EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

VLT

Astronomical Site Monitor

Control Software

Design Description

Doc.No. VLT-SPE-ESO-17441-1213

Issue 1.0/prep.3

Date 25.06.1997

Prepared for Review - INTERNAL USE ONLY

Prepared S.Sandrock, R.Amestica ....
Name Date Signature

Approved M.Sarazin, K.Wirenstrand ....
Name Date Signature

Released G.Raffi ....
Name Date Signature

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
1.0/prep.1	20.11.1996		Draft for review, with chapters: 1, 2, 3, 5, 10, 9
1.0/prep.2	02.12.1996		**Changed according to review comments.**
		2.3	Marked modules foreseen in 1st release.
		9.6	Added list of trk commands to be used.
1.0/prep.3	25.06.1997	All	**Updated for DIMM upgrade test.**

TABLE OF CONTENTS

1 INTRODUCTION

The **Astronomical Site Monitor (ASM)** provides TCS and external other packages with information about meteorological data, outside air, seeing, cloud coverage etc. The hardware of the ASM consists of an independent telescope with enclosure, technical CCD equipment, and several external sensor facilities like a meteorological station.

1.1 Purpose

This document contains the detailed design of the workstation and LCU control software included in the ASM subsystem. It shall provide the necessary information for implementation, maintenance, and partly also for usage and test. Future users of the ASM data may want to go directly to the User Interfaces chapter 21, where the access to the ASM on-line database is described.

1.2 Scope

The document describes the interface functions, database classes, and configuration possibilities of the ASM control software.

Knowledge of the Functional Specification [1] is required, since it includes a more global overview of the whole ASM system, which will not be the case in this document, which concentrates more on specific details of implementation.

The related hardware has partly not yet been designed. This implies that this description might change accordingly to reflect hardware evolution.

1.3 Applicable Documents

The following documents, of the exact issue shown, form a part of this document to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this document, the contents of this document shall be considered as a superseding requirement.

[1] ASM Control Software Functional Specification,	VLT-SPE-ESO-17441-1175, 1.0
[2] Telescope Control System Functional Specification,	VLT-SPE-ESO-11720-0001, 1.1, 02/03/95
[3] DIMM Upgrade Plan,	VLT-SPE-ESO-17410-1174, 1.0, 17/06/96
[4] LCU Software Specification,	VLT-SPE-ESO-17210-0002, 2.2, 26/10/93
[5] VLT Software Programming Standards,	VLT-PRO-ESO-10000-0228, 1.0, 10/03/93

1.4 Reference Documents

The following documents are referenced in this document.

[6] LCU Common Software User Manual,	VLT-MAN-SBI-17210-0001, 2.4, 10/08/95
[7] CCS User Manual,	VLT-MAN-ESO-17210-0619, 1.3, 17/08/95
[8] CCS On Line Database Loader User Manual,	VLT-MAN-ESO-17210-0707, 1.2, 30/07/95
[9] CCD Detectors Control SW Functional Spec.,	VLT-SPE-ESO-17240-0227, 1.0, 08/04/93
[10]Technical CCD Systems User Manual,	VLT-MAN-DJO-11700-0001, 1.0, 06/05/96
[11]LCU Axis Control Module Design Description	VLT-SPE-ESO-17130-1210, 1.0, 07/09/96
[12]TCS Tracking Design Description	VLT-SPE-ESO-17230-0755, 1.0, 16/05/95
[13]Heidenhain IK320 Driver User Manual	VLT-MAN-ESO-17210-0806, 1.2, 15.03.1996

1.5 Abbreviations And Acronyms

The following abbreviations and acronyms are used in this document:

ACE	Array Control Electronics (for CCD Cameras)
ADC	Analog to Digital Converter
ASM	Astronomical Site Monitor
CCD	Charge Coupled Device
CCS	Central Control Software
CDT	Command Definition Table
CI	Command Interpreter
CIT	Command Interpreter Table (for LCU CI)
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DB	Database
DIMM	Differential Image Motion Monitor
DIN	Deutsches Institut für Normung
FTP	File Transfer Protocol
GUI	Graphical User Interface
HW	Hardware
I/F	Interface
I/O	Input/Output
ISO	International Organization for Standardization
LAN	Local Area Network
LCC	LCU Common Software
LCU	Local Control Unit
N/A	Not Applicable
RMS	Root Mean Square
RTAP	Real Time Application Platform
SW	Software
TBC	To Be Confirmed/Clarified
TBD	To Be Defined
TCCD	Technical CCD
TCS	Telescope Control Software
TIM	Time Interface Module
TRS	Time Reference System
VLT	Very Large Telescope
VME	Versa Module Eurocard (Bus-System)
WS	Workstation
WWW	World Wide Web

1.6 Stylistic Conventions

The following styles are used:

bold
in the text, for commands, filenames, prefixes/suffixes as they have to be typed.

italic
in the text, for parts that have to be substituted with the real content before typing.

`teletype`
for examples.

`<name>`
in the examples, for parts that have to be substituted with the real content before typing.

bold and *italic* are also used to highlight words.

Points to be discussed during the Review (TBC/TBD) are highlighted with color and marked with change-bar. Comments on these points are particularly welcome.

1.7 Terminology

assert - a signal is set to its active state

negate - a signal is cleared to its inactive state

axes coordinates - position in an horizontal coordinate system denoted by angles suitable for the telescope mounting

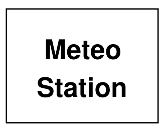
target coordinates - position of an object in an equatorial coordinate system

1.8 Diagram Conventions

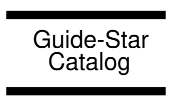
Control/data flow diagrams show the relations of a system with the environment, not in terms of physical location, but in an abstract form. The following symbolic notation is used for the elements of a diagram:

Software Data Process:

External Unit (HW or SW):

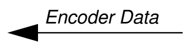


Database or File Unit:



Control Flow:

Data Flow:



2 OVERVIEW

2.1 Document Structure

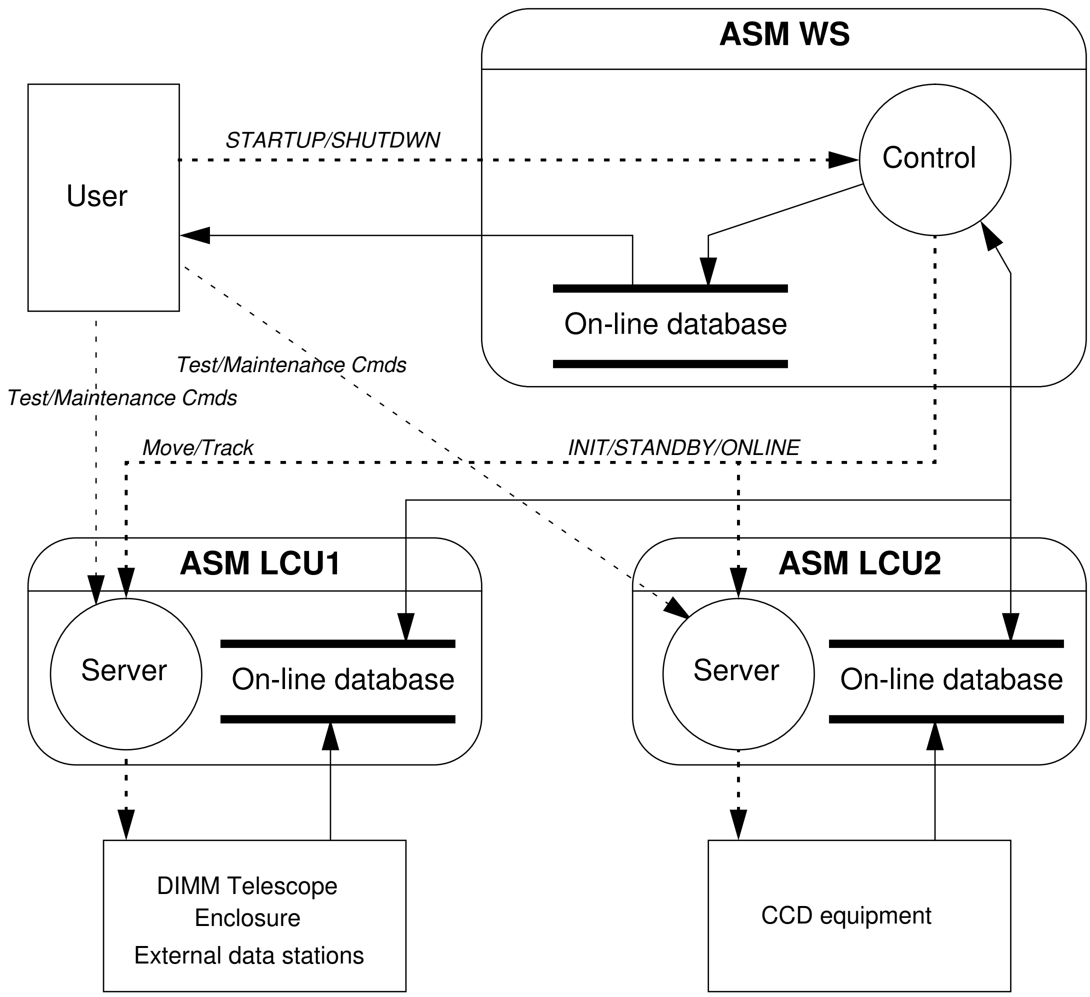
The ASM software is divided into several, reasonably small modules, each dealing with a specific unit of the system. This design description is therefore also structured in this way. After this **overview** chapter, which describes the features and requirements that apply to all modules, each specific **module** will then be described in a separate chapter.

Then the **environments** included in the ASM system, WS and LCU based, will be described, including their specific configuration. The **installation** of the ASM software is contained in chapter 22. The description of **interfaces** for external users of the ASM data is collected in chapter 21.

Finally the **reference** chapter 23 collects all source code extracts like man-pages and database classes in alphabetical order, so that they can be omitted during the main part of this document. Please consult the table of contents to locate a specific man-page or detailed source file.

2.2 Data Flow Overview

Fig. 1 Global data/control flow



The ASM workstation environment is the entry point for users during normal operation. It controls start-up and shutdown of the system, coordinates facilities during normal operation, and provides the measured data in its on-line database. Together with INS processes, it also takes care of converting and transferring the data to the VLT archive.

Two LCUs control the real-time part of the ASM, namely the tracking of the DIMM telescope, data reception from external data-sources (meteo-station etc.), and control of the seeing monitor CCD camera. A server process on each LCU accepts commands from the WS control process, and additionally from maintenance and test users.

2.3 ASM Module Decomposition

The module names and hierarchy are as stated in the functional specification [1]. There are:

- The module ***asm*** as a stand-alone interface for external users.
- Two top-level modules ***asmlcu*** and ***asmws*** implementing the entry-points to the ASM software (WS and LCU), see chapter 4 and 5.
- A base-level module asmi (ASM internal) implementing common functionality for LCU and WS respectively, see section 6.
- Several device-level modules implementing the specific software-device handling.
- Special modules at global level, containing build procedures, configuration files, etc.

The table contains the distribution on modules, and whether the functionality shall be implemented on LCU or workstation:

Table 1: ASM module overview

Purpose	Device ID [a]	Module ID	on WS	on LCU
Common Functionality (top/bottom level)	ASM	asm asmws asmlcu asmi	X X X	 X X
ASM-Telescope Enclosure	Enclosure	asmten		X
ASM-Telescope Motion Control	Telescope	asmtel		X
Target-Star List	Catalog	asmcat	X	
DIMM Control (incl. cloud monitor)	Dimm	asmdim	X	X
Meteo-Station	Meteo	asmmet		X
Engineering and Maintenance Tools	N/A	asmei	X	
VLT on-line archive interface	N/A	asmarc	X	

a. *Device ID* is used as name of the software-device, and root-point of the database branch.

2.4 ASM Module Requirements

General rules and standards to be applied to the asm modules are given in the function specification [1], chapter ‘System Attributes’.

The implementation shall be based on the May 1997 release of the VLT common software. WS-based modules shall make use of the event-handler *evh* [1]. All device-level modules should consist of a ‘ws’ and ‘lcu’ branch, even if the implementation is planned for only one platform (to make future extension easier). Care must be

1. Oh dear, how I regret this decision!

taken to avoid name duplication in ws and lcu branches; names of installed headers and dbl-classes must be unique. Furthermore, all ASM modules shall comply to the rules stated in the following subsections.

2.4.1 Makefiles

Where possible, all except the top-level modules *asmws/asmlcu* shall provide code installed as library rather than executable file. The top-level module will then bind the libraries to the executables. This applies to WS and LCU. Care must be taken that there is actually a reference from the asm module to all needed functions in the lower level modules, otherwise a function will not be extracted from the library at linking time, and will cause an *unresolved symbol* error at LCU loading time.

2.4.2 CDT and CIT

Only the top-level modules shall provide CDT and CIT for command interpreter processes. Lower level modules only provide library functions that are called from LCC compliant command handlers in the top-level module.

2.4.3 Standard Filenames

In order to achieve a common appearance of all modules, the following systematic file naming scheme shall be used.

Remark: 'xxx' corresponds to the module name prefix. Headers shall be installed except where otherwise noted.

- **include/xxx.h** - (mandatory) this header shall provide ***all*** declarations that are necessary for higher level ASM modules, but not any declarations that are only internally used by the module. Except the topmost asm.h there shall be a clear remark in the file that it is not for public, but only for internal use by ASM modules.
- **include/xxxDefines.h** - (optional) this header shall include macro definitions that are used also for database compilation, but exclude everything that cannot be interpreted by *dbl*.
- **include/xxxInternal.h** - (optional) this header shall include declarations that are only used internal to the module. It shall not be installed.
- **include/xxxPublic.h** - (optional) this header shall include public declarations that are provided to external SW, to be included into the global *asm.h*.
- **include/xxxErrors.h** - (mandatory) error-macros as produced by *errEditor*. It shall be included by *xxx.h*.
- **src/xxx.c** - (mandatory for LCU) this file shall define the version ID string, and contain the function *xxxId*.
- **src/xxx*Cmd<Command>*.c** - (optional, LCU only) contains the handler-function for a command and its interface function to the CI.
- **src/xxx*<Handler>*.c** - (optional, LCU only) several standard handler functions are usually needed in lower-level modules that are called indirectly by a global command handler of the *asm* module. Each function shall be contained in a separate source file with the same name.

2.4.4 Standard Command Handler Functions

All **LCC/CCS standard** commands are handled in the top-most asm module, which calls indirectly a function of a lower level module, if defined, see next section.

Each software-device can be individually put into **simulation** mode, by which all LCC standard commands except STATUS will be intercepted at *asmlcu* level.

On LCU, also a **verbose** mode is available on a per-device basis, as well as a global one applying to all devices. In verbose mode, intermediate replies shall be sent every 10 to 20 seconds during execution of a command, using the function *tooReplyf*. No intermediate replies shall be sent when not in verbose mode.

2.4.5 Indirect Standard Functions (LCU only)

The top-level *asmlcu* module calls several standard functions of the lower-level modules that are hooked up with function-tables in the files *asmlcuModules.c* (module-wise) and *asmlcuDevices.c* (for each software-device). A module is allowed to omit a handler function when no action is needed for a certain case.

Every module should provide the following functions:

- **xxxId** - return the version ID string of the module. The function definition shall be included in the file *xxx.c*. It shall consist of RCS ID and compilation time:

```
const char asmxxx_id[] = "@(#) $Id$ (Compiled "__DATE__" "__TIME__")";
```

- **xxxStart**[1] - start the module up, i.e. allocate permanent resources, configure database and signals, etc.

The module that handles a certain software-device may provide the following device-level standard functions:

- **xxxInit** - initialize the device as defined for Loaded state
- **xxxStandby** - operate the device as defined for Standby state
- **xxxOnline** - operate the device as defined for Online state
- **xxxOff** - operate the device as defined for Loaded state
- **xxxExit** - to release all previously allocated resources
- **xxxStop** - stop moving the device
- **xxxSelftst** - verify device without moving anything; must be possible in any state.
- **xxxTest** - extended device verification
- **xxxStatus** - send device information as reply

All standard functions must have a fixed prototype, as stated in the header *asmi.h*, see section 6.4.3.

All functions are in principle optional, i.e. when no device-specific function exists, the global CI will perform no action and simply return OK.

1. In the ASM modules there is no *InitAll* function. The Start function will initialize the module, but leaves the controlled software-devices in state *Off*. An *INIT* is therefore not included.

3 MODULE ‘asm’ - Stand-Alone Interface

- P R E L I M I N A R Y -

3.1 Purpose

This is the module through which external users shall access the ASM data. Other ASM modules may use it internally as bottom-level module. As such it is independent from all other asm modules.

3.2 Overview

3.3 Database Classes

3.4 Headers

The WS is the entry point to the ASM for all external users/software.

- **asm.h** - This header contains all public declarations that can be included by external SW:
  - ***TBD***

3.5 Functions

3.6 Commands

There is no command interface.

3.7 Configuration Files

There are no configuration files.

4 MODULE 'asmws' - WS Top-Level

4.1 Purpose

This is the top-most module on WS-side that binds all functionality of the lower-level modules together. It implements the ***asmwsControl*** application.

4.2 Overview

The *asmwsControl* process is running in the ASM WS environment, and controls the LCU environments of the ASM by sending state-switching commands to the respective *lccServer*, which will then distribute the commands to the application-specific servers that control the software-devices. Once initiated by a *START* command, it takes care of the automated operation of the ASM, which includes:

- start-up of connected LCU environments
- selecting *ONLINE* or *STANDBY* state for the LCUs depending on the time of day
- periodically passing control to functions of the sub-modules, e.g. to select guide-stars as positions for the DIMM telescope, to update the on-line database with received and calculated data, etc.
- forwarding selected data to the VLT on-line archive

The *asmws* module is dependent on all other *asm** modules, it must therefore be compiled and linked as last in the order.

