Created:June 25, 1996
Modified:February 3, 1997

Gemini Controls Group Report

Gemini Record Reference Manual

Bret Goodrich and Andy Foster

SPE-C-G0070/02

This document decribes the EPICS records created by the Gemini Project.

Table Of Contents

1.0 Introduction

1.1 Purpose

This document describes the EPICS records created by the Gemini 8M Telescopes Project for use in its telescope and instrument control databases.

1.2 Scope

This document defines the interface to only those records created by and for the Gemini Project. For a complete list of the standard EPICS records, and a description of the field summary tables, refer to the *EPICS Input Output Controller Record Reference Manual* [1].

1.3 References

1. *EPICS Input Output Controller Record Reference Manual*, Janet B. Anderson and Martin R. Kraimer, Argonne National Laboratory, Dec 1, 1994.
2. *ICD 1b — The Baseline Attribute/Value Interface* (gscg.grp.024/04), Kim Gillies, Steve Wampler, Bret Goodrich.
3. *EPICS Lookup Table Records* (gscg.bdg.003.lut/1), Bret Goodrich.
4. *The ‘mosub’ EPICS Record Reference Manual*, Andy Foster.
5. *The ‘genSub’ EPICS Record Reference Manual*, Andy Foster.

1.4 Revisions

1. Version 01, 8 November, 1996. Document created.
2. Version 02, February 3, 1997, Updated genSub section.

2.0 apply - Apply Record

The apply record executes data links to other records. Its primary purpose is to process CAD records in a fixed order, and return the results of processing those records. There may be up to eight sets of links to other records. The links in each set pass the directive field (DIR to OUTA through OUTH) and client ID field (CLID to OCLA through OCLH), and receive the result value (INPA through INPH to VAL) and error message (INMA through INMH to MESS).

The apply record accepts the same directives as the CAD, namely: MARK, CLEAR, PRESET, START, and STOP. Writing a value to the directive field (DIR) starts the processing of the record and subsequent processing of all attached records. Writing the START directive forces the PRESET directive to be sent to all links before the START directive is sent. This insures that all CAD records linked to the apply record have valid arguments.

Values returned through the INPx links are inspected by the apply record for non-zero results. If any link returns a non-zero value, the associated INMx link is read. The error value and error message are copied to the VAL and MESS fields, monitors are posted, and the processing halts. No further links are processed once an error has been returned.

The use of the apply record is required for all principal systems databases, including the TCS, CICS, and all instruments. There must be one, and only one, top level apply record in the database, although there may be cascaded apply records. All principal systems CAD records must be linked to the apply record and must be processed through this record. Links from the apply record outputs may go to records other than CAD records, such as calc, sub, or mosub records.

2.1 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	No	Yes	No
DIR	RECCHOICE	Yes	0	Yes	Yes	No	Yes
CLID	LONG	No	0	Yes	Yes	Yes	No
MESS	STRING	No	Null	Yes	Yes	Yes	No
OMSS	STRING	No	Null	Yes	No	No	No
OUTx	OUTLINK	No	0	No	No	No	No
OCLx	OUTLINK	No	0	No	No	No	No
INPx	INLINK	No	0	No	No	No	No
INMx	INLINK	No	0	No	No	No	No

2.2 Field Descriptions

Name	Summary	Description
VAL	Value	This is the return value from the input links. If any link returns a non-zero, processing stops and the last value is returned.
DIR	Directive	This value of this field is passed to all OUTx output links. If the directive is START, the directive PRESET is first passed to all output links.
CLID	Client ID	This number is incremented every time a directive is loaded. The value is passed to all OCLx output links.
MESS	Message	This is the return message from an INMx input link. If the return value is 0, this field is empty. Otherwise, it reads the error message from the INMx link.
OMSS	Old Message	This is the old message string.
OUTx	Output directive link	There are eight output links OUTA-OUTH which pass the value of the DIR field to a record field.
OCLx	Output client ID link	There are eight output links OUMA-OUMH which pass the value of the CLID field to a record field.
INPx	Input result link	There are eight input links INPA-INPH which read a value from a record field. A non-zero value halts the processing sequence.
INMx	Input message link	There are eight input links INMA-INMH which read a value from a record field. The link is read only for the corresponding INPx link which returned a non-zero value.

2.3 Record Support Routines

2.3.1 init_record

This routine initializes the apply record. All OUTx links are forced to be process passive; all OUMx, INPx, and INMx links are forced to be non-process passive.

2.3.2 process

See the next section.

2.3.3 get_value

This routine fills the values of `struct valueDes` so that they refer to VAL.

2.3.4 get_enum_str

This routine converts the long integer values 0 through 4 into the strings “MARK”, “CLEAR”, PRESET”, “START”, and “STOP”, respectively.

2.3.5 get_enum_strs

This routine returns all five of the above strings.

2.3.6 put_enum_str

This routine converts the above strings into the long integer values 0 through 4.

2.4 Record Processing

This routine processes the record whenever requested. Processing will occur whenever a value is written to the DIR field.

- All MARK directives are ignored and processing exits.
- The return message field is cleared.
- If the directive is START:
  - —increment the client ID,
  - —recursively call this procedure with PRESET,
  - —exit if an error occurred during PRESET.
- for each existing set of links A-H:
  - —send CLID and DIR to OCLx and OUTx links,
  - —get VAL from INPx link,
  - —if VAL is non-zero, get MESS from OUMx link and stop looping
- post monitor on VAL field.
- if VAL is non-zero and MESS is different than OMSS:
  - —post monitor on MESS field.

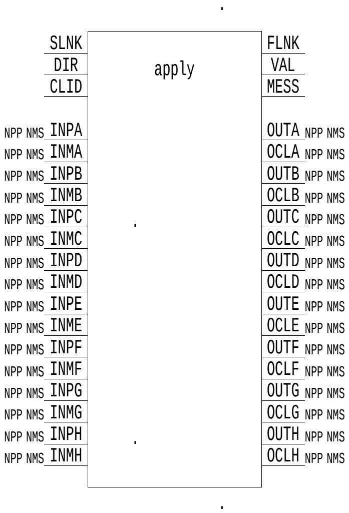
2.5 Device Support

There is no device support available.

2.6 CapFast

There is one CapFast symbol for the Apply record.

FIGURE 1. CapFast *eapply* symbol



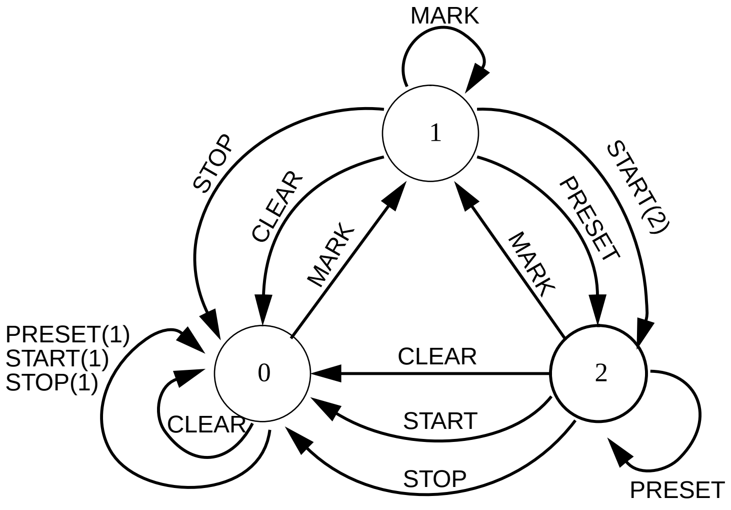
3.0 CAD - Command Action Directive Record

The CAD record initiates processing of Gemini commands. Attributes of the command are either loaded into fields A through T, or read from input links INPA through INPT. Processing begins when a directive is received in the DIR field. Subroutine calls may be made during initialization and processing. Each directive also has an associated link which is processed when the directive is received. A return value from the processing subroutine is returned in the VAL field.

There are five valid directives: MARK, CLEAR, PRESET, START, and STOP. The MARK directive forces the CAD state machine into state 1. This directive is also executed through special processing if any of fields A through T are modified. The CLEAR directive forces the state machine into state 0, clearing any prior mark or preset. The PRESET directive moves the state machine from state 1 to state 2. The START directive either moves the state machine from state 2 to state 0, or executes a PRESET directive then moves from state 2 to state 0. In all cases except PRESET, START, and STOP in state 0 the processing subroutine SNAM is called and the corresponding directive link is processed.

The CAD record can be in one of three states, given by the value of the MARK field. In state 0, the CAD record is considered to be cleared. In state 1, it is considered to be marked as ready to preset. In state 2, it is considered to be preset and ready to activate. Figure 1 shows the state diagram for the CAD record, with the edges identified as possible directive commands.

FIGURE 2. CAD Record state transition diagram



(1) No call to SNAM, no links processed.
(2) PRESET call and link processed first.

