DELPHI Collaboration

DELPHI 94–15 DAS 152
Revised 10th November 1994

Slow Control Database Manual

T.J. Adye, G. Gopal, G.R. Smith
Rutherford Appleton Laboratory
and
F. Fontanelli
INFN, Genoa

Abstract

This note describes the structure of the slow controls configuration database file, the format of the ASCII file used to create it, and the routines available for accessing the data.

An earlier version was issued as DELPHI note **88–86** DAS–89, *Storing Control Data for Slow-Control Channels*, on 8th December 1988.

Contents

1 Introduction

The Slow Controls Configuration Database, containing all data required for control and monitoring of the DELPHI detector partitions, is based on the DELPHI database management system (DBMS) — CARGO [1].

The slow controls (SC) monitoring data necessary for the offline reconstruction and analysis of event data [2] is considered here only in the definition of the correspondence between the two databases (see section 4). The problem of storage of general control and monitoring data (data logging) is not addressed in this document.

2 The Data

For each physical channel in a G64 crate, the slow controls data [3] falls into the following areas:

- Physical address, 3 words:
  1. G64 crate #
  2. Slot #
  3. Channel #

     Note: For CAEN high voltage channels (function flag = 5, below) the channel # is defined as follows:

     $$\text{channel \#} = 40 \times (\text{CAEN crate \#}) + (\text{channel \# in CAEN crate})$$

     with the CAEN crate # ranging from 0–99 and the channel # in each CAEN crate from 0–39, this gives a range of 0–3999 for the physical channel #.

- G64 crate and slot description, 7 words:
  1. Crate network address — 3 `CHARACTER*4` words.
  2. Slot card type — 3 `CHARACTER*4` words describing the type of card in a slot.
  3. Slot G64 bus address — 2 `INTEGER*4` words.

- Logical address, 2 words:
  1. SC subsystem name — `CHARACTER*4` — e.g. `VOLT`, `TEMP` etc. — names used by the individual detector groups.
  2. a logical channel # — integer, assigned sequentially from 1 to 9999 for a given SC subsystem.

- Function flag to indicate the function being performed by the channel. The flag values 0 to 10 are reserved for system-wide use:–

  0= Undefined function **or** dead/unused channel
  1= State monitoring
  2= Analog monitoring
  3= State setting
  4= Analog setting

5= CAEN high voltage channel
6= Analog setting & monitoring
7= Analog setting & monitoring discriminator channels

Other values can be assigned for channels with functions different from those described here. Users should consult the authors to define the specific record type and the data structure, so that specific modifications can be written into the software.

- Slow controls constants: The precise meaning of the values required by the EPs and G64s are given in [3]. The data pertaining to a given channel depends on its functions (defined by the function flag, above):

  1. State monitoring:
     (a) Desired state,
     (b) Error checking flag, and
     (c) Error routine # in G64.
  2. Analog monitoring:
     (a) Desired value,
     (b) Error limit,
     (c) Switch limit,
     (d) Error checking flag,
     (e) Gain value, and
     (f) Error routine # in G64.
  3. State setting: desired state.
  4. Analog setting: desired value.
  5. CAEN high voltage channels:
     (a) Setting & standby voltages,
     (b) Running, ramping & standby current limits,
     (c) Ramp rates (up & down)
     (d) Trip time.
     (e) Error checking flag, and
     (f) Error routine # in G64.
  6. Analog monitoring & setting: a combination of the data for analog monitoring & setting (items 2 and 4 above).
  7. Analog setting & monitoring discriminators channels: require
     (a) Six threshold values,
     (b) a dead time,
     (c) Monitoring value,
     (d) Currents,
     (e) Temperature, and
     (f) Status bits.

- Status Update Database file constants: this data is optional, depending on whether the actions/results of a given logical slow controls channel affect the functioning of one or more sensing device channels — and thus the offline analysis of event data. If required, the interfacing between the Slow Controls Configuration Database file and the Status Update Database file can be achieved if the following information is available:

  1. Status Update Database record identifier,
  2. The record type, and
  3. List of channel numbers of the sensing devices affected.

  See section 4 for details.

3 The Structure

The necessary information for all the SC parameters is stored in a single database file (called Slow Controls Configuration Database file) in identifiably different 'record types' — one for each distinct parameter. An example ASCII file, used to create this database file, is given in section 6.1.

In what follows, it might appear from the data record naming convention in CARGO that there is a hierarchical structure. However, the data records at any given level are independent of records at other levels except that all higher level data records in the tree have to exist on the database file before a data record at the current level can be entered. For read access CARGO gets the required data record at any level immediately — it does not have to read down the tree of data records to get to the one required.

All data is stored in a structure containing a single tree type called `SLOW` with `SCON` as the the root record type

Each detector has a single top-level data record. The name is built up from the the root record type (`SCON`) and a 4-character detector name). For example, the data record names for the TPC and the Outer Detector are:

```
SCON$TPC*
SCON$OD**
```

Note: The names used for the detectors correspond to those used in the Detector Description Database files [2].

This record has the following data fields:

`SPAR` a variable length field containing the detector-specific list of SC subsystem names.

`LISC` a variable length field containing the list of crate #s used in the slow controls subsystem for this detector. Each crate # is expected to be less than 9999.

`NAME` a character string describing the root level — ie. the detector name.

In the CARGO convention, the first word of each data field within a record is the word count. Any data word within a field, declared `INTEGER`, must be entered as an `INTEGER`.

3.1 Logical Addressing

Associated with each slow controls subsystem listed in the `SPAR` data field of the `SCON` record are three different items of data.

3.1.1 Subsystem Description

A textual description indicating what the physical aspect of the detector the subsystem name refers to. This is stored in a record (one per SC subsystem) of type `DESC`, which has one single field, called `TEXT`, containing the description as a character string. Examples of data record identifiers for this type are:

`SCON$OD**.DESC$VOLT` for voltages
`SCON$OD**.DESC$TEMP` for temperatures
`SCON$OD**.DESC$PRES` for gas pressures
etc. . .

3.1.2 Standard Constants

A set of standard constants that may apply to all the logical channels of this type. These are stored in a single variable length field of `REAL*4` words (called `STAN`) in a data record (one per SC subsystem) of type — `STCO` e.g.

`SCON$OD**.STCO$VOLT` for voltages
`SCON$OD**.STCO$TEMP` for temperatures
`SCON$OD**.STCO$PRES` for gas pressures
etc. . .