4.3 Database Classes

At this level only WS-based classes are needed. On the LCU side the database must be assembled at environment level by referencing the classes provided by the sub-modules, according to the software-devices handled by the environment; this is described in chapter 19 ff.

The following WS-based classes are contained in equally named files each:

- **asmwsBRANCH** - this is the whole ASM database branch of the ASM. It consists of the following sub-branches.
- **asmwsCONTROL** - contains attributes needed for the *asmControl* process. It inherits from the standard evh class *evhSTD_COMMANDS* and adds some more attributes:
  - *stdCmdTargets* - destinations for standard state switching commands
  - *lcuCmdTarget* - destination for LCU specific commands
  - *trkCmdTarget* - destination for presetting and tracking commands
  - *ccdCmdTarget* - destination for CCD specific commands
- **asmwsDEVICES** - contains all the private data in sub-branches per controlled device. It consists of provided branches of the lower-level modules.
- **asmwsPUBLIC** - contains all the provided data in sub-branches. It consists of all provided branches of the lower-level modules.

4.4 Headers

- **asmws.h** - This header contains all public declarations that can be included by external SW:
  - Declaration of public functions
  - Error definitions

4.5 Functions

The following executable functions are available on the WS:

- **asmwsControl** - starts the main ASM control process. This will then accept commands via CCS message-system, typically from UIF panels.

4.6 Commands

4.6.1 Commands of WS asmwsControl Process

The following commands are available:

Command		Cathegory	Action
START	(none)	Public	start-up of whole ASM system
ONLINE	-force (optional)	Maintenance	enter day-time On-Line state
STANDBY	-force (optional)	Maintenance	enter night-time Stand-By state

Other standard commands:

Command	Parameters	Cathegory	Action
INIT		Maintenance	initialize all devices
EXIT		Maintenance	release all resources allocated by device and exit
STOP		Maintenance	stops any movement of device; all devices which can move something need to support this command
STATE		Maintenance	returns current state string
STATUS		Maintenance	returns information about status as plain text
BREAK		Maintenance	send STOP to all controlled software-devices
KILL		Maintenance	send EXIT to all controlled software-devices, then terminate server
VERSION		Public	returns the version-IDs of all modules

4.7 Configuration Files

- **asmws.login** - can be sourced from the shell to define the most important shell variables for the ASM. It is installed into *$INTROOT/config*.

5 MODULE 'asmlcu' - LCU Top-Level

5.1 Purpose

This is the top-most module on LCU-side that binds all functionality of the lower-level modules together. It implements the ***asmlcuServer*** command-interpreter.

5.2 Overview

The *asmlcuServer* process is running in all ASM LCU environments (execpt currently CCD control LCUs), and controls the state of all ASM-specific software-devices that are present, based on commands that it receives through LCC from the *asmwsControl* process on the WS. For maintenance and test the *asmlcuServer* can also accept commands directly from users, but during normal operation this is not permitted. Note that the TCS modules *trk* and *axis*, which are used for telescope control, have their own server processes.

The *asmlcu* module is dependent on all other *asm** modules, it must therefore be compiled and linked as last in the order.

5.3 Database Classes

At this level no classes are defined. On the LCU side the database must be assembled at environment level by referencing the classes provided by the sub-modules, according to the software-devices handled by the environment; this is described in chapter 19 ff.

5.4 Headers

The WS is the entry point to the ASM for all external users/software. Therefore the *ws* part provides and installs all public headers, while the *lcu* branch only installs internal headers.

- **asmlcu.h** - exported definitions.
- **asmlcuInternal.h** - This header contains all declarations that are used inside the *asmlcu* module and shall not be exported.
- **tooInterface.h** - (developed for ASM, but moved to *too* module) This header contains macros to implement interface functions for the LCC command interpreter without the need for any additional code:
  - **tooINTERFACE_NOARG** - interface function without arguments
  - **tooINTERFACE_ARG** - interface function with one scalar argument
  - **tooINTERFACE_STR** - interface function with one string argument
  - **tooINTERFACE_STR_STR** - interface function with two string arguments
  - **tooINTERFACE_STR_ARG** - interface function with one string and one scalar argument

  More combinations of parameter types can be added when needed.

5.5 Functions

The first function to be invoked during LCU start-up from the bootScript is:

- **asmlcuStart** - initiates start-up of the *asmlcu*Server and all software-devices. Via a module table this will call all module-specific *asmxxxStart* functions in order.

The LCC standard commands are also available as public functions.

5.6 Commands

5.6.1 Commands of LCU asmlcuServer Process

LCC standard commands, each with the software-device name as parameter:

Command	Parameters	Cathegory	Action
INIT	device	Public	initialize the device
STANDBY	device	Public	bring device to day-time Stand-By state
ONLINE	device	Public	bring device to night-time On-Line state
OFF	device	Public	bring device to Loaded state
EXIT	device	Public	release all resources allocated by device and enter Off state
STOP	device	Public	stops any movement of device; all devices which can move something need to support this command
SELFTST	device	Public	performs partial test-sequence for device; must be possible in any state; is sent by LCC automatically during start-up
SIMULAT	device	Maintenance	set device into simulation-mode; all commands in this table except STATUS are then intercepted, the device-specific function is skipped and return OK.
STOPSIM	device	Maintenance	disables verbose-mode for device
TEST	device	Test	performs complete test-sequence for device
STATE	device	Public	returns state and simulation-mode setting for specific device (or 'all')
STATUS	device	Public	returns information about device status as plain text

The following are semi-standard commands:

Command		Cathegory	Action
BREAK	(none)	Maintenance	send STOP to all controlled software-devices
KILL	(none)	Maintenance	send EXIT to all controlled software-devices, then terminate server
VERBOSE	device, control	Public	enable/disable verbose-mode for specific device (or 'all') ; in verbose-mode additional intermediate replies are sent
VERSION	(none)	Public	returns the version-IDs of all modules

Device-specific commands will be listed under the respective sub-module. However, the CDT and CIT entries shall be implemented in the top-level asm module, as well as the LCC-compliant command handler function. The latter will only be a wrapper and calls a specific handler function which is then implemented in the sub-module.

5.7 Configuration Files

- **asmlcu.boot** - shell script to start up the LCU part. The installed version will contain only a call to *asmlcuStart*, but by placing the file in a higher prioritized directory according to *BOOTPATH*, specific configuration is possible.
- **asmlcuMonitor.config** - defines the behaviour of the monitor process.

6 MODULE 'asmi' - Internal Base-Level

6.1 Purpose

This module contains common ASM-internal definitions that can and shall be used by all other ASM modules. It must therefore be compiled and installed before any other higher level module.

6.2 Overview

All what is common to the ASM modules shall be collected here, so: **Whenever something is needed by more than one LCU module, it shall be moved to asmi in order to avoid code duplication.**

This includes macros, type declarations, functions, and (database) classes.

6.3 Database Classes

6.3.1 asmi.class

This includes several other files. In total the following classes are provided for WS and LCU:

- **asmiCLASS**: this shall be used instead of *BASE_CLASS* in all ASM LCU-based modules. At present both classes are identical, but they may deviate in future.
- **asmiBRANCH asmiDEVICE asmiTASK** - base-classes for the whole database branch that a module provides. Each module shall provide a single class derived from this type that contains all database attributes which the module needs.
- **asmiDATA**: Generic class for a public data point for a device. All data points for public access shall inherit from this.
- **asmiSCANDEV**: Replaces the standard CCS <VLT scan dev> class. Some attributes are overloaded for ASM-specific needs.
- **asmiCOMWS**: Generic class for a serial communication line; inherits from *tooCOM_PARAMS*. Same as *asmiCOM*, but for WS side with alarms.

6.3.2 asmiLCU.class

The following LCU-specific classes are provided:

- **asmiSIGNAL**: base-class for LCC signals, only for internal use. It has the attributes:
  - *value*: contains the value of the signal.
  - *value_status*: contains the non-zero value if LCC detects the signal as invalid.
  - *description*: contains a string for on-line help for the signal. This shall be overloaded with a meaningful text for each signal instance.
- **asmiSIG_DIGITAL**: digital LCC signal. This shall be used for all general purpose digital signals in an LCU. The type of *value* is *uint32*.
- **asmiSIG_ANALOG**: dto., but for analog signals. They differ in the type of the *value* attribute, here it is *float*.
- **asmiCOM**: Generic class for a serial communication line; inherits from *tooCOM_PARAMS*.

6.4 Headers

6.4.1 asmi.h

Public header of asmi module.

6.4.2 asmiDefines.h

This header contains defines that can be shared with dbl classes.

6.4.3 asmiLCU.h

This header shall be included by all modules' *xxxInternal.h*, and contains the following declarations:

- Types
  - **asmiCmdArg_t** - a pointer to this struct is passed as parameter to all functions that handle LCC standard commands. It has the following members:
    - **msg** - pointer to the message with which the command was sent. Shall be given as argument when sending intermediate replies with *tooReplyf*.
    - **devName** - name of the software-device.
    - **verbose** - verbose enable flag. Intermediate replies shall be sent only when this flag is asserted.
    - **simulat** - simulation mode flag. Note that when a device is in simulation mode, all calls to LCC standard commands will anyhow be intercepted by the global command-handler. It is included here for future extensions.
    - **oldState** - current device state. The module-function may decide upon the current state which action must be done, e.g. the actions for the *Standby* function will usually be different when invoked in state *Online* or *Loaded*.
    - **newState** - coming device state. The module-function may set this member to another value in order to set the device to a different state as previously expected. If the value is negative then no setting of the state will take place.
  - **asmiCmdFunc_t** - Prototype for LCC standard functions in ASM modules. All module-functions must comply with this prototype. The function has two parameters: a pointer to the *asmiCmdArg_t* struct as described above, and the *error* pointer.
- Command handling macros

  By default the last reply of a command is always the string 'OK'. The following macros allow to set the last reply explicitly:
  - **tooINTERFACE_REPLY_RETURN** - (developed for ASM but included in *too* module) sets the last reply to be returned by the CI and returns with *SUCCESS*.
- Error handling macros

  These macros take automatically care of the location-ID. 'error' must be used as pointer to the error-stack. Given error-macros must not include the '<mod>ERR_' prefix, this will be automatically added.
  - **asmiERROR_ADD**: add an error.
  - **asmiERROR_RETURN**: add error and return with *FAILURE*.
  - **asmiERROR_CONTINUE**: log current error and continue in loop.
  - **asmiERROR_ADD_CONTINUE**: add and log error and continue in loop.
  - **asmiERROR_BREAK**: log current error and break innermost loop.
  - **asmiERROR_ADD_BREAK**: add and log error and break innermost loop.
  - **asmiERROR_EXIT**: log current error, call *ccsExit*, return with *FAILURE*.

- **asmiERROR_ADD_EXIT**: add and log error, call *ccsExit*, return with *FAILURE*.

6.5 Workstation Classes

To be done by Rodrigo!

Some generic classes are provided, particularly for use with the *evh*:

- **asmiCLASS** - our base class, usually empty.
- **asmiCMD_RECEIVE** - class to handle user commands in an *evhHANDLER* application; inherits from *evhSIMPLE_TASK*.
- **asmiCMD_SEND** - dto., inherits from *evhCOMMAND*.
- **asmiDEVICE** - inherits from *asmiSTATE*.
- **asmiHISTORYTABLE** - wrapper around Rtap history table handling.
- **asmiMAIN** - inherits from *asmiSTATE*.
- **asmiSTATE** - inherits from *evhSIMPLE_TASK*.

6.6 LCU Functions

- **asmiId** - returns the version-ID string.
- **asmiTaskSpawn** - to spawn a task
- **asmiTaskDelete** - to terminate a task
- **tooReplyf** - (developed for ASM, but included in *too* module) shall be used to send intermediate replies. To simplify its usage, the parameter list consists only of the msg-pointer that has been passed through the CI, and a printf-style format/value list. Errors during the function are ignored.

6.7 Commands

None.

6.8 Configuration Files

None.

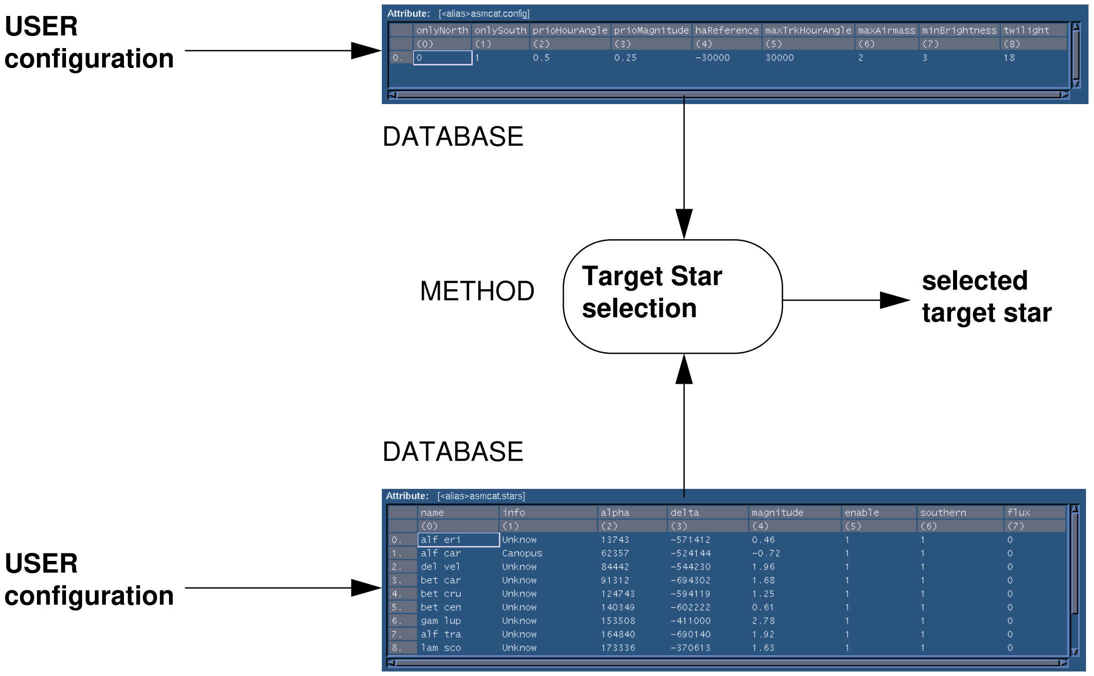
7 MODULE 'asmcat' - Target Star Catalog

7.1 Purpose

This module contains a list of target stars with their parameters, and methods to select one of them, based on certain criteria. It also takes care of the sun position.

7.2 Overview

The target star selection is based on the database contents that the user has configured. There are the list of possible stars from which to select, and the configuration with selection criteria, i.e. how to select. The software can then call a method to do the selection, and gets the parameters of the most suitable star.



7.3 Classes

The programmatic interface of the *asmcatDEVICE* class is as follows:

- SelectStar - automatic selection of the most suitable target star from the list.
- CalculateLST - get local siderial time.
- TimeToSunAtHorizon - for sun position calculations
- SetSiteCoordinates - define observatory location
- UpdateFlux - update the measured flux for the current target star in the list.

The *asmcatDEVICE* database class has the following attributes:

- **config** - configuration of the target star selection. This includes setting of the half-sky, priorities of criteria, etc. Tuning of the automatic selection must be done with these values.
- **stars** - this is the table containing the target star parameters: name, position, magnitude, etc. It also contains the flux value that shall be updated from the measurements done on the corresponding star.

7.4 Headers

- **asmcat.h** - public interface of module
- **asmcatDEVICE.h** - public interface of DEVICE class.

7.5 Functions

None.

7.6 Commands

None.

7.7 Configuration Files

- **asmcat.dbcfg** - file representation of the database contents. ***TBD***

8 MODULE 'asmdim' - DIMM Control

8.1 Purpose

This module controls the DIMM technical CCD and performs the image analysis calculations to provide the seeing and other values.

8.2 Overview

The DIMM control is distributed over the environments *wasm* on WS and *lasm2* on LCU:

- On LCU: image analysis, frame validation, stacking of all params of a sequence.
- On WS: setup and monitoring of exposure sequence, transfer of whole sequence params from LCU, end of sequence validation and statistical computation.

Most of the features of the standard CCD software are used for image analysis, i.e. bias reference frame substraction, statistical calculation (min, max) center of gravity calculation (incl. background substraction, thresholding). FWHM determination is indirectly called.

8.3 Classes

Most of the programmatic classes in this module directly match into a database class of the same name.

- asmdimDEVICE - this is the top-level class of the module containg all factilities.
- asmdimCONFIG - contains all user-changable configuration items of the DIMM. To have a single entry point for the user, the configuration is concentrated here, rather than spread over the individual DIMM objects. A reference to the configuration is passed to the other objects.
- asmdimSEQ - performs the exposure sequence for measurement. The CONFIG object is asked for current configuration, the CAMERA is set up accordingly, control is passed to the EXPOS object, and finally the CALC object is used to perform the calculation.
- asmdimSEQHISTORY - records the (internal) history of sequence parameters.
- asmdimEXPOS - handles the single exposures during a sequence. This class is actually distributed on WS and LCU; all image analysis is done on the LCU side and resulting values are stored in the EXPOS instance in the LCU database. The programmatic EXPOS class on the WS connects to the LCU database instance.
- asmdimCALC - performs the calculation onthe exposure sequence data.
- asmdimDATA - handles the public output of data.
- asmdimDATAHISTORY - records the history of public output data.
- asmdimCAMERA - represents the DIMM technical CCD camera. Methods are provided that directly match into commands of the CCD SW.
- asmdimAG - provides auto-guiding functionality.

8.4 Headers

Each class has its public header (same name) that must be included when an object is instanciated. Beside the following header exist:

- asmdim.h - including all classes.
- asmdimDefines.h - macros shared by programmatic and database classes.

8.5 Functions

LCU start-up and state swtiching functions are:

- asmdimStart - is called from the asmdim boot script to set up direct database addresses etc.
- asmdimInit - no operation.
- asmdimOff - no operation.

