GSI Multi-Branch System
Reference Manual

H.G. Essel, N. Kurz

October 28, 2003

GSI, Gesellschaft für Schwerionenforschung mbH
Planckstraße 1, D-64291 Darmstadt
Germany
Tel. (0 6159) 71–0
Chapter 1

Preface

GOOSY Copyright

The GOOSY software package has been developed at GSI for scientific applications. Any distribution or usage of GOOSY without permission of GSI is not allowed. To get the permission, please contact at GSI Mathias Richter (tel. 2394 or E-Mail ”M.Richter@gsi.de”) or Hans-Georg Essel (tel. 2491 or E-Mail ”H.Essel@gsi.de”).

MBS Copyright

The MBS software package has been developed at GSI for scientific applications. Any distribution or usage of MBS without permission of GSI is not allowed. To get the permission, please contact at GSI Mathias Richter (tel. 2394 or E-Mail ”M.Richter@gsi.de”) or Hans-Georg Essel (tel. 2491 or E-Mail ”H.Essel@gsi.de”), or Nikolaus Kurz (tel. 2979 or E-Mail ”N.Kurz@gsi.de”).

Conventions used in this Document

Examples in this manual show both system output (prompts, messages, and displays) and user input, which are all written in typewriter style. Names and keywords are also in typewriter style. Items to be replaced by actual values are enclosed in <>.

The change bars mark changes between MBS and SBS.

Registered Trademarks are not explicitly noted.
1.1 MBS Authors and Advisory Service

The authors of MBS and their main fields for advisory services are (phone GSI: + 6159 71 -):

**R. Barth** Fastbus SMI library (Tel. 2554)

**Y. Du** Event routines, DECunix port, AIX port (1995-1996)

**H.G. Essel** Multitasking, CI, message logger, ESONE, GOOSY connections (Tel. 2491)

**R. Fritzsche** TCP library.

**H. Göringer** Event server, PAW connections (Tel. 2553)

**J. Hoffmann** CVC hardware, trigger (Tel. 2494)

**F. Humbert** Esone library (1996)

**N. Kurz** Lynx, setup, readout, collector, multi-branch systems (Tel. 2979)


**W. Ott** Message logger, transport, stream server, taping (Tel. 2979)

**D. Schall** CVC hardware, trigger (left GSI in 1996)

The authors thank Margareta Hellström and Piotr Koczon for useful hints and for reviewing this manual resulting in substantial enhancements. Further suggestions to enhance the manual are welcome. The manuals are accessible also through WWW:

http://www-gsi-vms.gsi.de/img/doc or
http://www-gsi-vms.gsi.de/daq/home

H.Essel@gsi.de.
Chapter 2

MBS Keyword Summary
MBS_keywords

Keywords  In the following the MBS command keywords are listed with their occurrence in the commands.

1810_LECROY

SET SMI 1810_LECROY

1872A_LECROY

SET SMI 1872A_LECROY

1872_LECROY

SET SMI 1872_LECROY

1875A_LECROY

SET SMI 1875A_LECROY

1875_LECROY

SET SMI 1875_LECROY

1881M_LECROY

SET SMI 1881M_LECROY
1881_LECROY

SET SMI 1881_LECROY

1882F_LECROY

SET SMI 1882F_LECROY

1885F_LECROY

SET SMI 1885F_LECROY

ACCESS

ACCESS SMI
REMOTE ACCESS

ACQUISITION

START ACQUISITION
STOP ACQUISITION

ARECEIVER

SET ARECEIVER FLUSHTIME
START ARECEIVER
STOP ARECEIVER

ATTACH

ATTACH BASE

BASE
ATTACH BASE
CREATE BASE
DELETE BASE
DETACH BASE
DUMP BASE
RESTORE BASE

BROADCAST

READ SMI BROADCAST
WRITE SMI BROADCAST

CAMAC

CAMAC CNAF
CAMAC FILE

CHANNEL

SET HISTOGRAM CHANNEL

CLEAR

CLEAR DAQ_STATUS COUNTER
CLEAR DAQ_STATUS PROCTAB
CLEAR DAQ_STATUS STATUS
CLEAR HISTOGRAM
CLEAR PIPES
CLEAR POLYGON
CLEAR SMI FASTBUS
CLEAR TRIG_MOD
CLEAR WINDOW

CLOSE

CLOSE FILE
CNAF

CAMAC CNAF
CNAF

COMMANDS

DEFINE COMMANDS

COMMENT

COMMENT

CONDITION

CREATE CONDITION INCLUDE
CREATE CONDITION POLYGON
CREATE CONDITION WINDOW

CONNECT

CONNECT DISPATCHER
CONNECT TRANSPORT

COUNTER

CLEAR DAQ_STATUS COUNTER

CREATE

CREATE BASE
CREATE CONDITION INCLUDE
CREATE CONDITION POLYGON
CREATE CONDITION WINDOW
CREATE FILES
CREATE HISTOGRAM
CREATE POLYGON
CREATE WINDOW

CVC_CAM_IRQ

DISABLE CVC_CAM_IRQ
ENABLE CVC_CAM_IRQ

CVC_IRQ_MASK

DAQ_STATUS

CLEAR DAQ_STATUS COUNTER
CLEAR DAQ_STATUS PROCTAB
CLEAR DAQ_STATUS STATUS

DEFINE

DEFINE COMMANDS

DELAYED_EB

DISABLE DELAYED_EB
ENABLE DELAYED_EB

DELETE

DELETE BASE
DELETE HISTOGRAM

DEL_EB_COL

DISABLE DEL_EB_COL
ENABLE DET_EB_COL

DETACH

DETACH BASE

DISABLE

DISABLE CVC_CAM_IRQ
DISABLE DELAYED_EB
DISABLE DEL_EB_COL
DISABLE EVENT_COPY
DISABLE HISTOGRAM
DISABLE RECEIVE_DATA
DISABLE TCP
DISABLE TRIG_MOD

DISCONNECT

DISCONNECT DISPATCHER
DISCONNECT TRANSPORT

DISMOUNT

DISMOUNT TAPE

DISPATCHER

CONNECT DISPATCHER
DISCONNECT DISPATCHER
SET DISPATCHER
SET VERBOSE DISPATCHER

DR_FLUSHTIME

SET DR_FLUSHTIME
DUMP

DUMP BASE
DUMP HISTOGRAM
DUMP WINDOW

ENABLE

ENABLE CVC_CAM_IRQ
ENABLE DELAYED_EB
ENABLE DEL_EB_COL
ENABLE EVENT_COPY
ENABLE HISTOGRAM
ENABLE IRQ
ENABLE RECEIVE_DATA
ENABLE TCP
ENABLE TRIG_MOD

ENVIRONMENT

ESONE_SERV

SET MAXCLIENTS ESONE_SERV
SET PRINT ESONE_SERV
SET VERBOSE ESONE_SERV

EVENT

TYPE EVENT

EVENT_COPY

DISABLE EVENT_COPY
ENABLE EVENT_COPY
EVENT_SERV

SET EVENT_SERV
SET MAXCLIENTS EVENT_SERV
SET VERBOSE EVENT_SERV
START EVENT_SERV

FASTBUS

CLEAR SMI FASTBUS
INITIALIZE SMI FASTBUS
RELEASE SMI FASTBUS

FILE

CAMAC FILE
CLOSE FILE
OPEN FILE

FILEHEADER

SET FILEHEADER

FILES

CREATE FILES

FLUSHTIME

SET ARECEIVER FLUSHTIME
SET FLUSHTIME
SET RIRECEIVER FLUSHTIME
SET TORECEIVER FLUSHTIME
GLOBAL

SET VERBOSE GLOBAL

HELP

HELP

HISTOGRAM

CLEAR HISTOGRAM
CREATE HISTOGRAM
DELETE HISTOGRAM
DISABLE HISTOGRAM
DUMP HISTOGRAM
ENABLE HISTOGRAM
PROTECT HISTOGRAM
SET HISTOGRAM CHANNEL
SET HISTOGRAM TEXT
SET VERBOSE HISTOGRAM

INCLUDE

CREATE CONDITION INCLUDE

INITIALIZE

INITIALIZE SMI FASTBUS
INITIALIZE TAPE

INPUT

IRQ
ENABLE IRQ

LOAD

LOAD ML_Setup
LOAD MO_Setup
LOAD READOUT
LOAD SETUP
LOAD SLAVE_READOUT
LOAD SMI PEDESTAL
LOAD SMI_THRESHOLD

MAXCLIENTS

SET MAXCLIENTS ESONE_SERV
SET MAXCLIENTS EVENT_SERV

MESSAGE

START MESSAGE

ML_Setup

LOAD ML_Setup

MODULE

READ SMI MODULE
WRITE SMI MODULE

MOUNT

MOUNT TAPE
MO_SETUP

LOAD MO_SETUP

NEWS

NEWS

NODES

NTA

WRITE SMI NTA

OPEN

OPEN FILE

OUTPUT

START OUTPUT
STOP OUTPUT

PEDESTAL

LOAD SMI PEDESTAL
READ SMI PEDESTAL

PIPES

CLEAR PIPES
POLYGON

CLEAR POLYGON
CREATE CONDITION POLYGON
CREATE POLYGON
RESTORE POLYGON

PRINT

SET PRINT ESONE_SERV

PROCTAB

CLEAR DAQ_STATUS PROCTAB

PROMPT

SET VERBOSE PROMPT

PROTECT

PROTECT HISTOGRAM

PSHELL

PSHELL

RATE

READ

READ SMI BROADCAST
READ SMI MODULE
READ SMI PEDESTAL
READ SMI SEQUENCER
READ SMI SLAVE
READ SMI TPSCAN

READOUT

LOAD READOUT

RECEIVE_DATA

DISABLE RECEIVE_DATA
ENABLE RECEIVE_DATA

RELEASE

RELEASE SMI FASTBUS

REMOTE

REMOTE ACCESS
REMOTE RESET
REMOTE SHOW

REMOVE

REMOVE SEGMENTS

RESET

REMOTE RESET
RESET

RESTART
RESTART SMI SEQUENCER

RESTORE

RESTORE BASE
RESTORE POLYGON
RESTORE WINDOW

RIRECEIVER

SET RIRECEIVER FLUSHTIME
START RIRECEIVER
STOP RIRECEIVER

SEGMENTS

REMOVE SEGMENTS

SEQUENCER

READ SMI SEQUENCER
RESTART SMI SEQUENCER
WRITE SMI SEQUENCER

SET

SET ARECEIVER FLUSHTIME
SET DISPATCHER
SET DR_FLUSHTIME
SET EVENT_SERV
SET FILEHEADER
SET FLUSHTIME
SET HISTOGRAM CHANNEL
SET HISTOGRAM TEXT
SET MAXCLIENTS ESONE_SERV
SET MAXCLIENTS EVENT_SERV
SET PRINT ESONE_SERV
SET RI\textsc{receiver} FLUSH\textsc{time}  
SET SMI 1810\textunderscore L\textsc{ecroy}  
SET SMI 1872A\textunderscore L\textsc{ecroy}  
SET SMI 1872\textunderscore L\textsc{ecroy}  
SET SMI 1875A\textunderscore L\textsc{ecroy}  
SET SMI 1875\textunderscore L\textsc{ecroy}  
SET SMI 1881M\textunderscore L\textsc{ecroy}  
SET SMI 1881\textunderscore L\textsc{ecroy}  
SET SMI 1882F\textunderscore L\textsc{ecroy}  
SET SMI 1885F\textunderscore L\textsc{ecroy}  
SET STREAM\textunderscore SERV  
SET TASK  
SET TO\textsc{receiver} FLUSH\textsc{time}  
SET TRIG\textunderscore MOD  
SET VERBO\textsc{se} DISPATCH\textsc{er}  
SET VERBO\textsc{se} ES\textsc{one}\textunderscore SERV  
SET VERBO\textsc{se} EVENT\textunderscore SERV  
SET VERBO\textsc{se} GLOBAL  
SET VERBO\textsc{se} HISTOGRAM  
SET VERBO\textsc{se} PROMPT  
SET WINDOW  
SET XDISPLAY

\textbf{SETUP}

LOAD SETUP

\textbf{SHOW}

REMOTE SHOW  
SHOW

\textbf{SLAVE}

READ SMI SLAVE  
WRITE SMI SLAVE  
X\textsc{shell} SLAVE
SLAVE_READOUT

LOAD SLAVE_READOUT

SMI

ACCESS SMI
CLEAR SMI FASTBUS
INITIALIZE SMI FASTBUS
LOAD SMI PEDESTAL
LOAD SMI THRESHOLD
READ SMI BROADCAST
READ SMI MODULE
READ SMI PEDESTAL
READ SMI SEQUENCER
READ SMI SLAVE
READ SMI TPSCAN
RELEASE SMI FASTBUS
RESTART SMI SEQUENCER
SET SMI 1810_LECROY
SET SMI 1872A_LECROY
SET SMI 1872_LECROY
SET SMI 1875A_LECROY
SET SMI 1875_LECROY
SET SMI 1881M_LECROY
SET SMI 1881_LECROY
SET SMI 1882F_LECROY
SET SMI 1885F_LECROY
WRITE SMI BROADCAST
WRITE SMI MODULE
WRITE SMI NTA
WRITE SMI SEQUENCER
WRITE SMI SLAVE

START

START ACQUISITION
START ARECEIVER
START EVENT_SERV
START MESSAGE
START OUTPUT
START RIRECEIVER
START TASK

STATUS

CLEAR DAQ STATUS STATUS

STOP

STOP ACQUISITION
STOP ARECEIVER
STOP OUTPUT
STOP RIRECEIVER
STOP TASK

STREAM_SERV

SET STREAM_SERV

TAPE

DISMOUNT TAPE
INITIALIZE TAPE
MOUNT TAPE

TASK

SET TASK
START TASK
STOP TASK

TCP

DISABLE TCP
ENABLE TCP

TEXT

SET HISTOGRAM TEXT

THRESHOLD

LOAD SMI THRESHOLD

TO RECEIVER

SET TO RECEIVER FLUSH TIME

TP SCAN

READ SMI TP SCAN

TRANSPORT

CONNECT TRANSPORT
DISCONNECT TRANSPORT

TRIG_MOD

CLEAR TRIG.MOD
DISABLE TRIG.MOD
ENABLE TRIG.MOD
SET TRIG.MOD

TYPE

TYPE EVENT


**VERBOSE**

SET VERBOSE DISPATCHER
SET VERBOSE ESONE_SERV
SET VERBOSE EVENT_SERV
SET VERBOSE GLOBAL
SET VERBOSE HISTOGRAM
SET VERBOSE PROMPT

**VOID**

VOID

**WINDOW**

CLEAR WINDOW
CREATE CONDITION WINDOW
CREATE WINDOW
DUMP WINDOW
RESTORE WINDOW
SET WINDOW

**WRITE**

WRITE SMI BROADCAST
WRITE SMI MODULE
WRITE SMI NTA
WRITE SMI SEQUENCER
WRITE SMI SLAVE

**XDISPLAY**

SET XDISPLAY

**XSHELL**

XSHELL SLAVE
Chapter 3

MBS Command Summary

ACCESS SMI name [address]
(m_smi) (m_smi) Opens a shared segment to access smi

ATTACH BASE name
(m_histogram) (m_histogram) Attaches existing histogram data base.

CAMAC CNAF c n a f [d r] -LOG -NOPRINT
(m_esone_serv) (m_esone_serv) Executes local CAMAC cnaf.