3.1.3 Logical Channels Data

To relate the logical channels to their physical addresses, a record (one per SC subsystem) of type `LTOP` is used. Examples record identifiers are:

`SCON$OD**.LTOP$VOLT` for voltages
`SCON$OD**.LTOP$TEMP` for temperatures
`SCON$OD**.LTOP$FB**` for fastbus crate power supplies
`SCON$OD**.LTOP$PRES` for gas pressures
etc. . .

The logical-to-physical address correspondence table is stored in a single field, called `CORR`, containing 4 `INTEGER*4` words per logical channel:

1. G64 crate #,
2. Slot #,
3. Physical channel #, and
4. Function flag.

where the 1st four such words describe logical channel 1, the 2nd four are for channel 2, etc. There should be no gaps in the list, but unused logical channel numbers can be indicated with a function flag of 0.

A second data field, `USER`, is intended for user-specific data (`REAL*4`).

For each logical channel there is one data record required — its type depends on the function flag. There are 5 distinct types of records, `SETD`, `MOND`, `CAEN`, `ASMD`, and `DISC` (`ASMD` and `DISC` are not currently supported by the standard EP or G64 Skeleton), which are detailed here.

3.1.3.1 Control channels are described by the SETD (for **SET**ting **D**ata), branch to the LTOP record type (function flag = 3 or 4).

There will be one data record for each logical channel. Example control channel data record identifiers are:

SCON$OD**.LTOP$VOLT.SETD$nnnn for voltages
SCON$OD**.LTOP$FB**.SETD$nnnn for fastbus crate power supplies
etc...

where nnnn is the logical channel number. This allows for 9999 channels of a given type for each detector. If not sufficient, nnnn can be replaced by nnnnn giving a limit of 99999.

To make it applicable to both digital & analog setting channels the following fields have been defined for the SETD record type. Inapplicable fields (eg. DIGS for analog channels) can be left out.

DIGS a variable length field for state-setting data with the first 2 words defined — the desired state and a standard test state, both INTEGER*4 and each restricted to values 0 (= OFF) and 1 (= ON) only. These may be followed by user-specific INTEGER*4 words.

ANAS a variable length field for analog setting data with the first 2 words defined — the desired value and a standard test value, both REAL*4. These may be followed by user-specific REAL*4 words.

CFAC a variable length field containing the analog to digital REAL*4 conversion factors (required if it is an analog setting channel).

CARI field with strings of CHARACTER*4 words indicating the identifiers of records in the Status Update Database. Each string is ended by a CHARACTER*4 word which is either SENS or GLOB.

For example. suppose a given analog setting channel of the Outer Detector sets the voltages of two planks (1 & 2) and that one needs to record these settings as well as the status of the wires in these planks, then the following CARI field is required:–

```
*DBFIELD CARI
9, OD**, PL01, GLOB,
   OD**, PL02, GLOB,
   OD**, WIRE, SENS
```

SCGL field (only needed if the list of record identifiers in the CARI field contains at least one GLOB record type) with the locations in the SLOW field of the GLOB record(s) which need to be updated — e.g. for the above CARI field, this field could be:–

```
*DBFIELD SCGL
4           ! Total number of words following.
1           ! # of locations to update in SLOW field of GLOB$OD**.GLOB$PL01.
8           ! Location for the channel voltage value.
1           ! # of locations to update in SLOW field of GLOB$OD**.GLOB$PL02.
8           ! Location for the channel voltage value.
```

This field is required only if

1. the list of Status Update Database record identifiers in the CARI field above contains a GLOB record (as in the example above), **and**
2. the values of the slow controls subsystem being monitored are required for offline analysis of event data and have thus to be updated.

Note: If this field is absent **or** the first location indicated is equal to zero, then it is assumed that the controlling Elementary Process (SC EP) is to update the 'global' status (first word in the LEAD field of the GLOB record).

SCSD field (required if the list of records in the CARI field contains at least one SENS record type) with the channel numbers of the affected sensing devices whose status is altered by this SC channel — e.g. for the above CARI field, this field could be:–

```
*DBFIELD  SCSD
   3          ! Total number of words following.
   2          ! Number of Wires affected in GLOB$OD**.SENS$WIRE.
  51          ! Channel number of 1st wire.
 151          ! Channel number of 2nd wire.
```

The statuses are written to the STAT field in the GLOB$OD**.SENS$WIRE record. Unless this is stored in packed format, the sensing device channel numbers given are simply the word number within this field.

The SCSD field is required only if

1. the list of Status Update Database record identifiers in the CARI field above contains a SENS record, **and**
2. this SC channel affects any entity in the Status Update Database.

COMM an optional variable length field containing the comments (if any) in a string of CHARACTER*4 words. The standard SC EP uses this as a channel name.

USER an optional variable length field containing REAL*4 words that only a particular user is able to interpret. For example data for a user specified channel type/function flag could be stored in this field.

3.1.3.2 Monitoring channels are described by the MOND (for **MON**itoring **D**ata), branch to the LTOP record type (function flag = 1 or 2).

There will be one data record for each logical channel. Example monitoring channel data record identifiers are:

SCON$OD**.LTOP$VOLT.MOND$nnnn for voltages
SCON$OD**.LTOP$TEMP.MOND$nnnn for temperatures
SCON$OD**.LTOP$FB**.SETD$nnnn for fastbus crate power supplies
SCON$OD**.LTOP$PRES.MOND$nnnn for gas pressures
etc...

where nnnn is the logical channel number as before.

To make it applicable for digital & analog monitoring channels the following fields have been defined for the SETD record type. Inapplicable fields (eg. ANAM for digital channels) can be left out.

DIGM a variable length field for state-monitoring data with the first 3 words defined as follows:

Table 1: Basic SC Record MOND — Field DIGM		
Field Title: State-monitoring Constants		
Word	FMT	Description
1	I	Desired state (0=OFF & 1=ON)
2	I	Error checking flag (0=NO & 1=YES)
3	I	Error routine number in G64

If extra (INTEGER*4) words are required, these are entered as words 4 onwards and are user-specific.

ANAM a variable length field for analog monitoring data with the first 6 words defined as follows:

Table 2: Basic SC Record MOND — Field ANAM		
Field Title: Analog Monitoring Constants		
Word	FMT	Description
1	R	Desired value
2	R	Error limit
3	R	Switch limit
4	I	Error checking flag (0=NO & 1=YES)
5	R	Gain value (0.0, 1.0, 10.0, or 100.0)
6	I	Error routine number in G64

These may be followed by user-specific words (all REAL*4).

CFAC a variable length field containing the digital to analog REAL*4 conversion factors (required if it is an analog monitoring channel).