For the image analysis on the LCU teh following functions are dedicated:

- asmdimIpHook - this is called by the CCD SW for each window to be processed. The function is configured as user IP function.
- asmdimIpComputation - performs the calculation of all image analyssis parameters that are not directly provided by the standard CCD SW.
- asmdimExposEvaluation - called at the end of each exposure to perform the calculation of image-related parameters, frame validation, and storing into database.
- asmdimFrameValidation - performs the complete validation of one exposure in a sequence, consisting of two windows.

8.6 Commands

None.

8.7 Configuration Files

- asmdim.boot - this script is executed as part of the bootScript when the LCU starts up. Several variables are set here for tuning and debugging.

9 MODULE 'asmtel' - Telescope Control

9.1 Purpose

The DIMM telescope has an η–θ (eta-theta) mounting type, and the *asmtel* module will be the framework for standard TCS modules that control the motion of the two independent axes of the telescope. By sending an equatorial target position, the telescope will then preset to an astronomical object and track its apparent movement over the telescope site sky. It also allows to send offset-steps for auto-guiding.

9.2 Overview

Several TCS modules are re-used here:

- **axis**: module designed for the real-time control of the motor driven axes of the telescope.
- **trk**: tracking module which generates the appropriate position set-points in an alt/az coordinate system. The coordinates are converted by *asmtel* into the η–θ system.
- **sla**: library of astronomical calculation routines, used by *trk*.
- **acm**: programmatic interface between the application module *asmtel* and the *axis* and *trk* modules. Basically, *acm* provides a way to set pointers to data and functions that will be required by the *axis* and *trk* modules and that must be provided by the specific application module.

Fig. 2 Interaction of Telescope device with TCS modules

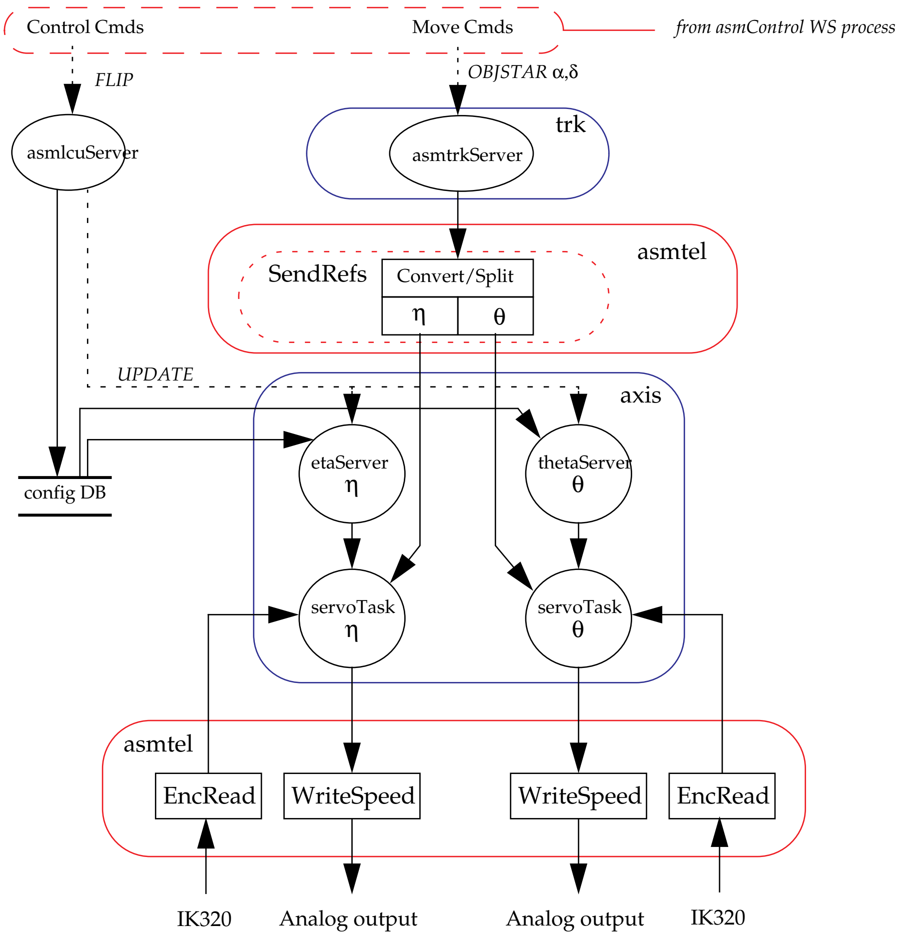


Fig. 2 shows the interaction between the asmtel module and the tracking and axis modules. The supervising process of the telescope software device is the common *asmlcuServer*, which is a Command Interpreter based application. Device start-up and LCC standard commands are managed by the asmServer, while the telescope movement commands are directed to the *asmtrkServer*, causing a preset of the position of the telescope and start the tracking of the object after the preset position is reached. Note that tracking commands do not pass through the *asmlcuServer*, but are sent directly from the WS control process to *asmtrkServer*.

Due to mechanical constraints only one half (north or south) of the sky can be covered under normal preset or offset commands. **One cannot track over the limits of a half-sky!** An explicit flip-over command must be given to *asmlcuServer*, and after that one is forced to stay within the same half-sky during tracking.

The Telescope software device shall provide the specific initialization and operation of the main hardware components of the telescope:

- the motor driven axes system and
- the encoder-based position measurement system.

The axis module foresees two hardware-dependent interfaces. In the present case, main features of these interface parts to the telescope hardware are:

- analog voltage speed reference amplifiers and
- single head reference encoder interfaced by an IK320 board.

The asmtel module shall provide functions that interface the axis module to these units.

9.3 Database Classes

9.3.1 asmtel.class

The classes of asmtel inherit from base-classes included in *asmi.class*, *axis.class* and *trk.class*.

- **asmtelDEVICE** - the complete database branch for the Telescope device. It consists of the two axes, and the tracking support.
- **asmtelAXIS** - represents an axis. It inherits from *axisAXIS*. It adds the components *ENCODER* and *AMPLIFIER*.
- **asmtelPOS_CTLR** - the position controller is part of an axis. It inherits from *axisPOS_CTLR*. As the base-class it contains the component *VELLOOP*. The *ABSENCODER* component of the base-class is not used, since no absolute encoder exists here. Instead the incremental encoder is placed as component of the *asmtelAXIS* class.
- **asmtelVEL_CTLR** - the velocity controller is part of the position-controller. It inherits the attributes from *axisSTD_VEL_CTLR* and overloads default configuration values.
- **asmtelENCODER** - represents an encoder. It inherits from *axisSTD_ENCODER*. It consists of the configuration values for the encoder.
- **asmtelAMPLIFIER** - represents an amplifier. It consists of the signals connected to the amplifier.

9.4 Headers

9.4.1 asmtel.h

This header contains the prototypes of exported functions for use by the top-level asm module:

- asmtelStart function for start-up
- the command-handler functions

9.4.2 asmtelInternal.h

This header contains private declarations:

- **asmtelAXESDATA** - structure to hold data common for both axes.
- **asmtelENCDATA** - structure to hold global data of encoder interface.
- **asmtelHWDATA** - structure to hold global data of amplifier interface
- Prototypes of interface functions to the axis module

9.5 Functions

The asmtel module provides two cathegories of functions:

1. direct command handlers for actions that are not covered by trk or axis
2. the 'glue-logic' between trk and axis, as well as between axis and the hardware

For the latter point the axis module design [12] foresees that the specific management of the hardware (amplifier and encoder) will be done via two different software modules (denoted as *hw* and *enc*), but in this case both parts of the hardware interface will be implemented within the same, the *asmtel*, module.

The code developed to deal with the hardware-specific interfaces, shall be identified by the following file naming convention:

- *asmtelHw* prefix for amplifier-dependent part
- *asmtelEnc* prefix for encoder-dependent part
- *asmtelTrk* prefix for interface to trk

9.5.1 Interface to *axis* and *trk*

- **asmtelFlip** - do a flip-over, i.e. select one half-sky for tracking

Due to mechanical constrains only one half (north or south) of the sky can be covered at a time. This means that one half-sky must be explicitly selected before issuing tracking commands. The flip-over sequence consists of the following steps:

1. Move eta-axis: $\eta = 0$ deg
2. Move theta-axis: for north $-90 < \theta < +90$; for south $90 < \theta < 270$

A special case must be considered for the encoder *Init* function: Due to the fact that at initialization-time the encoder cannot be used to get the actual telescope position, the next procedure must be followed during initiation:

- find out the actual half-sky pointed by the telescope by means a provided hardware signal (***TBC***)
- do a flip-over movement as above
- go to a default telescope position (***TBC***)

NOTE: the detection of the actual half-sky without having the encoder information is not clear yet!

During on-line movements hardware limits must be kept to avoid damage in the telescope structure. Because no vicinity limits are available in the DIMM hardware this must be achieved by means of software limits, which are normally set (as the trk specification defines) such that they are detected before the vicinity limits or interlocks. Depending in which half zone of the sky the telescope actually is currently pointing, a **dynamic reconfiguration of the software-limits** must be performed. In this sense the needed software limits are the following:

South:

- -100° < η < 0°
- 95° < θ < 265°

North:

- 0° < η < 100°
- -85° < θ < 85°

To implement the function that sets new software limits (asmtelSetLims) two aspects must considered:

- the new limits will be active after the next *UPDATE* command to the axis device
- database attributes with the new software limits must be provided.

The DB attributes in axis for the software limits are the following:

- <*axisName*>.posLims[4]
    - posLims[0] = lower limit (not used)
    - posLims[1] = lower limit (SW limit)
    - posLims[2] = upper limit (SW limit)
    - posLims[3] = upper limit (not used)

9.5.2 Interface *axis* - Encoder

The file *asmtelEnc.c* shall provide the functions interfacing the axis module to the encoder. These are:

- **asmtelEncStart** - start-up of encoder interface (a.k.a. *InitAll* function)
- **asmtelEncInit** - initialize the encoder:
    - Open channel to ikon device (attach)
    - Move to limit switch, taking the movement restrictions into account (see section 9.5.1)
    - Move in reverse direction until encoder finds the first reference-mark
- **asmtelEncOff** - uninitialize the encoder:
    - Close channel to ikon device (detach)
- **asmtelEncRead** - read the encoder
- **asmtelEncVersion** - return version string of asmtel module

Other functions foreseen in the axis interface are not needed in this application (i.e. *encExit*, *encCalib* can be defaulted).

9.5.3 Interface *axis* - Amplifier

The file *asmtelHw.c* shall provide the functions interfacing the axis module to the amplifier HW. These are:

- **asmtelHwStart** - start-up of amplifier interface (a.k.a. *InitAll* function)
- **asmtelHwWriteSpeed** - write velocity reference to the signal connected to the amplifier
- **asmtelHwCheckInterlock** - report the status of the interlock signals
- **asmtelHwOn** - write zero speed, assert ILENABO to enable interlock-chain
- **asmtelHwOff** - write zero speed, negate ILENABO to disable interlock-chain
- **asmtelHwVersion** - return version string of asmtel module

Other functions foreseen in the axis interface are not needed in this application (i.e. *hwInit*, *hwExit*, *hwOff*, *hwCyclic*, *asmtelHwCheckVicinityLimit* can be defaulted).

9.5.4 Interface *trk - axis*

The file *asmtelTrk.c* shall provide the function interfacing the *trk* to the *axis* module:

- **asmtelTrkSendRefs** - send new position set-points to each axis server. This includes:
  - conversion from alt/az to η–θ coordinate-systems.
  - Split the η–θ couple in η and θ values and send each one to the corresponding axis server.

This function is of particular importance for the ASM, because: (i) the mounting type of the DIMM telescope is η–θ and (ii) the architecture (depicted in Fig. 2) of the software system foresees only one tracking server for both axes of the telescope.

Because the output reference position of the tracking device is in the altazimutal coordinate system, a conversion function must be provided. This means a conversion between the alt/az angles an the θ/η angles. This conversion must be provided in the *asmtelTrkSendRefs* function. The corresponding equations are as follows:

$$\eta = f1\,(alt, az)$$

$$\theta = f2\,(alt, az)$$

NOTE: Any side-effects of the trk module when tracking over zenith are not clear yet! For eta-theta this is not a singularity as for alt/az.

9.6 Commands

Telescope movement commands are sent directly to trkServer and axisServers, not to asmServer. The corresponding design descriptions [11] and [12] contain details.

There is only one special command to be supported by the *asmServer*:

- **TELFLIP *north* | *south*** - do a flip-over to select a half-sky for tracking. No operation if the specified half-sky is already selected.

The following commands of *trk* will be used apart from the standard LCC commands; they are directed to the process *asmtrkServer*:

- **OBJSTAR** *alpha,delta* - define object and start tracking
- **OFFSAD** *offsetAlpha, offsetDelta* - give offset step

Additionally provided, but only used for test and maintenance, are:

- **OBJFIX** *alt,az* - go to alt/az position
- **OBJNAME** *posName* - go to named position
- **READPOS** - get current position
- **GETREM** - get remaining tracking time

Special commands of the *axisServer* are also available for test, but not used during normal operation.

9.7 Configuration Files

Signal configuration files with format as required by the axis module [11]:

- **asmtelDigital.config** - signal configuration for digital I/O
- **asmtelAnalog.config** - signal configuration for analog I/O

Encoder configuration file with format as required by the ikon driver [13]. The filename shall be specified in the *ikon.boot* file fo the respective environment:

- **asmtelIkon.config** - encoder parameters configuration, compatible with ikon driver

9.8 Differences to standard VLT Applications

The TCS modules are primarily designed for the VLT and NTT telescope tracking and axis control. The re-use for the ASM implies the following differences to those standard applications:

- Both axes are served by the same LCU. This implies that *axis* and *trk* must be capable of handling this, which means for ASM in detail:
  - Only one trkServer for both axes.
  - Two axisServers and servo-loops are running at the same time.
  - Two database branches, one for each axis (consider duplicate aliases).
  - Two sets of signals (with different names each) to be configured.
- The mounting system is not alt/az. Therefore the position set-points provided by *trk* must be converted to eta/theta, and it is not possible for *asmtel* to give feedback in alt/az coordinates.
- There is no absolute encoder, but only an incremental type, which must be initialized by moving to a limit-switch (which at the same time is an interlock).
- The encoder and hardware dependent module - as described in the axis design [11] - are merged here into the same module *asmtel*.
- There are no vicinity limits (consider behaviour for 'backoff').

10 MODULE 'asmten' - Enclosure Control

- PRELIMINARY -

10.1 Purpose

This module implements the automated operation of the DIMM telescope enclosure. The operation are the open and close actions of the enclosure due to VLT commands sent through the WS or LCU components of this system, and checking the hardware status condition.

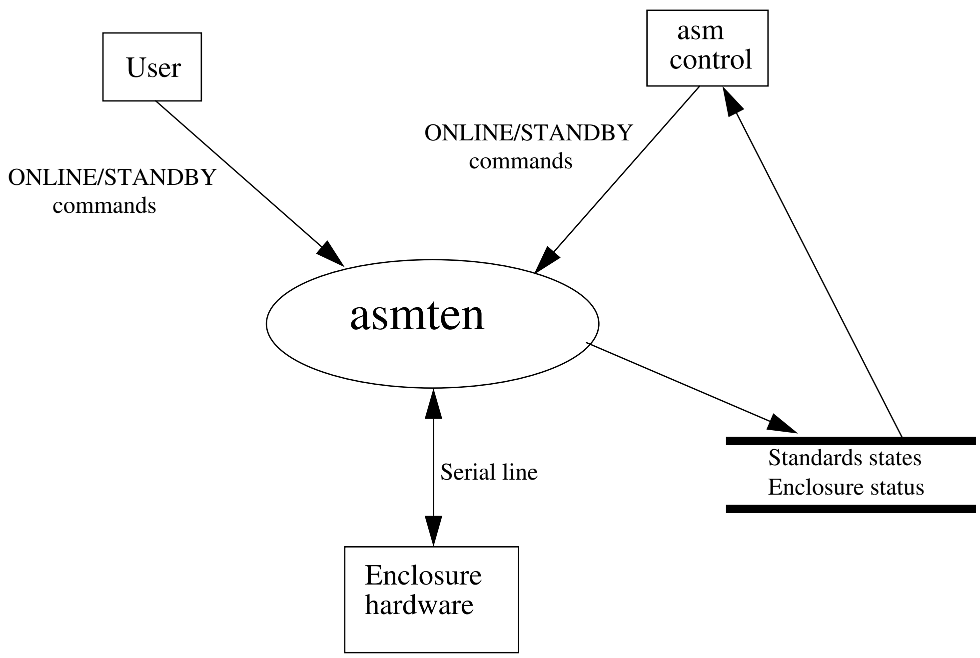
10.2 Overview

No dependencies are foreseen with respect to others modules, the only provided interface for the asmten module should be the LCC standards commands (INIT, STANDBY, ONLINE, EXIT). To Open or Close the enclosure the standard LCC ONLINE and STANDBY commands should be used.

The functionality of the enclosure software is depicted in the Fig. 3, where it's clear that the only way to access the enclosure hardware will be the enclosure software device, which is served by the common asmServer application (CI based application).

The "asm control" correspond to the workstation part of the asm module, that should be in charge of open and close the enclosure at an specific time in the afternoon and dawn, respective. In this way the fully automated operation of the Astronomical Site Monitor system is warranted.