3.1 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	No	Yes	No
SNAM	STRING	Yes	Null	Yes	No	No	No
SADR	LONG	No	0	No	No	No	No
STYP	SHORT	No	0	Yes	No	No	No
INAM	STRING	Yes	Null	Yes	No	No	No
DIR	RECCHOICE	Yes	1	Yes	Yes	No	Yes
ICID	LONG	No	0	Yes	Yes	No	No
MESS	STRING	No	Null	Yes	Yes	Yes	No
OMSS	STRING	No	Null	Yes	Yes	No	No
CTYP	SHORT	Yes	2	Yes	No	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
MLNK	FWDLINK	Yes	0	No	No	No	No
CLNK	FWDLINK	Yes	0	No	No	No	No
PLNK	FWDLINK	Yes	0	No	No	No	No
STLK	FWDLINK	Yes	0	No	No	No	No
SPLK	FWDLINK	Yes	0	No	No	No	No
OCID	LONG	No	0	Yes	Yes	Yes	No
OSIM	RECCHOICE	No	None	Yes	Yes	Yes	No
NARG	SHORT	Yes	0	Yes	Yes	No	No
MARK	SHORT	No	0	Yes	Yes	Yes	No
ERSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SIOL	INLINK	Yes	0	No	No	No	No
SVAL	LONG	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	No	No
SIMM	RECCHOICE	No	None	Yes	Yes	No	No
SIMS	GBLCHOICE	No	0	Yes	Yes	No	No
INPx	INLINK	Yes	0	No	No	No	No
OUTx	OUTLINK	Yes	0	No	No	No	No
A - T	STRING	No	0	Yes	Yes	No	Spec
VALx	NOACCESS	Yes	0	Yes	Yes	No	No
FTVx	GBLCHOICE	No	String	Yes	No	No	No

3.2 Field Descriptions

Name	Summary	Description
VAL	Return Error Code	The return value is set within the user-supplied processing subroutine. Conventionally, a return value of zero indicates success, while a non-zero value shows an error has occurred.
SNAM	Subroutine Name	The name of a VxWorks subroutine to execute during processing.
SADR	Subroutine Address	The internal representation of the subroutine address.
STYP	Subroutine symbol type	Not used.
INAM	Init Routine Name	The name of a VxWorks subroutine to execute during initialization.
DIR	CAD Directive	The directive to execute. This may be one of the following enumerated list values: MARK, CLEAR, PRESET, START, or STOP.
ICID	Client ID (In)	An integer value to be associated with the current command.
MESS	Message	A return message from the CAD. This string will be empty if the return value is zero.
OMSS	Old Message	The previous message.
CTYP	Number of CAD Args	This value should be set to the maximum number of arguments a CAD record will use. The value is usually set by the selected CapFast symbol.
PREC	Display Precision	This value is used for the precision of double-precision outputs.
MLNK	Mark Link	If the directive is MARK, this forward link is processed.
CLNK	Clear Link	If the directive is CLEAR, this forward link is processed.
PLNK	Preset Link	If the directive is PRESET, this forward link is processed.
STLK	Start Link	If the directive is START, this forward link is processed.
SPLK	Stop Link	If the directive is STOP, this forward link is processed.
OCID	Client ID (Out)	The input client ID is sent out this field.
OSIM	Simulation Mode (Out)	The simulation mode is sent out this field.
NARG	No. Inputs used	The actual number of arguments used is set in this field.
MARK	Is Record Preset?	This field shows the current state of the CAD. It can be zero, indicating no MARK has been done; one, showing a MARK; or two, showing a PRESET has been done.
ERSV	Error Alarm Severity	The severity of an alarm.
SIOL	Simulation Error Link	Simulation mode variables. Refer to reference [1], chapter 3.
SVAL	Simulation Error	

Name	Summary	Description
SIML	Simulation Mode Link	
SIMM	Simulation Mode	
SIMS	Simulation Mode Alarm Severity	The value for the simulated alarm.
INPx	Input Link A-T	The 20 input links for the arguments to the record.
OUTx	Output Link A-T	The 20 outputs are sent across these links.
A - T	Value of Input A-T	The 20 string input arguments.
VALx	Value of Output A-T	The 20 output values.
FTVx	Type of Value A-T	The types for the 20 output values. The type may be one of STRING, LONG, or DOUBLE.

3.3 Record Support Routines

3.3.1 init_record

During the first initialization pass, the output types are determined from the FTVx fields and appropriate space is created for the VALx fields. During the second pass, input and output links are initialized, the initialization routine is called, and the processing subroutine is readied. The directive field is set to CLEAR, and the mark field is set to zero.

3.3.2 process

See the next section.

3.3.3 special

The special processing is called when a value is put to one of the twenty fields A through T. The mark flag is set to one.

3.3.4 get_value

This routine fills the values of `struct valueDes` so that they refer to VAL.

3.3.5 get_precision

This routine retrieves PREC.

3.3.6 get_enum_str

This routine converts the long integer values 0 through 4 into the strings “MARK”, “CLEAR”, PRESET”, “START”, and “STOP”, respectively.

3.3.7 get_enum_strs

This routine returns all five of the above strings.

3.3.8 put_enum_str

This routine converts the above strings into the long integer values 0 through 4.

3.4 Record Processing

Record processing is very dependent upon the directive given to process and the value of the mark field in the state machine. The algorithm is:

- If PRESET, START, or STOP with MARK==0, then return
- If simulation, process simulation links.
- Process all input links.
- Process requested directive (see next sections for details).
- Enforce rule that MESS is empty if VAL is 0.
- Put the values on the output links.
- Raise monitors on fields VAL, MESS, OSIM, OCID, MARK.
- Process directive link (MLNK, CLNK, PLNK, STLK, SPLK).
- Process forward link.

3.4.1 Mark Directive Processing

- Call user subroutine, return value in VAL.
- Copy ICID to OCID.
- Set mark field to 1.

3.4.2 Clear Directive Processing

- Call user subroutine, return value in VAL.
- Copy ICID to OCID.
- Set mark field to 0.

3.4.3 Preset Directive Processing

- Call user subroutine, return value in VAL.
- Copy ICID to OCID.
- Set mark field to 2.

3.4.4 Start Directive Processing

- If mark field is 1:
  - set directive to PRESET.
  - Call user subroutine, return value in VAL.
  - Copy ICID to OCID.
  - Process PLNK link.
  - Set mark field to 2.
  - Put the values on the output links.
  - Raise monitors.

–set directive to START.

- Call user subroutine, return value in VAL.
- Copy ICID to OCID.
- Set mark field to 0.

3.4.5 Stop Directive Processing

- If mark field is not 0:
  - –Call user subroutine, return value in VAL.
  - –Copy ICID to OCID.
  - –Set mark field to 0.

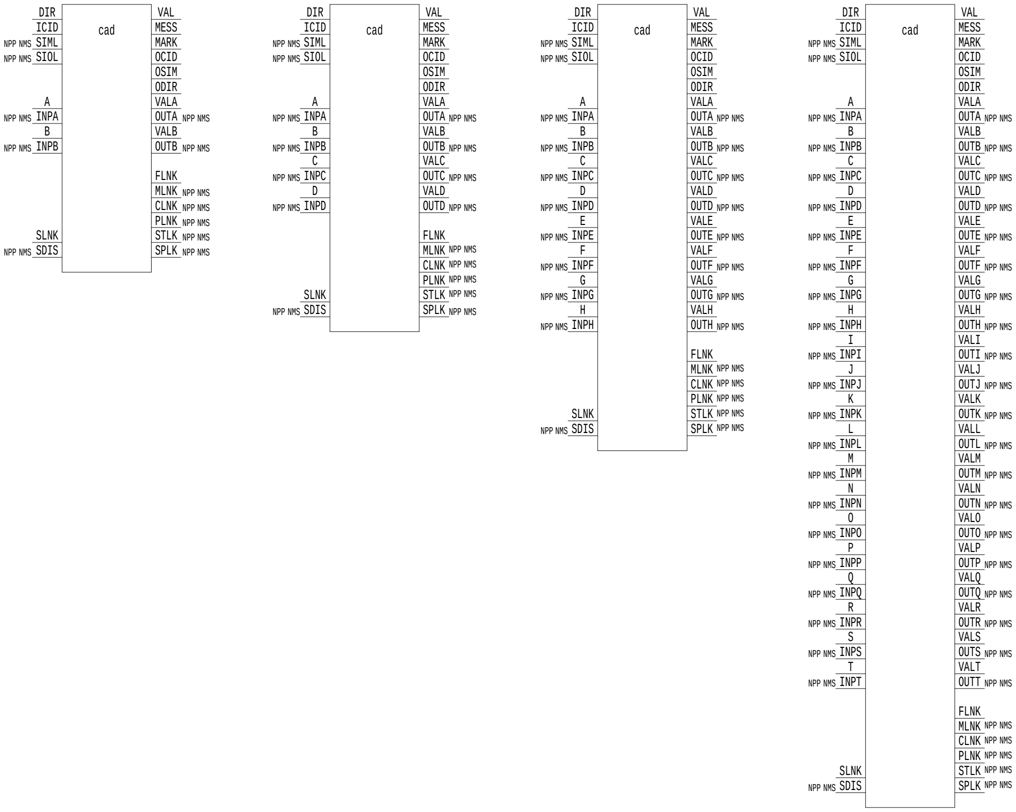
3.5 Device Support

There is no device support available.

3.6 CapFast

There are four CapFast symbols for the CAD record.

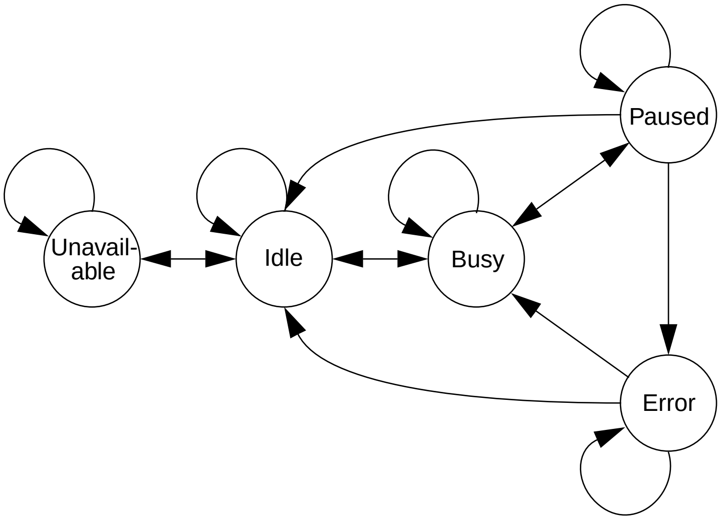
FIGURE 3. CapFast *ecad2, ecad4, ecad8, and ecad20* symbols


cad
DIR ICID SIML SIOL A INPA B INPB SLNK SDIS
VAL MESS MARK OCID OSIM ODIR VALA OUTA VALB OUTB FLNK MLNK CLNK PLNK STLK SPLK
NPP NMS
cad
DIR ICID SIML SIOL A INPA B INPB C INPC D INPD SLNK SDIS
VAL MESS MARK OCID OSIM ODIR VALA OUTA VALB OUTB VALC OUTC VALD OUTD FLNK MLNK CLNK PLNK STLK SPLK
cad
DIR ICID SIML SIOL A INPA B INPB C INPC D INPD E INPE F INPF G INPG H INPH SLNK SDIS
VAL MESS MARK OCID OSIM ODIR VALA OUTA VALB OUTB VALC OUTC VALD OUTD VALE OUTE VALF OUTF VALG OUTG VALH OUTH FLNK MLNK CLNK PLNK STLK SPLK
cad
DIR ICID SIML SIOL A INPA B INPB C INPC D INPD E INPE F INPF G INPG H INPH I INPI J INPJ K INPK L INPL M INPM N INPN O INPO P INPP Q INPQ R INPR S INPS T INPT SLNK SDIS
VAL MESS MARK OCID OSIM ODIR VALA OUTA VALB OUTB VALC OUTC VALD OUTD VALE OUTE VALF OUTF VALG OUTG VALH OUTH VALI OUTI VALJ OUTJ VALK OUTK VALL OUTL VALM OUTM VALN OUTN VALO OUTO VALP OUTP VALQ OUTQ VALR OUTR VALS OUTS VALT OUTT FLNK MLNK CLNK PLNK STLK SPLK

4.0 CAR - Command Action Response Record

The CAR record provides information about the state of a particular action occurring within a database. Possible states of the CAR record are UNAVAILABLE, IDLE, BUSY, PAUSED, and ERR. The allowable transitions between the states are illustrated in the figure below.