TORA an optional variable length field with the following first 2 words defined:

Table 3: Basic SC Record MOND — Field TORA		
Field Title: Monitoring Rate & Tolerance Limit		
Word	FMT	Description
1	R	Monitoring interval (in seconds)
2	R	Tolerance limit on desired value

These may be followed by user-specific words (all REAL*4).

Note 1: The monitoring interval is for use by the SC Elementary Process to interrogate the G64 program for current value of the parameter.

Note 2: The tolerance limit is intended to decide when the monitored value should be updated on the Status Update Database for offline analysis of event data.

CARI as in SETD record type.

SCGL as in SETD record type.

SCSD as in SETD record type.

COMM as in SETD record type.

USER as in SETD record type.

3.1.3.3 CAEN channels are described by the CAEN (for **CAEN** high voltage channel data), branch to the LTOP record type (function flag = 5).

For each logical channel there is one data record An example CAEN channel data record identifier is:

> SCON$OD**.LTOP$VOLT.CAEN$nnnn for high voltages
> etc. . .

where nnnn is the logical channel number as before.

The following fields have been defined for the CAEN record type.

CANS a variable length field with the first 12 words defined as follows:

Table 4: Basic SC Record CAEN — Field CANS		
Field Title: CAEN Analog Setting Constants		
Word	FMT	Description
1	R	Set voltage (V0 in Volts)
2	R	Running current limit (I0 in μA)
3	R	Ramping current limit (I0R)
4	R	Standby set voltage (V0S)
5	R	Standby current limit (I0S)
6	R	Ramp-up rate (Volts/sec) (RUP)
7	R	Ramp-down rate (RDW)
8	R	Trip time (TRIP in 0.1sec)
9	R	Set voltage for VSEL=1 (V1)
10	R	Running current limit for ISEL=1 (I1)
11	R	Ramping current limit for ISEL=1 (I1R)
12	R	Standby current limit for ISEL=1 (I1S)

Any words present in this field beyond the 12 described above are assumed to be user-specific and only the user(s) who provided the data knows what to do with them.

CANM a variable length field with the first two words as follows:

Table 5: Basic SC Record CAEN — Field CANM		
Field Title: CAEN Analog Monitoring Constants		
Word	FMT	Description
1	I	Error checking flag (0=OFF, 1=ON)
2	I	Error routine number in G64

Any words present in this field beyond the 2 described above are assumed to be user-specific and only the user(s) who provided the data knows what to do with them.

TORA an optional variable length field with the following first 3 words defined:

Table 6: Basic SC Record CAEN — Field TORA		
Field Title: Monitoring Rate & Tolerance Limit		
Word	FMT	Description
1	R	Monitoring interval (in seconds)
2	R	Tolerance limit on voltage
3	R	Tolerance limit on current

These may be followed by user-specific words (all REAL*4).

Note: The monitoring interval & tolerance values are for similar use as in MOND record types.

CARI as in SETD record type.

SCGL similar to the SCGL field in the SETD record type, but here both voltage and current monitored values can be specified. For example, with the appropriate record identifiers specified in the CARI field, we could have a SCGL field of

```
*DBFIELD SCGL
6                  ! Total number of words following.
2                  ! # of locations to update in SLOW field of GLOB$OD**.GLOB$PL01.
1                  ! Location for the voltage value.
2                  ! Location for the current value.
2                  ! # of locations to update in SLOW field of GLOB$OD**.GLOB$PL02.
1                  ! Location for the voltage value.
2                  ! Location for the current value.
```

SCSD as in SETD record type.

COMM as in SETD record type.

USER as in SETD record type.

3.1.3.4 Analog setting and monitoring channels are described by the ASMD (for **A**nalog **S**etting & **M**onitoring **D**ata), branch to the LTOP record type (function flag = 6).

There will be one data record for each logical channel. An example discriminator channel data record identifier is:

SCON$OD**.LTOP$VOLT.ASMD$nnnn for voltages
etc...

where nnnn is the logical channel number as before.

Channels of this type are not currently supported by the standard SC EP or G64 Skeleton. The following fields have been defined for the ASMD record type (both ANAS and ANAM should be specified).

ANAS as in SETD.

ANAM as in MOND.

CFAD a variable length field containing the analog to digital REAL*4 conversion factors.

CFDA a variable length field containing the digital to analog REAL*4 conversion factors.

TORA as in MOND record type.

CARI as in SETD record type.

SCGL as in SETD record type.

SCSD as in SETD record type.

COMM as in SETD record type.

USER as in SETD record type.

3.1.3.5 Discriminator channels are described by the DISC (for **DISC**riminator channels) with both analog setting & monitoring functions, branch to the LTOP record type (function flag = 7).

There will be one data record for each logical channel. Example discriminator channel data record identifiers are:

SCON$OD**.LTOP$DSCB.DISC$nnnn for voltages
etc. . .

where nnnn is the logical channel number as before.

Channels of this type are not currently supported by the standard SC EP or G64 Skeleton. The following fields have been defined for the DISC record type.

DSCS a variable length field for analog setting data with the first 14 words defined as follows:

Table 7: Basic SC Record DISC — Field DSCS

Field Title: Analog Discriminator Setting Constants

Word	FMT	Description
1	R	Threshold 1 ($0.0 < T_1 < 255.0$)
2	R	Threshold 2 ($0.0 < T_2 < 255.0$)
3	R	Threshold 3 ($0.0 < T_3 < 255.0$)
4	R	Threshold 4 ($0.0 < T_4 < 255.0$)
5	R	Threshold 5 ($0.0 < T_5 < 255.0$)
6	R	Threshold 6 ($0.0 < T_6 < 255.0$)
7	R	Dead time ($0.0 < t < 255.0$)
8	R	Monitoring channel ($0.0 < M_{\text{ch}} < 48.0$)
9	R	Monitoring value ($0.0 < M_{\text{va}} < 255.0$)
10	R	Preamp power (P_{st}: 0.0=OFF & 1.0=ON)
11	R	Discriminator status (STAT: 0.0=error & 1.0=OK)
12	R	Max. current V^+ (I_p in Amps.)
13	R	Max. current V^- (I_m in Amps.)
14	R	Max. temperature (TEMP in Celsius)

DSCM a variable length field for monitoring data with the first word (`INTEGER*4`) defined as the error checking flag. This may be followed by user-specific words (all `INTEGER*4`) from the second word onwards.

TORA as in `MOND` record type.

CARI as in `SETD` record type.

SCGL as in `SETD` record type.

SCSD as in `SETD` record type.

COMM as in `SETD` record type.

USER as in `SETD` record type.

3.2 Physical Addressing

Physical addressing is based on the G64 crate, slot, and channel numbers. It forms a parallel tree to the logical description described above.