Fig. 3 Enclosure device overview functionality

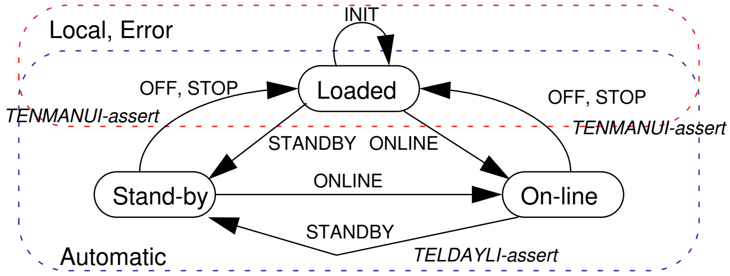


The provided Enclosure hardware is foreseen to be controlled only through a serial line plus an specific communication protocol, which is also a matter of this document. This serial line is the only way to get the status of the hardware as well. The major features of the hardware are the following:

- the dome consist of two independent halves
- the actual position of the two halves is determined by means of encoders, both position values correspond to integer degrees (***TBD***)
- hardware wise close when sun rise (daylight sensor)

The general state diagram of the enclosure system is as follow:

Fig. 4 'enclosure' state diagram



Application specifics substates are defined as follow:

- Open: both halves of the enclosure are complete opened, this condition is true whenever the open limit switches in both halves are asserted.
- Close: both halves of the enclosure are complete closed, this condition is true whenever the close limit switches in both halves are asserted.
- Intermediate: the enclosure is not moving and neither the open and close limit switches are asserted, so that the actual enclosure position is in some unknown place between the total open and total close position.
- Moving: the enclosure is actually moving to the open or close position.
- Malfunction: both open and close limit switches are active at the same time, which is physically impossible so a hardware malfunction is supposed

The description of the standard LCC-device states is as follow:

- Loaded: the state of the physical enclosure (substate) is determined acquiring the state of the hardware digital signals, internal variables are updated. If a malfunction is detected the Enclosure device shall not exit the Loaded state.
- Stand-by: the enclosure is completely closed,
- On-line: the enclosure is completely open

The substate Moving indicates that the device is waiting for the completion of an action, after which the LCC device state will be switched accordingly.

The feasible combination of LCC-device states and application specific substates is resumed in Table 2.

Table 2: 'Enclosure' LCC standard states

State/Substate	Implication
Off	Undefined
Loaded Loaded/Malfunction	Undefined The device could not leave the Loaded state
Stand-by/Close	Enclosure in the close position
On-line/Open	Enclosure in the open position

10.3 Serial line communication protocol

The communication between the ASM system and the enclosure hardware will be always initiated by the asmten module. The protocol used to control the hardware should be based in ascii string commands and one reply from the hardware for each received command. It's up to the asmten module to supervise the real completion of an old command, that is to say, wait for the corresponding reply.

To facilitate the testing of the system the replies from the hardware given an specific command, must be ascii strings ended whit a carriage return character. In this way the translation of the replies and its parameters should be easy using any given host with a serial interface.

Each reply to a command must contain as first parameter an status string indicating the acceptance or refuse of the command. The STATUS_STRING could be one of the following:

- OK: the command was received and the attached action is going on in the hardware
- FAILURE: the command was received but it can not be executed, for example: the sent command string is unknown or the hardware is under local manual control.

The foreseen serial line commands and it's reply are the following:

- ONLINE: both halves of the dome must be taken to the open position, no reply parameters. The reply to this serial command is sent back immediately and is the duty of the Enclosure device to check whether the enclosure was successfully opened or not.
- STANDBY: both halves of the dome must be taken to the close position, no reply parameters.The reply to this serial command is sent back immediately and is the duty of the Enclosure device to check whether the enclosure was successfully closed or not.
- STOP: any movement in the hardware must be stopped immediately, no reply parameters
- GETPOS: get the position of the encoders. Two reply parameters, one for each half part of the dome. The parameter strings should be integer angles in degrees.
- GETSTATUS: get the status of the limit switches in both halves of the dome. That is to say, four integers as replies indicating the assertion or not (0 or 1) of the open/close limit switches in both halves of the dome.
- CHECKLINE: The only acceptable response to this command is OK. If a time-out occurs means that the serial line is not working properly. No reply parameter.

10.4 Internal structure

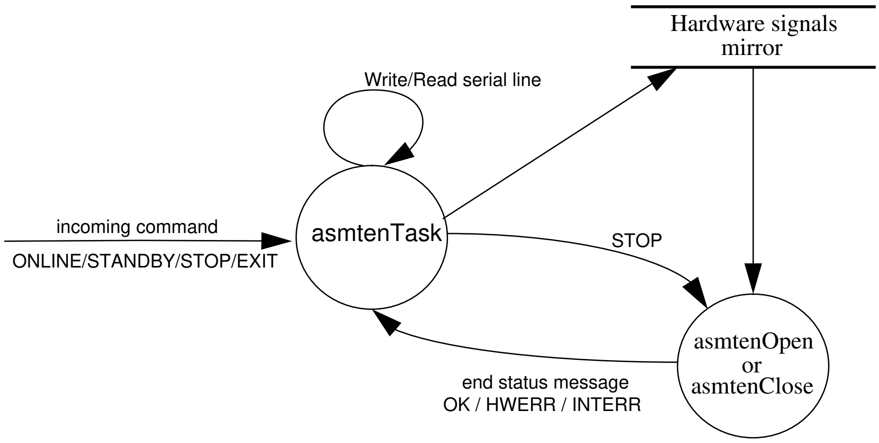
Due to the long time that takes the transition between the open and close position of the dome (aprox. 30 seconds), the implementation is suppose to be as an spawned-task approach, that is to say a main task ("asmtenTask") incharge of:

- <u>periodically supervision</u> of signals not direct involved in the on going action but which can change the actual state of the device. This signals are the manual switch and the daylight indicator (TENMANUI and TENDAYLI). The behaviour of the device after one of this signals is asserted must be the following:
  - TENMANUI: the actual action, if any, must be stopped and the device taken to the Loaded state.
  - TENDAYLI: the enclosure must be closed and then taken to the Standby state.
- <u>coordination of the task</u> needed to open or close the enclosure. In this way the main task (asmtenTask) could continue with its checking duty of the asynchronous signals TENMANI and TENDAYLI. Any needed communicating with the spawned task should be do via the Message System.
- <u>device state transitions</u> after the completion of some command or when one of the asynchronous signals TENMANUI and TENDAYLI is asserted.

The VLT message system should be used to achieve the above. That is to say, asmtenTask will be a task periodically checking if any command arrive to its queue and responding in accordance to the incoming commands and the actual state/substate of the device. In the case of the ONLINE and STANDBY commands, a dedicated task will be spawned which will command the hardware in the corresponding direction and will check the status completion of the hardware movement. The name of this task will be asmtenOpen in the case of the ONLINE command and asmtenClose for the case of the STANDBY command.

To fullfil the above description of the system, which must correspond to the LCC-device "Enclosure", in Fig.5 are shown the foreseen components of the device and their relation by means of messages communication.

Fig. 5 Task and intertasking components of the Enclosure software device



Whitin asmtenTask priorities must be implemented over the received commands. If an ONLINE or STANDBY command is going on (the enclosure is moving toward the corresponding position), then, the opposite command should not be accepted (STANDBY or ONLINE), that means that to reverse the moving direction of the enclosure first an STOP command must be requested. By the other hand, the STOP and EXIT standard LCC command have priority and must be attended immediately, stopping the hardware if correspond and taken the device to the Loaded or Off state.

During the transition time from the states Standby and Online or vice versa, the system must be able to react facing the next six situations:

- a malfunction is detected (this means that both open and close limit switches are actives at the same time). In such a case the reaction must be:
  - send the STOP command to the task which is moving the enclosure (asmtenOpen or asmtenClose)
  - the device state taken to Loaded and the substate to Malfunction
- an STOP or EXIT command of the moving hardware is requested, an STOP command must be passed to the moving task (asmtenOpen or asmtenClose). The final state should be Loaded,
- manual signal is asserted, equivalent to the STOP command,
- daylight signal is asserted, the enclosure must be closed and the final state must be Loaded. Special actions must be taken if the hardware is: (i) not moving, (ii) moving to the open position or (iii) moving to the close position. If the enclosure is already closed then not action must be performed.
- a time-out of the ONLINE or STANDBY command is reached under normal condition of the hardware, in such a case the device should be taken to the Loaded state with Malfunction substate,
- under normal conditions of the hardware the open or close substate is asserted, which means a successful completion of the command.

The code for the asmtenOpen and asmtenClose tasks are the same and only through it parameters the corresponding action is specified.

The asmtenTask will know about the completion status of the moving task (asmtenOpen or asmtenClose) through the VLT message system, expecting an ad-hoc message ID. As is shown in Fig.5 this messages replies are the following:

- OK: the moving action concluded successfully
- HWERR: while moving the domes halves a hardware error occurs,

- INTERR: while moving the dome halves a non determined error occurs in a function call within the code of the moving task.

10.5 Database Classes

The enclosure hardware system is completely operated by means of a serial line link, which encapsulate all the hardware signals. The purpose of the asmten database branch will be to mirror the hardware signals, periodically updated by the asmtenTask process. In this way the values contained in the database can be used as actual values of the signals and also events could be attached to this attributes.

From the software point of view, the input an output signals are the following:

Input:

- limit switches used to signalling the complete close or complete open position of each half part of the dome,
- switches activated by an operator or by the presence of day light
- angle position of each half of the dome

Output:

- relay-driven contact to move each half of the dome (TBC)

10.5.1 asmten.class

10.6 Headers

Three headers file are foreseen:

- signal configuration (asmtenSignals.h),
- internal used variables and functions prototypes (asmtenInternal.h).
- LCC standard commands handlers and handler for the ASM start function (asmten.h)

10.6.1 asmtenSignals.h

10.6.2 asmtenInternal.h

10.7 Functions

- **asmtenStandby** - closes the enclosure via serial line and checks it.
- **asmtenOnline** - opens the enclosure via serial line and checks it.
- **asmtenOpen** - checks via serial line if the enclosure is fully opened.
- **asmtenIsClose**- checks via serial line if the enclosure is fully closed.

The asmten module provides the following type of functions:

- handlers for standard LCC commands
- serial line hardware monitoring and control

In the last case, the monitoring task is performed by the asmtenTask and basically means the status supervision of the hardware through data interchange over the serial line link. The provided code should be in the asmtenTask.c file.

By the other hand, the movements control of the dome are supervised by the code provided in the asmtenMove function, contained in the asmtenMove.c file. This code will be spawned as the asmtenOpen or asmtenClose task, according to the situation. This means that the asmtenMove function should be parametrized in a clever way that allow the use of the same code for open or close the enclosure.

10.8 Commands

Because the open and close conditions of the dome coincides with the Online and Standby LCC standard state of the Enclosure software device, no special commands are foreseen for this purpose.

- **STANDBY Enclosure** - close both halves of the enclosure
- **ONLINE Enclosure** - open both halves of the enclosure
- **STOP Enclosure** - stop the actual movement of the enclosure (if any)
- **EXIT Enclosure** - equivalent to the LCC EXIT command
- **TENISOP** - checks if enclosure is open, sends error-reply if not
- **TENISCL** - checks if enclosure is closed, sends error-reply if not

10.9 Configuration Files

None.

- T O B E D O N E -

11 MODULE 'asmmet' - Meteo Station

12 MODULE 'asmsky' - All-Sky Monitor

13 MODULE 'asmsei' - Seismic Monitor

14 MODULE 'asmdus' - Dust Monitor

15 MODULE 'asmire' - Infrared Emissivity

16 MODULE 'asmei' - Engineering Interface

17 SPECIAL MODULES - BUILD ENV INS

17.1 Purpose

As indicated by the upper case letters in the module name, these are no usual code containing modules, but intended for special purposes. They contain procedures and configurations that are needed to build up the ASM system from scratch, but which cannot be assigned to a certain code module.

17.2 Overview

All modules typically contain elements that can be regarded as data. For the purpose of configuration control and proper archiving they are put here.

asmBUILD - contains the procedure to build the complete ASM software system. As such it can be regarged at the top-most module of all.

asmENV - contains all WS and LCU environments in the ASM system, which are located under the *VLTDATA* tree. They are generated in their initial default configuration.

asmINS - contains all instrumentation files that need to be present under the *INS_ROOT* tree. This includes CCD setup files, reference BIAS frames, etc.

Refer to the Reference chapter for more information.

17.3 Database Classes

None.

17.4 Headers

None.

17.5 Functions

None.

17.6 Commands

None.

17.7 Configuration Files

TBD

18 ENVIRONMENT 'wasm'

18.1 Purpose

The *wasm* workstation-based Rtap environment controls the whole ASM system. It is the entry point for external users.

18.2 Overview

The standard Rtap scheme starts up all processes defined for the environment, as listed in the RtapEnvTable. Then the *asmwsControl* process takes care of automatic start-up, coordination and continous operation of all ASM facilities.

The ASM data will be updated in the database of the environment, where they are available to external users via public attributes through standard CCS/Rtap access mechanisms, see section 18.5.

As new data become available, they will be sent to the VLT on-line archive through the *volac* process.

History data recording for internal purpose is performed with standard Rtap historian facilities.

18.3 Processes

Public processes (command interpreters):
- *asmwsControl* - main ASM control process
- *cmdManager logManager msgServer* - CCS standard

Internal processes:
- Rtap* - Rtap standard processes are not explicitly mentioned here
- *oslxServer* - to generate FITS files
- *volac* - online archive client

18.4 Configuration

RtapEnvTable

RtapDbStartup options - TBD, currently defaulted

RtapScheduler options - TBD, currently defaulted

18.5 Database

DATABASE.db

USER.db

Structure

Public attributes

19 ENVIRONMENT 'lasm1'

19.1 Purpose

The LCU environment *lasm1* incorporates the following control functions:

- Presetting and tracking motion control of both telescope axes
- Control of telescope enclosure movement
- Serial data communication with all external sensor boxes (meteo etc.)

19.2 Overview

General words about the functionality and how it is organized.

19.3 Modules

List of modules to be loaded:

System modules:
lcudrv lculog lqs acro aio iser ikon tim lcc cai scan

User modules:
inducer too sla tcsVcc axis trk asmlcu

19.4 Devices

Hardware devices:

Digital I/O:
/acro0 - limit switches, general purpose I/O

Analog I/O:
/aio0 - velocity reference to drive amplifiers

Encoder I/F:
/ikon0 - telescope axes encoders: eta-axis = channel X1, theta-axis = channel X2

Serial I/O:
/iser0 - Enclosure
/iser1 - Meteo station
...
/iser9 - (not assigned)

Time bus interface:
/tim0

Software-devices:

Monitor - background monitor process

Enclosure - telescope enclosure

Telescope - DIMM telescope

etaAxis - telescope eta axis

thetaAxis - telescope theta axis

trk - tracking calculation

Meteo - meteo station

Note that *Telescope* is not defined under LCC, but only the individual parts of it (*etaAxis*, thetaA*xis*, *trk*).

19.5 Processes

Public processes (command interpreters for normal operational access):
asmtrkServer - for telescope motion control
asmlcuServer - for ASM-specific software-device control

Public processes (command interpreters for test and maintenance access):
lccServer msgServer rdbServer - LCC standard
inducerServer - for driver test panels (test and maintenance only)
etaServer thetaServer - for eta/theta axes control

Internal processes:
asmMonitor - as background process to monitor and react on asynchronous events

19.6 Configuration

ikon0.dat

ikon.boot

trk.boot

devicesFile

userScript

asmMonitor.config

asmtelDigital.config

asmtelAnalog.config

19.7 Database

DATABASE.db

USER.db

Structure

Scanned attributes

20 ENVIRONMENT 'lasm2'

20.1 Purpose

The LCU environment *lasm2* incorporates the following control functions:

- Seeing monitor TCCD camera control
- Image processing for seeing calculation etc.

20.2 Overview

General words about the functionality and how it is organized.

20.3 Modules

List of modules to be loaded:

System modules:
lcudrv lculog lqs tim lcc cai scan b016

User modules:
inducer too ccd asmdim

20.4 Devices

Hardware devices:

CCD link devices
/bxvi00 - DIMM CCD
/bxvi01 - n/a
/bxvi02 - n/a
/bxvi03 - n/a

Time bus interface:
/tim0

Software-devices: none

20.5 Processes

Public processes (command interpreters):
lccServer msgServer rdbServer - LCC standard
inducerServer - for driver test panels (test and maintenance only)

Internal processes:
ccdcon_DIMM ccdip_DIMM ccdrdt_DIMM ccdtel_DIMM ccdtmp_DIMM - CCD control

20.6 Configuration

devicesFile

userScript

asmMonitor.config

20.7 Database

DATABASE.db

USER.db

Structure

Scanned attributes

21 USER INTERFACES

==== TBD ====

21.1 Public Database Attributes

@wasm<alias>*POINT.ATTR*(*ROW*,*FIELD*)

POINT - identifier of the specific instrument or data-station (*Dimm*, *Meteo*, etc.)

ATTR - identifier for the attribute under *POINT*, usually *data*

ROW - identifier for the data item, i.e. a row in a table. If omitted, the all rows are returned in order.

FIELD - identifier for the specific statistical value. If omitted, then all fields are returned in order.

POINT	ATTR	ROW	FIELD
Meteo	data	Temp Temp1 Temp2 Temp3	now, average, minimum, maximum, deviation
Dimm	data	Seeing	now, average, minimum, maximum, deviation

21.1.1 Access from C programs

21.1.2 Access from C++ programs

21.1.3 Access from sequencer scripts

seq_dbReadSymbolic @wasm<alias>Meteo.data(Temp0,average)

21.2 ASM Operation Control

21.2.1 Start-Up

21.2.2 Shut-Down

21.3 Monitoring

21.4 Engineering

22 SOFTWARE INSTALLATION

The installation of the complete ASM software consists of the following steps:

1. Install the VLT common software release in full Rtap version.
2. Install the VLT CCD software release (if not included in the common SW).
3. Install the needed TCS modules (as far as not included in the common SW). These are:
   a. sla - astronomical calculation library
   b. tcs - common TCS definitions
   c. msw - mode-switching (only public headers are needed, so *make install* is sufficient)
   d. acm - axis control module
   e. trk - tracking
   f. axis - axis control
4. Install the ASM software. A build-procedure is provided in the *asmBUILD* module. Invocation of it compiles and installs the complete SW.
5. Create and configure the ASM environments as described earlier in this document.