CAMAC FILE filenam -LOG -NOPRINT
(m_esone_serv) (m_esone_serv) Executes local CAMAC cnaf.

CLEAR DAQ STATUS COUNTER –
(m_util) (m_util) Clears status counters in the global daq status segment

CLEAR DAQ STATUS PROCTAB –
(m_util) (m_util) Clears daq process table in the global daq status segment

CLEAR DAQ STATUS STATUS –
(m_util) (m_util) Clears daq status bits in the global daq status segment

CLEAR HISTOGRAM name
(m_histogram) (m_histogram) Clears histogram.

CLEARPIPES –
(m_read_meb) (m_read_meb) Clears the subevent pipes (queue).

CLEAR POLYGON name
(m_collector) (m_collector) Clear polygon condition counters.

CLEAR SMIFASTBUS –
(m_smi) (m_smi) Removes all SMI signal assertion to Fastbus
CLEAR TRIG.MOD -
  (m_util) (m_util) Resets the trigger module.

CLEAR WINDOW name
  (m_collector) (m_collector) Clear window condition counters.

CLOSE FILE [tape]
  (m_transport) (m_transport) Closes file.

CNAF c n a f [d r]
  (m_camac) (m_camac) Executes local CAMAC cnaf.

COMMENT line -ERROR -INFO -COMMAND
  (m_dispatch) (m_dispatch) Writes line to log file.

CONNECT DISPATCHER [node] -ALL
  (m_prompt) (m_prompt) Connects prompter to remote dispatcher.

CONNECT TRANSPORT name
  (m_to) (m_to) connect to m_transport

CREATE BASE name histogram size
  (m_histogram) (m_histogram) Creates histogram data base.

CREATE CONDITION INCLUDE name -POLYGON -WINDOWS
  (m_collector) (m_collector) Creates include files.

CREATE CONDITION POLYGON polygons -NEW
  (m_collector) (m_collector) Initialize polygon conditions.

CREATE CONDITION WINDOW windows -NEW
  (m_collector) (m_collector) Initialize window conditions.

CREATE FILES project [-ANALYSIS -STARTUP -CREATE]
  (m_histogram) (m_lea) Create include files and procedure.

CREATE HISTOGRAM name dim type 1channels 1low 1up [1binsize 2channels 2low
  2up 2binsize]
  (m_histogram) (m_histogram) Creates histogram.

CREATE POLYGON name [description datafile]
  (m_collector) (m_collector) Create polygon condition.

CREATE WINDOW name [description x1 x2 y1 y2]
  (m_collector) (m_collector) Create window condition.

DEFINE COMMANDS task file -LOG
  (m_dispatch) (m_dispatch) Defines commands for task.
DELETE BASE  name
             (m_histogram) (m_histogram) Deletes histogram data base.

DELETE HISTOGRAM  name
                  (m_histogram) (m_histogram) Deletes histogram.

DETACH BASE  name
             (m_histogram) (m_histogram) Detaches histogram data base.

DISABLE CVC_CAM_IRQ  -
                   (m_util) (m_util) Disables CAMAC LAM and VSB interrupts

DISABLE DELAYED_EB  -
                     (m_ds) (m_ds) disables delayed event building

DISABLE DEL_EB_COL  -
                     (m_collector) (m_collector) disables delayed event building

DISABLE EVENT_COPY  -
                     (m_collector) (m_collector) Disables event copy during data acquisition

DISABLE HISTOGRAM  -
                     (m_collector) (m_collector) Disables histogramming during data acquisition

DISABLE RECEIVE_DATA  -
                       (m_dr) (m_dr) Disables data receive of a dr node

DISABLE TCP  -
             (m_transport) (m_transport) Disables tcp connection.

DISABLE TRIG_MOD  -
                  (m_util) (m_util) Disables trigger module on trigger bus

DISCONNECT DISPATCHER  [node] -ALL -KILL
                        (m_prompt) (m_prompt) Disconnects prompter from remote dispatcher.

DISCONNECT TRANSPORT  name
                       (m_to) (m_to) disconnect from m_transport

DISMOUNT TAPE  [tape] -UNLOAD
               (m_transport) (m_transport) Dismounts tape.

DUMP BASE  base file -COMPRESS -OVERWRITE
           (m_histogram) (m_histogram) Dumps base into file.

DUMP HISTOGRAM  name file -SEPARATE
                 (m_histogram) (m_histogram) Dumps histogram in file or files.
DUMP WINDOW file
 (m_collector) (m_collector) Dump window conditions.

ENABLE CVC_CAM_IRQ
 (m_util) (m_util) Enables CAMAC LAM and VSB interrupts

ENABLE DELAYED_EVENT
 lo mark hi mark
 (m_ds) (m_ds) enables delayed event building

ENABLE DELAYED_EVENT_COL
 lo mark hi mark
 (m_collector) (m_collector) enables delayed event building

ENABLE EVENT_COPY
 (m_collector) (m_collector) Enables event copy during data acquisition

ENABLE HISTOGRAM basename
 (m_collector) (m_collector) Enables histogramming during data acquisition

ENABLE IRQ
 (m_util) (m_util) Disables trigger module to send IRQ or LAM

ENABLE RECEIVE_DATA
 (m_dr) (m_dr) Enables data receive of a dr node

ENABLE TCP -INCLUSIVE -EXCLUSIVE
 (m_transport) (m_transport) Enables tcp connection.

ENABLE TRIG_MOD
 (m_util) (m_util) Enables trigger module on trigger bus

HELP [k1 k2 k3 k4 k5 library]
 (m_prompt) (m_prompt) Outputs help information by keywords.

INITIALIZE SMI FASTBUS
 (m_smi) (m_smi) Initializes FastBus

INITIALIZE TAPE label [tape]
 (m_transport) (m_transport) Initializes tape.

LOAD ML_SETUP usf_file
 (m_util) (m_util) Loads setup file of multi-layer multi-branch daq system

LOAD MO_SETUP usf_file
 (m_util) (m_util) Loads setup file multi-output/collector mbs system

LOAD READOUT usf_file
 (m_read_meb) (m_read_meb) Loads readout table.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LOAD SETUP</strong></td>
<td>usf file [crate, nr] (m_util) (m_util) Loads setup file.</td>
</tr>
<tr>
<td><strong>LOAD SLAVE READOUT</strong></td>
<td>usf file (m_read_cam_slav) (m_read_cam_slav) Loads readout table.</td>
</tr>
<tr>
<td><strong>LOAD SMI PEDESTAL</strong></td>
<td>slot range channel value (m_smi) (m_smi) Write to pedestal memory of SMI</td>
</tr>
<tr>
<td><strong>LOAD SMI THRESHOLD</strong></td>
<td>range value (m_smi) (m_smi) Write threshold to SMI</td>
</tr>
<tr>
<td><strong>MOUNT TAPE</strong></td>
<td>[tape] (m_transport) (m_transport) Mounts tape.</td>
</tr>
<tr>
<td><strong>NEWS</strong></td>
<td>[facility item path] -ALL (m_dispatch) (m_dispatch) Outputs news.</td>
</tr>
<tr>
<td><strong>OPEN FILE</strong></td>
<td>name [tape size number first inhead outhead] -AUTO -PROMPT -EDIT -DISK -RFIODISK -RFIOSERVER -MULTI -FIRST (m_transport) (m_transport) Opens file on tape.</td>
</tr>
<tr>
<td><strong>PROTECT HISTOGRAM</strong></td>
<td>name -UNPROTECT (m_histogram) (m_histogram) [Un]protect histogram from being cleared.</td>
</tr>
<tr>
<td><strong>PSHELL</strong></td>
<td>command [a1 a2 a3 a4] (m_prompt) (m_prompt) Executes shell command line.</td>
</tr>
<tr>
<td><strong>READ SMI BROADCAST</strong></td>
<td>type -CONTROL -DATA (m_smi) (m_smi) Broadcast read from FastBus</td>
</tr>
<tr>
<td><strong>READ SMI MODULE</strong></td>
<td>gad sad -CONTROL -DATA (m_smi) (m_smi) Reads from FastBus Module</td>
</tr>
<tr>
<td><strong>READ SMI PEDESTAL</strong></td>
<td>slot range channel (m_smi) (m_smi) Reads the pedestal memory of SMI</td>
</tr>
<tr>
<td><strong>READ SMI SEQUENCER</strong></td>
<td>address (m_smi) (m_smi) Read SMI sequencer instruction word</td>
</tr>
<tr>
<td><strong>READ SMI SLAVE</strong></td>
<td>– (m_smi) (m_smi) FastBus read from an attached slave</td>
</tr>
<tr>
<td><strong>READ SMI TPSCAN</strong></td>
<td>type (m_smi) (m_smi) Performs T-PIN scan and returns the pattern back</td>
</tr>
<tr>
<td><strong>RELEASE SMI FASTBUS</strong></td>
<td>– (m_smi) (m_smi) Release bus mastership</td>
</tr>
</tbody>
</table>
REMOTE ACCESS  [nodelist]  (m_prompt) (m_prompt) Grants command access from nodes.

REMOTE RESET  [node] -ALL  (m_prompt) (m_prompt) Cleans up remote nodes. Remote program is m_remote.

REMOTE SHOW  [node] [task] -LOCAL -TASKS -DAQ -NET  (m_prompt) (m_prompt) Shows remote info.

REMOVE SEGMENTS  –  (m_util) (m_util) Removes critical segments

RESET  [node] [task] -LOCAL  (m_remote) (m_remote) Resets remote node. Executed by alias remote.

RESET SMI FASTBUS  –  (m_smi) (m_smi) Activate RB on FastBus dataway

RESTART SMI SEQUENCER  –  (m_smi) (m_smi) Restart sequencer from an given address

RESTORE BASE  base file -OVERWRITE -NEW  (m_histogram) (m_histogram) Restores base from file.

RESTORE POLYGON  file -NEW -OVER -APPEND  (m_collector) (m_collector) Restore polygon condition from file.

RESTORE WINDOW  file -NEW -OVER -APPEND  (m_collector) (m_collector) Restore window condition from file.

SET RECEIVER FLUSHTIME  time  (m_ar) (m_ar) Sets stream flushtime

SET DISPATCHER  [node]  (m_prompt) (m_prompt) Sets terminal to remote dispatcher.

SET DR_FLUSHTIME  time  (m_dr) (m_dr) Sets stream flushtime

SET EVENT_SERV  [scale] [events] [maxclnt] -ALL  (m_event_serv) (m_event_serv) Sets m_event_serv parameters.

SET FLUSHTIME  time
   (m_collector) (m_collector) Sets stream flush time

SET HISTOGRAM CHANNEL  name value xchan [ychan] -INCREMENT
   (m_histogram) (m_histogram) Sets channel content of histogram.

SET HISTOGRAM TEXT  name [text] -TITLE -XTXT -YTXT -CONT
   (m_histogram) (m_histogram) Sets lettering text field of histogram.

SET MAXCLIENTS ESONE_SERV  maxchnt
   (m_esone_serv) (m_esone_serv) Sets maximum number of clients for m_esone_serv.

SET MAXCLIENTS EVENT_SERV  maxchnt
   (m_event_serv) (m_event_serv) Sets maximum number of clients for m_event_serv.

SET PRINT ESONE_SERV  -ON -OFF
   (m_esone_serv) (m_esone_serv) Sets terminal output Esone data for Esone Server.

SET RIRECEIVER FLUSHTIME  time
   (m_rirecc) (m_rirecc) Sets stream flush time

SET SMI 1810_LECROY  slot csr1 dsr0 dsr1 dsr2 dsr3 dsr4
   (m_smi) (m_smi) Initialize LeCroy Module 1810

SET SMI 1872A_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1872A

SET SMI 1872_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1872

SET SMI 1875A_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1875A

SET SMI 1875_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1875

SET SMI 1881M_LECROY  slot csr0 csr1 csr3 csr7 threshold
   (m_smi) (m_smi) Initialize LeCroy Module 1881M

SET SMI 1881_LECROY  slot csr0 csr1 csr3 csr7 threshold
   (m_smi) (m_smi) Initialize LeCroy Module 1881

SET SMI 1882F_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1882F

SET SMI 1885F_LECROY  slot csr0 csr1
   (m_smi) (m_smi) Initialize LeCroy Module 1885F
SET STREAM\_SERV  [scale] -[NO]SYNC -[NO]KEEP -SCALE\_KEEP -CLEAR
(m\_stream\_serv) (m\_stream\_serv) Selects scaledown of streams.

SET TASK  task pid -CLEAR
(m\_dispatch) (m\_dispatch) Sets task id.

SET TORECEIVER FLUSHTIME  time
(m\_to) (m\_to) set stream flush time

SET TRIG\_MOD  -SLAVE
(m\_util) (m\_util) Sets trigger module.

SET VERBOSE DISPATCHER  -ON -OFF
(m\_dispatch) (m\_dispatch) Sets verbosity for dispatcher.

SET VERBOSE ESONE\_SERV  -ON -OFF
(m\_esone\_serv) (m\_esone\_serv) Sets verbosity for Esone Server.

SET VERBOSE EVENT\_SERV  -ON -OFF
(m\_event\_serv) (m\_event\_serv) Sets verbosity for m\_event\_serv.

SET VERBOSE GLOBAL  -ON -OFF -NEUTRAL
(m\_util) (m\_util) Sets verbosity for all tasks

SET VERBOSE HISTOGRAM  -ON -OFF
(m\_histogram) (m\_histogram) Sets verbosity for m\_histogram.

SET VERBOSE PROMPT  -ON -OFF
(m\_prompt) (m\_prompt) Sets verbosity for m\_prompt.

SET WINDOW  name [x1 x2 y1 y2] -CURSOR
(m\_collector) (m\_collector) Set limits of window condition.

SET XDISPLAY  node
(m\_prompt) (m\_prompt) Set name of remote display.

SHOW  [node] [task] -LOCAL -TASKS -DAQ -NET
(m\_remote) (m\_remote) Shows remote info. Executed by alias remote.

SHOW ACQUISITION  [seconds] -SETUP -CRATES -SERVER -RATE -LOG
(m\_util) (m\_util) Shows acquisition.

SHOW BASE  [name] -FULL
(m\_histogram) (m\_histogram) Shows information about histogram data base.

SHOW COMMANDS  [task] -FULL -ALL
(m\_dispatch) (m\_dispatch) Shows known commands.
SHOW CVC_IRQ_MASK
  (m_util) (m_util) Reads irq mask of the CVC irq controller

SHOW DISPATCHER  [node]
  (m_prompt) (m_prompt) Shows connections to remote dispatchers.

SHOW ENVIRONMENT
  (m_dispatch) (m_dispatch) Shows environment parameters.

SHOW ESONE_SERV -FULL -LOG -CLIENT
  (m_esone_serv) (m_esone_serv) Shows status of m_esone_serv.

SHOW EVENT_SERV -FULL -LOG -CLIENT
  (m_event_serv) (m_event_serv) Shows status of m_event_serv.

SHOW HISTOGRAM  name [1offset] [1channels] [2offset] [2channels] -DATA -FULL
  (m_histogram) (m_histogram) Shows histogram info and content.

SHOW INPUT NODES
  (m_to) (m_to) show input nodes connected via tcp sockets

SHOW MESSAGE
  (m_dispatch) (m_dispatch) Shows status of internal message file.

SHOW ML_SETUP
  (m_util) (m_util) Shows multilayer setup parameters

SHOW POLYGON  name -DATA
  (m_collector) (m_collector) Show polygon conditions.