3.2.1 G64 Crate Description

Associated with each physical crate number listed in the `LISC` data field of the `SCON` record is a single data record of type `CRAT`, eg.

```
SCON$OD**.CRAT$cccc
```

where `cccc` is the crate #, allowing up 9999 crates in the SC system of a given detector (Outer Detector in the example above).

This record type has four data fields:

NETW a variable length field containing the crate's RPC address as a character string (up to 40 characters may be specified). An example is:

```
0@SKEL_17@08003000101D.IEEE
```

CHAN a variable length field containing 2 `INTEGER` words per active slot — the physical slot number and the maximum number of channels connected (ie. one more than the last channel number, since the physical channel numbers are numbered from zero). A maximum of 999 channels can be connected to any slot.

CARD a variable length field containing 3 `CHARACTER*4` words (10 characters maximum) per active slot describing the card type in the slot. The names of card types supported by the standard G64 Skeleton are described in section 3.2.1 of [3].

G64B a variable length field with 2 bitted (hexadecimal) words per active slot indicating the G64 bus addresses of the card in the slot (these are the 'base address' and 'auxiliary address').

3.2.2 Slot Description

For each slot in a given crate there is one data record containing the physical-to-logical address correspondence table for all channels in the slot. The record type is `SLOT`, giving record identifiers such as

```
SCON$OD**.CRAT$cccc.SLOT$SLss
```

where `cccc` is the crate # and `ss` is the (2-digit) slot #, allowing up 99 slots in a crate (physically, 32 is the maximum).

This record type has a single data field:

PTOL a variable length field with, for each channel, the following 4 words:

1. Physical channel #
2. SC subsystem name # (cf. list in `SCON` record)
3. Logical channel #
4. Channel function flag

4 Interfacing Slow Controls to the Status Update Database

The Status Update Database file is one part of the Detector Description Database [2] — the others being the Calibration and Geometry files.

The slow controls is responsible for updating the Status Update Database with

1. Physical information, such as temperatures, pressures etc. **and**
2. the status of detectors & their subelements

if this information is required for the offline analysis of event data.

To allow the SC Elementary Process do this, the information allowing a translation between each slow controls logical channel and the equivalent sensing devices in the Status Update Database should be included in the Slow Controls Configuration Database.

For each logical channel the slow controls data record (of type `SETD`, `MOND`, `ASMD`, `CAEN`, or `DISC`) should contain the following fields:

`CARI` A Status Update Database record identifier field, which is a list of `CHARACTER*4` names to allow the appropriate record identifier to be built-up. An example is

No. of words:	`4`
Word 1:	`OD**` — Top level name
Word 2:	`SEG1` — 2nd level name
Word 3:	`WIRE` — 3rd & lowest level name
Word 4:	`SENS` — Record type at lowest level.

From this information the SC Elementary Process will build the identifier of the appropriate Status Update Database record — viz.

```
GLOB$OD**.GLOB$SEG1.SENS$WIRE
```

Several such lists may be included in one `CARI` field, allowing several Status Update Database records to updated based on a single SC channel. The separate Status Update Database record identifiers are delimited by the final word of each, which gives the record type, and must be either `GLOB` or `SENS`.

If the Status Update Database record identifier is of type `GLOB`, the implication is that the SC EP is going to

1. update the detector subelement status word, which is the first word in the `LEAD` data field; **and/or**
2. update the `SLOW` data field of this record with monitoring information required for event analysis. For this the SC EP must be given the location of the relevant data word in the `SLOW` field. This location is given in the `SCGL` data field (see below).

If the Status Update Database record identifier is of type `SENS`, then the SC EP will set the status words of each of the sensing device channels based on the status of the given SC channel. For this it is necessary to have a list of the affected sensing device channel numbers — see field `SCSD` below.

SCGL Location(s) of the data word to be updated in the `SLOW` field of the `GLOB` record indicated by the `CARI` field above.

Note: this field need only be given if

1. the last word in the `CARI` is `GLOB`, **and**
2. the current value of the SC subsystem is required to be recorded for offline analysis of event data.

SCSD List of affected sensing device channel numbers, as they are defined in the Status Update Database record indicated by the `CARI` field above. From this information the SC EP will build-up and write a new `STAT` field for the Status Update Database record concerned.

This data field need only be entered if the record type indicated in the last word of the `CARI` field is `SENS`.

Details of the specification of these fields are given in section 3.1.3.1.

Note: These data fields are only required if the Status Update Database needs to be updated. For SC channels which are for monitoring the health/safety of the detector only (ie. produce no information required for offline analysis of event data) these fields are not needed.

To provide the data indicated above it is clearly important that the slow controls responsable be in **close contact** with the Status Update Database file responsable.

The standard SC EP makes use of this information, linking with DDAP (online detector description access package), to read the relevant information from the Status Update Database and write the necessary update record(s).

In setting a status word for either a sensing device channel (`STAT` field) or a detector subelement (`LEAD` field), the SC EP uses the following convention:

Status word = 0, ON
= 1, OFF

5 Data Access — SCAP

CARGO [1] provides the necessary Fortran-callable access routines, the most commonly used being:

DBINIT to initialize CARGO;

DBGETF to fetch a given named data field in an existing record;

DBOPER to initialize an empty record buffer for `DBNEWF`;

DBNEWF to add a field into a record buffer;

DBNEWR to write the record buffer to the database file; and

DBEND to close the database file.

However, it is much more convenient to use a purpose-written access package, called SCAP (for slow controls (database) application package) to access the slow controls configuration data. The SCAP routines, listed below, provide the required functionality to SC programs by using the above mentioned CARGO routines.

Unlike the CARGO routines themselves, it is possible to access the SCAP routines via RPC, allowing the entire slow controls configuration to be saved on a single database file. While this is certainly necessary for the Status Update Database (which is accessed by the online detector description application package, DDAP), at present it has seemed simpler for the application programs (eg. the SC Elementary Process) to access the slow controls configuration data by direct calls to SCAP routines.

Apart from the bureaucratic routines (`SC_INIT`, `SC_CLOSE`, and `SC_CHECK` — see sections 5.1 to 5.3), there are two methods of using SCAP. One can either call `SC_PUT_DETECTOR_INFO`, `SC_GET_DETECTOR_INFO`, `SC_PUT_PARA_DESC`, etc (sections 5.4 to 5.17), using SCAP to access the correct record/field and to format the field data, or one can use `SC_PUT_SLOW_FIELD` and `SC_GET_SLOW_FIELD` (sections 5.18 and 5.19) to access the SC Database fields directly, with the formatting of the field data left to the caller. The SC Elemementary Process now uses this latter method.