23 REFERENCE

All references are listed in alphabetical order within the following sections: [1]

1. Upper-case characters are inserted before all lower-case characters.

23.1 WORKSTATION APPLICATIONS

23.1.1 asmwsControl(1)

NAME

```
asmwsControl - astronomical site monitor control process
```

SYNOPSIS

```
asmwsControl [-r dbRoot] [-n processName]
```

DESCRIPTION

```
The asmwsControl process is part of the asm module. It receives and handles
all user and standard commands to the module. The process is responsible
for coordinating the actions of all astronomical site monitor facilities
by sending commands to the appropriate sub-modules.

The optional argument <dbRoot> is the root point of the ASM database
branch. If omitted it is defaulted to the local database root point.

The optional argument processName is the name under which the process'
instance registers with the CCS and hence the destination process name
for commands to it. If omitted, the instance registers with CCS with
its UNIX process name, as passed by the shell in argv[0].
```

FILES

```
asmws_ERRORS - r - error message definitions
```

COMPILATION FLAGS

```
TARGET_TCS Is used to build the application for VLT (VLT_TCS) or
           NTT (NTT_TCS). It is usually set to the correct value
           in the makefile
```

COMMANDS

```
START   starts the automatic operation of the ASM
and all standard commands
```

RETURN VALUES

```
0 if everything OK
1 on error
```

EXAMPLES

```
shell> asmwsControl -r :testDB:TCS &

Starts a process instance with process name asmwsControl and specifies
":testDB:TCS" as the TCS branch root point.
```

SEE ALSO

```
"asm Control Software Design", VLT-SPE-ESO-17441-1213
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:58
```

23.2 WORKSTATION CLASSES

23.2.1 asmcatDEVICE(4)

NAME

```
asmcatDEVICE - ASM target star catalog handling
```

SYNOPSIS

```
asmcatDEVICE::asmcatDEVICE(const dbSYMADDRESS dbroot,
                           const dbSYMADDRESS dbremotetarget);
asmcatDEVICE::~asmcatDEVICE();
ccsCOMPL_STAT asmcatDEVICE::Init(void);
ccsCOMPL_STAT SelectStar(vltDOUBLE &alpha, vltDOUBLE &delta,
                         vltDOUBLE &magnitude, vltDOUBLE &duration);
ccsCOMPL_STAT asmcatDEVICE::SelectStar(vltDOUBLE lstNow,
                                       vltDOUBLE &alpha,
                                       vltDOUBLE &delta,
                                       vltDOUBLE &magnitude,
                                       ccsTIMEVAL &duration);
ccsCOMPL_STAT asmcatDEVICE::UpdateFlux(double flux);
ccsCOMPL_STAT asmcatDEVICE::UpdateConfigBuffer();
ccsCOMPL_STAT asmcatDEVICE::SunPosition(vltDOUBLE  timeMJD,
                                        vltDOUBLE &alt, vltDOUBLE &az,
                                        vltLOGICAL &IsNight)
ccsCOMPL_STAT asmcatDEVICE::SunAltitude(vltDOUBLE &alt,
                                        vltDOUBLE  timeMJD)
ccsCOMPL_STAT asmcatDEVICE::RegulaFalsi(vltDOUBLE &timeZero,
                                        vltDOUBLE x1, vltDOUBLE x2)
ccsCOMPL_STAT asmcatDEVICE::SunAtHorizon(vltDOUBLE   timeMJD,
                                         vltDOUBLE  &timeSunAtHorizon,
                                         vltLOGICAL &IsNight)
ccsCOMPL_STAT TimeToSunAtHorizon(vltDOUBLE  &timeToSunAtHorizon,
                                 vltLOGICAL &IsNight)
ccsCOMPL_STAT asmcatDEVICE::CalculateHA(vltDOUBLE &hourAngle,
                                        vltDOUBLE lst,
                                        vltDOUBLE alpha);
ccsCOMPL_STAT asmcatDEVICE::calculateLST(vltDOUBLE  &lstTime,
                                         ccsTIMEVAL  timeNowUTC);
ccsCOMPL_STAT asmcatDEVICE::calculateLST(double *lstTime);
ccsCOMPL_STAT asmcatDEVICE::calculateLST(char *strNow);
ccsCOMPL_STAT asmcatDEVICE::calculateAirmass(vltDOUBLE  hourAngle,
                                             vltDOUBLE  delta,
                                             vltDOUBLE &zenithDistance,
                                             vltDOUBLE &airMass);
vltDOUBLE asmcatDEVICE::TrackingTime(vltDOUBLE starHA);
vltDOUBLE asmcatDEVICE::CriterionFunctionHA(vltDOUBLE  hourAngle);
vltDOUBLE asmcatDEVICE::CriterionFunctionMv(vltDOUBLE  magnitude);
ccsCOMPL_STAT asmcatDEVICE::CriterionFunction(vltDOUBLE  hourAngle,
                                              vltDOUBLE  magnitude,
                                              vltDOUBLE  prioTrktime,
                                              vltDOUBLE  prioMagnitude,
                                              vltDOUBLE &z);
ccsCOMPL_STAT asmcatDEVICE::SetSiteCoordinates(vltDOUBLE longitude,
                                               vltDOUBLE latitude,
                                               vltDOUBLE height);
vltINT32 asmcatDEVICE::GetStarsNumber();
ccsCOMPL_STAT asmcatDEVICE::GetStarData(vltDOUBLE   testTime,
                                        vltINT32    starIndex,
                                        vltDOUBLE  *starData);
```

PARENT CLASS

```
public eccsERROR_CLASS
```

DESCRIPTION

```
This class manages the target star list in the ASM database.
```

PUBLIC METHODS

```
The geographical site coordinates (in degrees) are provided in
the constructor of this class because they are needed in the
computation of the local sidereal time.

UpdateConfigBuffer - updates the content of the local variable
configBuffer with that already in the database. This function
should be called by any function that needs

Init - initialize the internal data of the object. No star selected
ion

updateConfigBuffer - updates the content of the local variable
configBuffer with that already in the database.

SelectStar - select target star from the database list

The selection is indicated in the `selected' attribute,
which is an index into the `stars' table: -1 means that none is
currently selected.
The controlling process shall select a new star when the
maximum tracking time has expired.
The selection algorithm proceeds over the start list (from the first
one till the last one) discarding the unsuitable ones: brightness
too small (minBrightness), airmass too high ( > maxAirmass) and
those which actually have an hour angle greater than the maximum
tracking hour angle (maxTrkHourAngle).
After this the algorithm evaluates the criterion function and
decides in a minimization way.
The criterion function has two parts: (i) hour angle reference: given
as an attribute in the database (haReference) this parameter is
interpreted as the prefereded starting hour angle of the selected star.
So, stars with minimum hour angle difference from the reference are
prefered. (ii) Magnitude parte: brighter stars are prefered.
Therefore, the criterion function has the following form:

z = a * f(HA) + b * g(Mv)

where:
    a = prioHourAngle
    b = prioMagnitude
f(HA) = hour angle criterion part
g(Mv) = magnitude criterion part

f(HA) = fabs(hourAngle - haReference)/DPI
g(Mv) = Mv/(2*minBrightness) + 0.5

( the scaling factors helps to mantain the f() and g() values in a
compatible range, in such a way that the addition of both in the
criterion function do not hide the influence of one term
over the other one. In the case of the g() function, is supposed
that the magnitudes of the stars in the list are in the range
[-minBrightness, minBrightness] (minBrightness > 0). If this is
true, then the range of the g() function will be [0,1]).

With this definition of the criterion function the best suitable
star will be the one that minimize the value of z.

UpdateFlux - update measured flux of selected star in database

The controlling process shall update the `flux' attribute
according to performed measurements with the selected star.
The passed value is updated in the database table when it
is higher than the currently stored value.
```

```
CalculateHA - given the local sidereal time and the right ascension
of an astronomical object, then the hour angle is calculated.
The return value is adjusted to the range [-PI,+PI].
The given parameters should be in radians.

CalculateLST(1) - using slalib functions the local sideral time is
calculated. The return result is in radians.
For star selection pourposes high precision is not mandatory. So, the
UTC-UT1 parameter is assumed to be zero.
The timeNowUTC parameter is the actual time used to do the calculation.
The calculation is based in the geographical position previously
set in the siteCoordinates array.

CalculateLST(2) - the LST time is calculated for the real actual
UTC time. To get that time a call to the timsGetUTC function is
performed.

CalculateLST(3) - idem. but the return LST is in the passed string with
the format HHMMSS.TTT

CalculateAirmass - calculate the zenith distance and airmass given the
hour angle and declination in radians of a specific object.
The latituded used to compute the result is that one contained in
the local variable siteCoordinates[asmcatLatitude].

TrackingTime - given an hour angle (rad) this routine computes the
remmaning time before the hour angle value contained in the
maxTrkHourAngle attribute of the configuration table.

CriterionFunctionHA - computes the hour angle criterion function part.
This part of the function measure the distance between the actual
hour angle (rad) of the star and the haReference member of the
config table. The absolute value of the difference is returned.

CriterionFunctionMv -

SetSiteCoordinates -

GetStarsNumber -

GetStarData -
```

```
- - - - - -
Last change:  22/06/97-18:01
```

23.2.1.1 asmcat.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmcat.class  (97/6/21  4:52)
#ifndef ASMCAT_CLASS
#define ASMCAT_CLASS
//************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmcat.class,v 1.12 1997/06/12 12:22:08 vltsccm Exp $"
//*
//* who        when       what
//* ---------- ---------- ------------------------------------------------
//* S.Sandrock 12.12.1996 created
//*/

#include "asmcatDefines.h"
#include "asmi.class"

// ---------------------------------------------------------------------------
// DEVICE BRANCH
//
CLASS "asmiDEVICE" "asmcatDEVICE"
BEGIN
    // CONFIGURATION
    ATTRIBUTE Table asmcatATTR_config (1,
        logical "onlyNorth",       // restrict star selection, only NORTH
        logical "onlySouth",       // restrict star selection, only SOUTH
        double  "prioHourAngle",   // priority of hour angle
        double  "prioMagnitude",   // priority of brightness (magnitude)
        double  "haReference",     // prefered hour angle (HHMMSS)
        double  "maxTrkHourAngle", // maximum tracking hour angle
        double  "maxAirmass",      // maximum starting airmass of the star
        double  "minBrightness",   // Minimun admissible brightness
        double  "twilight"         // twilight definition [deg]
        )
    BEGIN
        Value ((
                0, 1,           // by default only southern half-sky
                0.5, 0.25,      // priority values [0..1]
               -030000.0,       // hour angle referencepriority values [0..1]
                030000.0,       // maximum tracking hour angle
                2.0,            // maximum airmass
                3.0,            // minimum brightness
                18.0            // normal astronomical twilight
              ))
    END

    // TARGET-STAR LIST
    ATTRIBUTE Table asmcatATTR_stars (12,
        bytes8  "name",         // offical name of the star ("alf car")
        bytes20 "info",         // additional text ("Canopus")
        double  "alpha",        // right ascension [Hms]
        double  "delta",        // declination     [Dms]
        double  "magnitude",    // visual magnitude
        logical "enable",       // enable for this table-entry
        logical "southern",     // star lies in the southern half-sky
        double  "flux"          // measured flux (dynamically updated)
        )
    BEGIN
      Value (
      //  name      info              RA          Dec        Mv   Enb sh f
      //("alf can", "Sirius"   , 064509.000, -164258.000,  -1.5, 1,  0, 0),
        ("alf eri", "Unknow"   , 013743.000, -571412.000,  0.46, 1,  1, 0),
        ("alf car", "Canopus"  , 062357.000, -524144.000, -0.72, 1,  1, 0),
        ("del vel", "Unknow"   , 084442.000, -544230.000,  1.96, 1,  1, 0),
        ("bet car", "Unknow"   , 091312.000, -694302.000,  1.68, 1,  1, 0),
        ("bet cru", "Unknow"   , 124743.000, -594119.000,  1.25, 1,  1, 0),
        ("bet cen", "Unknow"   , 140349.000, -602222.000,  0.61, 1,  1, 0),
        ("gam lup", "Unknow"   , 153508.000, -411000.000,  2.78, 1,  1, 0),
```

```
        ("alf tra", "Unknow"   , 164840.000, -690140.000,  1.92, 1,  1, 0),
        ("lam sco", "Unknow"   , 173336.000, -370613.000,  1.63, 1,  1, 0),
        ("alf pav", "Unknow"   , 202539.000, -564407.000,  1.94, 1,  1, 0),
        ("alf gru", "Unknow"   , 220814.000, -465740.000,  1.74, 1,  1, 0),
        ("alf tuc", "Unknow"   , 221830.000, -601535.000,  2.86, 1,  1, 0)
      )
    END

    // SELECTED STAR (dynamically updated)
    ATTRIBUTE int32 asmcatATTR_selected -1 // index into "stars" (-1: none)

END

#endif /*!ASMCAT_CLASS*/
--->END IMPORTED FILE: ../dbl/asmcat.class  (97/6/21  4:52)
```

23.2.2 asmdimAG(4)

NAME

```
asmdimAG - ASM DIMM Autoguider
```

SYNOPSIS

```
#include "asmdimAG.h"

asmdimAG(const dbSYMADDRESS point, asmdimCAMERA *camera)
~asmdimAG();

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Find(double xRef, double yRef,
                   double & xArcsec, double & yArcsec)
ccsCOMPL_STAT Convert(double xRef, double yRef,
                      double xAct, double yAct,
                      double & xArcsec, double & yArcsec)
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This class provides the autoguider functionality of the DIMM.
It supports to take an exposure and determine the error-vector.
It uses the asmdimCAMERA class for all CCD accesses.
```

PUBLIC METHODS

```
Init -  set constant setup-file and angular pitch

Find -  make exposure and determine angular error to reference point.
        The returned error-vector is zero when it is already
        within the configured autoguiding radius.
        No correction is necessary then.

Convert - compute (Ref - Act) * Pitch
```

PUBLIC DATA MEMBERS

```
The following have to be set before calling Init():

fndSTRING AgSetupFile - name of the setup-file to be used.
double XPitch, YPitch - angular pitch per pixel in x/y direction.
```

SEE ALSO

```
asmiCLASS(4), asmdimCAMERA(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.2.1 asmdimAG.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimAG.class  (97/6/21  4:54)
#ifndef ASMDIM_AG_CLASS
#define ASMDIM_AG_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimAG.class,v 1.38 1997/06/20 15:24:51 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ------------------------------------------------
// S.Sandrock 02.05.1997 created
//

// headers:
#include "asmdimAG.h"

// referenced classes:
#include "asmi.class"

// ---------------------------------------------------------------------------
// AUTOGUIDER
//
CLASS   "asmiCLASS"     "asmdimAG"
BEGIN
  // CONFIGURATION:
  ATTRIBUTE double asmdimAG_ag_radius   10      // autoguiding radius [arcsec]

  // DATA:
  ATTRIBUTE double asmdimAG_x_offset    // position error in CCD-X [arcsec]
  ATTRIBUTE double asmdimAG_y_offset    // position error in CCD-Y [arcsec]
  ATTRIBUTE uint32 asmdimAG_new_data    // incremented upon new data update
END

#endif /*!ASMDIM_AG_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimAG.class  (97/6/21  4:54)
```

23.2.3 asmdimCALC(4)

NAME

```
asmdimCALC - ASM DIMM Seeing Computation
```

SYNOPSIS

```
#include "asmdimCALC.h"

asmdimCALC()
~asmdimCALC()

ccsCOMPL_STAT ComputeAll(void)

ccsCOMPL_STAT ImageMotionSeparations(void)
ccsCOMPL_STAT ImageMotionVariance(void)
ccsCOMPL_STAT ScintillationComputation(void)
ccsCOMPL_STAT StatisticalValidation(void)
ccsCOMPL_STAT SeeingComputation(void)
ccsCOMPL_STAT SeeingValidation(void)
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This class provides the DIMM calculation routines for
seeing and scintillation values.

Typically the user sets all IN parameters in the public struct p
(see below), calls the ComputeAll method, and the gets all OUT
parameters from p.
```

PUBLIC METHODS

```
Constructor and destructor do nothing.
No database is used.

ComputeAll -    calculate all output parameters.