SHOW RATE  [seconds] -OFF -ON
  (m_util) (m_util) Shows acquisition rate.

SHOW SETUP
  (m_util) (m_util) Shows setup parameters

SHOW STATUS
  (m_util) (m_util) Shows daqst parameters

SHOW STREAM_SERV -CLEAR
  (m_stream_serv) (m_stream_serv) Shows modes and counters.

SHOW TAPE  [tape]
  (m_transport) (m_transport) Shows tape information.

SHOW TASK  [task] -FULL -ALL
  (m_dispatch) (m_dispatch) Shows known tasks.
SHOW TRIG.MOD
(m_util) (m_util) Shows current setup of trigger module.

SHOW WINDOW name
(m_collector) (m_collector) Show window conditions.

START ACQUISITION
(m_util) (m_util) Starts acquisition.

START ARECEIVER
(m_ar) (m_ar) Enables data receive of a ar node

START EVENT_SERV [scale] [events] [maxclnt] -[NO]PORTS -VERB
(m_event_serv) (m_event_serv) Starts m_event_serv.

START MESSAGE
(m_dispatch) (m_dispatch) Starts the message logger.

START OUTPUT
(m_to) (m_to) starts time sorting and event output

START RIRECEIVER
(m_rirecc) (m_rirecc) Enables data receive of a rirec node

START TASK task [file]
(m_dispatch) (m_dispatch) Starts task.

STOP ACQUISITION
(m_util) (m_util) Stops acquisition.

STOP ARECEIVER
(m_ar) (m_ar) Disables data receive of a ar node

STOP OUTPUT
(m_to) (m_to) stops time sorting and event output

STOP RIRECEIVER
(m_rirecc) (m_rirecc) Disables data receive of a rirec node

STOP TASK [task] [pid] -ALL-KILL -ZOMBIE
(m_dispatch) (m_dispatch) Stops task by sending command "exit"

TYPE EVENT [events id] -SAMPLE -VERBOSE
(m_transport) (m_transport) Prints events.

VOID
(m_daq_rate) (m_daq_rate) place holder, do not execute.
WRITE SMI BROADCAST  type data -CONTROL -DATA
    (m_smi) (m_smi) Writes to FastBus Module

WRITE SMI MODULE  gad sad data -CONTROL -DATA
    (m_smi) (m_smi) Writes to FastBus Module

WRITE SMI NTA  sad
    (m_smi) (m_smi) Writes secondary address to an FastBus slave

WRITE SMI SEQUENCER  address [word] [table]
    (m_smi) (m_smi) Write SMI sequencer instruction word

WRITE SMI SLAVE  data
    (m_smi) (m_smi) FastBus write to an attached slave

XSHHELL  command [a1 a2 a3 a4]
    (m_dispatch) (m_dispatch) Executes shell command line.
Chapter 4

MBS Command Description
ATTACH

ATTACH BASE

ATTACH BASE name

PURPOSE  (m_histogram) Attaches existing histogram data base.

PARAMETERS

name      name of segment

Description

Function  Create or attach histogram data base.
Routine   f_his_cmd_attbas
Task      m_histogram
CAMAC

CAMAC CNAF

CAMAC CNAF c n a f [d r]-LOG -NOPRINT

PURPOSE
(m_esone_serv) Executes local CAMAC cnaf.

PARAMETERS
  c  crate
  n  station
  a  address
  f  function
  d  data
  r  repetition
  -LOG  write results to log file
  -NOPRINT  no terminal output

Description
  Function  Execute local CAMAC cnaf.
  Routine  f_es_cmd_cnaf
  Task  m_esone_serv

CAMAC FILE

CAMAC FILE filenam -LOG -NOPRINT

PURPOSE
(m_esone_serv) Executes local CAMAC cnaf.

PARAMETERS
<table>
<thead>
<tr>
<th><strong>filename</strong></th>
<th>filename with cnaf data: lines with c n a f d r like arguments of CAMAC CNAF COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-LOG</strong></td>
<td>write results to log file</td>
</tr>
<tr>
<td><strong>-NOPRINT</strong></td>
<td>no terminal output</td>
</tr>
</tbody>
</table>

**Description**

**Function**
Execute local CAMAC cnaf.

**Routine**
`fes_cmd_file`

**Task**
`m_esone_serv`
CLEAR

CLEAR DAQ_STATUS COUNTER

PURPOSE (m_util) Clears status counters in the global daq status segment
PARAMETERS
none

Description
Function Clear status counters in the global daq status segment
Routine f_util_clear_daqst
Task m_util

CLEAR DAQ_STATUS PROCTAB

PURPOSE (m_util) Clears daq process table in the global daq status segment
PARAMETERS
none

Description
Function Clear daq process table in the global daq status segment
Routine f_util_clear_daqst
Task m_util
CLEAR DAQ STATUS

PURPOSE (m_util) Clears daq status bits in the global daq status segment
PARAMETERS

none

Description
Function Clear daq status bits in the global daq status segment
Routine f_ut_clear_daqst
Task m_util

CLEAR HISTOGRAM

CLEAR HISTOGRAM name

PURPOSE (m_histogram) Clears histogram.
PARAMETERS

name name of histogram, also wildcard.

Description
Function Clear histogram.
Routine f_hist_cmd_clrhis
Task m_histogram

CLEAR PIPES

CLEAR PIPES

PURPOSE (m_read_meb) Clears the subevent pipes (queue).
PARAMETERS
Description

**Function**
Clear the subevent pipes (queue).
(not yet implemented)

**Routine**

**Task**
m_read_meb

**CLEAR TRIG_MOD**

**PURPOSE**
(\texttt{m\_util}) Resets the trigger module.

**PARAMETERS**

none

**Description**

**Function**
Reset the trigger module.

**Routine**
f\_ut\_op\_trig\_mod

**Task**
m\_util
CLOSE FILE

PURPOSE
(m_transport) Closes file.

PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tape</td>
<td>Optional tape number</td>
</tr>
</tbody>
</table>

Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Closes file on tape. The acquisition is not stopped!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routine</td>
<td>f_-</td>
</tr>
<tr>
<td>Task</td>
<td>m_transport</td>
</tr>
</tbody>
</table>
CNAF

CNAF c n a f [d r]

PURPOSE
(m_camac) Executes local CAMAC cnaf.

PARAMETERS
  c     crate
  n     station
  a     address
  f     function
  d     data
  r     repetition

Description
  Function    Execute local CAMAC cnaf.
  Routine     f_cnaf
  Task        m_camac
COMMENT

COMMENT line -ERROR -INFO -COMMAND

PURPOSE (m_dispatch) Writes line to log file.

PARAMETERS

command Line inserted in log file.
-ERROR Mark as error line
-INFO Mark as info line
-COMMAND Mark as command line

Description

Function Use F_ERROR to write line to log file and terminal.
Example COMM "Neue Schicht"
Routine f_disp_comment
Task m_dispatch
CONNECT

CONNECT DISPATCHER

CONNECT DISPATCHER [node]-ALL

PURPOSE  (m_prompt) Connects prompter to remote dispatcher.

PARAMETERS

node  Name of remote node.

-ALL  All known nodes

Description

Function  Opens TCP link to remote dispatcher which is started. Message client is started. The remote nodes must be declared as trustedhost and trusteduser in the .rhosts file on users directory.

Routine  f_ifa_new_connect

Task  m_prompt
CREATE

CREATE BASE

CREATE BASE name histogram size

PURPOSE (m_histogram) Creates histogram data base.

PARAMETERS

  name    name of segment
  histogram maximum number of histograms
  size    size of segment in kByte

Description

  Function    Create or attach histogram data base.
  Routine     f_his_cmd_crebas
  Task        m_histogram

CREATE FILES

CREATE FILES project [-ANALYSIS -STARTUP -CREATE]

PURPOSE (m_lea) Create include files and procedure.

PARAMETERS

  project used to name the files.
  -STARTUP not used
  -ANALYSIS create analysis function
  -CREATE not used
Description

Function Create three files:
hisdef_project.h Definition include file
hisini_project.h Initialization include file
hiscre_project.scom Procedure to create histograms
f_mbs_anal_project.c User function when -ANALYSIS was given.

Routine f_his_Cmd_crefil

Task m_histogram

CREATE HISTOGRAM

CREATE HISTOGRAM name dim type 1channels 1low 1up [1binsize 2channels 2low 2up 2binsize]

PURPOSE (m_histogram) Creates histogram.

PARAMETERS

name name of histogram
dim dimension of histogram (dim=1,2)
type type of histogram 'r' : real or 'i' : integer
chan1 number of channels 1st dimension
lo1 lower limit of histogram 1st dimension
up1 upper limit of histogram 1st dimension
chan2 number of channels 2nd dimension
lo2 lower limit of histogram 2nd dimension
up2 upper limit of histogram 2nd dimension
### Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Create histogram.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routine</td>
<td>f_his_cmd_crehis</td>
</tr>
<tr>
<td>Task</td>
<td>m_histogram</td>
</tr>
</tbody>
</table>
DEFINE

DEFINE COMMANDS

DEFINE COMMANDS task file -LOG

PURPOSE
(m_dispatch) Defines commands for task.

PARAMETERS

- task
  Task (program) name or * for all tasks

- file
  name of command definition file (cdf)

- -LOG
  Output definitions

Description

Function
Tasks, commands and arguments can be defined in text files like /mbs/deve/set/mbscom.c.

The definitions of such a file can be read by this command. The task which has to execute the commands, must specify functions for each command. This is done by calls to routines f_cmd_cdef and f_cmd_set_entry. See command interface description.

Routine
f_disp_def_cmd

Task
m_dispatch
DELETE

DELETE BASE

DELETE BASE name

PURPOSE (m_histogram) Deletes histogram data base.

PARAMETERS

name name of segment

Description

Function Delete histogram data base.
Routine f_his_cmd_delbas
Task m_histogram

DELETE HISTOGRAM

DELETE HISTOGRAM name

PURPOSE (m_histogram) Deletes histogram.

PARAMETERS

name name of histogram, also wildcard.

Description

Function Delete histogram in histogram manager.
Routine f_his_cmd_delhis
Task m_histogram
## DETACH

### DETACH BASE

**PURPOSE**
(m_histogram) Detaches histogram data base.

**PARAMETERS**

<table>
<thead>
<tr>
<th>name</th>
<th>name of segment</th>
</tr>
</thead>
</table>

**Description**

**Function**
Detach histogram data base.

**Routine**
_fhis_cmd_detbas_

**Task**
m_histogram
DISABLE

DISABLE CVC_CAM_IRQ

DISABLE CVC_CAM_IRQ

PURPOSE (m_util) Disables CAMAC LAM and VSB interrupts
PARAMETERS
none

Description
Function Disable CAMAC LAM and VSB interrupts of the CVC
Routine f_ut_cvc_cam_irq
Task m_util

DISABLE EVENT_COPY

DISABLE EVENT_COPY

PURPOSE (m_collector) Disables event copy during data acquisition
PARAMETERS

Description
Function disable event copy during data acquisition
Routine f_col_cmd_dis_evt_copy
Task m_collector
### DISABLE HISTOGRAM

**Purpose**
(mcollector) Disables histogramming during data acquisition

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Function</th>
<th>Routine</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>disable histogramming during data acquisition</td>
<td><code>f_col_cmd_dis_hist</code></td>
<td><code>m_collector</code></td>
</tr>
</tbody>
</table>

**Description**

- **Function**: disable histogramming during data acquisition
- **Routine**: `f_col_cmd_dis_hist`
- **Task**: `m_collector`

### DISABLE RECEIVE_DATA

**Purpose**
(mdr) Disables data receive of a dr node

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Function</th>
<th>Routine</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables data receive of a dr node</td>
<td><code>f_dr_cmd_dis_receive_data</code></td>
<td><code>m_dr</code></td>
</tr>
</tbody>
</table>

**Description**

- **Function**: Enables data receive of a dr node
- **Routine**: `f_dr_cmd_dis_receive_data`
- **Task**: `m_dr`

### DISABLE TCP

**Purpose**
(m_transport) Disables tcp connection.

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Function</th>
<th>Routine</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disables tcp connection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Description

Function    disables any tcp connection from GOOSY transport manager
Routine     f_
Task        m_transport

DISABLE TRIG_MOD

PURPOSE   (m_util) Disables trigger module on trigger bus
PARAMETERS  none

Description

Function    disables trigger module on the trigger bus to take NOT part in a multi trigger module environment
Routine     f_ut_op_trig_mod
Task        m_util
DISCONNECT

DISCONNECT DISPATCHER

DISCONNECT DISPATCHER [node]-ALL -KILL

PURPOSE (m_prompt) Disconnects prompter from remote dispatcher.

PARAMETERS

node Name of remote node.

-ALL All known nodes

-KILL Stop dispatcher task

Description


Routine f_ifa_disconnect

Task m_prompt
DISMOUNT

DISMOUNT TAPE

DISMOUNT TAPE [tape]-UNLOAD

PURPOSE (m_transport) Dismounts tape.

PARAMETERS

tape Optional tape number

-UNLOAD unloads tape after rewind

Description

Function Dismount tape.

Routine f_

Task m_transport
DUMP

DUMP BASE

DUMP BASE base file

PURPOSE (m_histogram) Dumps base into file.

PARAMETERS

  base    name of segment = name of base
  file    name of file

Description

  Function  Dump base into file.
  Routine   f_his_cmd_dmpbas
  Task      m_histogram

DUMP HISTOGRAM

DUMP HISTOGRAM name file -SEPARATE

PURPOSE (m_histogram) Dumps histogram in file or files.

PARAMETERS

  name    name of histogram, also wildcard.
  file    file name
  -SEPARATE  when using wildcard, file name equal histogram name
### Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dump histogram in file or files.</td>
<td></td>
</tr>
<tr>
<td>Routine</td>
<td>f.his.cmd.dmphis</td>
</tr>
<tr>
<td>Task</td>
<td>m.histogram</td>
</tr>
</tbody>
</table>
ENABLE

ENABLE CVC_CAM_IRQ

PURPOSE
(m_util) Enables CAMAC LAM and VSB interrupts

PARAMETERS
none

Description
Function
Enable CAMAC LAM and VSB interrupts.
this is normally done during boot time of the CVC

Routine
f_ut_cvc_cam_irq

Task
m_util

ENABLE EVENT_COPY

PURPOSE
(m_collector) Enables event copy during data acquisition

PARAMETERS

Description
Function
enable event copy during data acquisition

Routine
f_col_cmd_ena_evt_copy

Task
m_collector
ENABLE HISTOGRAM

ENABLE HISTOGRAM basename

PURPOSE (m_collector) Enables histogramming during data acquisition
PARAMETERS
basename    histogram base (segment) name

Description
Function    enable histogramming during data acquisition
Routine     f_col_cmd_ena_hist
Task        m_collector

ENABLE IRQ

ENABLE IRQ

PURPOSE (m_util) Disables trigger module to send IRQ or LAM
PARAMETERS
none

Description
Function    disable trigger module to send an IRQ or LAM on occurrence of a trigger
Routine     f_ut_op_trig_mod
Task        m_util

ENABLE RECEIVE_DATA

ENABLE RECEIVE_DATA

PURPOSE (m_dr) Enables data receive of a dr node
PARAMETERS
**Description**

**Function**
Enables data receive of a dr node

**Routine**
f_dr_cmd_ena_receive_data

**Task**
m_dr

**ENABLE TCP**

**ENABLE TCP -INCLUSIVE -EXCLUSIVE**

**PURPOSE**
(m_transport) Enables tcp connection.

**PARAMETERS**
- **-INCLUSIVE**
enables simultaneous output on tape and tcp
- **-EXCLUSIVE**
enables exclusive output on tape or tcp

**Description**

**Function**
enables tcp connection from GOOSY transport manager or other clients. Default is to write both, tape and TCP, if a client connects! To stop writing tape, when a client connects -EXCLUSIVE must be given.