FIGURE 4. CAR transitions



4.1 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	RECCHOICE	Yes	Idle	Yes	No	Yes	No
CLID	LONG	No	0	Yes	No	Yes	No
OMSS	STRING	No	Null	Yes	Yes	Yes	No
OERR	LONG	No	0	Yes	Yes	Yes	No
IVAL	LONG	No	0	Yes	Yes	No	Yes
ICID	INLINK	Yes	0	No	No	No	No
IMSS	STRING	No	Null	Yes	Yes	No	No
IERR	LONG	No	0	Yes	Yes	No	No
AVAL	LONG	No	0	Yes	Yes	No	No
MVAL	LONG	No	0	Yes	Yes	No	No
ACID	LONG	No	0	Yes	Yes	No	No
MCID	LONG	No	0	Yes	Yes	No	No
AMSS	STRING	No	Null	Yes	Yes	No	No
MMSS	STRING	No	Null	Yes	Yes	No	No
AERR	LONG	No	0	Yes	Yes	No	No
MERR	LONG	No	0	Yes	Yes	No	No
ERSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
SIOL	INLINK	Yes	0	No	No	No	No
SVAL	LONG	No	0	Yes	Yes	No	No
SIML	INLINK	Yes	0	No	No	No	No
SIMM	RECCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

4.2 Field Descriptions

Name	Summary	Description
VAL	State	Current state
CLID	Client ID	Value of the latest client ID
OMSS	Message (out)	Output message
OERR	Error Code (out)	Output error code
IVAL	State (in)	Input state transition
ICID	Client ID (in)	Link to client ID
IMSS	Message (in)	Input message
IERR	Error Code (in)	Input error code
AVAL	Last State Archived	The last state value which was archived

Name	Summary	Description
MVAL	Last State Monitored	The last state value which generated a monitor event
ACID	Last Client ID Archived	The last Client ID which generated an archive event
MCID	Last Client ID Monitored	The last Client ID which generated a monitor event
AMSS	Last Message Archived	The last message which generated an archive event
MMSS	Last Message Monitored	The last message which generated a monitor event
AERR	Last Error Code Archived	The last error code which generated an archive event
MERR	Last Error Code Monitored	The last error code which generated a monitor event
ERSV	Error Alarm Severity	The severity code of the error alarm
SIOL	Simulation Error Link	Simulation mode variables. Refer to reference [1], chapter 3.
SVAL	Simulation Error	
SIML	Simulation Mode Link	
SIMM	Simulation Mode	
SIMS	Simulation Mode Alarm Severity	

4.3 Record Support Routines

4.3.1 init_record

This routine sets the current state to IDLE and clears the input message and error code. Any simulation modes or links are also initialized.

4.3.2 process

See the next section.

4.3.3 get_value

This routine fills the values of `struct valueDes` so that they refer to VAL.

4.3.4 get_enum_str

This routine converts the long integer values 0 through 5 into the strings “UNAVAILABLE”, “IDLE”, “PAUSED”, “ERR”, “BUSY”, and “UNKNOWN” respectively.

4.3.5 get_enum_strs

This routine returns all of the above strings.

4.3.6 put_enum_str

This routine converts the state strings into values 0 through 5.

4.4 Record Processing

Record processing begins when a value is written into the IVAL field. The client ID is read from the ICID link and put into CLID. The input value is used to transition the CAR state machine and the input message and input error code are placed into the output message and output error code fields. If the state machine transitions to the ERROR state an alarm is generated.

Monitors are posted if the output value is different than the last output value or the client ID is different than the old client ID. Monitors are posted on the VAL, CLID, OMSS, and OERR fields.

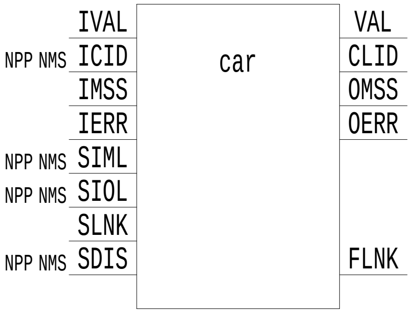
4.5 Device Support

There is no device support available.

4.6 CapFast

There is one CapFast symbol for the CAR record.

FIGURE 5. CapFast *ecars* symbol



5.0 SIR - Status Information Record

The Status Information Record provides a standard information-passing mechanism between Gemini principle systems. The SIR records for any Gemini system are expected to be combined into a separate database called the Status Alarm Database (SAD) and loaded into the SAD IOC.

SIR records combine three important parts of Gemini EPICS status reporting: a status value, status message, and alarms. The status value is read through the INP link, the status message is placed into the IMSS link. Alarms are propagated to the SIR record by the input link.

There are three valid output data types: long integer, double precision, and string. A subroutine supplied by the user can be attached to the SNAM field and called during record processing. This subroutine can convert the input, raise additional alarms, or enhance the output message string.

A full description of the SIR record's function is expected in the FDSC field. This field can hold a 40 character string which is used by the OCS when displaying the SIR value.

5.1 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
INP	INLINK	Yes	0	No	No	No	No
IMSS	STRING	No	Null	Yes	Yes	No	No
FDSC	STRING	Yes	Null	Yes	Yes	No	No
FTVL	GBLCHOICE	Yes	0	Yes	No	No	No
EGU	STRING	Yes	units	Yes	Yes	No	No
VAL	VOID *	No	Null	Yes	Yes	Yes	Yes
OMSS	STRING	No	Null	Yes	Yes	Yes	No
SNAM	STRING	Yes	Null	Yes	No	No	No
SADR	LONG	No	0	No	No	No	No
STYP	SHORT	No	0	Yes	No	No	No
PREC	SHORT	Yes	0	Yes	Yes	No	No
AVAL	VOID *	No	0	Yes	Yes	No	No
MVAL	VOID *	No	0	Yes	Yes	No	No
AMSS	STRING	No	Null	Yes	Yes	No	No
MMSS	STRING	No	Null	Yes	Yes	No	No
LALM	DOUBLE	No	0	XXX	No	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
BRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	XXX	Yes	No	No
ADEL	DOUBLE	Yes	0	XXX	Yes	No	No
MDEL	DOUBLE	Yes	0	XXX	Yes	No	No
SIOL	INLINK	Yes	0	No	No	No	No
SVAL	STRING	No	Null	Yes	Yes	No	Yes
SIML	INLINK	Yes	0	No	No	No	No
SIMM	RECCHOICE	No	0	Yes	Yes	No	No
SIMS	GBLCHOICE	Yes	0	Yes	Yes	No	No

5.2 Field Descriptions

Name	Summary	Description
INP	Input Link	A CA link to the value to be reported. This value must match the type declared in FTVL.
IMSS	Message In	The input message.
FDSC	Full Description	A 40 character description of the SIR record
FTVL	Type of value	The data type of the input link. May be either LONG, DOUBLE, or STRING.
EGU	Engineering Units	The units of the value.
VAL	Value Out	The input value converted to a string.
OMSS	Message Out	The output message.
SNAM	Subroutine Name	The VxWorks name of a subroutine to perform additional conversion between the input and output values.
SADR	Subroutine Address	The internal representation of the subroutine address.
STYP	Subroutine symbol type	Not used.
PREC	Display Precision	This value is used for the precision of double-precision output.
AVAL	Last Value Archived	This value is used to determine if the new value needs to generate an archive monitor.
MVAL	Last Value Monitored	This value is used to determine if the new value needs to generate a monitor.
AMSS	Last Message Archived	This string used to determine if the new message string needs to generate an archive monitor
MMSS	Last Message Monitored	This string used to determine if the new message string needs to generate a monitor.
LALM	Last Value Alarmed	This value saves the last value that caused an alarm.
HIHI	Hihi Alarm Limit	This sets the HI-HI alarm limit.
LOLO	Lolo Alarm Limit	This sets the LOW-LOW alarm limit.
HIGH	High Alarm Limit	This sets the HIGH alarm limit.
LOW	Low Alarm Limit	This sets the low alarm limit.
BRSV	Bad Sub Return Severity	This alarm is set if the user subroutine returns a non-zero value.
HHSV	Hihi Severity	This alarm is set if the HIGH-HIGH value is reached.
LLSV	Lolo Severity	This alarm is set if the LOW-LOW value is reached.
HSV	High Severity	This alarm is set if the HIGH value is reached.
LSV	Low Severity	This alarm is set if the LOW value is reached.