The rest of the methods correpond directly to the sections
6.5 to 6.8 in the ASM DIMM Upgrade Plan.
```

PUBLIC DATA MEMBERS

```
p - struct holding all I/O parameters

The member names in p correspond to the variable names listed in
the ASM DIMM Upgrade Plan. See there for a detailed description:

IN:     kc, kN, knx, kny, ks, D, d, lambda, gamma, px, py,
        x1[], y1[], x2[], y2[], A1[], A2[]
IN/OUT: N
OUT:    Xg, Yg, Wg, alphag, F1, F2, S1, S2, S,
        sigmag2para, sigmag2perp, FWHMpara, FWHMperp, FWHM
```

RETURN VALUES

```
Upon successful computation of all output parameters all methods
will return SUCCESS.
Failed validations or illegal input data will result in an error.
```

SEE ALSO

```
asmiCLASS(4),
ASM DIMM Upgrade Plan VLT-SPE-ESO-17410-1174
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.4 asmdimCAMERA(4)

NAME

```
asmdimCAMERA - ASM DIMM technical CCD Camera
```

SYNOPSIS

```
#include "asmdimCAMERA.h"

asmdimCAMERA(const ccdCAMERANAME name);
~asmdimCAMERA();

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Online(void)
ccsCOMPL_STAT Standby(void)
ccsCOMPL_STAT Off(void)
ccsCOMPL_STAT Setup(const char *setupstring)
ccsCOMPL_STAT Start(void)
ccsCOMPL_STAT Stop(void)
ccsCOMPL_STAT Wait(void)
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
The asmdimCAMERA class provides a 1-to-1 interface to the
control process of the CCD detector control software.
The argument `name' determines the camera to be controlled.
The corresponding control process must be running in the
local environment.
```

PUBLIC METHODS

```
Init    - initialize CCD camera and internal data members

Online  - set camera to on-line state
Standby - set camera to stand-by state
Off     - set camera to loaded state

Setup   - define exposure setup
Start   - start an exposure
Stop    - stop current exposure
Wait    - wait until the end of current exposure
```

PUBLIC DATA MEMBERS

```
They reflect directly the CCD public database values (LCU-side):

eccsDB_DOUBLE *DB_ipXCen
eccsDB_DOUBLE *DB_ipYCen
eccsDB_DOUBLE *DB_ipCenVal
eccsDB_DOUBLE *DB_ipFlux
```

PROTECTED METHODS

```
SendRecv - send a command and receive reply
```

PROTECTED DATA MEMBERS

```
ccdProcName - name of the CCD control process, derived from CCD name
```

ENVIRONMENT

```
CCDLENV - name of LCU environment controlling the CCD camera
```

SEE ALSO

```
asmiCLASS(4)
```

BUGS

```
CCDLENV is implicitly assumed for the LCU. No parameter is provided.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.4.1 asmdimCAMERA.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimCAMERA.class  (97/6/21  4:54)
#ifndef ASMDIM_CAMERA_CLASS
#define ASMDIM_CAMERA_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimCAMERA.class,v 1.38 1997/06/20 15:24:50 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 14.04.1997 created
//

// headers:
#include "asmdimCAMERA.h"

// referenced classes:
#include "ccdCAMERA.class"

// ---------------------------------------------------------------------------
// CAMERA
//
CLASS   "ccdCAMERA"     "asmdimCAMERA"
BEGIN
END

#endif /*!ASMDIM_CAMERA_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimCAMERA.class  (97/6/21  4:54)
```

23.2.5 asmdimCONFIG(4)

NAME

```
asmdimCONFIG - ASM DIMM Configuration
```

SYNOPSIS

```
#include "asmdimCONFIG.h"

asmdimCONFIG(const dbSYMADDRESS alias)
~asmdimCONFIG()

ccsCOMPL_STAT Init(void)
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This contains all configuration parameters of the DIMM.
```

PUBLIC METHODS

```
Init - read all attributes from the database.
```

PUBLIC DATA MEMBERS

```
Mirrors of the database values in memory:

vltBYTES80                              log_file;
vltBYTES80                              last_file;
vltBYTES80                              setup_file_ag;
vltBYTES80                              setup_file_bg;
vltBYTES80                              setup_file_seq;
asmdimCONFIG_mode_select_t              mode_select;
asmdimCONFIG_hardware_constants_t       hardware_constants;
asmdimCONFIG_mode_setup_t               mode_setup;
asmdimCONFIG_validation_limits_t        validation_limits;
```

SEE ALSO

```
asmiCLASS(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.5.1 asmdimCONFIG.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimCONFIG.class  (97/6/21  4:54)
#ifndef ASMDIM_CONFIG_CLASS
#define ASMDIM_CONFIG_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimCONFIG.class,v 1.38 1997/06/20 15:24:47 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ------------------------------------------------
// S.Sandrock 02.12.1996 created
//

/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 * C A U T I O N:
 *
 * This header contains database attributes that map into type definitions.
 * They have to be kept aligned with the type definitions in include/asmdim*.h
 *+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

// headers:
#include "asmdimCONFIG.h"

// referenced classes:
#include "asmi.class"

// ---------------------------------------------------------------------------
// CONFIGURATION DATA
//
CLASS "asmiCLASS"      "asmdimCONFIG"
BEGIN
    ATTRIBUTE bytes16  asmdimCONFIG_mode_select "DIMM" // "DIMM" or "COHERENCE"

    ATTRIBUTE bytes80   asmdimCONFIG_log_file           "dimm.log"
    ATTRIBUTE bytes80   asmdimCONFIG_last_file          "dimm.last"
    ATTRIBUTE bytes80   asmdimCONFIG_setup_file_ag      "dimmAG.det"
    ATTRIBUTE bytes80   asmdimCONFIG_setup_file_bg      "dimmBG.det"
    ATTRIBUTE bytes80   asmdimCONFIG_setup_file_seq     "dimmSEQ.det"

    ATTRIBUTE Table     asmdimCONFIG_hardware_constants (asmdimCONFIG_hardware_constants_N,
        bytes32 "parameter", double asmdimCONFIG_value, bytes16 "unit"  )
    BEGIN
      Value (
        ("Telescope Diameter"           ,   0.35       , "m"          ),
        ("Telescope F Ratio"            ,  15.0        , ""           ),
        ("Objective Focal Length"       ,   0.04       , "m"          ),
        ("Pupil Mask Prism Angle"       ,   1.0        , "degree"     ),
        ("Mask Hole Diameter"           ,   0.0009     , "m"          ),
        ("Mask Hole Separation"         ,   0.015      , "m"          ),
        ("Pupils Diameter"              ,   0.118      , "m"          ),
        ("Pupils Separation"            ,   0.200      , "m"          ),
        ("X CCD total size"             , 400          , "pixel"      ),
        ("Y CCD total size"             , 290          , "pixel"      ),
        ("CCD noise floor"              ,   0          , "A/D count"  ),
        ("X Pixel Size"                 ,  22E-6       , "m"          ),
        ("Y Pixel Size"                 ,  22E-6       , "m"          ),
        ("X Pixel Angular Pitch (nom)"  ,   0.84,      , "arcsec"     ),
        ("Y Pixel Angular Pitch (nom)"  ,   0.84,      , "arcsec"     ),
        ("X Pixel Angular Pitch (act)"  ,   0.81,      , "arcsec"     ),
        ("Y Pixel Angular Pitch (act)"  ,   0.81,      , "arcsec"     ),
        ("Spot Axis Rotation (vs CCD-x)",   0          , "radian"     ),
        ("Spot separation (nominal)"    ,  32          , "pixel"      ),
        ("Wavelength (FWHM)"            ,   0.5e-6     , "m"          )
      )
    END

    ATTRIBUTE Table asmdimCONFIG_dimm_mode_setup(asmdimCONFIG_mode_setup_N,
```

```
        bytes32 "parameter", double asmdimCONFIG_value, bytes16 "unit"  )
    BEGIN
      Value (
        ("Exposure time"                ,   0.01        , "s"           ),
        ("Exposures per frame"          ,   1           , "#"           ),
        ("Frames per sequence"          , 400           , "#"           ),
        ("X CCD sub-array size"         ,  20           , "pixel"       ),
        ("Y CCD sub-array size"         ,  20           , "pixel"       ),
        ("X Autoguiding target"         , 193           , "pixel"       ),
        ("Y Autoguiding target"         , 145           , "pixel"       )
      )
    END

    ATTRIBUTE Table asmdimCONFIG_coherencemode_setup(asmdimCONFIG_mode_setup_N,
        bytes32 "parameter", double asmdimCONFIG_value, bytes16 "unit"  )
    BEGIN
      Value (
        ("Exposure time"                ,   0.001       , "s"           ),
        ("Exposures per frame"          ,  19           , "#"           ),
        ("Frames per sequence"          , 400           , "#"           ),
        ("X CCD sub-array size"         ,  20           , "pixel"       ),
        ("Y CCD sub-array size"         ,  20           , "pixel"       ),
        ("X Autoguiding target"         , 200           , "pixel"       ),
        ("Y Autoguiding target"         , 145           , "pixel"       )
      )
    END

    ATTRIBUTE Table     asmdimCONFIG_validation_limits (asmdimCONFIG_validation_limits_N,
        bytes32 "parameter", double asmdimCONFIG_value, bytes16 "unit"  )
    BEGIN
      Value (
        ("Saturation Limit"             ,   0.9         , "A/D count"   ),
        ("S/N Ratio Limit"              ,   5           , ""            ),
        ("Spot Elongation Limit"        ,  10           , "%"           ),
        ("Spot Decentering Limit"       ,   2           , "Spot fwhm"   ),
        ("Gaussian Clipping Limit"      ,   3           , "rms of distr."),
        ("Clipped Exposures Limit"      ,   3           , "%"           ),
        ("Instrumental Noise X"         ,   0.01        , "pixel^2"     ),
        ("Instrumental Noise Y"         ,   0.01        , "pixel^2"     ),
        ("Statistical Error Limit"      ,   1           , "1/sqrt(N-1)" )
      )
    END

END

#endif /*!ASMDIM_CONFIG_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimCONFIG.class  (97/6/21  4:54)
```

23.2.6 asmdimDATA(4)

NAME

```
asmdimDATA - ASM DIMM Provided Public Data
```

SYNOPSIS

```
#include "asmdimDATA.h"

asmdimDATA(const dbSYMADDRESS dataPoint)
~asmdimDATA()

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Update(const char *tag, double value)
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This class handles the DIMM public data in the on-line database
and the log-file.
```

PUBLIC METHODS

```
Init - initialize, open LogFile and LastFile
Update - update specified tag in database and possibly files
```

PUBLIC DATA MEMBERS

```
fndSTRING LogFile  - name of the log-file to be used, "" = none.
fndSTRING LastFile - name of the last-file to be used, "" = none.

Both names must be set before calling Init().
Only the seeing value is written to the log-file (appended) and
to the last-file (overwritten). Both files are provided only for
compatibility with the La-Silla-DIMM.
```

FILES

```
Specified log-file is opened in append mode.
Specified last-file is opened in write mode.
They are created if not yet existing.
```

SEE ALSO

```
asmiCLASS(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.6.1 asmdimDATA.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimDATA.class  (97/6/21  4:54)
#ifndef ASMDIM_DATA_CLASS
#define ASMDIM_DATA_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimDATA.class,v 1.38 1997/06/20 15:24:47 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 02.12.1996 created
//

// headers:
#include "asmdimDefines.h"
#include "asmdimDATA.h"

// referenced classes:
#include "asmi.class"
#include "asmdimDATAHISTORY.class"


// ---------------------------------------------------------------------------
// PROVIDED DATA
//
CLASS "asmiDATA"        "asmdimDATA"
BEGIN
    ATTRIBUTE Table     asmdimDATA_data (asmdimDATA_data_N)
    BEGIN
      Value (
        // Values of seeing-monitor unit:
        //         tag            loc    description                unit
        (asmdimTAG_Seeing       ,"DIMM","Seeing"                  ,"arcsec"),
        (asmdimTAG_Wavelength   ,"DIMM","Wavelength"              ,"m"),
        (asmdimTAG_Airmass      ,"DIMM","Airmass at line-of-sight" ,""),
        (asmdimTAG_TimespanDimm ,"DIMM","Time span seeing estimate","s"),

        // Values of sky-monitor unit:
        //         tag            loc    description              unit
        (asmdimTAG_Scintillation,"Sky" ,"Scintillation Index"   ,"%"),
        (asmdimTAG_Extinction   ,"Sky" ,"Extinction"            ,"Magn/airm"),
        (asmdimTAG_ExtinctionRMS,"Sky" ,"rms Extinction"        ,"Magn/airm"),
        (asmdimTAG_PosAlpha     ,"Sky" ,"RA  of line-of-sight"  ,"HHMMSS.TTT"),
        (asmdimTAG_PosDelta     ,"Sky" ,"Dec of line-of-sight"  ,"DDMMSS.TTT"),
        (asmdimTAG_TimespanSky  ,"Sky" ,"Time span sky-monitor" ,"s")
        )
    END

    ATTRIBUTE "asmdimDATAHISTORY"       asmdimDATA_History
    BEGIN
        Alias           "DimmHistory"
        Class           65411 // <history point>
        Residence       Disk
        Categories      11111111111111111111111111111111
        CE Indicator    Disabled
        CE Order        Natural
    END

END

#endif /*!ASMDIM_DATA_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimDATA.class  (97/6/21  4:54)
```

23.2.6.2 asmdimDATAHISTORY.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimDATAHISTORY.class  (97/6/21  4:54)
#ifndef ASMDIM_DATAHISTORY_CLASS
#define ASMDIM_DATAHISTORY_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimDATAHISTORY.class,v 1.38 1997/06/20 15:24:48 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 07.03.1997 created
//

#ifndef MAKE_VXWORKS // only on WS-side!

// headers:
#include "CCS.class"


CLASS "<history point>"  "asmdimDATAHISTORY"
BEGIN

  ATTRIBUTE Table "history config"     (1,          // table size (CONSTANT)

                                        ..., // CONSTANT PART from base-class

                                        // VARIABLE PART:
                                        rtDB_XREF       "item 1",
                                        rtDB_XREF       "item 2",
                                        rtDB_XREF       "item 3",
                                        rtDB_XREF       "item 4",
                                        rtDB_XREF       "item 5",
                                        rtDB_XREF       "item 6",
                                        rtDB_XREF       "item 7",
                                        rtDB_XREF       "item 8",
                                        rtDB_XREF       "item 9",
                                        rtDB_XREF       "item 10"
                                        )
  BEGIN
        Value ((
                // CONSTANT PART:
                0,0,0,0,0,0,0,0,0,0,[],0,0,0,0,
                // VARIABLE PART:
                [^.data(0,value)],      //
                [^.data(1,value)],      //
                [^.data(2,value)],      //
                [^.data(3,value)],      //
                [^.data(4,value)],      //
                [^.data(5,value)],      //
                [^.data(6,value)],      //
                [^.data(7,value)],      //
                [^.data(8,value)],      //
                [^.data(9,value)]       //
                ))
  END

  ATTRIBUTE Table "setups"             (2)          // table size (CONSTANT)
  BEGIN
   Value ((1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, [], 2, 0, 2, 0, 0, 0, [], 0, 1, 0),
          (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, [], 0, 0, 0, 0, 0, 0, [], 0, 0, 0))
  END

  ATTRIBUTE Table "history"            (500,     // max. number of records

                                        ..., // CONSTANT PART from base-class

                                        // VARIABLE PART:
```

```
                                        // to be extended with:
                                        uint8   "quality 1",
                                        double  "Seeing",
                                        uint8   "quality 2",
                                        double  "Wavelength",
                                        uint8   "quality 3",
                                        double  "Airmass",
                                        uint8   "quality 4",
                                        double  "TimespanDimm",
                                        uint8   "quality 5",
                                        double  "Scintillation",
                                        uint8   "quality 6",
                                        double  "Extinction",
                                        uint8   "quality 7",
                                        double  "ExtinctionRMS",
                                        uint8   "quality 8",
                                        double  "PosAlpha",
                                        uint8   "quality 9",
                                        double  "PosDelta",
                                        uint8   "quality 10",
                                        double  "TimespanSky"
                                        )

  ATTRIBUTE Table "statistics"         (10)     // number of recorded items

END

#endif // !MAKE_VXWORKS

#endif /*!ASMDIM_DATAHISTORY_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimDATAHISTORY.class  (97/6/21  4:54)
```

23.2.7 asmdimDEVICE(4)

NAME

```
asmdimDEVICE - ASM DIMM Device
```

SYNOPSIS

```
asmdimDEVICE(const dbSYMADDRESS devicePoint,
             const dbSYMADDRESS dataPoint)
~asmdimDEVICE()

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Find(double &xArcsec, double &yArcsec)
ccsCOMPL_STAT Cycle(double alpha, double delta, double magnitude,
                    double gamma,
                    evhOBJ_CALLBACK endCB)

double Xarcsec, Yarcsec;
```

PARENT CLASS

```
asmiDEVICE
```

DESCRIPTION

```
This class coordinates data and sequences related to the DIMM.