**Routine**
f_

**Task**
m_transport

**ENABLE TRIG_MOD**

**ENABLE TRIG_MOD**

**PURPOSE**
(m_util) Enables trigger module on trigger bus

**PARAMETERS**
none
## Description

<table>
<thead>
<tr>
<th><strong>Function</strong></th>
<th>enables trigger module on the trigger bus to take part in a multi trigger module environment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Routine</strong></td>
<td>f_ut_op_trig_mod</td>
</tr>
<tr>
<td><strong>Task</strong></td>
<td>m_util</td>
</tr>
</tbody>
</table>
HELP

HELP [k1 k2 k3 k4 k5 library]

PURPOSE (m_promise) Outputs help information by keywords.

PARAMETERS

    key#    Optional keys.
    library Help library in VAX text format. A .hlp is appended. Default is /mbs/deve/lib/mbs.hlp

Description

    Function Help similar to VMS.
    Routine f_ifa help
    Task m_promise
INITIALIZE

INITIALIZE TAPE

INITIALIZE TAPE label [tape]

PURPOSE  (m_transport) Initializes tape.

PARAMETERS

label        ANSI label of tape.
tape         Optional tape number

Description

Function      Writes ANSI label to tape. Any information on the tape is lost.
Routine       f_
Task          m_transport
LOAD

LOAD ML_SETUP

LOAD ML_SETUP usf_file

PURPOSE (m_util) Loads setup file of multi-layer multi-branch daq system
PARAMETERS
usf_file User setup file.

Description
Function load content of mb ml setup file into memory.
Routine f_ut_load_ml_setup
Task m_util

LOAD MO_SETUP

LOAD MO_SETUP usf_file

PURPOSE (m_util) Loads setup file multi-output/collector mbs system
PARAMETERS
usf_file User setup file.

Description
Function load content of mo setup file into memory.
Routine f_ut_load_mo_setup
Task m_util
LOAD READOUT

LOAD READOUT  usf_file

PURPOSE       (m_read_meb) Loads readout table.
PARAMETERS    
usf_file       Definition file.

Description
  Function      Load readout table for master readout into memory.
               This must be done at least once but as often
               the user wants since the startup of the
               readout task. The previous readout table will be
               completely overwritten.
  See also     Command LOAD SLAVE.READOUT.
  Routine      f_ut_load_readout
  Task         m_read_meb

LOAD SETUP

LOAD SETUP  usf_file  [crate_nr]

PURPOSE       (m_util) Loads setup file.
PARAMETERS    
usf_file       User setup file.
crate_nr       Optional crate number (slave readout)

Description
  Function      load content of setup file into memory.
               When executing on a slave, the crate number
               of the slave must be specified.
  Routine      f_ut_load_setup
  Task         m_util
LOAD SLAVE_READOUT

LOAD SLAVE_READOUT usf_file

PURPOSE (m_read_cam_slav) Loads readout table.

PARAMETERS
usf_file Definition file.

Description
Function Load readout table for slave readout into memory.
This must be done at least once but as often
the user wants since the startup of the
readout task. The previous readout table will be
completely overwritten.

See also Command LOAD READOUT.
Routine f_ut_load_readout
Task m_read_cam_slav
# MOUNT

## MOUNT TAPE

<table>
<thead>
<tr>
<th>MOUNT TAPE [tape]</th>
</tr>
</thead>
</table>

**PURPOSE** (m_transport) Mounts tape.

**PARAMETERS**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tape</td>
<td>Optional tape number</td>
</tr>
</tbody>
</table>

**Description**

<table>
<thead>
<tr>
<th>Function</th>
<th>Mount tape and skip files up to logical end of tape. The tape must be initialized.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routine</td>
<td>f_</td>
</tr>
<tr>
<td>Task</td>
<td>m_transport</td>
</tr>
</tbody>
</table>
NEWS

NEWS [facility item path]-ALL

PURPOSE  (m_dispatch) Outputs news.

PARAMETERS

facility  Optional name of facility, i.e. mbs. If not specified, a listing of available facilities is printed. Facilities are files with .inf postfix.

item  Optional number of item to be printed.

path  Optional path to look for facility files. Default is /mbs/deve/lib.

-ALL  If specified, all items are listed, if not, only unseen ones. When called from shell, switch -a lists all items, i.e. news mbs -a.

Description

Function  News similar to VMS.

Routine  f_disp_info

Task  m_dispatch
OPEN

OPEN FILE

OPEN FILE name [tape size number first inhead outhead]
-AUTO -PROMPT -EDIT -DISK

PURPOSE
(m_transport) Opens file on tape.

PARAMETERS

name
File name. if no extension ist given, .lmd ist appended

tape
Optional tape number

size
file size in MB, default value in auto mode is 50MB

-AUTO
automatic file creation. Names of the form namexxx.lmd are created
with consecutive numbers xxx. The first number is either given with
the first=yyy parameter or read from the file filenum.set

number
number of automatically generated files

first
sets the first file number to start with. If first is not specified, the first
file number is taken from the file filenum.set, which contains always the
last used file number

inhead
reads a goosy file header from the file specified Information from this file
is used in the GOOSY file header written to tape except of Filename,
Tape label and file creation date

outhead
writes goosy file header data to the specified file

-PROMPT
prompt for goosy file header data

-EDIT
this switch can be used together with the inhead command
option to edit a goosy file header read from a file

-DISK
Write to disk file.
### Description

**Function**
When the file is opened, incoming buffers are written. Normally one first opens the file and start the acquisition then. Opening and closing files does not affect the acquisition status.

**Routine**
f

**Task**
m\_transport
PROTECT

PROTECT HISTOGRAM

PROTECT HISTOGRAM name -UNPROTECT

PURPOSE (m_histogram) [Un]protect histogram from being cleared.

PARAMETERS

  name        name of histogram, also wildcard.
  -UNPROTECT  Enable histogram clearing

Description

  Function     [Un]protect histogram from being cleared.
  Routine      f_his_cmd_prohis
  Task         m_histogram
PSHELL

<table>
<thead>
<tr>
<th>PSHELL command [a1 a2 a3 a4]</th>
</tr>
</thead>
</table>

**PURPOSE**  
(m_prompt) Executes shell command line.

**PARAMETERS**

- **command**  
  Command to be executed.

- **a#**  
  Further arguments

**Description**

- **Function**  
  Use function system() to fork a command.

- **Example**  
  PSH ps -af

- **Routine**  
  f_ifa_shell

- **Task**  
  m_prompt
REMOTE

REMOTE ACCESS

REMOTE ACCESS [nodelist]

PURPOSE  (m_prompt) Grants command access from nodes.
PARAMETERS
nodelist     Nodelist

Description
  Function    Grant nodes command access by f_pr_send.
  Routine     f_pr_access
  Task        m_prompt

REMOTE RESET

REMOTE RESET [node]-ALL

PURPOSE  (m_prompt) Cleans up remote nodes. Remote program is m_remote.
PARAMETERS
node       Reset one node
-ALL       Reset all known nodes.
See also   RESET command of program remote.
Description

**Function**

Resets nodes by following steps:
1. Disconnect all dispatchers.
2. Kill all MBS tasks of user (remote only).
   - m dispatched
   - m_col_vme
   - m_collector
   - m_resone_serv
   - m_event_serv
   - m_read_cam_slav
   - m_read_meb
   - m_stream_serv
   - m_transport
   - m_util
   - m_msg_log
3. Create new msg files (remote only)
4. Clear DAQ status (remote only)

After RESET REM-ALL the tasks on the local node are still there. One must use standalone program m_remote "remote reset" to cleanup the local node, too. Other remote commands:

- **remote show**
  - Show remote tasks, message files and netstat.
- **remote reset**
  - Reset specified nodes.
- **Routine**
  - `f_ifa_reset`
- **Task**
  - `m_prompt`

### REMOTE SHOW

```
REMOTE SHOW [node][task]-LOCAL -TASKS -DAQ -NET
```

**PURPOSE**

(m_prompt) Shows remote info.

**PARAMETERS**

- **node**
  - Node name or @file (one name per line)
- **task**
  - Optional task for ps command
- **-LOCAL**
  - Local node
### -TASKS
- show task list

### -DAQ
- show DAQ status and message status

### -NET
- show net status

### See also
- SHOW command of program remote

### Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Shows remote info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routine</td>
<td>f_ifa_show_rem</td>
</tr>
<tr>
<td>Task</td>
<td>m_prompt</td>
</tr>
</tbody>
</table>
REMOVE

REMOVE SEGMENTS

PURPOSE (m_util) Removes critical segments
PARAMETERS
  none

Description
  Function remove critical segments
  Routine f_ut_remove_segments
  Task m_util
# RESET

**RESET**

<table>
<thead>
<tr>
<th>Command: RESET [node][task]-LOCAL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PURPOSE</strong></td>
</tr>
<tr>
<td><strong>PARAMETERS</strong></td>
</tr>
<tr>
<td>node</td>
</tr>
<tr>
<td>task</td>
</tr>
<tr>
<td>- m_dispatch</td>
</tr>
<tr>
<td>- m_col_vme</td>
</tr>
<tr>
<td>- m_collector</td>
</tr>
<tr>
<td>- m_esone_serv</td>
</tr>
<tr>
<td>- m_event_serv</td>
</tr>
<tr>
<td>- m_read_cam_slav</td>
</tr>
<tr>
<td>- m_read_meb</td>
</tr>
<tr>
<td>- m_stream_serv</td>
</tr>
<tr>
<td>- m_transport</td>
</tr>
<tr>
<td>- m_util</td>
</tr>
<tr>
<td>- m_msg_log</td>
</tr>
<tr>
<td>-LOCAL</td>
</tr>
<tr>
<td><strong>See also</strong></td>
</tr>
</tbody>
</table>

**Description**

**Function** | Kills MBS tasks, creates new message files and clears DAQ status. |
**Example** | shell> remote reset -l |
**Routine** | f_remote.c |
**Task** | m_remote |
RESTORE

RESTORE BASE

RESTORE BASE base file

PURPOSE (m_histogram) Restores base from file.

PARAMETERS

base name of segment = name of base
file name of file

Description

Function Restore base from file.
Routine f_his_cmd_resbas
Task m_histogram
SET

SET DISPATCHER

SET DISPATCHER [node]

PURPOSE (m_prompt) Sets terminal to remote dispatcher.

PARAMETERS

node Name of remote node. If omitted use local.

Description

Function Set prompt to node name, and route terminal IO to/from node through TCP.
Note, that for single commands the node can be specified by a node:: prefix, i.e.
cvc10::shotask
To send commands to all connected nodes, use *::command

Routine f_ifa_set_host
Task m_prompt

SET EVENT_SERV

SET EVENT_SERV [scale][events][maxclnt]-ALL

PURPOSE (m_event_serv) Sets m_event_serv parameters.

PARAMETERS

scale scaledown streams to analyse. 1: take all
events maximum number of events to copy from a stream
maxclnt maximum number of clients
-ALL take all events per selected stream
Description

**Function**
Set max events/stream in m_event_serv. This command can be used to reduce the load produced by the event server. The event server processes only the number of events specified per stream and frees the stream.

**Function**
Set max number of clients on m_event_serv.

**Function**
Similar to SET EVENTS this command may reduce the load produced by the event server. Specified number of streams is skipped without processing.

**Routine**

**Task**

---

**SET FILEHEADER**

```
SET FILEHEADER string [line]-RUNID -EXPERIMENT
-COMMENT -LABEL -FILENAME -USERNAME -CLEAR
```

**PURPOSE**
(m_transport) Specify fileheader information.

**PARAMETERS**

- **string**: Text according qualifier
- **-EXPERIMENT**: Experiment
- **-RUNID**: Run id.
- **-COMMENT**: Comment line (line=# optionally overwrites line).
- **-LABEL**: Label (default)
- **-FILENAME**: Filename (default)
- **-USERNAME**: Username (default)
- **-CLEAR**: Disables this info.
Description

Function  Specify fileheader information for next 'open file' command. This information can be input interactively by 'open file -prompt'.

Routine  f_

Task  m_transport

SET FLUSHTIME

SET FLUSHTIME time

PURPOSE  (m_collector) Sets stream flush time

PARAMETERS

time  stream flush time in seconds

Description

Function  set stream flush time

Routine  f_col_set_flush_time

Task  m_collector

SET HISTOGRAM CHANNEL

SET HISTOGRAM CHANNEL name value xchan [ychan]-INCREMENT

PURPOSE  (m_histogram) Sets channel content of histogram.

PARAMETERS

name  histogram name

value  value or increment

xchan  x-channel (1st dim)

ychan  y-channel (2nd dim)

-INCREMENT  Add value.
Description

Function Set or increment channel content of histogram.
Routine f_his_cmd_setcha
Task m_histogram

SET HISTOGRAM TEXT

SET HISTOGRAM TEXT name [text]-TITLE -XTXT -YTXT -CONT

PURPOSE (m_histogram) Sets lettering text field of histogram.

PARAMETERS

name histogram name
text value or increment
-TITLE Lettering on histogram title
-XTXT Lettering on x-axis
-YTXT Lettering on y-axis
-CONT Lettering on data content

Description

Function Sets lettering text field of histogram.
Routine f_his_cmd_settxt
Task m_histogram

SET MAXCLIENTS ESONE_SERV

SET MAXCLIENTS ESONE_SERV maxclnt

PURPOSE (m_esone_serv) Sets maximum number of clients for m_esone_serv.

PARAMETERS

maxclnt maximum number of clients
Description

**Function** Set max number of clients on m_esone_serv.

**Routine** \texttt{f_escmd_setmaxclnt}

**Task** m_esone_serv

**SET MAXCLIENTS EVENT_SERV**

\begin{verbatim}
SET MAXCLIENTS EVENT_SERV maxclnt

PURPOSE (m\_event\_serv) Sets maximum number of clients for m\_event\_serv.

PARAMETERS
\begin{itemize}
  \item \texttt{maxclnt} maximum number of clients
\end{itemize}
\end{verbatim}

Description

**Function** Set max number of clients on m\_event\_serv.

**Routine** \texttt{f_ev\_cmd_setmaxclnt}

**Task** m\_event\_serv

**SET PRINT ESONE_SERV**

\begin{verbatim}
SET PRINT ESONE_SERV -ON -OFF

PURPOSE (m\_esone_serv) Sets terminal output Esone data for Esone Server.

PARAMETERS
\begin{itemize}
  \item \texttt{-ON} print C N A F on.
  \item \texttt{-OFF} print C N A F off (default)
\end{itemize}
\end{verbatim}

Description

**Function** Set terminal output Esone data for m\_esone_serv.

**Routine** \texttt{f_escmd_setprint}

**Task** m\_esone_serv
SET STREAM_SERV

SET STREAM_SERV [scale]-[NO]SYNC-[NO]KEEP-SCALED_KEEP-CLEAR

PURPOSE (m_stream_serv) Selects scaledown of streams.

PARAMETERS

scale  scaledown streams to client. 1: send all, when requested. Default after startup is 2!

