 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:

- PACT, DPVT, UDF  See Chapter titled Fields Common to All Record Types for description.
- VAL This field is set by the device support routines.
- INP This field is used by the device support routines to locate its input.

26.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, paddr)

Not currently used.
init()

This routine is called once during IOC initialization.

init_record(precord)

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

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(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.

26.6 Device support for soft records

This module places a value directly in VAL.

If the INP link type is constant, the double constant, if nonzero, 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 Soft Input in Fields Common to Many Record Types for details.

If the return status of recGblGetLinkValue is zero, then read_stringin sets UDF to False. The status of recGblGetLinkValue is returned.
CHAPTER 27  Stringout – String Output

This record is used to output an arbitrary ASCII string.

27.1  Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>STRING</td>
<td>Yes</td>
<td>0</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OVAL</td>
<td>STRING</td>
<td>No</td>
<td>3</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OMSL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOA</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>IVOV</td>
<td>DOUBLE</td>
<td>Yes</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

27.2  Field Descriptions

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 (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.

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.

27.3 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 nonzero, 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.

27.4 Record Processing

Routine process implements the following algorithm:

1  Checks 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 another process 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 Soft Input in Fields Common to Many Record Types 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 Invalid Alarm Output Action and Simulation Mode described in Fields Common to Many Record Types 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  Checks 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. Scans forward link if necessary, sets PACT False, and returns.

27.5 Device Support

27.5.1 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:

PACT, DPVT See Chapter titled Fields Common to All Record Types for description.

VAL This is the field written by the device support routines.

OUT This field is used by the device support routines to locate its output.

NSEV, NSTA See Chapter titled Fields Common to All Record Types for description.

27.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, padr)

Not currently used.

init()

This routine is called once during IOC initialization.

init_record(precord)

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

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 form 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(precord)

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

0 Success.
other Error.

27.6 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 Soft Output in Fields Common to Many Record Types for details.
CHAPTER 28 Subroutine

This record provides a subroutine escape mechanism.

### 28.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>INAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNAM</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SADR</td>
<td>NOACCESS</td>
<td>NO</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STYP</td>
<td>SHORT</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>INPA</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPB</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPC</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPD</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPE</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPF</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPG</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPH</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPI</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPJ</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPK</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>B</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>C</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>D</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>E</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>F</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>G</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>H</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>I</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>J</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>K</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>L</td>
<td>DOUBLE</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LA</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LB</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LC</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LD</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LE</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LF</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LG</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LH</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LI</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>LJ</td>
<td>DOUBLE</td>
<td>No</td>
<td>3</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
28.2 Field Descriptions

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 not used by record support other than to supply a value when get_precision is called.

EGU  Engineering Units

ASCII string describing Engineering units. This field is not used by record support other than to supply a units description string when get_units is called.

HOPR  High Operating Range
LOPR  Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

HIHI  Hihi Alarm Limit
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

These fields specify the alarm limits and severities.

HYST  Alarm Deadband
ADEL  Archive Deadband
MDEL  Monitor, i.e. value change, 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.

LALM  Value when last monitors for alarm were triggered
ALST  Value when last monitors for archiver were triggered
MLST  Value when last monitors for value changes were triggered

These fields are used to implement the hysteresis factors for monitors.

28.3 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

28.4 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

28.5 Example Synchronous Subroutine

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

```c
#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+=;
```
28.6 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. The routine callbackRequest is described in Part 1 of this manual.
   d Returns a value of 1. 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.

```c
#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);
}
```
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);
            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 29  Timer

The functions of this record have been replaced by the pulseCounter, pulseDelay, pulseTrain, and Event records. The Timer record type is included for upward compatibility.

This record type interacts with timer modules.