It uses the following other classes:

asmdimAG       - for autoguiding functionality
asmdimCAMERA   - for all accesses to the CCD camera
asmdimCONFIG   - for all configuration parameters
asmdimDATA     - for public data output
asmdimSEQ      - for measurement sequence handling
```

PUBLIC METHODS

```
Init - read database and initialize used obejcts
Find - find star in field and return error-vector
Cycle - execute a single DIMM measurement cycle
```

PUBLIC DATA MEMBERS

```
Xarcsec, Yarcsec - error-vector for autoguiding resulting from Cycle
```

SEE ALSO

```
asmiDEVICE(4),
asmdimAG(4), asmdimCAMERA(4), asmdimCONFIG(4),
asmdimDATA(4), asmdimSEQ(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.7.1 asmdimDEVICE.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimDEVICE.class  (97/6/21  4:54)
#ifndef ASMDIM_DEVICE_CLASS
#define ASMDIM_DEVICE_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimDEVICE.class,v 1.38 1997/06/20 15:24:51 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 02.12.1996 created
//

// headers:
#include "asmdimDEVICE.h"

// referenced classes:
#include "asmi.class"
#include "asmdimCONFIG.class"
#include "asmdimCAMERA.class"
#ifndef MAKE_VXWORKS
#include "asmdimAG.class"
#include "asmdimSEQ.class"
#else
#include "asmdimEXPOS.class"
#endif

// ---------------------------------------------------------------------------
// DEVICE BRANCH
//
CLASS           "asmiDEVICE"    "asmdimDEVICE"
BEGIN
    ATTRIBUTE   "asmdimCONFIG"  "Configuration"
    BEGIN
        Alias   asmdimDEVICE_ALIAS_CONFIG
        Residence Disk
    END

#   ifndef MAKE_VXWORKS
    // Autoguider exists only on WS-side!
    ATTRIBUTE   "asmdimAG"      "Autoguider"
    BEGIN
        Alias   asmdimDEVICE_ALIAS_AG
        Residence RAM
    END

    // Sequence exists only on WS-side!
    ATTRIBUTE   "asmdimSEQ"     "Sequence"
    BEGIN
        Alias   asmdimDEVICE_ALIAS_SEQ
        Residence RAM
    END
#   else
    // Exposures exists only on LCU-side!
    ATTRIBUTE   "asmdimEXPOS"   "Exposures"
    BEGIN
        Alias   asmdimDEVICE_ALIAS_EXPOS
        Residence RAM
    END
#   endif

    ATTRIBUTE   "asmdimCAMERA"  "Camera"
    BEGIN
        Alias   asmdimDEVICE_ALIAS_CAMERA
        Residence RAM
    END
```

```
END

#endif /*!ASMDIM_DEVICE_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimDEVICE.class  (97/6/21  4:54)
```

23.2.8 asmdimEXPOS(4)

NAME

```
asmdimEXPOS - ASM DIMM Exposure Handling
```

SYNOPSIS

```
#include "asmdimEXPOS.h"

asmdimEXPOS(const dbSYMADDRESS aliasLCU,
            asmdimCAMERA       *camera,
            const asmdimCONFIG *config);
~asmdimEXPOS();

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Start(asmdimEXPOS_EndCB_t ExposEndCB)
ccsCOMPL_STAT Read(void)
ccsCOMPL_STAT Read(int index, asmdimEXPOS_parameters_t &p)

double x1[asmdimEXPOS_MAX], y1[asmdimEXPOS_MAX];
double x2[asmdimEXPOS_MAX], y2[asmdimEXPOS_MAX];
double xg[asmdimEXPOS_MAX], yg[asmdimEXPOS_MAX];
double A1[asmdimEXPOS_MAX], A2[asmdimEXPOS_MAX];
double n_1[asmdimEXPOS_MAX],n_2[asmdimEXPOS_MAX];
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This class handles the exposures to be taken for a DIMM cycle.
This includes the entire communication with the CCD LCU.
Upon completion of all exposures the image analysis data is
provided in data vectors.
```

PUBLIC METHODS

```
Init - read database
Start - start exposure sequence
Read - read result-vectors from LCU database
```

PUBLIC DATA MEMBERS

```
Mirrors of the database values in memory:

x1,y1,x2,y2: center of gravity x/y in window 1/2
A1,A2: flux in window 1/2
n_1,n_2: illuminated pixels in window 1/2
```

SEE ALSO

```
asmiCLASS(4), asmdimCAMERA(4), asmdimCONFIG(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.8.1 asmdimEXPOS.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimEXPOS.class  (97/6/21  4:54)
#ifndef ASMDIM_EXPOS_CLASS
#define ASMDIM_EXPOS_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimEXPOS.class,v 1.38 1997/06/20 15:24:50 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ------------------------------------------------
// S.Sandrock 02.12.1996 created
//

// headers:
#include "asmdimEXPOS.h"
#include "asmdimDefines.h"

// referenced classes:
#include "asmi.class"


// ---------------------------------------------------------------------------
// EXPOSURE PARAMETERS
//
// Exposure parameters are computed, validated, and stored for the whole
// sequence in the LCU DB. After completion of the sequence they are
// copied to the WS DB.
//
CLASS "asmiCLASS"       "asmdimEXPOS"
BEGIN
    // Status flag of frame validation
    // set by WS to enable frame evaluation, monitored by LCU after each expo
    ATTRIBUTE uint32    asmdimEXPOS_run         0

    // Status flag of frame validation
    // set by LCU in case of failed validation or completion, monitored by WS
    ATTRIBUTE uint32    asmdimEXPOS_status      0

    // Number of accepted exposures (only accepted expos are counted)
    // set to 0 by WS, incremented by LCU after each exposure
    ATTRIBUTE uint32    asmdimEXPOS_count       0

    // Max number of exposures in the sequence
    // set to N by WS, checked by LCU after each exposure
    ATTRIBUTE uint32    asmdimEXPOS_limit       0

    // Spot 1 center of gravity x/y [pixel]
    ATTRIBUTE Vector asmdimEXPOS_spot1_cdg_x (asmdimEXPOS_MAX, double)
    ATTRIBUTE Vector asmdimEXPOS_spot1_cdg_y (asmdimEXPOS_MAX, double)

    // Spot 2 center of gravity x/y [pixel]
    ATTRIBUTE Vector asmdimEXPOS_spot2_cdg_x (asmdimEXPOS_MAX, double)
    ATTRIBUTE Vector asmdimEXPOS_spot2_cdg_y (asmdimEXPOS_MAX, double)

    // Image center of gravity x/y (for subarray trk) [pixel]
    ATTRIBUTE Vector asmdimEXPOS_image_cdg_x (asmdimEXPOS_MAX, double)
    ATTRIBUTE Vector asmdimEXPOS_image_cdg_y (asmdimEXPOS_MAX, double)

    // Spot 1/2 flux [A/D count]
    ATTRIBUTE Vector asmdimEXPOS_spot1_flux  (asmdimEXPOS_MAX, double)
    ATTRIBUTE Vector asmdimEXPOS_spot2_flux  (asmdimEXPOS_MAX, double)

    // Spot 1/2 number of pixels illuminated
    ATTRIBUTE Vector asmdimEXPOS_spot1_illum (asmdimEXPOS_MAX, double)
    ATTRIBUTE Vector asmdimEXPOS_spot2_illum (asmdimEXPOS_MAX, double)

END
```

```
#endif /*!ASMDIM_EXPOS_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimEXPOS.class  (97/6/21  4:54)
```

23.2.9 asmdimSEQ(4)

NAME

```
asmdimSEQ - ASM DIMM Sequence Handling
```

SYNOPSIS

```
#include "asmdimSEQ.h"

asmdimSEQ(const dbSYMADDRESS point,
          const dbSYMADDRESS aliasExposData,
          asmdimCAMERA      *camera,
          asmdimCONFIG      *config);
~asmdimSEQ();

ccsCOMPL_STAT Init(void)
ccsCOMPL_STAT Start(asmdimSEQ_EndCB_t SeqEndCB)

asmdimSEQ_sequence_input_t      sequence_input;
asmdimSEQ_sequence_output_t     sequence_output;
```

PARENT CLASS

```
asmiCLASS
```

DESCRIPTION

```
This class coordinates the measurement sequence of the DIMM.

It uses the following other classes:

asmdimCAMERA    - for all accesses to the CCD camera
asmdimCONFIG    - for all configuration parameters
asmdimEXPOS     - for exposure storage and LCU communication
asmdimCALC      - for calculation and processing of exposure data
```

PUBLIC METHODS

```
Init - initialize object
Start - start measurement sequence
```

PUBLIC DATA MEMBERS

```
sequence_input  - input data to be set before calling Start()
sequence_output - ouput data provided after end of sequence
```

SEE ALSO

```
asmiCLASS(4),
asmdimCALC(4), asmdimCAMERA(4), asmdimCONFIG(4), asmdimEXPOS(4)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.2.9.1 asmdimSEQ.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimSEQ.class  (97/6/21  4:54)
#ifndef ASMDIM_SEQ_CLASS
#define ASMDIM_SEQ_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
//"@(#) $Id: asmdimSEQ.class,v 1.38 1997/06/20 15:24:49 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 02.12.1996 created
//

// headers:
#include "asmdimSEQ.h"

// referenced classes:
#include "asmi.class"
#include "asmdimEXPOS.class"
#include "asmdimSEQHISTORY.class"

// ---------------------------------------------------------------------------
// SEQUENCE PARAMETERS
//
CLASS "asmiCLASS"       "asmdimSEQ"
BEGIN
    ATTRIBUTE Table     asmdimSEQ_sequence_status (asmdimSEQ_sequence_status_N,
        bytes32 "explanation", uint32 asmdimSEQ_value)
    BEGIN
      Value (
        ("Elongation too high"          , asmdimSeqStatusFWHM   ),
        ("Spot not centered"            , asmdimSeqStatusCENTER ),
        ("Poor signal-noise ratio"      , asmdimSeqStatusSN     ),
        ("Undefined internal error"     , asmdimSeqStatusError  )
      )
    END

    ATTRIBUTE Table     asmdimSEQ_sequence_input (asmdimSEQ_sequence_input_N,
        bytes32 "parameter", double asmdimSEQ_value, bytes16 "unit"     )
    BEGIN
      Value (
        // updated from selected star:
        ("Star alpha coordinate"        ,   0           , "radian"      ),
        ("Star delta coordinate"        ,   0           , "radian"      ),
        ("Star zenith distance"         ,   0           , "radian"      ),
        ("Star magnitude"               ,   0           , "V band"      ),
        ("Airmass"                      ,   0           , ""            ),
        ("Sky background"               ,   0           , "A/D count"   )
      )
    END

    ATTRIBUTE Table     asmdimSEQ_sequence_output (asmdimSEQ_sequence_output_N,
        bytes32 "parameter", double asmdimSEQ_value, bytes16 "unit"     )
    BEGIN
      Value (
        ("Number validated"             ,   0           , "exposures"   ),
        ("Image cdgX (for autoguiding)" ,   0           , "pixel"       ),
        ("Image cdgY (for autoguiding)" ,   0           , "pixel"       ),
        ("Spot Separation (measured)"   ,   0           , "pixel"       ),
        ("Spot Axis Rotation (vs CCD-x)",   0           , "radian"      ),
        ("Spot1 Flux"                   ,   0           , "A/D count"   ),
        ("Spot2 Flux"                   ,   0           , "A/D count"   ),
        ("Spot1 Scintillation Index"    ,   0           , "%"           ),
        ("Spot2 Scintillation Index"    ,   0           , "%"           ),
        ("Scintillation Index avg."     ,   0           , "%"           ),
        ("Variance of diff motion para" ,   0           , "radian^2"    ),
        ("Variance of diff motion perp" ,   0           , "radian^2"    ),
```

```
        ("Seeing parallel"              ,   0           , "arcsec"      ),
        ("Seeing perpendicular"         ,   0           , "arcsec"      ),
        ("Seeing avg."                  ,   0           , "arcsec"      )
      )
    END

    ATTRIBUTE   "asmdimSEQHISTORY"      asmdimSEQ_History
    BEGIN
        Alias           "DimmSequenceHistory"
        Class           65411 // <history point>
        Residence       Disk
        Categories      11111111111111111111111111111111
        CE Indicator    Disabled
        CE Order        Natural
    END

END

#endif /*!ASMDIM_SEQ_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimSEQ.class  (97/6/21  4:54)
```

23.2.9.2 asmdimSEQHISTORY.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmdimSEQHISTORY.class  (97/6/21  4:54)
#ifndef ASMDIM_SEQHISTORY_CLASS
#define ASMDIM_SEQHISTORY_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmdimSEQHISTORY.class,v 1.38 1997/06/20 15:24:49 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ------------------------------------------------
// S.Sandrock 07.03.1997 created
//

#ifndef MAKE_VXWORKS // only on WS-side!

// referenced classes:
#include "CCS.class"

CLASS "<history point>"  "asmdimSEQHISTORY"
BEGIN

  ATTRIBUTE Table "history config"     (1,         // table size (CONSTANT)

                                        ..., // CONSTANT PART from base-class

                                        // VARIABLE PART:
                                        rtDB_XREF       "item 1",
                                        rtDB_XREF       "item 2",
                                        rtDB_XREF       "item 3",
                                        rtDB_XREF       "item 4",
                                        rtDB_XREF       "item 5",
                                        rtDB_XREF       "item 6",
                                        rtDB_XREF       "item 7",
                                        rtDB_XREF       "item 8",
                                        rtDB_XREF       "item 9",
                                        rtDB_XREF       "item 10",
                                        rtDB_XREF       "item 11",
                                        rtDB_XREF       "item 12",
                                        rtDB_XREF       "item 13",
                                        rtDB_XREF       "item 14",
                                        rtDB_XREF       "item 15",
                                        rtDB_XREF       "item 16",
                                        rtDB_XREF       "item 17",
                                        rtDB_XREF       "item 18"
                                        )
  BEGIN
        Value ((
                // CONSTANT PART:
                0,0,0,0,0,0,0,0,0,0,[],0,0,0,0,
                [^.sequence_input(0,value)],    //
                [^.sequence_input(1,value)],    //
                [^.sequence_input(2,value)],    //
                [^.sequence_input(3,value)],    //
                [^.sequence_input(4,value)],    //
                [^.sequence_input(5,value)],    //
                // VARIABLE PART:

                [^.sequence_output(0,value)],   //
                [^.sequence_output(1,value)],   //
                [^.sequence_output(2,value)],   //
                [^.sequence_output(3,value)],   //
                [^.sequence_output(4,value)],   //
                [^.sequence_output(5,value)],   //
                [^.sequence_output(6,value)],   //
                [^.sequence_output(7,value)],   //
                [^.sequence_output(8,value)],   //
                [^.sequence_output(9,value)],   //
```

```
                [^.sequence_output(10,value)],  //
                [^.sequence_output(11,value)],  //
                [^.sequence_output(12,value)],  //
                [^.sequence_output(13,value)],  //
                [^.sequence_output(14,value)]   //
                ))
  END

  ATTRIBUTE Table "setups"             (2)         // table size (CONSTANT)
  BEGIN
   Value ((1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, [], 2, 0, 2, 0, 0, 0, [], 0, 1, 0),
          (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, [], 0, 0, 0, 0, 0, 0, [], 0, 0, 0))
  END

  ATTRIBUTE Table "history"            (500,    // max. number of records

                                        ..., // CONSTANT PART from base-class

                                        // VARIABLE PART:
                                        // to be extended with:
                                        uint8   "quality 1",
                                        double  "StarAlpha",
                                        uint8   "quality 2",
                                        double  "StarDelta",
                                        uint8   "quality 3",
                                        double  "StarGamma",
                                        uint8   "quality 4",
                                        double  "StarMagnitude",
                                        uint8   "quality 5",
                                        double  "StarAirmass",
                                        uint8   "quality 6",
                                        double  "SkyBackground",

                                        uint8   "quality 7",
                                        double  "ValidExposures",
                                        uint8   "quality 8",
                                        double  "cdgX",
                                        uint8   "quality 9",
                                        double  "cdgY",
                                        uint8   "quality 10",
                                        double  "SpotSeparation",
                                        uint8   "quality 11",
                                        double  "SpotAxisRotation",
                                        uint8   "quality 12",
                                        double  "Spot1Flux",
                                        uint8   "quality 13",
                                        double  "Spot2Flux",
                                        uint8   "quality 14",
                                        double  "Spot1Scintillation",
                                        uint8   "quality 15",
                                        double  "Spot2Scintillation",
                                        uint8   "quality 16",
                                        double  "Scintillation",
                                        uint8   "quality 17",
                                        double  "VariancePara",
                                        uint8   "quality 18",
                                        double  "VariancePerp",
                                        uint8   "quality 19",
                                        double  "SeeingParallel",
                                        uint8   "quality 20",
                                        double  "SeeingPerpendicular",
                                        uint8   "quality 21",
                                        double  "SeeingAverage"
                                        )

  ATTRIBUTE Table "statistics"         (21)     // number of recorded items

END
```

```
#endif // !MAKE_VXWORKS

#endif /*!ASMDIM_SEQHISTORY_CLASS*/
--->END IMPORTED FILE: ../dbl/asmdimSEQHISTORY.class  (97/6/21  4:54)
```

23.2.10 asmiCMD_RECEIVE(4)

NAME

```
asmiCMD_RECEIVE - class for handling user commands
```

SYNOPSIS

```
#include <asmiCMD_RECEIVE.h>
asmiCMD_RECEIVE cmdandler(dbroot, cmdName, callback)
```

PARENT CLASS

```
asmSTATE
```

DESCRIPTION

```
Class to handle user commands in an evhHANDLER application. It provides
the following features:
  - command call-back installation and de-installation
  - checking whether execution of command is allowed
  - state switching between IDLE and BUSY
  - storing the user message
  - sending replies (normal or error) to the originator
  - internal usage of the command function

To use the class for a specific command you can either overload the
Execute() method or provide an external callback for that purpose.

The constructor installs a generic call-back for the given command which
performs execution checking and calls the command executor function if
command execution is allowed. The destructor de-installs that call-back.

The checking for execution permit uses the following algorithm:
  - If the state is not an ONLINE sub-state, the command is rejected.
  - If the object is still active, i.e. the previous same command has not
    yet sent a last reply (or error reply), the new command is rejected.
  - If any other command is still active, the command is accepted.

If the command is accepted, the state is set to BUSY if not yet there.
On command completion the state is set to IDLE, if no other command is
still active. However, this is done only while in an ONLINE sub-state.

The object keeps track whether it is active, i.e. whether the execution of
a command is in progress. The according flag is set when a command is
received and unset when an error reply or last reply is sent.
THEREFOR IT IS IMPORTANT ALWAYS TO USE THE SendReply() METHOD OF THIS CLASS
FOR SENDING REPLIES, OTHERWISE THE OBJECT WOULDN'T KNOW THAT THE COMMAND
IS COMPLETED!

The class is also designed to allow usage of the command function not only
by the evhHandler but from inside the application (high-level commands use
the functionality of low-level commands). To do so, the application sends
the low-level command to itself.

When called from internally the SendReply() method hence sends replies to
the own process. Thus, using own commands internally is just like sending
a command to a sub-process.
```

PUBLIC METHODS

```
asmiCMD_RECEIVE(const dbSYMADDRESS dRoot, const msgCMD cmdName,
                 const evhCALLBACK *cb = NULL);
    The constructor installs the generic command call-back. It takes
    the command name and the DB support point of the process as arguments.
    If the optional callback cb is given, it is replaces the built-in
    command execution function.

virtual ~asmiCMD_RECEIVE();
```

```
    De-installs the command call-back and cleans everything.

virtual Init();
    Initializes the object.

virtual Reset();
    Aborts the command execution, if active and resets the object to
    inactive state.

virtual Cancel();
    If the command is active sends back a last reply with the error
    CMD_STOPPED. For the time being this function is only used to
    give priority to the STOP command. That is to say, if the STOP
    command is received by the control process then the STOP callback
    will cancel all pending commands received by the control process.

const char *CmdName();
    Returns the associated command name.

vltLOGICAL IsActive();
    Returns ccsTRUE if a command execution is currently in progress.

evhCB_COMPL_STAT SendReply(const char *buffer, msgLENGTH size = 0,
                           vltLOGICAL last = ccsTRUE);
    Sends a normal reply to the originator and sets the command state to
    "not active" if flag <last> is set.

evhCB_COMPL_STAT SendReply(const ccsERROR &newErrmsg);
    Send an error reply back to the originator and sets the command state
    to "not active".

msgMESSAGE &MsgCaller();
    Returns the original user command.

const char *MsgCallerName();
    Returns the name of the command originator process as @env:process.

vltLOGICAL InternalCall();
    Returns ccsTRUE if currently active due to an internal invokation.
```

PROTECTED METHODS

```
evhCB_COMPL_STAT CommandCB(msgMESSAGE &msg, void *udata);
    This is the generic callback which is installed at the event handler.
    It first checks if execution of the command is allowed in the current
    state and, if so, calls the overloaded execution function.

virtual ccsCOMPL_STAT Execute(msgMESSAGE &msg, void *udata) = 0;
    Executes the connected command function. Gets the original message and
    udata from the CommandCB.

vltLOGICAL ExecutionPermitted(msgMESSAGE &msg);
    Check if the new command can be executed in the current state and if it
    conflicts with a pending command.

void SetActive(vltLOGICAL flag=TRUE);
    Sets the object command state to "active" or "not active" as given in
    flag. Used internally when a command is received and accepted. When
    the state was IDLE it is set to BUSY on a command becoming active and
    vice versa.

void MsgCaller(msgMESSAGE &msg);
    The given message is stored as the original user message.
```

PRIVATE DATA MEMBERS

```
msgCMD       command;         // name of command handled by this instance
msgMESSAGE   orgMsg;          // originator command message
evhCALLBACK *userCB;          // user's command callback
```

```
vltLOGICAL   cmdActive;       // flag if a command is active
static vltINT32 activeCount;  // number of commands active in parallel
```

FILES

None

ENVIRONMENT

None

COMMANDS

An object of this class can be connected to any command.