-[ NO] SYNC  synchronously mode: send all scaled, wait for request.

-[ NO] KEEP  Keep streams when scale=1 and when there are > 3 free streams.

-SCALED_KEEP  Keep scaled down streams when there are > 3 free streams.

-CLEAR  Clear counters.

REMARKS  Mode after startup: SET STREAM 2 -NOKEEP-NOSYNC

Description

Function  This command may reduce the load produced by the stream server. Specified number (scale) of streams is skipped without processing. There are several strategies to avoid the problem that between spills all streams are passed, and the clients do not get any more streams, even if both, client and mbs, are idle. The general mechanism is that the server checks if there are enough (3) free streams. If yes it yields and gives the client the chance to request the stream. Between spills the client would then get all streams, but also, if the streams are always free (slow data rate). This behavior is modified in two variants: When -KEEP is used, free streams are only checked, if scale=1. Otherwise no streams are kept and the client will not get streams between spills. When -SCALE is used, for any nth stream the number of free streams is checked. This is a compromise to give the client the chance to get streams between spills, but still use scaling for not slowing down the acquisition. Which method is the best depends very much on the special situation. One should play with the parameters to get the desired behavior (transfer rate against DAQ slow down). The default mode after startup is

SET STREAM 2 -NOKEEP-NOSYNC
REMARK -SYNC switches off any keep mechanism.

NOTE Arguments and switches not specified are not changed.

Routine f_str_setstr

Task m_stream_serv

SET TASK

SET TASK task pid -CLEAR

PURPOSE (m_dispatch) Sets task id.

PARAMETERS

  task  Task (program) name
  pid   pid of task. Normally tasks write their pid in s_daqst control structure and m_dispatch reads it from there. If not, SET TASK can be used to set the pid which is used for command channels.

-CLEAR Removes tasks from the command list which are marked not active in the daq status table.

NOTE: This qualifier MUST NOT given as long as there are tasks starting! It should be used when tasks abort abnormally.

Description

Function This command is not needed under normal conditions. It could be used, if the task is known to the dispatcher but is started elsewhere. Then its pid can be set in the dispatcher and the task accepts command from dispatcher. The task MUST NOT, however, stopped by STOP TASK command.

Routine f_disp_set_tsk

Task m_dispatch
SET TRIG_MOD

SET TRIG_MOD -SLAVE

PURPOSE (m_util) Sets trigger module.

PARAMETERS

-SLAVE Trigger module is set to slave.

Description

Function Set the trigger module parameters.
Routine f_ut_op_trig_mod
Task m_util

SET VERBOSE DISPATCHER

SET VERBOSE DISPATCHER -ON -OFF

PURPOSE (m_dispatch) Sets verbosity for dispatcher.

PARAMETERS

-ON verbosity on.
-OFF verbosity off (default)

Description

Function Set verbosity of messages for dispatcher.
Routine f_dispVerbose
Task m_dispatch
SET VERBOSE ESONE_SERV

SET VERBOSE ESONE_SERV -ON -OFF

PURPOSE (m_esone_serv) Sets verbosity for Esone Server.

PARAMETERS
  -ON verbosity on.
  -OFF verbosity off (default)

Description
Function Set verbosity of messages for m_esone_serv.
Routine f_es_cmd_setverb
Task m_esone_serv

SET VERBOSE EVENT_SERV

SET VERBOSE EVENT_SERV -ON -OFF

PURPOSE (m_event_serv) Sets verbosity for m_event_serv.

PARAMETERS
  -ON verbosity on.
  -OFF verbosity off (default)

Description
Function Set verbosity of messages for m_event_serv.
Routine f_ev_cmd_setverb
Task m_event_serv
SET VERBOSE GLOBAL

PURPOSE (m_util) Sets verbosity for all tasks

PARAMETERS

-ON verbosity on.
-OFF verbosity off (default)
-NEUTRAL verbosity is like set in specific task

Description

Function Set verbosity of all tasks
Routine f_ut_set_gleob_vebose
Task m_util

SET VERBOSE HISTOGRAM

PURPOSE (m_histogram) Sets verbosity for m_histogram.

PARAMETERS

-ON verbosity on.
-OFF verbosity off (default)

Description

Function Set verbosity of messages for m_histogram.
Routine f_his_cmd_setverb
Task m_histogram
SET VERBOSE PROMPT

SET VERBOSE PROMPT -ON -OFF

PURPOSE (m_prompt) Sets verbosity for m_prompt.

PARAMETERS
- -ON verbosity on.
- -OFF verbosity off (default)

Description
- Function Set verbosity of messages for m_prompt.
- Routine f_pr_verbose
- Task m_prompt

SET XDISPLAY

SET XDISPLAY node

PURPOSE (m_prompt) Set name of remote display.

PARAMETERS
- node Node where a remote display runs.

Description
- Function Set name of remote display.
- Routine f_pr_set_disp
- Task m_prompt
SHOW

SHOW [node][task]-LOCAL -TASKS -DAQ -NET

PURPOSE  (m(remote)) Shows remote info. Executed by alias remote.

PARAMETERS

node  Node name or @file (one name per line)
task  optional task for ps command
-LOCAL  local node
-TASKS  show task list
-DAQ  show DAQ status and message status
-NET  show net status
See also  REMOTE SHOW command of prompter

Description
Function  Shows remote info.
Example  shell> remote show -l
Routine  f_remote.s
Task  m_remote

SHOW ACQUISITION

SHOW ACQUISITION [seconds]-SETUP -CRATES -SERVER -RATE -LOG

PURPOSE  (m_util) Shows acquisition.

PARAMETERS
seconds wait n seconds to show rates (def=1).

-SETUP Show setup
-CRATES Show crates
-SERVER Show data transfers
-RATE Show data rates
-LOG write output also to log file

Description
Function Show acquisition. When no switch is given, all are output.
Routine f_ut_show_acq
Task m_util

SHOW BASE

SHOW BASE [name]

PURPOSE (m_histogram) Shows information about histogram data base.
PARAMETERS
name name of segment

Description
Function Show information about histogram data base.
Routine f_his.cmd_shobas
Task m_histogram
SHOW COMMANDS

SHOW COMMANDS [task]-FULL -ALL

PURPOSE  (m_dispatch) Shows known commands.

PARAMETERS
  task          Optional task name to show commands
  -FULL         Commands with arguments.
  -ALL          All commands with arguments

Description
  Function      Show all known commands (including the commands of inactive tasks by -ALL).
  Routine       f_disp_sho_cmd
  Task          m_dispatch

SHOW CVC_IRQ_MASK

SHOW CVC_IRQ_MASK

PURPOSE  (m_util) Reads irq mask of the CVC irq controller

PARAMETERS
  none

Description
  Function      Read irq mask of the CVC irq controller
  Routine       f_ut_cvc_cam_irq
  Task          m_util
SHOW DISPATCHER

SHOW DISPATCHER [node]

PURPOSE (m_prompt) Shows connections to remote dispatchers.

PARAMETERS

node Name of remote node.

Description

Function Show connections to remote dispatchers.
Routine f_ifa_show_connect
Task m_prompt

SHOW ENVIRONMENT

SHOW ENVIRONMENT

PURPOSE (m_dispatch) Shows environment parameters.

PARAMETERS

Description

Function Show environment parameters of dispatcher.
Routine f_disp_sho_env
Task m_dispatch

SHOW ESONE_SERV

SHOW ESONE_SERV -FULL -LOG -CLIENT

PURPOSE (m_esone_serv) Shows status of m_esone_serv.

PARAMETERS
-FULL  full output
-LOG   output to log file
-CLIENT show status of each client

Description
Function  Show status of m_event_serv.
Routine   f_es_cmd_showsts
Task      m_esone_serv

SHOW EVENT_SERV

SHOW EVENT_SERV -FULL -LOG -CLIENT

PURPOSE  (m_event_serv) Shows status of m_event_serv.
PARAMETERS
-FULL   full output
-LOG    output to log file
-CLIENT show client info

Description
Function  Show status of m_event_serv.
Routine   f_ev_cmd_showsts
Task      m_event_serv

SHOW HISTOGRAM

SHOW HISTOGRAM name -DATA -FULL

PURPOSE  (m_histogram) Shows histogram info and content.
PARAMETERS
name  name of histogram, also wildcard.
-DATA  show histogram data
-FULL  show full histogram parameters

Description
Function  Show histogram info and content.
Routine  f_his.cmd_shohis
Task  m_histogram

SHOW MESSAGE

PURPOSE  (m_dispatch) Shows status of internal message file.
PARAMETERS

Description
Function  There are two message files: One for commands and one for messages. These files are used for intertask communication.
Routine  f_disp_sho_mes
Task  m_dispatch

SHOW ML_SETUP

PURPOSE  (m_util) Shows multilayer setup parameters
PARAMETERS
  none
SHOW

Description
Function parameters from memory
Routine f_ut_print_ml_setup
Task m_util

SHOW RATE

SHOW RATE [seconds]-OFF -ON

PURPOSE (m_util) Shows acquisition rate.

PARAMETERS

seconds If 0, show counters.
If > 0, wait n seconds and show rates.
Also sets interval for rate program.

-OFF switches rate program off
-ON switches rate program on

Description
Function Show counters and rates.
Routine f_ut_rate
Task m_util

SHOW SETUP

SHOW SETUP

PURPOSE (m_util) Shows setup parameters

PARAMETERS none
Description

Function       SHOW SETUP parameters from memory
Routine        f_ut_print_setup
Task           m_util

SHOW STATUS

SHOW STATUS

PURPOSE  (m_util) Shows daqst parameters
PARAMETERS  none

Description

Function       daqst parameters from memory
Routine        f_ut_print_daqst
Task           m_util

SHOW STREAM_SERV

SHOW STREAM_SERV -CLEAR

PURPOSE  (m_stream_serv) Shows modes and counters.
PARAMETERS
   -CLEAR        Clear counters.

Description

Function       This command shows counters and modes of the stream server. The
counters are incremented only if a client is connected.
Routine        f_str_shostr
Task           m_stream_serv
SHOW TAPE

SHOW TAPE [tape]

PURPOSE
(m_transport) Shows tape information.

PARAMETERS

   tape  Optional tape number

Description

Function
Shows current tape device, current tape label, file name and number of kBytes written already to tape.

Routine
f_show_tape

Task
m_transport

SHOW TASK

SHOW TASK [task]-FULL -ALL

PURPOSE
(m_dispatch) Shows known tasks.

PARAMETERS

   task  Optional task (program) name. When specified, show commands of this task.

-FULL
Show commands with arguments

-ALL
Show also inactive tasks

Description

Function
Outputs list of known tasks. When the task is marked "running" it accepts commands and could be terminated by command STOP TASK. Otherwise it is not started or runs "detached", i.e. does not execute commands.

Routine
f_disp_sho_tsk

Task
m_dispatch
SHOW TRIG_MOD

PURPOSE (m_util) Shows current setup of trigger module.

PARAMETERS
  none

Description
  Function       returns current setup of trigger module.
  Routine        f_ut_op_trig_mod
  Task           m_util
START

START ACQUISITION

PURPOSE (m_util) Starts acquisition.

PARAMETERS

xxx

Description

Function

This command assumes that the trigger module is setup properly (by command SET TRIG_MOD or after STOP ACQUIS). Readout tables must be loaded. A software trigger number 14 is delivered and the deadtime is reset. The readout tasks get the trigger 14 as first event. They may do specific actions, collect some data and write a start event.

Routine            f_ut_op_trig_mod
Task               m_util

START EVENT_SERV

START EVENT_SERV [scale][events][maxclnt]
- [NO]PORTS -VERB

PURPOSE (m_event_serv) Starts m_event_serv.

PARAMETERS

scale      scaledown streams to analyse. 1: take all
events     maximum number of events to copy from a stream.
maxclnt    max number of clients
-NOPORTS  do not send information to portserver. (default)
-PORTS    send information to portserver.
-VERB     verbosity on.

Description
- Function abc
- Routine f.ev.cmd.start
- Task   m.event_serv

START MESSAGE

START MESSAGE

PURPOSE (m.dispatch) Starts the message logger.

Description
- Function The message logger is required before any other task can be started or any command can be executed.
- Routine f.disp.sta.msg
- Task   m.dispatch

START TASK

START TASK task [file]

PURPOSE (m.dispatch) Starts task.

PARAMETERS
- task  Task (program) name. Optionally with path.
  Default path is /MBSROOT/bin_HOSTTYPE.
- file  optional path
Description

Function
Start a task (program) by forking. When the task is known to the dispatcher through command definitions, the program is forked directly and the status is set to running. These tasks can be terminated by command STOP TASK. When the task is not known, it is forked through an intermediate task which terminates. The new task is entered to the task list, but not marked running but detached thus indicating that it does not accept commands. These tasks can be stopped by command STOP TASK dummy pid -KILL. It is assumed that they exit when their job is done.

Routine
f disp sta tsk

Task
m dispatch
STOP

STOP ACQUISITION

PURPOSE (m_util) Stops acquisition.

PARAMETERS

none

Description

Function This command delivers software trigger 15. Then the readout tasks recognise this event as last event. The master readout keeps the dead-time thus blocking further triggers. The collector frees the current buffer stream when event type 15 occurs. The status of the system is set to STOP PENDING until the transport acknowledges the last event and finishes the acquisition stop.

Routine f_ut_op_trig_mod

Task m_util

STOP TASK

STOP TASK [task][pid]-ALL -KILL -ZOMBIE

PURPOSE (m_dispatch) Stops task by sending command "exit"

PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task</td>
<td>Task (program) name</td>
</tr>
<tr>
<td>pid</td>
<td>Optional pid used with -KILL if task not known.</td>
</tr>
<tr>
<td>-ALL</td>
<td>Stop all known tasks except message logger.</td>
</tr>
</tbody>
</table>
-KILL Stop task by calling kill function with signal SIGTERM.

-ZOMBIE Remove status blocks of terminated tasks to let them die (Z to RIP).

Description

Function
Without -KILL: Sends command "exit" to the task. The task command thread exits and the main program exits. Only tasks running commands can be terminated by this way. Other tasks can only be stopped with -KILL.

zombies
When a task terminates by itself, it goes into status Z(ombie), because the father must remove the termination status block. With the -Z switch the dispatcher looks through the task table and tries to remove pending termination status blocks. This is only possible for childs. When the dispatcher has been quit, all childs get the init task as father. Now the restarted dispatcher cannot remove their status block anymore.

Routine
fdisp_sto_tsk

Task
mdispatch
TYPE

TYPE EVENT

TYPE EVENT [events id]-SAMPLE -VERBOSE

PURPOSE 
(m_transport) Prints events.

PARAMETERS

events 
number of events to print, default = 1

id
subevent id for printing only subevents of type id

-SAMPLE
print only first events of every buffer

-VERBOSE
print also event data

Description

Function
Prints event headers and event data

Routine
f_

Task
m_transport
VOID

VOID

PURPOSE     (m\_daq\_rate) place holder, do not execute.