Name	Summary	Description
HYST	Alarm Deadband	This sets the hysteresis range for alarm reporting.
ADEL	Archive Deadband	This set the archive hysteresis range.
MDEL	Monitor Deadband	This sets the monitor hysteresis range.
SIOL	Simulation Value Link	In simulation mode this link value is read and returned in VAL.
SVAL	Simulation Value	The simulation value is stored here.
SIML	Simulation Mode Link	This link determines if the record is in simulation mode or not.
SIMM	Simulation Mode	The simulation mode is stored here.
SIMS	Simulation Mode Alarm Severity	This is the value of the simulation mode alarm severity.

5.3 Record Support Routines

5.3.1 init_record

On the first pass, space is allocated for the VAL, SVAL, MVAL, AVAL fields, based upon the type of FTVL field (DBF_STRING, DBF_DOUBLE, or DBF_LONG). On the second pass, the simulation mode is checked and if true, the simulation mode and input link are set. The address of the process subroutine (SNAM) is found.

5.3.2 process

See the next section.

5.3.3 get_value

This routine fills the values of `struct valueDes` so that they refer to VAL.

5.3.4 get_precision

This routine retrieves PREC.

5.3.5 get_units

This routine returns EGU.

5.3.6 cvt_dbaddr

This routine converts the VAL field to either string, double precision, or long integer based upon the value of FTVL.

5.3.7 get_alarm_double

This routine sets the following values:

upper_alarm_limit = HIHI
upper_warning_limit = HIGH

lower_warning_limit = LOW
lower_alarm_limit = LOLO

5.4 Record Processing

In simulation mode, the simulation value is fetched from the link. Otherwise the values is fetched from the INP link. The user subroutine (SNAM) is executed and the return value is checked. For return values of zero, the input message IMSS is copied to the output message OMSS, the record is time stamped, alarms and monitors are checked, and the forward link is processed.

An alarm is set if the value is outside the preset range limits (HIHI, HIGH, LOW, LOLO) and outside of the hysteresis range (HYST).

Logging and archive monitors are raised on VAL and OMSS if their value changes. For double precision and long integer values of VAL the change must be greater than the hysteresis ranges (MDEL and ADEL).

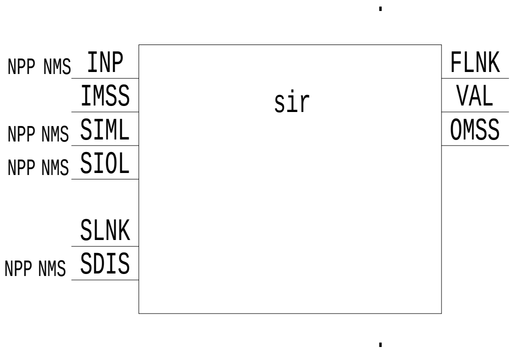
5.5 Device Support

No device support is available.

5.6 CapFast

There is one CapFast symbol for the SIR record.

FIGURE 6. CapFast *esirs* symbol



6.0 lutout - Lookup Table Output Record

The lutout record converts a character string tag into up to four output values. The output values may be either string, double-precision, or long integer, and are selected through the FTVA, FTVB, FTVC, and FTVD fields. The conversion is performed on the value found in the lookup table for the appropriate output, thus if the conversion is to a long integer type the lookup table value should be an ASCII string representation of a long integer. Conversion of string values consists of copying the lookup table value to the output field. A character string tag may have outputs with mixed types, however each output field's type is fixed at initialization.

The lutout record may selectively send values to the output fields by using the SELB field. This field is a bit mask for the enabled output fields. If the bit mask is 1, then only the VALA field will be written, if it is 3, then both VALA and VALB will be written. A value of 15 will write to all four output fields. The number of fields successfully written will be placed in the NVAL field. This value is the intersection of the bits set in SELB and the number of values in a tag's lookup table entry. The SEVR and STAT alarm fields are set to INVALID and SOFT if an input configuration string is not recognized

The lookup table can be reread from the file by writing a one to the LOAD field.

A lutout record can reference another lutout or lutin record's lookup table by making a connection between its LLNK field and the other record's LTBL field. The record can now find a tag in either its lookup table or the table from the other record. This feature can only work within an IOC; it will not work across channel access. *[This feature is not yet implemented]*.

More information and examples can be found in the lutout manual[3].

6.1 Field Summary

Field	Type	Initial	Access	Modify	Monitor	PP
VAL	STRING	Null	Yes	Yes	Yes	Yes
OVAL	STRING	Null	Yes	No	No	No
FDIR	STRING	Null	Yes	Yes	No	No
FNAM	STRING	Null	Yes	Yes	No	No
NVAL	LONG	0	Yes	No	Yes	No
ONVL	LONG	0	Yes	No	No	No
SELB	LONG	15	Yes	Yes	No	No
LOAD	LONG	0	Yes	Yes	No	Special
LTBL	NOACCESS	0	No	No	No	No
LLNK	INLINK	0	No	No	No	No
PREC	LONG	2	Yes	Yes	No	No
VALA	NOACCESS	0	Yes	No	Yes	No
VALB	NOACCESS	0	Yes	No	Yes	No
VALC	NOACCESS	0	Yes	No	Yes	No
VALD	NOACCESS	0	Yes	No	Yes	No
OLDA	NOACCESS	0	No	No	No	No
OLDB	NOACCESS	0	No	No	No	No
OLDC	NOACCESS	0	No	No	No	No
OLDD	NOACCESS	0	No	No	No	No
OUTA	OUTLINK	0	No	No	No	No
OUTB	OUTLINK	0	No	No	No	No
OUTC	OUTLINK	0	No	No	No	No
OUTD	OUTLINK	0	No	No	No	No
FTVA	GBLCHOICE	STRING	Yes	No	No	No
FTVB	GBLCHOICE	STRING	Yes	No	No	No
FTVC	GBLCHOICE	STRING	Yes	No	No	No
FTVD	GBLCHOICE	STRING	Yes	No	No	No

6.2 Field Description

TABLE 1. Lutout Record Field Description

Name	Summary	Description
VAL	Input string	An ASCII string which corresponds to a tag entry in the lookup table.
OVAL	Old tag	The previous value.

Name	Summary	Description
FDIR	Initialization file directory	The name of the directory of the lookup table.
FNAM	Initialization file name	The name of the file for the lookup table.
NVAL	Number of outputs	This is the number of values found in the lookup table for the appropriate input tag. It is between 0 and 4.
ONVL	Old number	The previous value of NVAL. Used for triggering monitors.
SELB	Selection Bits	This is a bit mask for the values which are to be converted. If SELB is 1, only VALA will be written; if 3, VALA and VALB will be written. A value of 15 will write all values.
LOAD	Reload configuration file	Any nonzero value written to this field will force the reloading of the lookup table from the configuration file defined by FDIR and FNAM.
LTBL	Lookup Table	This is the location where the conversion table is stored in memory.
LLNK	Lookup Table Link	The link to another record’s lookup table (LTBL). This link only works within an IOC.
PREC	Precision	This is the level of precision given for double-precision conversion.
VALA	Output value	The lookup table values are put on this port. A monitor event is sent if the value changed.
VALB	Output value	The lookup table values are put on this port. A monitor event is sent if the value changed.
VALC	Output value	The lookup table values are put on this port. A monitor event is sent if the value changed.
VALD	Output value	The lookup table values are put on this port. A monitor event is sent if the value changed.
OLDA	Old value	The old value. Used to trigger monitors.
OLDB	Old value	The old value. Used to trigger monitors.
OLDC	Old value	The old value. Used to trigger monitors.
OLDD	Old value	The old value. Used to trigger monitors.
OUTA	Output link	The lookup table values are sent out these links.
OUTB	Output link	The lookup table values are sent out these links.
OUTC	Output link	The lookup table values are sent out these links.
OUTD	Output link	The lookup table values are sent out these links.
FTVA	Output value type	May be one of STRING, DOUBLE, or LONG. This field forces the conversion of the appropriate table value to the correct type.
FTVB	Output value type	May be one of STRING, DOUBLE, or LONG. This field forces the conversion of the appropriate table value to the correct type.

Name	Summary	Description
FTVC	Output value type	May be one of STRING, DOUBLE, or LONG. This field forces the conversion of the appropriate table value to the correct type.
FTVD	Output value type	May be one of STRING, DOUBLE, or LONG. This field forces the conversion of the appropriate table value to the correct type.

6.3 Record Support Routines

6.3.1 init_record

During the first initialization pass the ASCII configuration file is read and stored in LTBL. Space is allocated for the VAL[A-D] and OLD[A-D] fields. On the second pass, device support initialization is performed.

6.3.2 process

See next section.

6.3.3 special

If the LOAD field is nonzero, the existing lookup table is cleared and a new table is read from the file described by the FDIR and FNAM fields.

6.3.4 get_value

This routine fills in the values of `struct valueDes` so that they refer to VAL.

6.3.5 get_precision

This routine returns the value of PREC.

6.3.6 get_enum_strs

All tags are returned by this routine.

6.3.7 cvt_dbaddr

This routine converts the output strings into the appropriate data type defined by FTV[A-D].

6.4 Record Processing

The processing algorithm performs the following steps:

1. Checks to see 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.
2. The device support write function is called. The return status value is set from the return value of this function.
3. If the device support did not complete processing, the PACT field is left TRUE and processing terminates. It is assumed the device support callback will reenter the processing routine and complete at a later time.
4. The processing flag is set to TRUE.

5. Check for alarms. If the NVAL field is equal to zero, the SOFT_ALARM is set to INVALID.
6. Check for monitors. If any value in the fields VAL, NVAL, VALA, VALB, VALC, VALD has changed a monitor event is posted for that field.
7. Process the forward link.
8. Set PACT to FALSE and exit.