The SCAP/CARGO system automatically timestamps all updates, so that the configuration at any past time can be recovered if so desired.

Each SCAP routine can be called as a function and will make use of the VAX/VMS convention for error reporting (see section 5.20). An odd return value indicates successful completion.

5.1 SC_INIT

Routine to

1. register the calling process for a given detector (or all detectors), with appropriate read and/or write access, **and**
2. check that the database file has a complete and consistent set of data records for the given detector for all SC subsystems or for a named SC subsystem.

The calling sequence is as follows:

```
IERROR = SC_INIT (PASWORD,DETNAME,SUBSYS,DEBUG)
```

Input:

PASWORD	Database file password
DETNAME	Detector name (CHARACTER*4)
	'****' = all detectors
SUBSYS	Subsystem name (CHARACTER*4)
	'****' = all subsystems
	' ' = no check
DEBUG	CHARACTER*3 variable ('ON' or 'OFF') for debug output

Output:

IERROR	Error flag

5.2 SC_CLOSE

Routine to de-register the calling process. Once a process has finished all requests (read/write) for slow controls data a call to SC_CLOSE is essential.

The calling sequence is as follows:

```
IERROR = SC_CLOSE ()
```

Output:

IERROR	Error flag

5.3 SC_CHECK

Routine to check the consistency of the SC database file for a given detector (or all detectors) and a given subsystem (or all). It normally called by `SC_INIT`.

The calling sequence is as follows:

```
IERROR = SC_CHECK (DETNAME,SUBSYS)
```

Input:

`DETNAME`	Detector name (`CHARACTER*4`)
	`'****'` = all detectors
`SUBSYS`	Subsystem name (`CHARACTER*4`)
	`'****'` = all subsystems

Output:

`IERROR`	Error flag

5.4 SC_PUT_DETECTOR_INFO

Routine to write the appropriate SCON record for the given detector. The calling sequence is:

```
IERROR = SC_PUT_DETECTOR_INFO (DETNAME,NPARA,PARAS,
                    NCRATS,CRATES,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
NPARA	Number of SC parameter types
PARAS	List (CHARACTER*4) of SC parameter types
NCRATS	Number of crates
CRATES	List of physical crate #s
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.5 SC_GET_DETECTOR_INFO

Routine to fetch the list of SC subsystem names and the list of physical crate #s for the given detector. The calling sequence is:

```
IERROR = SC_GET_DETECTOR_INFO (DETNAME,NMXPAR,NMXCRAT,
           VDTIME,NPARA,PARAS,NCRATS,CRATES)
```

Input:

DETNAME	Detector name (CHARACTER*4)
NMXPAR	Maximum number of SC subsystems expected
NMXCRAT	Maximum number of crates expected
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NPARA	Number of SC parameter types; if NPARA $>$ NMXPAR, first NMXPAR names returned
PARAS	List (CHARACTER*4) of SC parameter types
NCRATS	Number of crates; if NCRATS $>$ NMXCRAT, first NMXCRAT #s returned
CRATES	List of physical crate #s
IERROR	Error flag

5.6 SC_PUT_PARA_DESC

Routine to write the appropriate DESC record for a specified slow controls subsystem in a given detector. The calling sequence is:

```
IERROR = SC_PUT_PARA_DESC (DETNAME,SCPARA,NDLEN,
                          DESCR,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
NDLEN	Number of characters in description string
DESCR	Description string
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.7 SC_GET_PARA_DESC

Routine to fetch the appropriate DESC record for a specified slow controls subsystem in a given detector. The calling sequence is:

```
IERROR = SC_GET_PARA_DESC (DETNAME,SCPARA,MAXLEN,
                     VDTIME,NDLEN,DESCR)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
MAXLEN	Maximum length of description string
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NDLEN	Actual length of description string
DESCR	Description string
IERROR	Error flag

5.8 SC_PUT_PARA_STCO

Routine to write the appropriate STCO record for a specified slow controls subsystem in a given detector. The calling sequence is:

```
IERROR = SC_PUT_PARA_STCO (DETNAME,SCPARA,NWDS,
                          STCON,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
NWDS	Number of standard constants
STCON	REAL*4 array of standard constants
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.9 SC_GET_PARA_STCO

Routine to fetch the appropriate STCO record for a specified slow controls subsystem in a given detector. The calling sequence is:

```
IERROR = SC_GET_PARA_STCO (DETNAME,SCPARA,NMXWD,
                          VDTIME,NWDS,STCON)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
NMXWD	Maximum number of constants expected
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NWDS	Actual number of standard constants
STCON	REAL*4 array of standard constants
IERROR	Error flag

5.10 SC_PUT_LOG_PHYS

Routine to write the logical–physical address table in the appropriate LTOP record for a specified SC parameter type for the given detector. The information required to write the LTOP record is used to build up the SLOT record. This is compared with the existing SLOT record and, if necessary, an update SLOT record is witten. The calling sequence is:

```
IERROR = SC_PUT_LOG_PHYS (DETNAME,SCPARA,LOGCHAN,
                  LPHYADR,LFUNC,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
LOGCHAN	Number of channels
LPHYADR	INTEGER*4 (3,LOGCHAN) array containing the physical addresses for all logical channels
LFUNC	INTEGER*4 (LOGCHAN) array containing the function flags for each logical channel
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.11 SC_GET_LOG_PHYS

Routine to fetch the logical–physical address table for a specified SC parameter type for the given detector. The calling sequence is:

```
IERROR = SC_GET_LOG_PHYS (DETNAME,SCPARA,NMAXCH,
            VDTIME,LOGCHAN,LPHYADR,LFUNC)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
NMAXCH	Expected number of logical channels
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

LOGCHAN	Actual number of logical channels; if LOGCHAN > NMAXCH, first NMAXCH logical addresses and function flags returned
LPHYADR	Array (3,NMAXCH) containing the physical addresses for all logical channels
LFUNC	Array (NMAXCH) containing the function flags for each logical channel
IERROR	Error flag

5.12 SC_PUT_CHAN_FIELD

Routine to write a specific field in the appropriate data record (SETD, MOND, CAEN, ASMD, or DISC) for a given logical channel for the given detector. The calling sequence is:

```
IERROR = SC_PUT_CHAN_FIELD (DETNAME,SCPARA,LOGCHAN,
                FLDNAM,NWDF,FDARAY,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
LOGCHAN	Logical channel #
FLDNAM	Field name (CHARACTER*4)
NWDF	Number of data words in the field
FDARAY	Array (REAL*4) with field data
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.13 SC_GET_CHAN_FIELD