PARAMETERS

Description
  Function       This command must not be executed.
  Routine
  Task          m\_daq\_rate
X SHELL

**X SHELL command [a1 a2 a3 a4]**

<table>
<thead>
<tr>
<th><strong>PURPOSE</strong></th>
<th>(m_dispatch) Executes shell command line.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PARAMETERS</strong></td>
<td></td>
</tr>
<tr>
<td>command</td>
<td>Command to be executed.</td>
</tr>
<tr>
<td>a#</td>
<td>Further arguments</td>
</tr>
</tbody>
</table>

**Description**

<table>
<thead>
<tr>
<th><strong>Function</strong></th>
<th>Use function system() to fork a command.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example</strong></td>
<td>X ps -af</td>
</tr>
<tr>
<td><strong>Routine</strong></td>
<td>f_disp_shell</td>
</tr>
<tr>
<td><strong>Task</strong></td>
<td>m_dispatch</td>
</tr>
</tbody>
</table>
Chapter 5

Related GOOSY Commands
CAMAC

CAMAC CNAF

CAMAC CNAF C=c N=n A=a F=f DATA=d Branch=b

**PURPOSE**
Perform a single CAMAC action

**PARAMETERS**

- **C=c**
  Crate number, between 1 and 7.
  Replaceable default = 1

- **N=n**
  Station number, between 1 and 31.
  Replaceable default = 1

- **A=a**
  Subaddress, between 0 and 15.
  Replaceable default = 0

- **F=f**
  Function code, between 0 and 31.
  Replaceable default = 0

- **DATA=d**
  Data word. Should be specified if the function code is between 16 and 23 and will be ignored otherwise.
  NOTE: The data word may be entered also in binary (e.g. %B101),
  octal (e.g. %O123), or hexadecimal (e.g. %X1A1B) if convenient.
  Replaceable default = 0

- **Branch=b**
  Branch number. Used for the translation of
  CAMAC\_BRANCH\_b

**EXAMPLE**
CAMAC CNAF C=1 N=12 A=3 F=16 DATA=%O777

**Action rout.**
I$MCCNA

**Author**
Walter F.J. Mueller

**Remarks**

**File name**
I$MCCNA.PPL

**REMARKS**
-

110
Version 4.3 October 28, 2003
Description

CALLING

@CALL I$MCCNA(I_C,I_N,I_A,I_F,L_DATA);

ARGUMENTS

I_C BIN FIXED(15) [INPUT]
    Crate number, between 1 and 7.

I_N BIN FIXED(15) [INPUT]
    Station number, between 1 and 31.

I_A BIN FIXED(15) [INPUT]
    Subaddress, between 0 and 15.

I_F BIN FIXED(15) [INPUT]
    function code, between 0 and 31.

L_DATA BIN FIXED(31) [INPUT]
    Data word to be written if function is between 16 and 23, ignored otherwise.

I_B BIN FIXED(15) [INPUT]
    Branch number. This number is used for the translation of CAMAC_BRANCH_b.

FUNCTION

This procedure calls I$CFSA to execute the given CAMAC action and displays the result.

REMARKS

-

EXAMPLE

@CALL I$MCCNA(1,1,0,16,1);
CONNECT

CONNECT STREAM

CONNECT STREAM node /NOSTART

PURPOSE Connect to stream server of Multi Branch System.

PARAMETERS

node string replace default=""
   Internet node

/NOSTART Do not start acquisition

FUNCTION Connect to stream server and waits for a 4 longword buffer. 1.=endian (=1), 2.=buffer size, 3.=buffers per stream, 4.=streams. When buffer size does not match, disconnect. When there are less free buffers then in stream, disconnect, otherwise start acquisition.

CALLING @CALL I$ACTCP_DISCON(CV_node);

PURPOSE Called by command DISCONNECT TRANSPORT.

Description

FUNCTION Connect to stream server of Multi Branch System.

File name I$ACTCP.PPL

Action rout. I$ACTCP_CONS

Version 1.01

Author H.G.Essel

Last Update 20-Jul-1994
CONNECT TRANSPORT

CONNECT TRANSPORT node /NOSTART

PURPOSE
Connect to transport task of Multi Branch System.

PARAMETERS

node     string replace default=""
          Internet node
/NOSTART Do not start acquisition

FUNCTION
Connect to transport task and waits for a 4 longword buffer. 1.=endian (=1), 2.=buffer size, 3.=buffers per stream, 4.=streams. When buffer size does not match, disconnect, otherwise start acquisition.

CALLING
@CALL I$ACTCP_CONS(CV_node,INostart);

PURPOSE
Connect to MBS stream server.

Description

FUNCTION
Connect to transport task of Multi Branch System.

File name    I$ACTCP.PPL
Action rout.  I$ACTCP_CON
Version       1.01
Author        H.G.Essel
Last Update   20-Jul-1994
DISCONNECT

DISCONNECT STREAM

**DISCONNECT STREAM node**

**PURPOSE**
Disconnect from stream server and stop acquisition.

**PARAMETERS**

- **node**
  - string replace default="Internet node"

Description

**FUNCTION**
Disconnect from stream server of Multi Branch System.

**File name**
I$ACTCP.PPL

**Action rout.**
I$ACTCP_DISCON

**Version**
1.01

**Author**
H.G.Essel

**Last Update**
20-Jul-1994

DISCONNECT TRANSPORT

**DISCONNECT TRANSPORT node**

**PURPOSE**
Disconnect from transport and stop acquisition.

**PARAMETERS**

- **node**
  - string replace default="Internet node"

**CALLING**
@CALL I$ACTCP_DISCON(CV_node);

**PURPOSE**
Called by command DISCONNECT STREAM.
**Description**

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>Disconnect from transport of Multi Branch System.</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>I$ACTCP.PPL</td>
</tr>
<tr>
<td>Action rout.</td>
<td>I$ACTCP_DISCON</td>
</tr>
<tr>
<td>Version</td>
<td>1.01</td>
</tr>
<tr>
<td>Author</td>
<td>H.G.Essel</td>
</tr>
<tr>
<td>Last Update</td>
<td>20-Jul-1994</td>
</tr>
</tbody>
</table>
INITIALIZE

INITIALIZE ACQUISITION

INITIALIZE ACQUISITION mailbox size count
in_buffers out_buffers
node command data
/VME /TCP /MBD /J11 /FILE /RANDOM /FOREIGN
/PAGE /BYTE /KBYTE
/MBX

PURPOSE
Init data taking

PARAMETERS

mailbox string replace
String to build mailbox name:
GOOSY_mailbox_1,2,3
Three mailboxes are created.
Default is the environment name. The name must be unique. This is
ensured by using the environment name.

size integer replace default=8192
Size of listmode data buffers in bytes, must be
between 512 and 28672. The size should be a multiple of 512 or even
better a multiple of 4096. Together with the /PAGE /KBYTE qualifier
this quantity can be specified in different units than bytes. If zero,
defaults to 16384 for VME, 8192 others.

count integer default=8
Number of listmode data buffers, must be between 4
and 32. The default is normally adequate.

in_buffers integer default=4
Number of buffers queued to MBD.
The default is normally adequate.

out_buffers integer default=0
Number of buffers hold before writing to the output
device is started. The default is normally adequate.
node  string replace
       Node of J11 (only needed for /J11)

command  string replace default=CMDERR
       Command component of J11 (/J11 only)
The default is adequate.
data  string replace default=TMR11S
       Data component of J11 (/J11 only)
The default is adequate.

/MBX  Open input mailbox. String to build mailbox name:
       GOOSY_mailbox_I1

/VME  Input channel VME

/TCP  Input channel is TCP server (single branch system)

/MBD  CAMAC interface is MBD

/J11  CAMAC interface is J11

.FILE  Input from file (use OPEN-CLOSE FILE)

/FOREIGN  Input from unknown source.

/RANDOM  Generates random data type 4/1.

/PAGE  Buffer size in pages (512 bytes)

/BYTE  Buffer size in bytes

/KBYTE  Buffer size in Kbytes (1024 bytes)

EXAMPLE  INIT ACQU ANNA SIZE=8192 COUNT=12 IN_BUF=6
       creates mailboxes GOOSY_ANNA_1,2,3

Description

FUNCTION  This procedure initializes the buffer pool used for listmode data. The
given number of buffers is allocated and locked in the workingset.
NOTE: The number and size of buffers cannot be changed afterwards
whereas the queuing parameters may be modified.
       In J11 mode the links to the J11 are opened.
he mailboxes are created. The size of the mailboxes ust be 8192 for the
J11 input. This value is efaulted with the /J11 qualifier. With /MBD
the ailbox size default is 4096.
<table>
<thead>
<tr>
<th><strong>File name</strong></th>
<th>I$ACQ_INI.ACQ.PPL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Action rout.</strong></td>
<td>I$ACQ_INI.ACQ</td>
</tr>
<tr>
<td><strong>Version</strong></td>
<td>1.01</td>
</tr>
<tr>
<td><strong>Author</strong></td>
<td>Walter F.J. Mueller</td>
</tr>
<tr>
<td><strong>Last Update</strong></td>
<td>12-APR-1985</td>
</tr>
</tbody>
</table>
START

START ACQUISITION

START ACQUISITION buffers events
    skip_buf skip_event
    /CLEAR /NET /STOP /RESET

PURPOSE
Start data taking

PARAMETERS

buffers
    integer default=0
    Number of buffers to process (0 = infinite)
    (file input only)

events
    integer default=0
    Number of events to process (0 = infinite)
    (file input only)

skip_buf
    integer default=0
    Number of buffers to skip
    (file input only)

skip_event
    integer default=0
    Number of events to skip
    (file input only)

/NET
    switch
    Start input from net (VME)

/CLEAR
    switch
    Clear counter. Same effect as SHOW/CLEAR.
    Some counters are cleared by START ACQUISITION, others by SHOW/CLEAR
    or START ACQ/CLEAR.

/STOP/RESET
    switch
    Break stopping status. Using VME frontends, the
    STOP ACQ command will enter stopping status which terminated by
    last buffer. When the frontend system does not send buffers stopping
status would never be terminated. In this case, /STOP or /RESET can be used to terminate this status.

EXAMPLE  STA ACQU

Description

FUNCTION  This procedure starts the data acquisition. The MBD and all front end processors are started. MBD is initialized by I$ACQ_IMBD. Input is queued by delivering I$ACQ_QMBD on AST level.

For a J11 system the J11 is started.
The buffer and event counters are cleared.

File name  I$ACQ_STA_ACQ.PPL
Action rout.  I$ACQ_STA_ACQ
Dataset  -
Version  1.01
Author  Walter F.J. Mueller
Last Update  12-APR-1985

START INPUT EVENT

START INPUT EVENT server events filter sample
   flush port
   /PROMPT
   /RESET

PURPOSE  Open event stream from server

PARAMETERS

  server  Name of server, or node where server runs [update]
  events  Number of events to process [update]
  filter  Filter file.
  sample  Sampling event rate
  flush  Buffer flashing time (sec)
<table>
<thead>
<tr>
<th><strong>port</strong></th>
<th>Port number, when server is specified as node or 0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>/PROMPT</strong></td>
<td>Prompt for filter</td>
</tr>
<tr>
<td><strong>/RESET</strong></td>
<td>When stop did not complete</td>
</tr>
</tbody>
</table>

**EXAMPLE**

**Description**

**FUNCTION**

This procedure opens a link to event server to get events.

**File name**

I$ANACM.PPL

**Action rout.**

I$ANACM_CON

**Version**

1.01

**Author**

H.G.Essel

**Last Update**

12-APR-1985
STOP

STOP ACQUISITION

STOP ACQUISITION /ABORT /CLOSE /STOP/RESET

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>Stop data taking</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARAMETERS</td>
<td>-</td>
</tr>
<tr>
<td>/ABORT</td>
<td>Delete links to J11 and cleanup (J11 only). All buffers in the J11 are lost.</td>
</tr>
<tr>
<td>/CLOSE</td>
<td>Close input file (File input only)</td>
</tr>
<tr>
<td>/STOP/RESET</td>
<td>Do not wait for last buffer.</td>
</tr>
</tbody>
</table>

**FUNCTION**

With the MBD as frontend all data in the MBD is sent to the VAX. All data is written to file, if output is active. With a single crate J11 system the /ABORT qualifier clears all data in the J11.

**Description**

**FUNCTION**

MBD inputs are canceled. J11 is stopped. Mailbox QIO’s are canceled.

**File name**

I$ACQ_STO_ACQ.PPL

**Action rout.**

I$ACQ_STO_ACQ

**Dataset**

-

**Version**

1.01

**Author**

Walter F.J. Mueller

**Last Update**

12-APR-1985
STOP INPUT EVENT

STOP INPUT EVENT server

PURPOSE               Close event stream from server
PARAMETERS
    server           Name of server, or node where server runs
EXAMPLE

Description
    FUNCTION        This procedure closes a link to event server
    File name       I$ANACM.PPL
    Action rout.    I$ANACM.DISCON
    Version         1.01
    Author          H.G.Essel
    Last Update     12-APR-1985
Chapter 6

Histogramming and Analysis Functions
f_his_anal

\[
\text{l_status = f_his_anal(ps_form)}
\]

**PURPOSE**
Template for analysis program.

**ARGUMENTS**
- **ps_form**
  (struct s_form *) input
  pointer to structure

**Return type**
int

- **ERR_SUCCESS**
  Successfull completion
- **ERR_ERROR**
  Error: Analysis should be disabled
- **ERR_HIS_EANAL**
  Error in analysis, no disable
- **ERR_HIS_SKIPEVT**
  Skip event
- **ERR_HIS_SKIPSEV**
  Skip subevent
- **other error value**
  see **ERR_ERROR**

**Description**

**CALLING**
\[
l_status = f_his_accu(ps_form)
\]

**FUNCTION**
Template for analysis program.
f_his_anal_ini

\[
l\_status = \text{f\_his\_anal\_ini}(\text{ps\_head}, \text{pps\_spe}, \text{ppl\_data}, \text{c\_hisname})
\]

**PURPOSE**

Initialize histogram for analysis routine.

**ARGUMENTS**

- **ps\_head**
  - (struct \_head *) input
  - pointer to header structure
- **pps\_spe**
  - (struct \_spe **) output
  - address of pointer to spectrum header
- **ppl\_data**
  - (long **) output
  - address of pointer to data block
- **c\_hisname**
  - (char *) input
  - pointer to histogram name

**Return type**

int

**Description**

**CALLING**

\[
l\_status = \text{f\_his\_anal\_ini}(\text{ps\_head}, \text{pps\_spe}, \text{ppl\_data}, \text{c\_hisname})
\]

**FUNCTION**

Initialize histogram for analysis routine.
f_his_accu1

\[
\text{l\_status} = \text{f\_his\_accu1}(\text{ps\_spe}, \\
\quad \text{pl\_data}, \\
\quad \text{r\_energy}, \\
\quad \text{r\_incr})
\]

**PURPOSE**
Accumulate histogram dim=1 in analysis routine.

**ARGUMENTS**

- **ps\_spe** (struct spe *) input
  pointer to spectrum header
- **pl\_data** (long *) input
  pointer to data block
- **r\_energy** (float) input
  energy value. (channel will be calculated)
- **r\_incr** (float) input
  increment value.

**Return type**
int ERR\_SUCCESS : always

**Description**

**CALLING**
\[
\text{l\_status} = \text{f\_his\_accu1}(\text{ps\_spe}, \\
\quad \text{pl\_data}, \\
\quad \text{r\_energy}, \\
\quad \text{r\_incr})
\]

**FUNCTION**
Accumulate histogram dim=1 in analysis routine.
f_his_accu2

\[
l_{\text{status}} = \text{f}_\text{his}_\text{accu2}(\text{ps}_\text{spe}, \\
                   \text{pl}_\text{data}, \\
                   \text{r}_\text{energy1}, \\
                   \text{r}_\text{energy2}, \\
                   \text{r}_\text{incr})
\]

**PURPOSE**
Accumulate histogram dim=2 in analysis routine.

**ARGUMENTS**

- **ps**
  - `ps` (struct s_spe *) input
  - pointer to spectrum header

- **pl**
  - `pl` (long *) input
  - pointer to data block

- **r_energy1**
  - (float) input
  - energy value dim=1. (channel will be calculated)

- **r_energy2**
  - (float) input
  - energy value dim=2. (channel will be calculated)

- **r_incr**
  - (float) input
  - increment value.