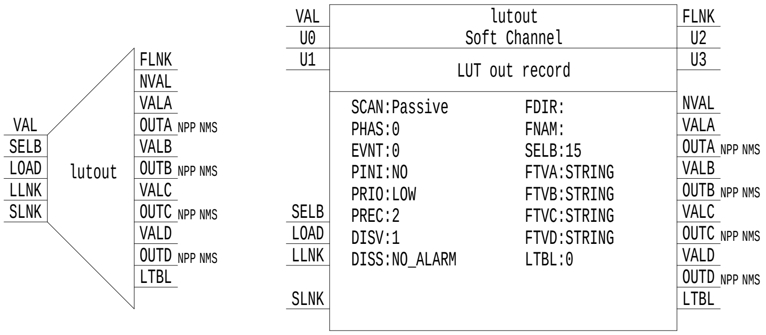
6.5 Device Support

The only device currently supported is Soft Channel.

6.6 CapFast

There are two CapFast symbols for the lutout record: *elutouts.sym* and *elutout.sym*. These symbol are included in the epics3.12.2Gem.4 and later distributions. The symbols are shown in Figure 7 on page 26.

FIGURE 7. CapFast *elutouts* and *elutout* symbols



6.7 Future Enhancements

Some future enhancements the lutout record may be:

- Access of LTBL across a network.
- Support for more data types.
- Per-item additions to the lookup table.
- Better return of the list of tags.

7.0 lutin - Lookup Table Input Record

The lutin record converts up to four input values into a character string. The input values can be any of string, double-precision, or long integer data types, determined from the FTVA, FTVB. FTVC, and FTVD fields. The conversion type must match the value found in the lookup table, thus if the type is a long integer, the corresponding lookup table value must also be an integer. For integer and double-precision values, the value is matched to within the low and high tolerances given in the lookup table. This feature supports any hysteresis or jitter associated with the sampled value. The low and high tolerances can be individually adjusted to accommodate nonlinear or preloaded systems.

The SELB field allows only the specified input fields to be used in the conversion. If the selection bit mask is 1, the first configuration that matches only the VALA field is returned. If the mask is 3, the VALA and VALB fields are the only ones tested. For a mask of 15, all input fields are used. The NVAL field returns the number of fields that were successfully matched. An NVAL field of zero will generate a SOFT alarm of INVALID.

The lookup table can be reread from the file by writing a one to the LOAD field.

A lutin record can reference another lutout or lutin record’s lookup table by making a connection between its LLNK field and the other record’s LTBL field. The record can now find a tag in either its lookup table or the table from the other record. This feature can only work within an IOC; it will not work across channel access. [*This feature is not yet implemented]*.

More information and examples can be found in the lutin manual[3].

7.1 Field Summary

TABLE 2. Lutin Record Field Summary

Field	Type	Initial	Access	Modify	Monitor	PP
VAL	STRING	Null	Yes	No	Yes	No
OVAL	STRING	Null	Yes	No	No	No
NVAL	LONG	0	Yes	No	Yes	No
ONVL	LONG	0	Yes	No	No	No
SELB	LONG	15	Yes	Yes	No	No
PREC	LONG	2	Yes	Yes	No	No
FDIR	STRING	Null	Yes	Yes	No	No
FNAM	STRING	Null	Yes	Yes	No	No
LTBL	NOACCESS	0	No	No	No	No
LLNK	INLINK	0	No	No	No	No
LOAD	LONG	0	Yes	Yes	No	Special
INPA	INLINK	0	No	No	No	No

Field	Type	Initial	Access	Modify	Monitor	PP
INPB	INLINK	0	No	No	No	No
INPC	INLINK	0	No	No	No	No
INPD	INLINK	0	No	No	No	No
VALA	NOACCESS	0	Yes	Yes	Yes	Yes
VALB	NOACCESS	0	Yes	Yes	Yes	Yes
VALC	NOACCESS	0	Yes	Yes	Yes	Yes
VALD	NOACCESS	0	Yes	Yes	Yes	Yes
OLDA	NOACCESS	0	No	No	No	No
OLDB	NOACCESS	0	No	No	No	No
OLDC	NOACCESS	0	No	No	No	No
OLDD	NOACCESS	0	No	No	No	No
FTVA	GBLCHOICE	STRING	Yes	No	No	No
FTVB	GBLCHOICE	STRING	Yes	No	No	No
FTVC	GBLCHOICE	STRING	Yes	No	No	No
FTVD	GBLCHOICE	STRING	Yes	No	No	No

7.2 Field Description

TABLE 3. Lutin Record Field Description

Name	Summary	Description
VAL	Result	The character string of the configuration that matches the inputs.
OVAL	Old Result	The old VAL. Used to trigger monitors.
NVAL	Result Code	The number of values that were used in the match.
ONVL	Old Result Code	The old NVAL. Used to trigger monitors.
SELB	Selection Bits	The bit mask of the input values to use for testing.
PREC	Precision	The precision of double-precision values.
FDIR	Init File Directory	The name of the directory of the ASCII configuration file.
FNAM	Init File Name	The name of the file of the ASCII configuration file.
LTBL	Lookup Table	The address of the storage of the lookup table.
LLNK	Lookup Table Link	A link to another record’s lookup table.
LOAD	Reload Table	Any nonzero value written to this field will force the reloading of the lookup table from the configuration file defined by FDIR and FNAM.
INPA	Input	The input link.
INPB	Input	The input link.
INPC	Input	The input link.

Name	Summary	Description
INPD	Input	The input link.
VALA	Value of Input	The input value.
VALB	Value of Input	The input value.
VALC	Value of Input	The input value.
VALD	Value of Input	The input value.
OLDA	Old Value of Input	The old value. Used to trigger monitors.
OLDB	Old Value of Input	The old value. Used to trigger monitors.
OLDC	Old Value of Input	The old value. Used to trigger monitors.
OLDD	Old Value of Input	The old value. Used to trigger monitors.
FTVA	Type of Value	The data type of the input value. May be either STRING, DOUBLE, or LONG.
FTVB	Type of Value	The data type of the input value. May be either STRING, DOUBLE, or LONG.
FTVC	Type of Value	The data type of the input value. May be either STRING, DOUBLE, or LONG.
FTVD	Type of Value	The data type of the input value. May be either STRING, DOUBLE, or LONG.

7.3 Record Support Routines

7.3.1 init_record

During the first initialization pass the ASCII configuration file is read and stored in LTBL. Space is allocated for the VAL[A-D] and OLD[A-D] fields. On the second pass, device support initialization is performed.

7.3.2 process

See next section.

7.3.3 special

If the LOAD field is nonzero, the existing lookup table is cleared and a new table is read from the file described by the FDIR and FNAM fields.

7.3.4 get_value

This routine fills in the values of `struct valueDes` so that they refer to VAL.

7.3.5 get_precision

This routine returns the value of PREC.

7.3.6 get_enum_strs

All tags are returned by this routine.

7.3.7 cvt_dbaddr

This routine converts the input strings into the appropriate data type defined by FTV[A-D].

7.4 Record Processing

The processing algorithm performs the following steps:

1. Checks to see 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.
2. The device support read function is called. The return status value is set from the return value of this function.
3. If the device support did not complete processing, the PACT field is left TRUE and processing terminates. It is assumed the device support callback will reenter the processing routine and complete at a later time.
4. The processing flag is set to TRUE.
5. Check for alarms. If the NVAL field is equal to zero, SOFT_ALARM is set to INVALID.
6. Check for monitors. If any value in the fields VAL, NVAL, VALA, VALB, VALC, VALD has changed a monitor event is posted for that field.
7. Process the forward link.
8. Set PACT to FALSE and exit.

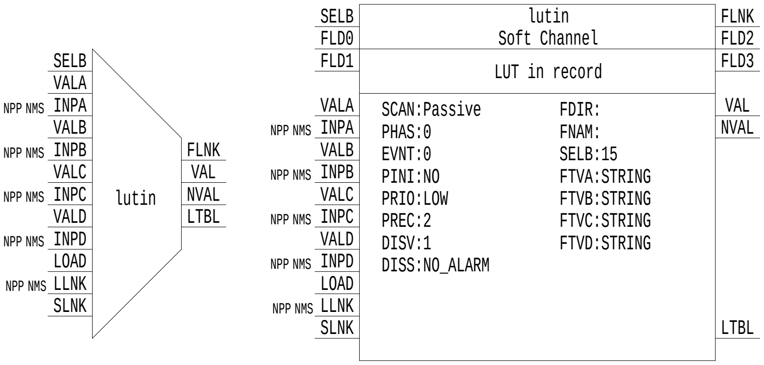
7.5 Device Support

The only device currently supported is Soft Channel.

7.6 CapFast

There are two CapFast symbols for the lutin record: *elutins.sym* and *elutin.sym*. These symbols are part of the epics3.12.2Gem.4 and later distributions. The symbols are shown in Figure 8 on page 30.

FIGURE 8. CapFast *elutins* and *elutin* symbols



7.7 Future Enhancements

Some future enhancements the lutin record may be:

- Access of LTBL across a network.
- Support for more data types.
- Per-item additions to the lookup table.
- Better return of the list of tags.

8.0 mosub - Multiple Output Subroutine Record

The material presented here is taken from [4].

8.1 Introduction

The Multiple Output Subroutine record was developed for two reasons. Firstly, to provide a record which has more than one output field and secondly, as a record which can handle the transfer of ‘strings’ across the database. Traditional EPICS records only have one output value. This is very restricting when dealing with an application which needs to transfer large amounts of data between records and leads to unnecessarily complicated database schematics. It is also true that with the ‘standard’ set of EPICS records, there is no way of passing a ‘string’ between two records. For example, there are ‘stringin’ and ‘stringout’ records, but the problem is, how do you get the string into the record in the first place? The Multiple Output Subroutine record solves both of these problems by adding extra functionality to the original EPICS ‘sub’ record.