Routine to fetch a specific field in a logical channel data record for a given logical channel for the given detector. The calling sequence is:

```
IERROR = SC_GET_CHAN_FIELD (DETNAME,SCPARA,LOGCHAN,
         FLDNAM,NWMAXF,VDTIME,NWDF,FDARAY)
```

Input:

DETNAME	Detector name (CHARACTER*4)
SCPARA	SC parameter name (CHARACTER*4)
LOGCHAN	Logical channel #
FLDNAM	Field name (CHARACTER*4)
NWMAXF	Max length of field
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NWDF	Number of data words in field
FDARAY	Array (REAL*4) with field data
IERROR	Error flag

5.14 SC_GET_CHAN_DATA

Routine to fetch the setting and/or monitoring data, the comments and the user specific data for a given logical channel for the given detector. The calling sequence is:

```
IERROR = SC_GET_CHAN_DATA (DETNAME,SCPARA,LOGCHAN,
    VDTIME,NMAXR,RECORD,LOCFLDS,LENFLDS,LFUNC)
```

Input:

`DETNAME`	Detector name (`CHARACTER*4`)
`SCPARA`	SC parameter name (`CHARACTER*4`)
`LOGCHAN`	Logical channel #
`VDTIME`	Validity date & time for data; if (0,0) current date & time are used
`NMAXR`	Maximum length of `RECORD`

Output:

`RECORD`	Output buffer
`LOCFLDS`	Array (`INTEGER*4`) with pointers to each field as given by CARGO
`LENFLDS`	Array (`INTEGER*4`) with length of each field; if ≤ 0, field is missing
`LFUNC`	Function flag for the channel
`IERROR`	Error flag

Note: The data fields are returned in the order specified in section 3.1.3, and depend on the value of the function flag for the channel.

`LOCFLDS` and `LENFLDS` currently cater for up to 15 data fields in a given channel record.

5.15 SC_GET_PHYS_LOG

Routine to fetch the physical–logical address table for all channels in a specified slot of a given crate for the given detector. The calling sequence is:

```
IERROR = SC_GET_PHYS_LOG (DETNAME,CRATNO,SLOTNO,
              NMAXCH,VDTIME,NCHAN,ITABLE)
```

Input:

DETNAME	Detector name (CHARACTER*4)
CRATNO	Physical crate #
SLOTNO	Physical slot #
NMAXCH	Number of channels in slot
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NCHAN	Actual number of channels in slot; if NCHAN > NMAXCH, first NMAXCH logical addresses returned
ITABLE	Array (4,NMAXCH) containing 4 words for each physical channel (see PTOL in sec 3.2) in the slot
IERROR	Error flag

5.16 **SC_PUT_CRATE_INFO**

Routine to write the appropriate CRAT record for a specified crate in the given detector. The calling sequence is:

```
IERROR = SC_PUT_CRATE_INFO (DETNAME,CRATNO,NETADDR,
    NACTSL,NOSLOT,ICHANOS,SLCARD,G64BAD,VDTIME)
```

Input:

DETNAME	Detector name (CHARACTER*4)
CRATNO	Physical crate #
NETADDR	Network address (CHARACTER*40)
NACTSL	Number of active slots in crate
NOSLOT	INTEGER*4 (32) array containing the physical slot #s of the active slots
ICHANOS	INTEGER*4 (32) array containing the number of channels in each active slot
SLCARD	CHARACTER*12 (32) array containing the card type information for each active slot
G64BAD	Array (INTEGER*4 (2,32)) containing the G64 bus addresses of each active slot
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.17 SC_GET_CRATE_INFO

Routine to fetch the crate's network address and the list of card types in the 32 slots in a given crate. The calling sequence is:

```
IERROR = SC_GET_CRATE_INFO (DETNAME,CRATNO,VDTIME,
   NETADDR,NACTSL,NOSLOT,ICHANOS,SLCARD,G64BAD)
```

Input:

DETNAME	Detector name (CHARACTER*4)
CRATNO	Physical crate #
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

NETADDR	Network address (CHARACTER*40)
NACTSL	Number of active slots in crate
NOSLOT	INTEGER*4 (32) array containing the physical slot #s of the active slots
ICHANOS	INTEGER*4 (32) array containing the number of channels in each active slot
SLCARD	CHARACTER*12 (32) array containing the card type information for each active slot
G64BAD	Array (INTEGER*4 (2,32)) containing the G64 bus addresses of each active slot
IERROR	Error flag

5.18 SC_PUT_SLOW_FIELD

Routine to write a field to the SC Database. Specification of the record and field identifiers, and filling of the data (see section 3) buffer are left to the caller. The calling sequence is:

```
IERROR = SC_PUT_SLOW_FIELD (ELEMS,NELEMS,FLDNAM,
                   FDARAY,NWDF,VDTIM)
```

Input:

ELEMS	CHARACTER*4 array with detector name and all elements of record identifier (eg. 'OD**','LTOP','TEMP','MOND','0024')
NELEMS	Number of words in ELEMS
FLDNAM	Field name (eg. 'ANAM')
FDARAY	Field data. Can be of any type, but must be an integral number of 4-byte words long. CHARACTER data must be passed by reference.
NWDF	Number of 4-byte words in FDARAY
VDTIME	Validity date & time for data; if (0,0) current date & time are used

Output:

IERROR	Error flag

5.19 **SC_GET_SLOW_FIELD**

Routine to read a field from the SC Database. Specification of the record and field identifiers, and interpretation of the data (see section 3) are left to the caller. The calling sequence is:

```
IERROR = SC_GET_SLOW_FIELD (ELEMS,NELEMS,FLDNAM,VDTIM,
                  NWMAXF,FDARAY,NWDF,PERIOD)
```

Input:

ELEMS	CHARACTER*4 array with detector name and all elements of record identifier (eg. 'OD**','CRAT','0002')
NELEMS	Number of words in ELEMS
FLDNAM	Field name (eg. 'NETW')
VDTIME	Validity date & time for data; if (0,0) current date & time are used
NWMAXF	Maximum number of words to be filled into FDARAY

Output:

FDARAY	Field data. Can be of any type, but must be an integral number of 4-byte words long. CHARACTER data must be passed by reference.
NWDF	Number of 4-byte words returned in FDARAY; if NWDF $>$ NWMAXF, first NWMAXF words returned
PERIOD	Validity period (start & end date & time: REAL*4 (4))
IERROR	Error flag