**Return type**
int ERR_SUCCESS : always

**Description**

**CALLING**
\[
l_{\text{status}} = \text{f}_\text{his}_\text{accu2}(\text{ps}_\text{spe}, \\
                   \text{pl}_\text{data}, \\
                   \text{r}_\text{energy1}, \\
                   \text{r}_\text{energy2}, \\
                   \text{r}_\text{incr})
\]

**FUNCTION**
Accumulate histogram dim=2 in analysis routine.
f_his_gethist

<table>
<thead>
<tr>
<th>Function: f_his_gethist</th>
<th>(CHARS *pc_node, CHARS *pc_basename, CHARS *pc_hisname, CHARS **ppc_buf, INTS4 *pl buflen)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purpose</td>
<td>Get histogram data from histogram server.</td>
</tr>
<tr>
<td>Arguments:</td>
<td></td>
</tr>
<tr>
<td>pc_node</td>
<td>(CHARS *) input</td>
</tr>
<tr>
<td></td>
<td>pointer to node name</td>
</tr>
<tr>
<td>pc_basename</td>
<td>(CHARS *) input</td>
</tr>
<tr>
<td></td>
<td>pointer to shared memory segment’s name</td>
</tr>
<tr>
<td>pc_hisname</td>
<td>(CHARS *) input</td>
</tr>
<tr>
<td></td>
<td>pointer to histogram name</td>
</tr>
<tr>
<td>ppc_buf</td>
<td>(CHARS **) output</td>
</tr>
<tr>
<td></td>
<td>pointer to pointer to s.pe, the histogram header,</td>
</tr>
<tr>
<td></td>
<td>must be NULL when calling, it’s user’s responsibity to release this memory</td>
</tr>
<tr>
<td></td>
<td>Histogram data follows the header.</td>
</tr>
<tr>
<td>pl buflen</td>
<td>(INTS4) output</td>
</tr>
<tr>
<td></td>
<td>the returned buffer’s length</td>
</tr>
<tr>
<td>Return type</td>
<td>INTS4</td>
</tr>
</tbody>
</table>

Description

**FUNCTION**

Get histogram data from histogram server. In the returned buffer there might be several histograms. Each histogram has a header (s.pe) and a data part.

**Status codes**

- GETHIS_SUCCESS: success
- GETHIS_BUFNN: ppc_buf is not NULL.
- GETHIS_NOMEM: malloc() fail.
- STC_***: network error
f_his_getdir

f_his_getdir(CHARS *pc_node, CHARS *pc_basename, CHARS **ppc_buf, INTS4 *pl_buflen)

PURPOSE
Get histogram list in a base from histogram server.

ARGUMENTS

pc_node
(CHARS *) input
pointer to node name

pc_basename
(CHARS *) input
pointer to shared memory segment’s name

ppc_buf
(CHARS **) output
pointer to pointer to s_head, which is the header of a histogram base. It is followed by the histogram headers, i.e. structures s_spe. Pointer must be NULL when calling, it’s user’s responsibility to release this memory

pl_buflen
(INTS4) output
the returned buffer’s length

Return type
INTS4

Description

FUNCTION
Get histogram directory in a base from histogram server. The structure is
s_head
s_spe
s_spe
...

Status codes
GETHIS_SUCCESS success
GETHIS_BUFNN ppc_buf is not NULL.
GETTHIS_NOMEM    malloc() fail.
STC_***          network error
Chapter 7

Esone Client Functions
### int cccc(long EXT)

**PURPOSE**

generate dataway clear

**Description**

**ARGUMENTS**

- **EXT**
  
  external address of a camac crate in encode format produced by function cdreg

- **return value**
  
  return -1 if X = 0, otherwise return 0

**Function**

This procedure causes a Dataway Clear (C) to be generated in the crate specified by EXT.
int cccd(long EXT, int L)

**PURPOSE**
Enable or Disable Crate Demand

**Description**

**ARGUMENTS**

- **EXT**
  external address of a camac crate in encode format produced by function cdreg

- **L**
  flag indicating whether demand is to enable F_ENB or to disable F_DIS
  (F_ENB, F_DIS defined in camdef.h)

**return value**
return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the Crate Demand to be enabled in the crate specified by EXT if L is F_ENB and to be disabled if L is F_DIS.
ccci

int ccci(long EXT, int L)

PURPOSE
Set or clear Dataway inhibit.

Description
ARGUMENTS

EXT  
external address of a camac crate in encode format produced by function cdreg

L  
flag indicating whether inhibit is to set F_ENB or to clear F_DIS (F_ENB, F_DIS defined in camdef.h)

return value  
return -1 if X = 0, otherwise return 0

Function
This procedure causes the Dataway Inhibit to be set in the crate specified by EXT if L is F_ENB and to be clear if L is F_DIS.
int cccz(long EXT)

PURPOSE
generate dataway initialize

Description

ARGUMENTS
EXT
external address of a camac crate in encode format produced by function cdreg

return value
return -1 if X = 0, otherwise return 0

Function
This procedure causes a Dataway initialize (Z) to be generated in the crate specified by EXT.
ccopen

```c
int ccopen(char *HOSTNAME, long *HOSTADD)
```

**PURPOSE**
Enable or Disable Crate Demand

**Description**

**ARGUMENTS**

- **HOSTNAME**
  hostname as character pointer e.g. e7, e19, cve8

- **HOSTADD**
  external address of host in encoded format returned by function ccopen

- **return value**
  return -1 if error occurs, otherwise return 0

**Function**

This procedure create an encoded address for use with function cdreg from the input parameter HOSTNAME. This function has to be called only once for each host at the begin of the user programm !!!
int cdreg(long *EXT, long B, int C, int N, int A)

PURPOSE
Declare CAMAC register

Description

ARGUMENTS

EXT  external address of a camac crate in encoded format produced by function cdreg
B    external host address in encoded format produced by function ccopen
C    crate number (0 - 15)
N    station number (0 - 31)
A    subaddress (0 - 15)

return value return 0

Function
This procedure combines the host address, the crate number, the station number and the subaddress into a convenient format for later usage and stores the result in the struct camacadd referenced by the pointer EXT.
cerror

cchar *cerror(int ERROR, int FLAG)

**PURPOSE**  
Simple error handling routine

**Description**

**ARGUMENTS**

- **ERROR**: error number if error number < 0 the last internal errno number (_cam_errno) is use for output; for values >= 0 the output corresponds to the given error number. error numbers and error messages are defined in es-one/include/camdef.h
- **FLAG**: if FLAG <> 0 date and time are inserted in the output.
- **return value**: char pointer to error message

**Function**

This function performs a simple error message handling. With the global variable _cam_errno it’s possible to create your own error handling together with this function.
int cfga(int *FUNC, long EXT, int *DATA, int *Q,
        int *CB)

PURPOSE General multiple CAMAC action

Description

ARGUMENTS

FUNC array of camac function codes (0 - 31)

EXT array of external addresses of a camac crates in encode format produced
by function cdreg

DATA array of camac data words (read or write dep. of camac function codes)

Q array of Q/X responses. Stored in an integer element as X = bit 1, Q = bit 0, all others are zero.

CB control block. the contents of the four array elements are:
   cb(0) -> repeat_count (1 - 32768)
   cb(1) -> tally
   cb(2) -> lam_id (not implemented !!!!)
   cb(3) -> mode (not used)

The repeat count specifies the number of individual
CAMAC actions or the maximum number of data words to be trans-
ferred. The tally is the number of actions actually performed or the
number of words actually transferred. If the action is terminated by the
exhaustion of the repeat count, the tally will be equal to the repeat
count, otherwise it may be less. The LAM identification is not used in
this implementation and must be zero.

return value return -1 if X = 0, otherwise return 0
Function

This procedure causes a sequence of CAMAC functions specified in successive elements of FUNC to be performed at a corresponding sequence of CAMAC addresses specified in successive elements of EXT. Any read or write function in FUNC causes a CAMAC data word be transferred between the corresponding element of DATA and the specified CAMAC register. The X/Q response of each individual action is stored in the corresponding element of Q. The number of actions to be performed and the minimal dimension of DATA and Q is given by the value contained in the first element of CB.
int cfmad(int FUNC, long EXT, int *DATA, int *CB)

**PURPOSE**

Address scan mode

**Description**

**ARGUMENTS**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FUNC</strong></td>
<td>function code (0 - 31)</td>
</tr>
<tr>
<td><strong>EXT</strong></td>
<td>two external addresses of a camac crates in encode format produced by function cdreg (start (EXT[0]) and stop (EXT[1]) addresses)</td>
</tr>
<tr>
<td><strong>DATA</strong></td>
<td>array of camac data words (read or write dep. of camac function codes)</td>
</tr>
</tbody>
</table>
| **CB** | control block. the contents of the four array elements are:  
  CB(0) -> repeat_count (1 - 32768)  
  CB(1) -> tally  
  CB(2) -> lam_id (not implemented !!!)  
  CB(3) -> mode (not used)  
  The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero. |

**return value**

return -1 if X = 0, otherwise return 0

**Function**

The address scan mode is used when a block of registers or modules are to be read (or written onto) sequentially. The modules involved need not be located at consecutive addresses, however,
subaddresses within a given module must be so. The Q-response is used to determine if an address is occupied or not. For occupied $Q = 1$ is returned. With $N$ (station number) held constant, the subaddress is then incremented and transfers made until $Q = 0$ is returned. $A_{\text{subadd.}}$ is then reset to $A(0)$ and $N$ is incremented, etc.
int cfnsa(int FUNC, long EXT, int *DATA, int *Q)

PURPOSE
Perform single CAMAC action

Description
ARGUMENTS
  FUNC  camac function code (0 - 31)
  EXT   external address of a camac crate in encode format produced by function cdreg
  DATA  camac data word (read or write dep. of camac function code)
  Q     Q/X response
  return value  return -1 if X = 0, otherwise return 0

Function
This procedure causes the CAMAC specified by the function code to be performed at the CAMAC address specified by ext. If the function is a read or write code, a twenty-four-bit data transfer occurs between the CAMAC register and the given variable. The state of X and Q resulting from the operation is stored in q.
cfubc

```c
int cfubc(int FUNC, long EXT, int *DATA,
    int *CB)
```

**PURPOSE**
Controller synchronized block transfer

**Description**

**ARGUMENTS**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FUNC</strong></td>
<td>function code (0 - 31)</td>
</tr>
<tr>
<td><strong>EXT</strong></td>
<td>external addresse of a camac crate in encode format produced by function cdreg</td>
</tr>
<tr>
<td><strong>DATA</strong></td>
<td>array of camac data words (read or write dep. of camac function codes)</td>
</tr>
<tr>
<td><strong>CB</strong></td>
<td>control block. the contents of the four array elements are: CB(0) -&gt; repeat_count (1 - 32768) CB(1) -&gt; tally CB(2) -&gt; lam_id (not implemented !!!) CB(3) -&gt; mode (fixed !!!) The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.</td>
</tr>
</tbody>
</table>

**return value**
return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the single CAMAC function given by the value FUNC to be executed at the CAMAC address specified by the value of EXT. In this mode the CAMAC address is
never changed, but the single register is expected to supply or accept many words of data. It is assumed able to supply or accept a data word whenever the controller addresses it until the block is exhausted or the controller terminates the process because the number of data transfers exceeds the limit given by the first element of CB. The module indicates that the block is exhausted by its Q response. It stops when Q = 0.
cfubr

```c
int cfubr(int FUNC, long EXT, int *DATA,
          int *CB)
```

**PURPOSE**
Controller synchronized block transfer

**Description**

**ARGUMENTS**

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>FUNC</td>
<td>function code (0 - 31)</td>
</tr>
<tr>
<td>EXT</td>
<td>external address of a camac crate in encode format produced by function cdreg</td>
</tr>
<tr>
<td>DATA</td>
<td>array of camac data words (read or write dep. of camac function codes)</td>
</tr>
<tr>
<td>CB</td>
<td>control block. the contents of the four array elements are:</td>
</tr>
<tr>
<td></td>
<td>CB(0) -&gt; repeat_count (1 - 32768)</td>
</tr>
<tr>
<td></td>
<td>CB(1) -&gt; tally</td>
</tr>
<tr>
<td></td>
<td>CB(2) -&gt; lam_id (not implemented !!!)</td>
</tr>
<tr>
<td></td>
<td>CB(3) -&gt; mode (fixed !!!)</td>
</tr>
</tbody>
</table>

The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.

**return value**

return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the single CAMAC function specified by the contents of FUNC to be executed at the CAMAC address specified by the contents of EXT with the usage of the Q
response for the Repeat mode. In this mode the CAMAC address is never changed, but the single register is expected to supply or accept many words of data. Q is used as a timing signal. \( Q=1 \) indicates that the previously executed function succeeded; \( Q=0 \) indicates that the module was not ready to execute the function and that the controller should try again. Any data words transferred are placed into or taken from the array data. If the response is \( Q=0 \), no transfer took place, and the index into the array DATA is not changed. The number of \( Q=1 \) responses expected is given by the contents of the first element of CB.
cgreg

int cgreg(long EXT, long *B, int *C, int *N, int *A)

**PURPOSE**
Analyze register identifier

**Description**

**ARGUMENTS**

- **EXT**: external address of a camac crate in encoded format produced by function cdreg
- **B**: external host address in encoded format produced
- **C**: crate number (0 - 15)
- **N**: station number (0 - 31)
- **A**: subaddress (0 - 15)

**return value**: return 0

**Function**

This procedure decodes the CAMAC address identifier into its component parts host address, crate number, station number and subaddress. This procedure exactly reverses the process performed by cdreg, all parameters have the same interpretation and form. This routine is the only function which don’t reset the errno number !!!

Version 4.3 October 28, 2003
csga

int csga(int *FUNC, long EXT, int *DATA, int *Q, int *CB)

PURPOSE
General multiple CAMAC action (16-bit words)

Description
ARGUMENTS

FUNC
array of camac function codes (0 - 31)

EXT
array of external addresses of a camac crates in encode format produced by function cdreg

DATA
array of camac data words 16-bit(read or write dep. of camac function codes)

Q
array of Q/X responses. Stored in an integer element as X = bit 1, Q = bit 0, all others are zero.

CB
control block. the contents of the four array elements are:
CB(0) -> repeat_count (1 - 32768)
CB(1) -> tally
CB(2) -> lam_id (not implemented !!!)
CB(3) -> mode (not used)

The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.

return value
return -1 if X = 0, otherwise return 0
Function

This procedure causes a sequence of CAMAC functions specified in successive elements of FUNC to be performed at a corresponding sequence of CAMAC addresses specified in successive elements of EXT. Any read or write function in FUNC causes a CAMAC data word be transferred between the corresponding element of DATA and the specified CAMAC register. The X/Q reponse of each individual action is stored in the corresponding element of Q. The number of actions to be performed and the minimal dimension of DATA and Q is given by the value contained in the first element of CB.
csmad

```c
int csmad(int FUNC, long EXT, int *DATA,
          int *CB)
```

**PURPOSE**

Address scan mode 16-bit

**Description**

**ARGUMENTS**

- **FUNC**
  - function code (0 - 31)
- **EXT**
  - two external addresses of a CAMAC crate in encode format produced by function cdreg (start (ext[0]) and stop (ext[1]) addresses)
- **DATA**
  - array of CAMAC data words 16-bit (read or write dep. of CAMAC function codes)
- **CB**
  - control block. the contents of the four array elements are:
    - CB(0) -> repeat_count (1 - 32768)
    - CB(1) -> tally
    - CB(2) -> lam_id (not implemented !!!)
    - CB(3) -> mode (not used)
  - The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.