8.2 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	DOUBLE	No	0	Yes	Yes	Yes	Yes
INAM	STRING	Yes	Null	Yes	No	No	No
SNAM	STRING	Yes	Null	Yes	No	No	No
SADR	NOACCESS	No	0	No	No	No	No
STYP	SHORT	No	0	Yes	No	No	No
INPA	INLINK	Yes	0	No	No	N/A	No
INPB	INLINK	Yes	0	No	No	N/A	No
INPC	INLINK	Yes	0	No	No	N/A	No
INPD	INLINK	Yes	0	No	No	N/A	No
INPE	INLINK	Yes	0	No	No	N/A	No
INPF	INLINK	Yes	0	No	No	N/A	No
INPG	INLINK	Yes	0	No	No	N/A	No
INPH	INLINK	Yes	0	No	No	N/A	No
INPI	INLINK	Yes	0	No	No	N/A	No
INPJ	INLINK	Yes	0	No	No	N/A	No
INPK	INLINK	Yes	0	No	No	N/A	No
INPL	INLINK	Yes	0	No	No	N/A	No
INPM	INLINK	Yes	0	No	No	N/A	No
OUTA	OUTLINK	Yes	0	No	No	N/A	No
OUTB	OUTLINK	Yes	0	No	No	N/A	No
OUTC	OUTLINK	Yes	0	No	No	N/A	No
OUTD	OUTLINK	Yes	0	No	No	N/A	No
OUTE	OUTLINK	Yes	0	No	No	N/A	No
OUTF	OUTLINK	Yes	0	No	No	N/A	No
OUT1	OUTLINK	Yes	0	No	No	N/A	No
OUT2	OUTLINK	Yes	0	No	No	N/A	No
OUT3	OUTLINK	Yes	0	No	No	N/A	No
OUT4	OUTLINK	Yes	0	No	No	N/A	No
OUT5	OUTLINK	Yes	0	No	No	N/A	No
OUT6	OUTLINK	Yes	0	No	No	N/A	No
EGU	STRING	Yes	Null	Yes	Yes	No	No
HOPR	FLOAT	Yes	0	Yes	Yes	No	No
LOPR	FLOAT	Yes	0	Yes	Yes	No	No
HIHI	FLOAT	Yes	0	Yes	Yes	No	Yes
LOLO	FLOAT	Yes	0	Yes	Yes	No	Yes
HIGH	FLOAT	Yes	0	Yes	Yes	No	Yes

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LOW	FLOAT	Yes	0	Yes	Yes	No	Yes
PREC	SHORT	Yes	0	Yes	Yes	No	No
BRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HHSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LLSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
LSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
HYST	DOUBLE	Yes	0	Yes	Yes	No	No
ADEL	DOUBLE	Yes	0	Yes	Yes	No	No
MDEL	DOUBLE	Yes	0	Yes	Yes	No	No
A	DOUBLE	No	0	Yes	Yes	Yes	Yes
B	DOUBLE	No	0	Yes	Yes	Yes	Yes
C	DOUBLE	No	0	Yes	Yes	Yes	Yes
D	DOUBLE	No	0	Yes	Yes	Yes	Yes
E	DOUBLE	No	0	Yes	Yes	Yes	Yes
F	DOUBLE	No	0	Yes	Yes	Yes	Yes
G	DOUBLE	No	0	Yes	Yes	Yes	Yes
H	DOUBLE	No	0	Yes	Yes	Yes	Yes
I	DOUBLE	No	0	Yes	Yes	Yes	Yes
J	DOUBLE	No	0	Yes	Yes	Yes	Yes
K	DOUBLE	No	0	Yes	Yes	Yes	Yes
L	DOUBLE	No	0	Yes	Yes	Yes	Yes
M	STRING	No	Null	Yes	Yes	Yes	Yes
VALA	DOUBLE	No	0	Yes	Yes	Yes	No
VALB	DOUBLE	No	0	Yes	Yes	Yes	No
VALC	DOUBLE	No	0	Yes	Yes	Yes	No
VALD	DOUBLE	No	0	Yes	Yes	Yes	No
VALE	DOUBLE	No	0	Yes	Yes	Yes	No
VALF	DOUBLE	No	0	Yes	Yes	Yes	No
STR1	STRING	No	Null	Yes	Yes	Yes	No
STR2	STRING	No	Null	Yes	Yes	Yes	No
STR3	STRING	No	Null	Yes	Yes	Yes	No
STR4	STRING	No	Null	Yes	Yes	Yes	No
STR5	STRING	No	Null	Yes	Yes	Yes	No
STR6	STRING	No	Null	Yes	Yes	Yes	No
LA	DOUBLE	No	0	Yes	No	No	No
LB	DOUBLE	No	0	Yes	No	No	No
LC	DOUBLE	No	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
LD	DOUBLE	No	0	Yes	No	No	No
LE	DOUBLE	No	0	Yes	No	No	No
LF	DOUBLE	No	0	Yes	No	No	No
LG	DOUBLE	No	0	Yes	No	No	No
LH	DOUBLE	No	0	Yes	No	No	No
LI	DOUBLE	No	0	Yes	No	No	No
LJ	DOUBLE	No	0	Yes	No	No	No
LK	DOUBLE	No	0	Yes	No	No	No
LL	DOUBLE	No	0	Yes	No	No	No
LM	STRING	No	Null	Yes	No	No	No
LVA	DOUBLE	No	0	Yes	No	No	No
LVB	DOUBLE	No	0	Yes	No	No	No
LVC	DOUBLE	No	0	Yes	No	No	No
LVD	DOUBLE	No	0	Yes	No	No	No
LVE	DOUBLE	No	0	Yes	No	No	No
LVF	DOUBLE	No	0	Yes	No	No	No
LS1	STRING	No	0	Yes	No	No	No
LS2	STRING	No	0	Yes	No	No	No
LS3	STRING	No	0	Yes	No	No	No
LS4	STRING	No	0	Yes	No	No	No
LS5	STRING	No	0	Yes	No	No	No
LS6	STRING	No	0	Yes	No	No	No
LALM	DOUBLE	No	0	Yes	No	No	No
ALST	DOUBLE	No	0	Yes	No	No	No
MLST	DOUBLE	No	0	Yes	No	No	No

8.3 Field Descriptions

Name	Summary	Description
VAL	Value Field	This field is not used.
INAM	Initialisation Routine	This is the name of the initialisation routine to be called once, at iocInit.
SNAM	Process Routine	This is the name of the routine to be called when the record processes.
SADR	Subroutine Address	Filled in by record processing.
STYP	Subroutine Symbol Type	Filled in by record processing.
INPA,..., INPM	Input Link A,..., Input Link M	The input links from where the values of A,...,M are fetched during record processing.
OUTA,..., OUTF	Output Link A,.. Output Link F	The output links on which the DOUBLE values located at VALA,...,VALF are placed during record processing.
OUT1,..., OUT6	Output Link 1,.. Output Link 6	The output links on which the STRINGS located at VAL1,...,VAL6 are placed during record processing.
EGU	Engineering Units	ASCII string describing Engineering units. This field is used by record support to supply a units description string when *get_units* is called.
HOPR	High Operating Range	These fields determine the upper and lower display limits for graphical displays and the upper and lower control limits for control displays. The fields are used in the record support routines: *get_graphic_double* and *get_control_double*.
LOPR	Low Operating Range	
HIHI	Hihi Alarm Limit	These fields specify the alarm limits and severities. Note that they are used as range limits for checks against the VAL field. Since the VAL field is not used by this record, these fields are redundant.
LOLO	Lolo Alarm Limit	
HIGH	High Alarm Limit	
LOW	Low Alarm Limit	
BRSV	Severity for a subroutine return value less than 0.	
HHSV	Severity for a Hihi Alarm.	
LLSV	Severity for a Lolo Alarm.	
HSV	Severity for a High Alarm.	
LSV	Severity for a Low Alarm	

Name	Summary	Description
HYST	Alarm Deadband	These parameters specify hysteresis factors for raising alarms and posting logging and monitor events for the VAL field. As above, since VAL is not used by this record, these fields are redundant.
ADEL	Archive Deadband	
MDEL	Monitor Deadband	
A,...,L	A,...,L	The input fields which hold the DOUBLE values fetched in across the input links INPA,...,INPL.
M	M	The input field which holds the STRING value fetched in across the input link INPM.
VALA,..., VALF	VALA,..., VALF	These fields can hold any DOUBLE value that the user desires. They can be set from within the subroutine called at process time. The DOUBLES are placed on the output links: OUTA,...,OUTF when the record processes.
STR1,..., STR6	STR1,...,STR6	These fields can hold any STRING value that the user desires. They can be set from within the subroutine called at process time. The STRINGS are placed on the output links: OUT1,...,OUT6 when the record processes.
LA,...,LL	Last A,.., Last L	Previous input DOUBLE values. These fields are used to decide when to trigger monitors on A,...,L.
LM	Last M	Previous input STRING value. This field is used to decide when to trigger a monitor on M.
LVA,..., LVF	Last VALA,..., Last VALF	Previous output DOUBLE values. These fields are used to decide when to trigger monitors on VALA,...,VALF.
LS1,..., LS6	Last STR1,..., Last STR6	Previous output STRING values. These fields are used to decide when to trigger monitors on STR1,...,STR6.
LALM	Last Alarm Monitor Trigger Value	These fields are used to implement the hysteresis factors for monitors. Again, since VAL is not used by this record, these fields are redundant.
ALST	Last Archiver Monitor Trigger Value	
MLST	Last Value Change Monitor Trigger Value	

8.4 Record Support Routines

8.4.1 init_record

The VAL field is set equal to 0. For each constant input link to a DOUBLE value, the corresponding field is initialised with the constant value. For each input link that is of type PV_LINK, a channel access link is created. The STRING input field ‘M’ is initialised as a blank string and a channel access link is created for INPM. For each output DOUBLE and STRING link, a channel access link is created.

The user initialisation routine, whose name is specified in INAM, is located and called. Note that the record assumes that an initialisation routine will always be specified. The record fails if no routine name is given.

The routine specified in SNAM is located and its address and type are stored in SADR and STYP. These are used to call the routine when the record processes.

8.4.2 process

The record can be made to process by writing to any of the input fields A-M. The process routine implements the following algorithm:

- If PACT is FALSE, set PACT = TRUE and process the input links.
- Call the routine specified in SNAM.
- Process the output links.
- Get the time stamp for this processing.
- Check for alarms. Alarms are irrelevant in this record because the VAL field is not used.
- Post events for value changes in the input fields and the output fields.
- Process the record on the end of the forward link.
- Set PACT = FALSE.

