Created: March 12, 2003
Modified:March 17, 2004

EPICS Record Reference Manual

The EPICS genSub Record Reference Manual

Andy Foster

This document decribes the EPICS genSub Record.

1.0 genSub - The General Subroutine Record

1.1 Introduction

This record is an enhancement to the standard EPICS subroutine record. It allows the easy passage of arrays, scalars and user defined structures between records within the same database or in separate IOCs. The advantage of using arrays when transferring data between IOCs, 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 the consistency of the data at the receiving end. This is important in many applications. Versions of EPICS prior to release R3.14 limited the amount of data which could be transferred with a single *ca_put* to 16kB. In EPICS release R3.14 the environment variable EPICS_CA_MAX_ARRAY_BYTES can be used to raise this limit.

Other features of the 'genSub' record include the following:

- Up to 21 input fields.
- Up to 21 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.

1.2 Field Summary

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
VERS	DOUBLE	No	1.4	Yes	No	No	No
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
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
INPN	INLINK	Yes	0	No	No	N/A	No
INPO	INLINK	Yes	0	No	No	N/A	No
INPP	INLINK	Yes	0	No	No	N/A	No
INPQ	INLINK	Yes	0	No	No	N/A	No
INPR	INLINK	Yes	0	No	No	N/A	No
INPS	INLINK	Yes	0	No	No	N/A	No
INPT	INLINK	Yes	0	No	No	N/A	No
INPU	INLINK	Yes	0	No	No	N/A	No
UFA	STRING	Yes	Null	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
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
UFK	STRING	Yes	Null	Yes	No	No	No
UFL	STRING	Yes	Null	Yes	No	No	No
UFM	STRING	Yes	Null	Yes	No	No	No
UFN	STRING	Yes	Null	Yes	No	No	No
UFO	STRING	Yes	Null	Yes	No	No	No
UFP	STRING	Yes	Null	Yes	No	No	No
UFQ	STRING	Yes	Null	Yes	No	No	No
UFR	STRING	Yes	Null	Yes	No	No	No
UFS	STRING	Yes	Null	Yes	No	No	No
UFT	STRING	Yes	Null	Yes	No	No	No
UFU	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
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
K	NOACCESS	No	0	No	Yes	No	No
L	NOACCESS	No	0	No	Yes	No	No
M	NOACCESS	No	0	No	Yes	No	No
N	NOACCESS	No	0	No	Yes	No	No
O	NOACCESS	No	0	No	Yes	No	No
P	NOACCESS	No	0	No	Yes	No	No
Q	NOACCESS	No	0	No	Yes	No	No
R	NOACCESS	No	0	No	Yes	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
S	NOACCESS	No	0	No	Yes	No	No
T	NOACCESS	No	0	No	Yes	No	No
U	NOACCESS	No	0	No	Yes	No	No
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
FTK	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTL	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTM	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTN	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTO	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTP	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTQ	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTR	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTS	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTT	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTU	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
NOK	ULONG	Yes	1	Yes	No	No	No
NOL	ULONG	Yes	1	Yes	No	No	No
NOM	ULONG	Yes	1	Yes	No	No	No
NON	ULONG	Yes	1	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
NOO	ULONG	Yes	1	Yes	No	No	No
NOP	ULONG	Yes	1	Yes	No	No	No
NOQ	ULONG	Yes	1	Yes	No	No	No
NOR	ULONG	Yes	1	Yes	No	No	No
NOS	ULONG	Yes	1	Yes	No	No	No
NOT	ULONG	Yes	1	Yes	No	No	No
NOU	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
OUTK	OUTLINK	Yes	0	No	No	N/A	No
OUTL	OUTLINK	Yes	0	No	No	N/A	No
OUTM	OUTLINK	Yes	0	No	No	N/A	No
OUTN	OUTLINK	Yes	0	No	No	N/A	No
OUTO	OUTLINK	Yes	0	No	No	N/A	No
OUTP	OUTLINK	Yes	0	No	No	N/A	No
OUTQ	OUTLINK	Yes	0	No	No	N/A	No
OUTR	OUTLINK	Yes	0	No	No	N/A	No
OUTS	OUTLINK	Yes	0	No	No	N/A	No
OUTT	OUTLINK	Yes	0	No	No	N/A	No
OUTU	OUTLINK	Yes	0	No	No	N/A	No
UFVA	STRING	Yes	Null	Yes	No	No	No
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

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
UFVK	STRING	Yes	Null	Yes	No	No	No
UFVL	STRING	Yes	Null	Yes	No	No	No
UFVM	STRING	Yes	Null	Yes	No	No	No
UFVN	STRING	Yes	Null	Yes	No	No	No
UFVO	STRING	Yes	Null	Yes	No	No	No
UFVP	STRING	Yes	Null	Yes	No	No	No
UFVQ	STRING	Yes	Null	Yes	No	No	No
UFVR	STRING	Yes	Null	Yes	No	No	No
UFVS	STRING	Yes	Null	Yes	No	No	No
UFVT	STRING	Yes	Null	Yes	No	No	No
UFVU	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	Yes/No	No
VALJ	NOACCESS	No	0	No	Yes	Yes/No	No
VALK	NOACCESS	No	0	No	Yes	Yes/No	No
VALL	NOACCESS	No	0	No	Yes	Yes/No	No
VALM	NOACCESS	No	0	No	Yes	Yes/No	No
VALN	NOACCESS	No	0	No	Yes	Yes/No	No
VALO	NOACCESS	No	0	No	Yes	Yes/No	No
VALP	NOACCESS	No	0	No	Yes	Yes/No	No
VALQ	NOACCESS	No	0	No	Yes	Yes/No	No
VALR	NOACCESS	No	0	No	Yes	Yes/No	No
VALS	NOACCESS	No	0	No	Yes	Yes/No	No
VALT	NOACCESS	No	0	No	Yes	Yes/No	No
VALU	NOACCESS	No	0	No	Yes	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