5.20 SCAP Error List

In all these SCAP routines a common error scheme has been adopted. The possible symbolic error codes are as follows:

```
.title          SCAPERR Error messages for SCAP
.ident          'V3.03'
.facility       SCAP, 3 /PREFIX=SCAP__
.base           1793                    ! 701 hex

.severity       SUCCESS
SUCCESS         "operation correctly completed"

.severity       INFORMATION
READONLY        "read access only is granted"
CONSCHKF        "consistency check failed"
NOINITCALL      "SC_INIT not yet called"
NOACTPROC       "no SCAP process initialised yet"

.severity       ERROR
WRONGARG        "wrong arguments in calling sequence"
WRONGDATA       "data outside allowed range for this field"
ILLFUNFL        "illegal function flag (<0 or >7)"
NULLFUNFLG      "function flag set to 0"
NPARGTNMXP      "NPARA > NMXPAR"
NCRAGTNMXC      "NCRATS > NMXCRAT"
BOTHGTNMX       "both NPARA>NMXPAR & NCRATS>NMXCRAT"
RECNOTFOUND     "data record not found"
RECORDALREXST   "record already exists"
NCHGTNMXCH      "NCHAN > NMAXCH"
NMXCHTOOBIG     "NMAXCH > 2000: record too big, call expert"
UNKNOSCPAR      "unknown SC parameter name"
UNKNOFIELD      "unknown field name"
MISSEDFIELD     "record doesn't contain the wanted field"
LOGCGTNMXCH     "LOGCHAN > NMAXCH"
NDLGTMAXL       "NDLEN > MAXLEN"
NWDSGTNMX       "NWDS > NMXWD"
INVCHAN         "invalid channel"
UNKNODETEC      "unknown detector name"
NOTEXSTCRT      "crate number doesn't exist"
UNKNOPROC       "unknown process"

.severity       FATAL
INTBUFFOVFL     "internal buffer overflow"
NOROOTREC       "no root record for given date and time"
UNKNOROOT       "unknown root"
TOOACTPROC      "too many active processes; ie. >20"
MEMOVRWR        "memory overwritten?"
```

6 Detector-dependent ASCII Input Files

The format/structure of an ASCII input file acceptable to CARGO is described in the CARGO Manual [1]. Each detector group should provide an ASCII file containing the required data for each SC channel in the detector.

6.1 Example Slow Controls Configuration Database ASCII File

We present here a short example ASCII file. It describes a fictitious detector (named `TIM`) with 4 CAEN high voltage channels and 2 fastbus control channels. A fuller example may be found on the online cluster in
`DELPHI$ONLINE:[SLOW_CONTROL.ELEMENTARY_PROCESS.EXAMPLES.TIM]TIM_SCON.ASC`

```
!     Example Slow Controls Configuration Database ASCII File
!     ======= ==== ======== ============= ======== ===== ====
!
! Start of tree SCON$TIM*
! ===== == ==== =========
!
*DBRECORD
SLOW SCON$TIM*.B
0,0,910702,110000
*DBFIELD SPAR            ! SC parameters for this detector
2,
    HV**
    FB**
*DBFIELD LISC            ! List of crates
1,
    1
*DBFIELD NAME (I5,/,4X,(20A4))
5,
    Tim's test detector
!
! MAC-64 crate and cards
! ====== ===== === =====
!
*DBRECORD
SLOW SCON$TIM*.CRAT$0001.B
0,0,910702,110000
*DBFIELD NETW (I2,/,4X,8A4) ! Crate RPC address (includes ethernet addr)
 7,
    0@SKEL_07@080030001009.IEEE
*DBFIELD CHAN               ! 2 slots; slot number, max chans in slot
4,
     7, 40
     9, 32
*DBFIELD CARD (I2/(4X,3A4)) ! Which sort of cards in these slots
```

```
6,
    caen
    relay_32fb
*DBFIELD G64B               ! G64 bus address for each slot
4,
    E100, 0000
    E220, 0000
!
! Physical addressing for each slot
! ======== ========== === ==== ====
!
! For each card in above slots: the correspondence between physical
! channels and logical channels
!
*DBRECORD               ! Slot 4: CAEN interface - para 1 (HV**)
SLOW SCON$TIM*.CRAT$0001.SLOT$SL07.B
0,0,910702,110000
*DBFIELD PTOL           ! 4 chans; phys chan, SC para, log chan, func flag
16,
    0,1,1,5
    1,1,2,5
    2,1,3,5
    3,1,4,5
*DBRECORD               !slot  9: Fastbus control
SLOW SCON$TIM*.CRAT$0001.SLOT$SL09.B
0,0,910702,110000
*DBFIELD PTOL           ! 2 chans; phys chan, SC para, log chan, func flag
8,
   0,3, 1,3
   2,3, 2,3
!
! High Voltage Data -- SC parameter 1 (HV**)
! ==== ======= ====    == ========= = ======
!
*DBRECORD               ! Parameter description
SLOW SCON$TIM*.DESC$HV**.B
0,0,910702,110000
*DBFIELD TEXT (I2,/,4X,5A4)
5,
    High voltage (CAEN)
*DBRECORD               ! Standard constants for all 4 channels
SLOW SCON$TIM*.STCO$HV**.B
0,0,910702,110000
*DBFIELD STAN
1,
    1.0                 ! Dummy data (not used by standard SC EP)
*DBRECORD               ! Correlation table: logical to physical
```

```
SLOW SCON$TIM*.LTOP$HV**.B
0,0,910702,110000
*DBFIELD CORR           ! 4 chans; crate, slot, phys chan, func flag
16,
    1,7,0,5
    1,7,1,5
    1,7,2,5
    1,7,3,5
*DBRECORD               ! Values for logical channel 1
SLOW SCON$TIM*.LTOP$HV**.CAEN$0001.B
0,0,910702,110000
*DBFIELD CANS           ! V0,I0,I0R,V0S,I0S,RUP,RDW,TRIP,V1,I1,I1R,I1S
12,
    428, 3, 20, 100, 3, 100, 100, 0, 0, 0, 3, 3
*DBFIELD CANM           ! ERR flag, ERR routine
2,
    1,0
*DBFIELD TORA           ! Monitoring rate(s), Tolerance on VMON, IMON
3,
    300, 50, 3
*DBFIELD CARI           ! Calibration record identifiers
6,
    TIM*, SEC1, GLOB
    TIM*, WIRE, SENS
*DBFIELD SCGL           ! Locations in SLOW field of GLOB record
3,
    2,  1,2
*DBFIELD SCSD           ! Locations in STAT field of SENS record
2,
    1,  1
*DBFIELD COMM (I5,/,4X,(20A4))  ! Channel name
4
    Tim's anode 1
*DBRECORD               ! Values for logical channel 2
SLOW SCON$TIM*.LTOP$HV**.CAEN$0002.B
0,0,910702,110000
*DBFIELD CANS           ! V0,I0,I0R,V0S,I0S,RUP,RDW,TRIP,V1,I1,I1R,I1S
12,
    420, 3, 20, 100, 3, 100, 100, 0, 0, 0, 3, 3
*DBFIELD CANM           ! ERR flag, ERR routine
2,
    1,0
*DBFIELD TORA           ! Monitoring rate(s), Tolerance on VMON, IMON
3,
     60, 50, 3
*DBFIELD CARI           ! Calibration record identifiers
6,
```

```
    TIM*, SEC1, GLOB
    TIM*, WIRE, SENS
*DBFIELD SCGL            ! Locations in SLOW field of GLOB record
3,
    2,  3,4
*DBFIELD SCSD            ! Locations in STAT field of SENS record
2,
    1,  2
*DBFIELD COMM (I5,/,4X,(20A4))  ! Channel name
4
    Tim's anode 2
*DBRECORD                ! Values for logical channel 3
SLOW SCON$TIM*.LTOP$HV**.CAEN$0003.B
0,0,910702,110000
*DBFIELD CANS            ! V0,I0,I0R,V0S,I0S,RUP,RDW,TRIP,V1,I1,I1R,I1S
12,
    500, 3, 20, 100, 3, 100, 100, 0, 0, 0, 3, 3
*DBFIELD CANM            ! ERR flag, ERR routine
2,
    1,0
*DBFIELD TORA            ! Monitoring rate(s), Tolerance on VMON, IMON
3,
    300, 50, 3
*DBFIELD CARI            ! Calibration record identifiers
6,
    TIM*, SEC2, GLOB
    TIM*, WIRE, SENS
*DBFIELD SCGL            ! Locations in SLOW field of GLOB record
3,
    2,  1,2
*DBFIELD SCSD            ! Locations in STAT field of SENS record
2,
    1,  3
*DBRECORD                ! Values for logical channel 4
SLOW SCON$TIM*.LTOP$HV**.CAEN$0004.B
0,0,910702,110000
*DBFIELD CANS            ! V0,I0,I0R,V0S,I0S,RUP,RDW,TRIP,V1,I1,I1R,I1S
12,
    600, 3, 20, 100, 3, 100, 100, 0, 0, 0, 3, 3
*DBFIELD CANM            ! ERR flag, ERR routine
2,
    1,0
*DBFIELD TORA            ! Monitoring rate(s), Tolerance on VMON, IMON
3,
    300, 50, 3
*DBFIELD CARI            ! Calibration record identifiers
6,
```

```
    TIM*, SEC2, GLOB
    TIM*, WIRE, SENS
*DBFIELD SCGL           ! Locations in SLOW field of GLOB record
3,
    2,  3,4
*DBFIELD SCSD           ! Locations in STAT field of SENS record
2,
    1,  4
!
! Fastbus control data -- SC parameter 2 (FB**)
! ======= ======= ====    == ========= = ======
!
*DBRECORD               ! Parameter description
SLOW SCON$TIM*.DESC$FB**.B
0,0,910702,110000
*DBFIELD TEXT (I2,/,4X,7A4)
7,
    Fastbus control
*DBRECORD               ! Correlation table: logical to physical
SLOW SCON$TIM*.LTOP$FB**.B
0,0,910702,110000
*DBFIELD CORR           ! 2 chans; crate, slot, phys chan, func flag
8,
    1, 9, 0,3
    1, 9, 2,3
!
! Values required for Fastbus control
! ====== ======== === ======= =======
!
*DBRECORD               ! Values for logical channel 1
SLOW SCON$TIM*.LTOP$FB**.SETD$0001.B
0,0,910702,110000
*DBFIELD DIGS           ! State, and standard test state
2,
    1,0
*DBRECORD               ! Values for logical channel 2
SLOW SCON$TIM*.LTOP$FB**.SETD$0002.B
0,0,910702,110000
*DBFIELD DIGS           ! State, and standard test state
2,
    1,0
```