8.4.3 get_value

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

8.4.4 get_units

Retrieves the Engineering Units string.

8.4.5 get_precision

Retrieves the value of the PREC field.

8.4.6 get_graphic_double

Sets the upper and lower display limits for a field. If the field is one of A-D, LA-LD, VALA-VALF, HIHI, HIGH, LOW or LOLO, the limits are set to HOPR and LOPR.

8.4.7 get_control_double

Sets the upper and lower control limits for a field. If the field is one of A-D, LA-LD, VAL, HIHI, HIGH, LOW or LOLO, the limits are set to HOPR and LOPR.

8.4.8 get_alarm_double

Sets the following values:

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

9.0 GenSub - The General Subroutine Record

9.1 Introduction

This record, known as ‘GenSub’, has been designed as a replacement for the standard EPICS subroutine record. It allows the easy passage of arrays, scalars and user defined structures between records existing within the same database and between those which exist in separate IOC’s. The advantage of using arrays when transferring data between IOC’s, rather than a set of values from individual records, is that Channel Access guarantees to write the whole array with one *ca_put*. The atomicity of this operation insures that data at the receiving end is consistent at any given time. This is important in many applications. The current Channel Access limit for the amount of data which can be transferred with a single *ca_put* is 16kB.

Other features of the ‘GenSub’ record include the following:

- Up to 10 input fields.
- Up to 10 output fields.
- The types of both input and output fields are completely configurable by the user.
- The routine to be called at process time can be changed dynamically after the database has been loaded. The name of the routine can either be fetched in over a link from another record or written directly into the SNAM field.
- The user can configure the record to decide when events will be posted for the output fields. This can be: Never, Always or just when any element of an array changes value.
- The VAL field holds the value returned from the routine called during record processing.

9.2 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VAL	LONG	No	0	Yes	Yes	Yes	No
OVAL	LONG	No	0	Yes	Yes	No	No
SADR	LONG	No	0	Yes	No	Yes	No
OSAD	LONG	No	0	Yes	No	No	No
LFLG	RECCHOICE	Yes	Ignore	Yes	Yes	No	No
EFLG	RECCHOICE	Yes	Always	Yes	Yes	No	No
SUBL	INLINK	Yes	0	No	No	N/A	No
INAM	STRING	Yes	Null	Yes	No	No	No
SNAM	STRING	Yes	Null	Yes	Yes	No	No
ONAM	STRING	Yes	Null	Yes	No	No	No
STYP	SHORT	No	0	Yes	No	No	No
BRSV	GBLCHOICE	Yes	0	Yes	Yes	No	Yes
PREC	SHORT	Yes	0	Yes	Yes	No	No
INPA	INLINK	Yes	0	No	No	N/A	No
INPB	INLINK	Yes	0	No	No	N/A	No
INPC	INLINK	Yes	0	No	No	N/A	No
INPD	INLINK	Yes	0	No	No	N/A	No
INPE	INLINK	Yes	0	No	No	N/A	No
INPF	INLINK	Yes	0	No	No	N/A	No
INPG	INLINK	Yes	0	No	No	N/A	No
INPH	INLINK	Yes	0	No	No	N/A	No
INPI	INLINK	Yes	0	No	No	N/A	No
INPJ	INLINK	Yes	0	No	No	N/A	No
UFA	STRING	Yes	Null	Yes	No	No	No
UFB	STRING	Yes	Null	Yes	No	No	No
UFC	STRING	Yes	Null	Yes	No	No	No
UFD	STRING	Yes	Null	Yes	No	No	No
UFE	STRING	Yes	Null	Yes	No	No	No
UFF	STRING	Yes	Null	Yes	No	No	No
UFG	STRING	Yes	Null	Yes	No	No	No
UFH	STRING	Yes	Null	Yes	No	No	No
UFI	STRING	Yes	Null	Yes	No	No	No
UFJ	STRING	Yes	Null	Yes	No	No	No
A	NOACCESS	No	0	No	Yes	No	No
B	NOACCESS	No	0	No	Yes	No	No
C	NOACCESS	No	0	No	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
D	NOACCESS	No	0	No	Yes	No	No
E	NOACCESS	No	0	No	Yes	No	No
F	NOACCESS	No	0	No	Yes	No	No
G	NOACCESS	No	0	No	Yes	No	No
H	NOACCESS	No	0	No	Yes	No	No
I	NOACCESS	No	0	No	Yes	No	No
J	NOACCESS	No	0	No	Yes	No	Yes
FTA	GBLCHOICE	Yes	Double	Yes	No	No	No
FTB	GBLCHOICE	Yes	Double	Yes	No	No	No
FTC	GBLCHOICE	Yes	Double	Yes	No	No	No
FTD	GBLCHOICE	Yes	Double	Yes	No	No	No
FTE	GBLCHOICE	Yes	Double	Yes	No	No	No
FTF	GBLCHOICE	Yes	Double	Yes	No	No	No
FTG	GBLCHOICE	Yes	Double	Yes	No	No	No
FTH	GBLCHOICE	Yes	Double	Yes	No	No	No
FTI	GBLCHOICE	Yes	Double	Yes	No	No	No
FTJ	GBLCHOICE	Yes	Double	Yes	No	No	No
NOA	ULONG	Yes	1	Yes	No	No	No
NOB	ULONG	Yes	1	Yes	No	No	No
NOC	ULONG	Yes	1	Yes	No	No	No
NOD	ULONG	Yes	1	Yes	No	No	No
NOE	ULONG	Yes	1	Yes	No	No	No
NOF	ULONG	Yes	1	Yes	No	No	No
NOG	ULONG	Yes	1	Yes	No	No	No
NOH	ULONG	Yes	1	Yes	No	No	No
NOI	ULONG	Yes	1	Yes	No	No	No
NOJ	ULONG	Yes	1	Yes	No	No	No
OUTA	OUTLINK	Yes	0	No	No	N/A	No
OUTB	OUTLINK	Yes	0	No	No	N/A	No
OUTC	OUTLINK	Yes	0	No	No	N/A	No
OUTD	OUTLINK	Yes	0	No	No	N/A	No
OUTE	OUTLINK	Yes	0	No	No	N/A	No
OUTF	OUTLINK	Yes	0	No	No	N/A	No
OUTG	OUTLINK	Yes	0	No	No	N/A	No
OUTH	OUTLINK	Yes	0	No	No	N/A	No
OUTI	OUTLINK	Yes	0	No	No	N/A	No
OUTJ	OUTLINK	Yes	0	No	No	N/A	No
UFVA	STRING	Yes	Null	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
UFVB	STRING	Yes	Null	Yes	No	No	No
UFVC	STRING	Yes	Null	Yes	No	No	No
UFVD	STRING	Yes	Null	Yes	No	No	No
UFVE	STRING	Yes	Null	Yes	No	No	No
UFVF	STRING	Yes	Null	Yes	No	No	No
UFVG	STRING	Yes	Null	Yes	No	No	No
UFVH	STRING	Yes	Null	Yes	No	No	No
UFVI	STRING	Yes	Null	Yes	No	No	No
UFVJ	STRING	Yes	Null	Yes	No	No	No
VALA	NOACCESS	No	0	No	Yes	Yes/No	No
VALB	NOACCESS	No	0	No	Yes	Yes/No	No
VALC	NOACCESS	No	0	No	Yes	Yes/No	No
VALD	NOACCESS	No	0	No	Yes	Yes/No	No
VALE	NOACCESS	No	0	No	Yes	Yes/No	No
VALF	NOACCESS	No	0	No	Yes	Yes/No	No
VALG	NOACCESS	No	0	No	Yes	Yes/No	No
VALH	NOACCESS	No	0	No	Yes	Yes/No	No
VALI	NOACCESS	No	0	No	Yes	No	No
VALJ	NOACCESS	No	0	No	Yes	No	No
OVLA	NOACCESS	No	0	No	No	No	No
OVLB	NOACCESS	No	0	No	No	No	No
OVLC	NOACCESS	No	0	No	No	No	No
OVLD	NOACCESS	No	0	No	No	No	No
OVLE	NOACCESS	No	0	No	No	No	No
OVLF	NOACCESS	No	0	No	No	No	No
OVLG	NOACCESS	No	0	No	No	No	No
OVLH	NOACCESS	No	0	No	No	No	No
OVLI	NOACCESS	No	0	No	No	No	No
OVLJ	NOACCESS	No	0	No	No	No	No
FTVA	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVB	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVC	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVD	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVE	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVF	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVG	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVH	GBLCHOICE	Yes	Double	Yes	No	No	No
FTVI	GBLCHOICE	Yes	Double	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
FTVJ	GBLCHOICE	Yes	Double	Yes	No	No	No
NOVA	ULONG	Yes	1	Yes	No	No	No
NOVB	ULONG	Yes	1	Yes	No	No	No
NOVC	ULONG	Yes	1	Yes	No	No	No
NOVD	ULONG	Yes	1	Yes	No	No	No
NOVE	ULONG	Yes	1	Yes	No	No	No
NOVF	ULONG	Yes	1	Yes	No	No	No
NOVG	ULONG	Yes	1	Yes	No	No	No
NOVH	ULONG	Yes	1	Yes	No	No	No
NOVI	ULONG	Yes	1	Yes	No	No	No
NOVJ	ULONG	Yes	1	Yes	No	No	No
TOVA	ULONG	Yes	0	Yes	No	No	No
TOVB	ULONG	Yes	0	Yes	No	No	No
TOVC	ULONG	Yes	0	Yes	No	No	No
TOVD	ULONG	Yes	0	Yes	No	No	No
TOVE	ULONG	Yes	0	Yes	No	No	No
TOVF	ULONG	Yes	0	Yes	No	No	No
TOVG	ULONG	Yes	0	Yes	No	No	No
TOVH	ULONG	Yes	0	Yes	No	No	No
TOVI	ULONG	Yes	0	Yes	No	No	No
TOVJ	ULONG	Yes	0	Yes	No	No	No