29.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TORG</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT</td>
<td>OUTLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAL</td>
<td>SHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TSRC</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PTST</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TEVT</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TIMU</td>
<td>RECCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MAIN</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RDT1</td>
<td>FLOAT</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RDW1</td>
<td>FLOAT</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PDLY</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DUT1</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OPW1</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DUT2</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OPW2</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DUT3</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OPW3</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DUT4</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OPW4</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DUT5</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>OPW5</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>T1DL</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T1WD</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>T2DL</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T2WD</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T3DL</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T3WD</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T4DL</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T4WD</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T5DL</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T5WD</td>
<td>DOUBLE</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T1TD</td>
<td>FLOAT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T1LD</td>
<td>FLOAT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T2TD</td>
<td>FLOAT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T2LD</td>
<td>FLOAT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>T3TD</td>
<td>FLOAT</td>
<td>No</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
29.2 Field Descriptions

TORG  Trigger Delay Origin, an 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 only 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

RDW1  Reboot width of 1.

DUTi  Delay width for trigger i, in timer units.

OPWi  Output pulse width for trigger i, in timer units.

TiDL  Delay width for trigger i, in seconds.

TiWD  Pulse width of trigger i, in seconds.
TiTD  Trailing delay of trigger i. ( TiLD+OPWi )

TiLD  Leading delay of trigger i. ( DUTi+TRDL )

TRDL  Trigger delay, obtained from trigger delay origin TORG.

TDIS  Timing pulse disable

### 29.3 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.

### 29.4 Record Processing

This section not yet written.

### 29.5 Device Support

Currently device support is intimately combined with record support.
CHAPTER 30  Waveform

This record type stores arrays is data. The array can contain any of the supported data types.

30.1 Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>DCT</th>
<th>Interest</th>
<th>Initial</th>
<th>Access</th>
<th>Modify</th>
<th>Monitor</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>See FTVL</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RARM</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PREC</td>
<td>SHORT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>INP</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EGU</td>
<td>STRING</td>
<td>Yes</td>
<td>1</td>
<td>null</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>HOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>LOPR</td>
<td>FLOAT</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>NELM</td>
<td>ULONG</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>FTVL</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>BPTR</td>
<td>NOACCESS</td>
<td>No</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NORD</td>
<td>ULONG</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>BUSY</td>
<td>SHORT</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SIOL</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIML</td>
<td>INLINK</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIMM</td>
<td>GBLCHOICE</td>
<td>No</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>SIMS</td>
<td>GBLCHOICE</td>
<td>Yes</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

30.2 Field Descriptions

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 not used by record support other than 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 not used by record support other than to supply a units description string when get_units is called.

HOPR  High Operating Range  
LOPR  Low 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 not used by the record support routines themselves other than to honor calls to get_graphic_double or get_control_double.

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?  

### 30.3 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:

\[
\begin{align*}
\text{upper Disp Limit} &= \text{hopr} \\
\text{lower Disp Limit} &= \text{lopr}
\end{align*}
\]

get_control_double Sets the following values:

\[
\begin{align*}
\text{upper Ctrl Limit} &= \text{hopr} \\
\text{lower Ctrl Limit} &= \text{lopr}
\end{align*}
\]

### 30.4 Record Processing

Routine process implements the following algorithm:

1. Checks 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 another process 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. Checks 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. Scans forward link if necessary, sets PACT False, and returns.

### 30.5 Device Support

#### 30.5.1 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:
PACT, DPVT  See Chapter titled Fields Common to All Record Types for description.

INP  This field is used by the device support routines to locate its input.

NSEV,NSTA  See Chapter titled Fields Common to All Record Types for description.

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.

NORD  Number of elements read. Device support must set this value when it completes.

BUSY  Is device busy?

30.5.2 Device Support routines

Device support consists of the following routines:

report(FILE fp, paddr)

Not currently used.

init()

This routine is called once during IOC initialization.

init_record(precord)

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

get_ioint_info(int cmd,struct dbCommon *prevcord,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(precord)

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

0      Success.
other  Error.

30.6 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.

30.7 Device support for soft records

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.