**return value**

- return -1 if X = 0, otherwise return 0

**Function**

The address scan mode is used when a block of registers or modules are to be read (or written onto) sequentially. The modules involved need not be located at consecutive addresses, however,
subaddresses within a given module must be so. The Q-response is used to determine if an address is occupied or not. For occupied Q = 1 is returned. With N(station number) held constant, the subaddress is then incremented and transfers made until Q = 0 is returned. A(subadd.) is then reset to A(0) and N is incremented, etc.
int cssa(int FUNC, long EXT, int *DATA, int *Q)

**PURPOSE**
Perform single CAMAC action with 16 bit data words

**Description**

**ARGUMENTS**

- **FUNC**
  camac function code (0 - 31)

- **EXT**
  external address of a camac crate in encode format produced by function cdreg

- **DATA**
  camac data word 16 bit (read or write dep. of camac function code)

- **Q**
  Q/X response

- **return value**
  return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the CAMAC specified by the function code to be performed at the CAMAC address specified by EXT. If the function is a read or write code, a 16-bit data transfer occurs between the CAMAC register and the given variable. The state of X and Q resulting from the operation is stored in q.
csubc

```c
int csubc(int FUNC, long EXT, int *DATA,
           int *CB)
```

**PURPOSE**  
Controller synchronized block transfer (16-bit)

**Description**

**ARGUMENTS**

- **FUNC**  
  function code (0 - 31)

- **EXT**  
  external address of a camac crate in encode format produced by function cdreg

- **DATA**  
  array of camac data words 16-bit (read or write dep. of camac function codes)

- **CB**  
  control block. the contents of the four array elements are:
  - CB(0) -> repeat_count (1 - 32768)
  - CB(1) -> tally
  - CB(2) -> lam_id (not implemented !!!)
  - CB(3) -> mode (fixed !!!)

  The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.

**return value**  
return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the single CAMAC function given by the value FUNC to be executed at the CAMAC address specified by the value of EXT. In this mode the CAMAC address is
never changed, but the single register is expected to supply or accept many words of data. It is
assumed able to supply or accept a data word whenever the controller addresses it until the block
is exhausted or the controller terminates the process because the number of data transfers exceeds
the limit given by the first element of CB. The module indicates that the block is exhausted by
its Q response. It stops when Q = 0.
csubr

```c
int csubr(int FUNC, long EXT, int *DATA,
    int *CB)
```

**PURPOSE**

Controller synchronized block transfer (16-bit)

**Description**

**ARGUMENTS**

- **FUNC**
  - function code (0 - 31)
- **EXT**
  - external addresse of a camac crate in encode format produced by function cdreg
- **DATA**
  - array of camac data words 16-bit (read or write dep. of camac function codes)
- **CB**
  - control block. the contents of the four array elements are:
    - CB(0) -> repeat_count (1 - 32768)
    - CB(1) -> tally
    - CB(2) -> lam_id (not implemented !!!)
    - CB(3) -> mode (fixed !!!)

  The repeat count specifies the number of individual CAMAC actions or the maximum number of data words to be transferred. The tally is the number of actions actually performed or the number of words actually transferred. If the action is terminated by the exhaustion of the repeat count, the tally will be equal to the repeat count, otherwise it may be less. The LAM identification is not used in this implementation and must be zero.

- **return value**
  - return -1 if X = 0, otherwise return 0

**Function**

This procedure causes the single CAMAC function specified by the contents of FUNC to be executed at the CAMAC address specified by the contents of EXT with the usage of the Q
response for the Repeat mode. In this mode the CAMAC address is never changed, but the single register is expected to supply or accept many words of data. Q is used as a timing signal. Q=1 indicates that the previously executed function succeeded; Q=0 indicates that the module was not ready to execute the function and that the controller should try again. Any data words transferred are placed into or taken from the array data. If the response is Q=0, no transfer took place, and the index into the array DATA is not changed. The number of Q=1 responses expected is given by the contents of the first element of CB.
ctcd

```c
int ctcd(long EXT, int *L)
```

**PURPOSE**
Test crate demand enabled

**Description**

**ARGUMENTS**

- **EXT**
  external address of a camac crate in encode format produced by function cdreg
- **L**
  flag returning demand enable status
- **return value**
  return -1 if X = 0, otherwise return 0

**Function**

This procedure sets the value of L to 0x1 if crate demand is enabled in the crate specified by ext and sets L to 0x0 if crate demand is disabled
int ctcilong EXT, int *L)

PURPOSE
Test Dataway inhibit

Description
ARGUMENTS
EXT
external address of a camac crate in encode format produced by function cdreg

L
flag returning inhibit status

return value
return -1 if X = 0, otherwise return 0

Function
This procedure sets the value of L to OX1 if dataway inhibit is set in the crate specified by ext and sets L to OX0 if dataway inhibit is not set
ctgl

```
int ctgl(long EXT, int *L)
```

**PURPOSE**
Test Crate Demand present

**Description**

**ARGUMENTS**

- **EXT**
  external address of a camac crate in encode format produced by function cdreg

- **L**
  flag returning crate demand status

- **return value**
  return -1 if $X = 0$, otherwise return 0

**Function**

This procedure sets the value of $L$ to 0x1 if any demand is present in the crate specified by $ext$ and sets $L$ to 0x0 if no demand is present.
ctstat

int ctstat(int *L)

PURPOSE test status of preceding camac action

Description

ARGUMENTS

L status code

return value L (same as parameter)

Function

This procedure stores an integer status code into the parameter L. The status code reflects the results of the last action executed by another function call to the esone library (libcamcli). After a cfga call the status reflects the worst case response of all camac accesses. The status code has the following meaning:

<table>
<thead>
<tr>
<th>L</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

If some errors occurs the errno number specific to this implementation (errno numbers are defined in esone/include/camdef.h) is shifted by two bits and coded in the first 30 bit’s of the integer L. E.g. errno = EC_IFUNCODE (illegal function code) = 6, X = 1, Q = 0 -> L = 0000 0000 0001 1001 = 0x19
Chapter 8

Event Routines
f_getevt_open

f_getevt_open(long l_mode, char &c_server[], s_channel &s_chan, char **ppc_server_info)

PURPOSE  

f_getevt_open opens an event stream from specified channel.

ARGUMENTS  

l_mode  
Type of server:

GETEVT_FILE Input from file
GETEVT_STREAM Input from SBS stream server
GETEVT_TRANS Input from SBS transport
GETEVT_EVENT Input from SBS event server (NYI)
GETEVT_STDIN Input from stdin (NYI)

c_server  
Node of server or file name.

s_chan  
structure s_channel, will be returned.

ppc_server_info  
address of pointer. If it is not NULL, then try to return file header or other information about server. If it is NULL, then returns nothing.

Return type  
int.

Description  

FUNCTION  
Opens the input channel and save context in s_chan.
f_getevt_get

f_getevt_get(s_channel &s_chan, long **ppl_buffer, long **ppl_goobuf)

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>f_getevt_get returns address of event</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARGUMENTS</td>
<td></td>
</tr>
<tr>
<td>s_chan</td>
<td>Input channel from open.</td>
</tr>
<tr>
<td>ppl_buffer</td>
<td>Address of pointer. Returns address of event.</td>
</tr>
<tr>
<td>ppl_goobuf</td>
<td>Address of pointer. Returns address of buffer.</td>
</tr>
<tr>
<td>Return type</td>
<td>int.</td>
</tr>
</tbody>
</table>

Description

FUNCTION Get next event and returns pointer. The pointer may point to the event in the buffer or internal event buffer (spanned events). The content of the pointer may be destroyed by next call.
f_getevt_close

f_getevt_close( s_channel &s_chan)

<table>
<thead>
<tr>
<th>PURPOSE</th>
<th>f_getevt_close closes event stream of specified channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARGUMENTS</td>
<td></td>
</tr>
<tr>
<td>s_chan</td>
<td>Input channel from open.</td>
</tr>
<tr>
<td>Return type</td>
<td>int.</td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>FUNCTION</td>
<td>Closes the specified input channel.</td>
</tr>
<tr>
<td>REMARKS</td>
<td>-</td>
</tr>
</tbody>
</table>


**f\_getevt\_error**

\[
f\_getevt\_error( \text{ long l\_error }, \text{ char &c\_string[] }, \text{ long l\_out } )
\]

**PURPOSE**  
f\_getevt\_error displays error messages.

**ARGUMENTS**

- **l\_error**  
The error id as returnred from other calls

- **c\_string**  
The string into f\_getevt\_error() copies the message.

- **l\_out**  
specifies the output device for the error message.
  
  - **out = 1**  
    error message is copied to string.
  
  - **out = 0**  
    error message is printed on terminal.

**Return type**  
int (longword).

**Description**

**FUNCTION**  
f\_getevt\_error displays the error message for the error id ( l\_error ). If out = 1 the error message is copied into string, else f\_getevt\_error prints the message on terminal.
f_putevt_open

f_putevt_open(char *c_file[], long l_size, long l_stream, long l_type, long l_subtype, s_channel *ps_chan, char *ps_filhe)

PURPOSE  
f_putevt_open opens an event output stream.

ARGUMENTS

c_file       Name of file.
l_size       Size of output buffers in bytes.
l_stream     Number of buffers with spanning events.
l_type       Buffer type number
l_subtype     Buffer subtype number
ps_chan      Address of channel structure which will be returned.
ps_filhe     Address of user specified file header

Return type  int.

Description

FUNCTION  
Opens the output channel and save context in s_channel structure.

REMARKS  
-
f_putevt_put

f_putevt_put(s_channel *ps_chan, long &la_evt_buf[])

PURPOSE  f_putevt_put outputs event

ARGUMENTS

ps_chan  Address of channel structure as returned from f_putevt_open.

la_evt_buf  event data array. Standard GSI event structure.

Return type  int.

Description

FUNCTION  Copies current event into output buffer. Writes buffer to file, when full.

REMARKS  -
f_putevt_close

f_putevt_close(s_channel *ps-chan)

PURPOSE
f_putevt_close closes specified channel.

ARGUMENTS
channel
Channel number.

Return type
int.

Description

FUNCTION
Closes the specified output channel after writing last buffer.

REMARKS
-
f_putevt_error

f_putevt_error( long l_error, char &c_string[], long l_out )

PURPOSE
f_putevt_error displays error messages.

ARGUMENTS

l_error
The error id.

c_string
The string into f_putevt_error() copies the message.

l_out
specifies the output device for the error message.

    out = 1  error message is copied to string.
    out = 0  error message is printed on terminal.

Return type
int (longword).

Description

FUNCTION
f_putevt_error displays the error message for the error id (error). If out = 1 the error message is copied into string, else f_putevt_error prints the message on terminal.

REMARKS
-

Version 4.3 October 28, 2003
# Contents

1 Preface ................................................................. 1  
   1.1 MBS Authors and Advisory Service ................................. 2

2 MBS Keyword Summary .................................................. 3  
   MBS keywords .......................................................... 4

3 MBS Command Summary .................................................. 23

4 MBS Command Description ................................................ 35  
   ATTACH ................................................................. 36
   CAMAC ................................................................. 37
   CLEAR ................................................................. 39
   CLOSE ................................................................. 42
   CNAF ................................................................. 43
   COMMENT .............................................................. 44
   CONNECT .............................................................. 45
   CREATE ............................................................... 46
   DEFINE ............................................................... 49
   DELETE ............................................................... 50
   DETACH ............................................................... 51
   DISABLE ............................................................. 52
   DISCONNECT .......................................................... 55
   DISMOUNT ............................................................. 56
   DUMP ................................................................. 57
   ENABLE ............................................................... 59
   HELP ................................................................. 63
   INITIALIZE ........................................................... 64
   LOAD ................................................................. 65
   MOUNT ............................................................... 68
   NEWS ................................................................. 69
   OPEN ................................................................. 70
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROTECT</td>
<td>72</td>
</tr>
<tr>
<td>PSHELL</td>
<td>73</td>
</tr>
<tr>
<td>REMOTE</td>
<td>74</td>
</tr>
<tr>
<td>REMOVE</td>
<td>77</td>
</tr>
<tr>
<td>RESET</td>
<td>78</td>
</tr>
<tr>
<td>RESTORE</td>
<td>79</td>
</tr>
<tr>
<td>SET</td>
<td>80</td>
</tr>
<tr>
<td>SHOW</td>
<td>91</td>
</tr>
<tr>
<td>START</td>
<td>101</td>
</tr>
<tr>
<td>STOP</td>
<td>104</td>
</tr>
<tr>
<td>TYPE</td>
<td>106</td>
</tr>
<tr>
<td>VOID</td>
<td>107</td>
</tr>
<tr>
<td>XSHELL</td>
<td>108</td>
</tr>
</tbody>
</table>

5 Related GOOSY Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAMAC</td>
<td>110</td>
</tr>
<tr>
<td>CONNECT</td>
<td>112</td>
</tr>
<tr>
<td>DISCONNECT</td>
<td>114</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>116</td>
</tr>
<tr>
<td>START</td>
<td>119</td>
</tr>
<tr>
<td>STOP</td>
<td>122</td>
</tr>
</tbody>
</table>

6 Histogramming and Analysis Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>f_his_anal</td>
<td>126</td>
</tr>
<tr>
<td>f_his_anal_init</td>
<td>127</td>
</tr>
<tr>
<td>f_his_accu1</td>
<td>128</td>
</tr>
<tr>
<td>f_his_accu2</td>
<td>129</td>
</tr>
<tr>
<td>f_his_get_hist</td>
<td>130</td>
</tr>
<tr>
<td>f_his_getdir</td>
<td>131</td>
</tr>
</tbody>
</table>

7 Esone Client Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cccc</td>
<td>134</td>
</tr>
<tr>
<td>cccd</td>
<td>135</td>
</tr>
<tr>
<td>ccci</td>
<td>136</td>
</tr>
<tr>
<td>ccccz</td>
<td>137</td>
</tr>
<tr>
<td>ccopen</td>
<td>138</td>
</tr>
<tr>
<td>cdreg</td>
<td>139</td>
</tr>
<tr>
<td>cerror</td>
<td>140</td>
</tr>
<tr>
<td>cfga</td>
<td>141</td>
</tr>
<tr>
<td>cfmad</td>
<td>143</td>
</tr>
<tr>
<td>cfsa</td>
<td>145</td>
</tr>
<tr>
<td>cfubc</td>
<td>146</td>
</tr>
</tbody>
</table>
8 Event Routines

8.1 Event Routines

f_getevt_open .................................................. 166
f_getevt_get ................................................... 167
f_getevt_close .................................................. 168
f_getevt_error .................................................. 169
f_putevt_open .................................................. 170
f_putevt_put .................................................... 171
f_putevt_close .................................................. 172
f_putevt_error .................................................. 173