9.3 Field Descriptions

Name	Summary	Description
VAL	Value returned from process routine	This field holds the value returned from the user defined process routine.
OVAL	Old VAL	Previous VAL, used to decide when to post events.
SADR	Subroutine Address	The address of the routine called at process time.
OSAD	Old SADR	Previous SADR, used to decide when to post events.
LFLG	Link Flag	Tells the record whether to read or ignore the SUBL link. If the value is READ, then the name of the subroutine to be called at process time is read from SUBL. If the value is IGNORE, the name of the subroutine is that currently held in SNAM.
EFLG	Event Flag	Tells the record when to post events on the output fields VALA,...,VALJ. If the value is NEVER, events are never posted. If the value is ALWAYS, events are posted everytime the record processes. If the value is ON CHANGE, events are posted when any element of an array changes value. Archiving and Value Change events are posted in each case.
SUBL	Subroutine Link	Where to get the subroutine name from.
INAM	Initialisation Routine	This is the name of the initialisation routine to be called once, at iocInit.
SNAM	Process Routine	This is the name of the routine to be called when the record processes. Note, this can be overwritten by the SUBL link, if LFLG is set to READ.
ONAM	Process Routine	Old process subroutine name.
STYP	Subroutine Symbol Type	Filled in by record processing.
BRSV	Severity for a subroutine return value less than 0.	Specifies the Alarm severity.
PREC	Display Precision	Specifies the number of decimal places with which to display the values of the fields VALA,...,VALJ.
INPA,..., INPJ	Input Link A,..., Input Link J	The input links from where the values of A,...,J are fetched during record processing.
UFA,..., UFJ	User Function A User Function J	These are the names of functions which return the sizes of any user defined structures to be received in the input fields A,...,J.
A,...,J	Input Fields	The input fields which hold the scalar values or arrays fetched in across the input links INPA,...,INPJ.
FTA,..., FTJ	Field Type of A Field Type of J	Field types of the input values. These can be CHAR, STRING, DOUBLE, LONG, etc.

Name	Summary	Description
NOA..., NOJ	Number of elements in A,.. Number of elements in J	The number of elements in each input field. Default is 1 (scalar value). An array is specified by setting this field to greater than 1.
OUTA,..., OUTJ	Output Link A,.. Output Link J	The output links on which the scalars or arrays located at VALA,...,VALJ are placed during record processing.
UFVA,..., UFVJ	User Function VALA,..., User Function VALJ	These are the names of functions which return the sizes of any user defined structures which are to passed out of the record from the fields VALA,...,VALJ.
VALA,.., VALJ	Output Fields	The output fields which hold the scalar values or arrays pushed out across the output links OUTA,...,OUTJ.
FTVA,..., FTVJ	Field Type of VALA Field Type of VALJ	Field types of the output values. These can be CHAR, STRING, DOUBLE, LONG, etc.
OVLA,.., OVLJ	Previous Outputs	The previous values of the outputs. These are used to decide when to post events if EFLG is set to ON CHANGE.
NOVA,..., NOVJ	Number of elements in VALA Number of elements in VALJ	The number of elements in each output field. Default is 1 (scalar value). An array is specified by setting this field to greater than 1.
TOVA,..., TOVJ	Total Number of bytes in VALA Total Number of bytes in VALJ	The total number of bytes in each output field. These are used internally by record processing and do not concern the user.

9.4 Record Support Routines

9.4.1 init_record

This routine is called twice at *iocInit*. On the first call it does the following:

- Look for any user functions defined in the fields UFA-UFJ and UFVA-UFVJ. If they have been defined, call them to get the size of the structure which is to passed across the link. If they are not defined, no routine is called.
- Calloc sufficient space to hold the number of input scalars and/or arrays defined by the settings of the fields FTA-FTJ and NOA-NOJ. If a user function has been defined, calloc the space required by the multiple of the number of elements and size returned from the user function.
- Calloc sufficient space to hold the number of output scalars and/or arrays defined by the settings of the fields FTVA-FTVJ and NOVA-NOVJ. If a user function has been

defined, calloc the space required by the multiple of the number of elements and size returned from the user function. For the output fields, also calloc space to hold the previous value of a field. This is required when the decision is made on whether or not to post events.

On the second call, it does the following:

- Create the SUBL link.
- Create each input link.
- Create each output link.
- If the field INAM is set, look-up the address of the routine and call it.
- If the field LFLG is set to IGNORE and SNAM is defined, look-up the address of the process routine.

9.4.2 process

This routine implements the following algorithm:

- Set PACT to TRUE.
- If the field LFLG is set to READ, get the subroutine name from the SUBL link. If the name is not NULL and it is not the same as the previous subroutine name, look-up the subroutine address. Set the old subroutine name, ONAM, equal to the current name, SNAM.
- Fetch the values from the input links.
- Call the routine specified by SNAM.
- Set VAL equal to the return value from the routine specified by SNAM.
- Place the output values on the output links.
- Get the time of processing and put it into the timestamp field.
- If the subroutine address has changed, post a change-of-value event and a log event for the SADR field. If VAL has changed, post a change-of value and log event for this field. If EFLG is set to ALWAYS, post change-of-value and log events for every output field. If EFLG is set to ON CHANGE, post change-of-value and log events for every output field which has changed. In the case of an array, an event will be posted if any single element of the array has changed. If EFLG is set to NEVER, no change-of-value or log events are posted for the output fields.
- Process the record on the end of the forward link, if one exists.
- Set PACT to FALSE.

9.4.3 get_value

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

9.4.4 get_precision

Sets the display precision to the value of PREC for any of the output fields VALA,..., VALJ. This routine could be called for any of these fields.

9.4.5 cvt_dbaddr

The purpose of this routine is to fill in the struct *dbAddr* for the field of the record for which it has been called. Typically, the number of elements in the field, the field type and the size of the field will be set in this routine. For arrays, this record support routine is essential.

9.4.6 get_array_info

This routine returns the current number of elements and the offset of the first value for an array. For this record, the offset field is always 0.

9.4.7 put_array_info

This routine is called after new values have been placed in an array.

9.4.8 special

This routine is called whenever the SNAM field changes. It is called twice, once before the change and once after. On the first call, the routine simply returns. On the second call, after SNAM has changed, it implements the following algorithm:

- If LFLG is set to IGNORE and SNAM is not NULL, then look-up the address of the routine specified by SNAM. Set the SADR field equal to the subroutine address.
- Post change-of-value and log events for the SADR field, if this has changed.

9.5 Use of the ‘GenSub’ Record

Two ‘GenSub’ records can be used to transfer data between one another. These records can be located in the same IOC or in separate IOC’s. The data can be a scalar of any type, an array, a user defined structure or an array of user defined structures.

Let us name the ‘GenSub’ record which is sending data, record A, and the ‘GenSub’ record which is receiving the data, record B. There are some fields which must be set-up correctly in each record before the data transfer can occur without error. The output field types and the number of elements in the output fields of record A must match the input field types and number of elements in the input fields of record B. Thus, combining the two records is rather like a jigsaw. Let us take an example.

If 5 doubles are to be passed from the VALA field of record A to the A field of record B, then the following settings are necessary:

Record A should have: FTVA = DOUBLE, NOVA = 5.
Record B should have: FTA = DOUBLE, NOA = 5.

9.6 Use of User Defined Structures

It is possible for the user to define a structure which is to be passed between two ‘GenSub’ records. As an example, let us imagine that the following structure needs to be transferred between the VALB field of record A and the B field of record B:

```
struct pinfo
{
  int      age;
  char     name[128];
  char     posn[128];
  double  salary;
};
```

The user must supply a function which will return the size of this structure. An example, and template for such a function is given below:

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

#include <dbEvent.h>
#include <dbDefs.h>
#include <dbCommon.h>
#include <recSup.h>
#include <GenSubRecord.h>
#include <pinfo.h>

long setup( struct GenSubRecord *pgsub )
{
   return( sizeof(struct pinfo) );
}
```

The user should set the following fields:

```
Record A:  UFVB:setup
           FTVB:CHAR
           NOVB: 1
Record B:   UFB: setup
           FTB: CHAR
           NOB: 1
```

These settings indicate that a single structure, of the size of what is returned from *setup*, will be passed from VALB of record A to B of record B. Inside the process routine called from record B, the user should cast the B field as a pointer to the structure, thus:

```
struct pinfo *ex;
ex = (struct pinfo *)pgsub->b;
```

The elements of the structure then become available to the routine. Note that the user is responsible for ensuring that the two ‘GenSub’ records, which may be in separate IOC’s, share identical layouts of the structure.

It is also worth pointing out here, that because we are packing the structure into a stream of characters, character arrays within the structure are not limited to the 40 character limit imposed on strings for normal record fields. In this example, we have used character arrays dimensioned to 128.

The total size of the structure must be less than 16kB. Internal ‘GenSub’ record code checks for this.

9.7 Dynamically Changing the User Routine called during Record Processing

The ‘GenSub’ record allows the user to dynamically change which routine is called when the record processes. This can be done in two ways:

- The LFLG field can be set to READ so that the name of the routine is read from the SUBL link. Thus, whatever is feeding this link can change the name of the routine before the ‘GenSub’ record is processed. In this case, the record looks in the symbol table for the symbol name whenever the name of routine fetched from the link changes.
- The LFLG field can be set to IGNORE. In this case, the rotine called during record processing is that specified in the SNAM property field. Under these conditions, the SNAM field can be changed by a Channel Access write to the SNAM field. Thus, during development work, when it is often required to run a modified version of the routine, it will no longer be a requirement to reboot the IOC and reload the database. A new routine will be called during record processing if the routine is loaded with the vxWorks *ld* command, and *cau* is used to put the name of the routine into the record’s SNAM field. After the SNAM field has been changed, the record automatically looks up the symbol name in the symbol table. Note that, if the same routine name is used, this is not a problem. The record finds the latest version of the code which has been loaded. Obviously, one needs to take care of the amount of memory which is used, if no *unld* command is ever used.