6.2 Example Status Update Database ASCII File

The CARI, SCGL, and SCSD fields above reference records in the Status Update Database, such as those defined by the following example ASCII file [2]. A fuller example may be found on the online cluster in
DELPHI$ONLINE:[SLOW_CONTROL.ELEMENTARY_PROCESS.EXAMPLES.TIM]TIM_CALB.ASC

```
!    Example Status Update Database ASCII File
!    ======= ====== ====== ======== ===== ====
!
! Start of tree GLOB$TIM*
! ===== == ==== =========
!
*DBRECORD                       ! Dummy root record
CALB GLOB$TIM*.B
 0, 0,  891202,  201026

*DBRECORD                       ! Sector 1
CALB GLOB$TIM*.GLOB$SEC1.B
 0, 0,  891202,  201027
*DBFIELD COMM (I4/4X,(20A4))
2,
    Sector 1
*DBFIELD LEAD
5,
    0, 0, 0, 0, 0
*DBFIELD SLOW
4,                              ! Channel 1 and 2 values written here:
    0.0, 0.0, 0.0, 0.0          ! VMON1, IMON1, VMON2, IMON2

*DBRECORD                       ! Sector 2
CALB GLOB$TIM*.GLOB$SEC2.B
 0, 0,  891202,  201027
*DBFIELD COMM (I4/4X,(20A4))
2,
    Sector 2
*DBFIELD LEAD
5,
    0, 0, 0, 0, 0
*DBFIELD SLOW
4,                              ! Channel 3 and 4 values written here:
    0.0, 0.0, 0.0, 0.0          ! VMON3, IMON3, VMON4, IMON4

*DBRECORD
CALB GLOB$TIM*.SENS$WIRE.B
 0, 0,  891202,  201027
*DBFIELD LEAD
7,
```

```
    4, 1, 0, 0, -1, 19, 0         ! word 5=-1 => STAT field is packed
*DBFIELD STAT
1,
    0                             ! Bad channels written here
```

References

[1] Yu. Belokopytov and V. Perevozchikov, *CARGO Database Management Package,* DELPHI note **93–5** PROG–195, 29th March 1993.

[2] Yu. Belokopytov, G. Gopal, V. Perevozchikov, H. Klein, W. Venus, *The DELPHI Database Description: Calibration & Detector Status File,* DELPHI note **93–4** PROG–194 DAS–138, 22nd November 1993.

[3] G. Smith, *DELPHI Slow Controls G64 Microcomputers Skeleton Program,* Version 4.2, DELPHI note **94–13** DAS–150, revised 24th June 1994.