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
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
OVLK	NOACCESS	No	0	No	No	No	No
OVLL	NOACCESS	No	0	No	No	No	No
OVLM	NOACCESS	No	0	No	No	No	No
OVLN	NOACCESS	No	0	No	No	No	No
OVLO	NOACCESS	No	0	No	No	No	No
OVLP	NOACCESS	No	0	No	No	No	No
OVLQ	NOACCESS	No	0	No	No	No	No
OVLR	NOACCESS	No	0	No	No	No	No
OVLS	NOACCESS	No	0	No	No	No	No
OVLT	NOACCESS	No	0	No	No	No	No
OVLU	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
FTVJ	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVK	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVL	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVM	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVN	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVO	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVP	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVQ	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVR	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVS	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVT	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
FTVU	GBLCHOICE	Yes	DOUBLE	Yes	No	No	No
NOVA	ULONG	Yes	1	Yes	No	No	No
NOVB	ULONG	Yes	1	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
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
NOVK	ULONG	Yes	1	Yes	No	No	No
NOVL	ULONG	Yes	1	Yes	No	No	No
NOVM	ULONG	Yes	1	Yes	No	No	No
NOVN	ULONG	Yes	1	Yes	No	No	No
NOVO	ULONG	Yes	1	Yes	No	No	No
NOVP	ULONG	Yes	1	Yes	No	No	No
NOVQ	ULONG	Yes	1	Yes	No	No	No
NOVR	ULONG	Yes	1	Yes	No	No	No
NOVS	ULONG	Yes	1	Yes	No	No	No
NOVT	ULONG	Yes	1	Yes	No	No	No
NOVU	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
TOVK	ULONG	Yes	0	Yes	No	No	No
TOVL	ULONG	Yes	0	Yes	No	No	No
TOVM	ULONG	Yes	0	Yes	No	No	No
TOVN	ULONG	Yes	0	Yes	No	No	No
TOVO	ULONG	Yes	0	Yes	No	No	No
TOVP	ULONG	Yes	0	Yes	No	No	No
TOVQ	ULONG	Yes	0	Yes	No	No	No
TOVR	ULONG	Yes	0	Yes	No	No	No
TOVS	ULONG	Yes	0	Yes	No	No	No

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
TOVT	ULONG	Yes	0	Yes	No	No	No
TOVU	ULONG	Yes	0	Yes	No	No	No

1.3 Field Descriptions

Name	Summary	Description
VERS	Version number of the genSub record code	This field holds the version number of the genSub record code. The current version is 1.6 and applies to the EPICS release 3.14.
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,...,VALU. 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,...,VALU.
INPA,..., INPU	Input Link A,..., Input Link U	The input links from where the values of A,...,U are fetched during record processing.
UFA,..., UFU	User Function A User Function U	These are the names of functions which return the sizes of any user defined structures to be received in the input fields A,...,U.
A,...,U	Input Fields	The input fields which hold the scalar values or arrays fetched in across the input links INPA,...,INPU.

Name	Summary	Description
FTA,..., FTU	Field Type of A Field Type of U	Field types of the input values. These can be CHAR, STRING, DOUBLE, LONG, etc.
NOA..., NOU	Number of elements in A,.. Number of elements in U	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,..., OUTU	Output Link A,.. Output Link U	The output links on which the scalars or arrays located at VALA,...,VALU are placed during record processing.
UFVA,..., UFVU	User Function VALA,..., User Function VALU	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,...,VALU.
VALA,.., VALU	Output Fields	The output fields which hold the scalar values or arrays pushed out across the output links OUTA,...,OUTU.
FTVA,..., FTVU	Field Type of VALA Field Type of VALU	Field types of the output values. These can be CHAR, STRING, DOUBLE, LONG, etc.
OVLA,.., OVLU	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,..., NOVU	Number of elements in VALA Number of elements in VALU	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,..., TOVU	Total Number of bytes in VALA Total Number of bytes in VALU	The total number of bytes in each output field. These are used internally by record processing and do not concern the user.

1.4 Record Support Routines

1.4.1 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-UFU and UFVA-UFVU. 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-FTU and NOA-NOU. 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-FTVU and NOVA-NOVU. 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:

- Initializes SUBL if it is a constant link.
- Initializes each constant input 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.

1.4.2 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.

1.4.3 get_value

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

1.4.4 get_precision

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

1.4.5 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.

1.4.6 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.

1.4.7 put_array_info

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

1.4.8 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.

1.5 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 IOCs. 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.

1.6 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 <time.h>
#include <stdlib.h>
#include <stdio.h>

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

long setup( 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 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 IOCs, 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.

1.7 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 routine called during record processing is that specified in the SNAM field. Under these conditions, the SNAM field can be changed by a Channel Access write to that 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 used in the system, if the vxWorks *unld* command is never used.