SEE ALSO

evhHANDLER, asmSTATE

\- - - - - -

Last change: 22/06/97-18:20

23.2.11 asmiCMD_SEND(4)

NAME

```
asmiCMD_SEND - tbd
```

```
- - - - - -
Last change:  22/06/97-18:20
```

23.2.12 asmiDEVICE(4)

NAME

```
       asmiDEVICE - tbd



- - - - - -
Last change:  22/06/97-18:20
```

23.2.13 asmiEVENT(4)

NAME

```
       asmiEVENT - tbd




- - - - - -
Last change:  25/06/97-11:45
```

23.2.14 asmiMAIN(4)

NAME

```
       asmiMAIN - tbd
```

```
- - - - - -
Last change:  22/06/97-18:20
```

23.2.15 asmiSTATE(4)

NAME

```
asmiSTATE - process state handling
```

SYNOPSIS

```
#include <asmiSTATE.h>
asmiSTATE asmiSTATE(dbPoint);
```

PARENT CLASS

```
evhSIMPLE_TASK (to get fndOBJECT and eccsERROR_CLASS inheritance)
```

DESCRIPTION

```
This class provides reading and writing of the process state on database,
as well as value to name translation.
```

PUBLIC METHODS

```
asmiSTATE(dbSYMADDRESS dbPoint);
  The constructor creates an eccsDB_INT32 object to access the database
  attribute of the state, expected to be the <dbPoint>.state attribute.

vltINT32 State();
  Returns the current state.

ccsCOMPL_STAT State(vltINT32 newState);
  Sets the state.

const char *StateName();
  Returns the name of the current state.

vltLOGICAL IsOnline();
  Returns ccsTRUE if the current state is an ONLINE sub-state, i.e.
  IDLE or BUSY.

const char *DbRoot();
  Returns the module DB root point.
```

PRIVATE DATA MEMBERS

```
fndSTRING     dbRoot;        // module DB root point
eccsDB_INT32  state;         // state attribute
```

```
- - - - - -
Last change:  22/06/97-18:20
```

23.2.16 asmwsCON_MAIN_TASK(3)

NAME

```
asmwsCON_MAIN_TASK - enclosure coordination main object
```

SYNOPSIS

```
#include <asmwsCON_MAIN_TASK.h>
asmwsCON_MAIN_TASK asmwsConMain(dbSYMADDRESS point);
```

PARENT CLASS

```
asmwsSTATE
```

DESCRIPTION

```
This class forms the main object of the asmwsControl process.

It is also the main handler of the INIT command, all other objects have
not installed INIT command handlers, instead each of them provides an
Init() method. These Init() methods are called in proper sequence from
the asmwsCON_MAIN_TASK class which also sends the reply.
```

PUBLIC METHODS

```
asmwsCON_MAIN_TASK(dbSYMADDRESS point);
    The constructor takes the name of the asmwsControl process DB root
    point as argument.

~asmwsCON_MAIN_TASK();
    The destructor deletes all dynamically created objects.

ccsCOMPL_STAT Recover();
    This method is called from the event handler whenever a callback
    returns the FAILURE bit set, meaning fatal error. It tries to recover
    by re-initializing the process, calling the Init() procedure. If this
    does not succeed the function exits with FAILURE, causing the event
    handler to leave the event loop with error state and the process
    terminates abnormally.
```

PROTECTED METHODS

```
ccsCOMPL_STAT Init();
    This is the excecution method of the INIT command, being provided as
    a hook to the INIT command handler. It calls the Init() methods of all
    managed objects in a proper sequence.
```

COMMANDS

```
INIT    - read configuration from database, bring process to initial state
ONLINE  -
STANDBY -
EXIT    - terminate the process
START   - starts the automatic operation of the ASM
```

OFF

```
STOP    - Stop any hardware movement. The LCCSTOP command is used for
          this pourpose. This command is sent to all the LCU environments.
          If the command comes from the outside world then old pending
          commands are canceled. This means that this command has
          priority over the other, and if it sent then it will realy
          stop any hardware activity. If the command was internaly
          sent then old pending commands will not be canceled.
          The state of the asmwsControl process will switch depending on
          it's actual state.
GETSID  - get an string with the actual local sideral time in radians.
          The asmcat::CalculateLST function is called.
ASMTEST - get the name of a functionality to test and the required
```

```
                 arguments to do so.


- - - - - -
Last change: ASM-XXX 21/06/97-04:58
```

23.2.17 asmwsCON_STD_COMMANDS(3)

NAME

```
asmwsCON_STD_COMMANDS - standard command handling of asmwsControl process
```

SYNOPSIS

```
#include <asmwsCON_STD_COMMANDS.h>
asmwsCON_STD_COMMANDS asmwsConStdCmds(dbPoint);
```

PARENT CLASS

```
evhSTD_COMMANDS
```

DESCRIPTION

```
This class overloads the evh standard command handling where necessary
to implement asm specific behaviour. In detail this includes:
- STATUS to return process specific status data

Further the commands INIT, EXIT, STOP, ONLINE, STANDBY and OFF are
disabled. They are handled by the asmwsCON_MAIN_TASK class since they
affect the currently on-going action.
```

PUBLIC METHODS

```
asmwsCON_STD_COMMANDS(const char *);
evhCB_COMPL_STAT StatusCB(msgMESSAGE &msg, void *udata);
    The STATUS command provides also the status of various sum-modules.
```

COMMANDS

```
All standard commands, except INIT, STOP, EXIT, ONLINE, STANDBY, OFF
```

SEE ALSO

```
evhSTD_COMMANDS
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:58
```

23.3 DATABASE CLASSES

23.3.1 asmiBASE.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmiBASE.class  (97/6/21  4:52)
#ifndef ASMI_BASE_CLASS
#define ASMI_BASE_CLASS
//****************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmiBASE.class,v 1.22 1997/06/20 23:30:07 vltsccm Exp $"
//*
//* who        when       what
//* ---------- ---------- -------------------------------------------------
//* S.Sandrock 12.09.1996 created
//*/

#include "evhSTD_COMMANDS.class"

#include "asmiDefines.h"

//----------------------------------------------------------------------------
// Generic base-class
// Use this always instead of BASE_CLASS!
//
CLASS "BASE_CLASS"      "asmiCLASS"
BEGIN
END

//----------------------------------------------------------------------------
// Generic class for a software-device
//
CLASS "evhSTD_COMMANDS" "asmiTASK"
BEGIN
    ATTRIBUTE rtINT32   "state"
    ATTRIBUTE rtINT32   "substate"
END

CLASS "asmiCLASS"       "asmiDEVICE"
BEGIN
    ATTRIBUTE rtINT32   "state"
    ATTRIBUTE rtINT32   "substate"
END

CLASS "asmiCLASS"       "asmiBRANCH"
BEGIN
END

#endif /*!ASMI_BASE_CLASS*/
--->END IMPORTED FILE: ../dbl/asmiBASE.class  (97/6/21  4:52)
```

23.3.2 asmiCOMWS.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmiCOMWS.class  (97/6/21  4:52)
#ifndef ASMI_COMWS_CLASS
#define ASMI_COMWS_CLASS
//***************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmiCOMWS.class,v 1.22 1997/06/20 23:30:06 vltsccm Exp $"
//*
//* who        when       what
//* ---------- ---------- -------------------------------------------------
//* S.Sandrock 12.09.1996 created
//*/

#include "asmiDefines.h"
#include "asmiBASE.class"

#include "alrmALARM.class"

#define asmiCOM_ALARM_POINT_NAME        "Alarm"
#define asmiCOM_ALARM_SCOPE             LOCAL
#define asmiCOM_ALARM_TIMEOUT           60

//---------------------------------------------------------------------------
// Generic class for a serial communication line
//
CLASS "asmiCLASS"       "asmiCOMWS"
BEGIN
  // when "control" is OFF then the running task is halted (but not deleted)
  // and continues as soon is it goes to ON again:
  ATTRIBUTE uint32      "control"       asmiCOM_CONTROL_OFF

  // "status" is modified by the running task corresponding to "control",
  // or when an error occurs according to the alarm values below.
  // The error indication is self-clearing before each loop begins,
  // so that it allows to use ACK-alarms:
  ATTRIBUTE uint32      "status"        asmiCOM_STATUS_OFF

  alrmDiscrete5st(asmiCOM_ALARM_POINT_NAME,             \
                  status,                               \
                  WARNING,     asmiCOM_STATUS_OFF,      \
                  NORMAL,      asmiCOM_STATUS_ON,       \
                  WARNING_ACK, asmiCOM_STATUS_WARN,     \
                  SERIOUS_ACK, asmiCOM_STATUS_ERR,      \
                  FATAL_ACK,   asmiCOM_STATUS_DOWN,     \
                  asmiCOM_ALARM_SCOPE,                  \
                  asmiCOM_ALARM_TIMEOUT)

  // "counter" is set to 0 when the task begins running, is incremented
  // internally once per loop, but only written to the database when all
  // accesses have been processed successfully:
  ATTRIBUTE uint32      "counter"       0

END

#endif /*!ASMI_COMWS_CLASS*/
--->END IMPORTED FILE: ../dbl/asmiCOMWS.class  (97/6/21  4:52)
```

23.3.3 asmiDATA.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmiDATA.class  (97/6/21  4:52)
#ifndef ASMI_DATA_CLASS
#define ASMI_DATA_CLASS
//*************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmiDATA.class,v 1.22 1997/06/20 23:30:06 vltsccm Exp $"
//*
//* who        when       what
//* ---------- ---------- -------------------------------------------------
//* S.Sandrock 12.09.1996 created
//*/

#include "asmiDefines.h"
#include "asmiBASE.class"


//-------------------------------------------------------------------------
// Generic class for a provided-data point
//
// Note: The data shall be accessible by the user eg. under "Meteo.data(...)",
//
CLASS "asmiCLASS" "asmiDATA"
BEGIN
    ATTRIBUTE Table      asmiATTR_data (1,       // size to be overloaded
                // CONSTANT PART:
                bytes20  asmiFIELD_tag,
                bytes8   asmiFIELD_loc,
                bytes32  asmiFIELD_description,
                bytes16  asmiFIELD_unit,
                // INSTANT DATA:
                double   asmiFIELD_value,
                abs_time asmiFIELD_timestamp,
                // SHORT-TERM STATISTICS:
                double   asmiFIELD_timespan_stat,
                double   asmiFIELD_average,
                double   asmiFIELD_minimum,
                double   asmiFIELD_maximum,
                double   asmiFIELD_deviation)
    BEGIN
        // standard access protection for the data-table:
        WriteGroups  0000
        ReadGroups   1111
    END

    ATTRIBUTE   uint32  "new_data"      0
END

#endif /*!ASMI_DATA_CLASS*/
--->END IMPORTED FILE: ../dbl/asmiDATA.class  (97/6/21  4:52)
```

23.3.4 asmiLCU.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmiLCU.class  (97/6/21  4:52)
#ifndef ASMI_LCU_CLASS
#define ASMI_LCU_CLASS
//*******************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmiLCU.class,v 1.22 1997/06/20 23:30:02 vltsccm Exp $"
//*
//* who        when       what
//* ---------- ---------- -------------------------------------------------
//* S.Sandrock 12.09.1996 created
//*/

#include "asmiDefines.h"
#include "tooCom.class"

//-----------------------------------------------------------------------------
// Basic classes for signals
//
CLASS "asmiCLASS"       "asmiSIGNAL"
BEGIN
    // Mandatory attributes with predefined names, for handling by LCC
    ATTRIBUTE rtLOGICAL "value_status"

    // Overloaded at instantiation time to provide on line help:
    ATTRIBUTE rtBYTES80 "description"
END

CLASS "asmiSIGNAL"      "asmiSIG_DIGITAL"
BEGIN
    ATTRIBUTE rtUINT32  "value"
END

CLASS "asmiSIGNAL"      "asmiSIG_ANALOG"
// Identical to the digital signal, except for the type of value.
BEGIN
    ATTRIBUTE rtFLOAT   "value"
END


//-----------------------------------------------------------------------------
// Generic class for a serial communication line
//
CLASS "tooCOM_PARAMS"   "asmiCOM"
BEGIN
  // when "control" is OFF then the running task is halted (but not deleted)
  // and continues as soon is it goes to ON again:
  ATTRIBUTE uint32      "control"       hbCOM_CONTROL_OFF

  // "status" is modified by the running task corresponding to "control",
  // or when an error occurs according to the alarm values below.
  // The error indication is self-clearing before each loop begins,
  // so that it allows to use ACK-alarms:
  ATTRIBUTE uint32      "status"        hbCOM_STATUS_OFF

  // "counter" is set to 0 when the task begins running, is incremented
  // internally once per loop, but only written to the database when all
  // accesses have been processed successfully:
  ATTRIBUTE uint32      "counter"       0

  ATTRIBUTE int32 period   6000 /* communication cycle period [ticks]   */

  // The communication parameters are read from the database when the
  // serial device is opened and the task is started:
  ATTRIBUTE int32 debug       0 /* record transceived data in file [0,1]*/
  ATTRIBUTE int32 baudrate 9600 /* baudrate to use      [1200..9600]    */
  ATTRIBUTE int32 charbits    8 /* bits per character   [5..8]          */
```

```
  ATTRIBUTE int32 stopbits    1 /* number of stop bits  [1..2]          */
  ATTRIBUTE int32 parity      0 /* parity type [1:even, 0:none, -1:odd] */
  ATTRIBUTE int32 handshake   3 /* handshake [0:DTR,1:Xon,2:RTS,3:none] */
  ATTRIBUTE int32 reclen      0 /* iser record length [0:max, 1..1023]  */
  ATTRIBUTE int32 rxmode      0 /* iser receive mode (see iser) [0..7]  */
  ATTRIBUTE int32 timeout0  100 /* iser 1st-char timeout [0,1..64K ms]  */
  ATTRIBUTE int32 timeout1   50 /* iser ch-to-ch timeout [0,1..64K ms]  */

END

//------------------------------------------------------------------------------

#endif /*!ASMI_LCU_CLASS*/
--->END IMPORTED FILE: ../dbl/asmiLCU.class  (97/6/21  4:52)
```

23.3.5 asmiSCANDEV.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmiSCANDEV.class  (97/6/21  4:52)
#ifndef asmiSCANDEV_H
#define asmiSCANDEV_H
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmiSCANDEV.class,v 1.22 1997/06/20 23:30:07 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ---------------------------------------------
// S.Sandrock 09.12.1996 created
//

#include "CCS.class"

#define asmiSCANDEV_MAX 100 // max. number of scanned attributes per LCU

//
// The scan-points inherits all attributes from the standard CCS,
// and overloads the table-sizes as appropriate for ASM
//
CLASS "<VLT scan dev>" "asmiSCANDEV"
BEGIN
#   ifndef MAKE_VXWORKS
    ATTRIBUTE Table "scan input"  (asmiSCANDEV_MAX) // table size overloaded
#   endif
    ATTRIBUTE Table "scan output" (asmiSCANDEV_MAX) // table size overloaded
END

#endif /*!asmiSCANDEV_H*/
--->END IMPORTED FILE: ../dbl/asmiSCANDEV.class  (97/6/21  4:52)
```

23.3.6 asmtel.class

```
--->BEGIN IMPORTED FILE: ../dbl/asmtel.class  (97/6/21  4:56)
#ifndef ASMTEL_CLASS
#define ASMTEL_CLASS
//*************************************************************************
//* E.S.O. - VLT project
//*
//*"@(#) $Id: asmtel.class,v 1.35 1997/06/20 15:34:42 vltsccm Exp $"
//*
//* who       when      what
//* --------  --------  ------------------------------------------------
//* ramestic  28/10/96  created, derived from nttaltaz.class
//*/

#define TARGET_TCS VLT_TCS

#include "axis.class"
#include "trk.class"
#include "asmiLCU.class"

#ifdef TEST_USE_IKENC
#include "ikenc.class" // normally ikenc is NOT used!
#endif

// ---------------------------------------------------------------------------
// ENCODER
//
CLASS axisSTD_ENCODER   "asmtelENCODER"
BEGIN
  // Parameters of initialization movement:
  ATTRIBUTE Vector      "initSpeed"     (3, rtDOUBLE)
  BEGIN
        //        +--------------- 0: high speed during flip-over
        //        |     +--------- 1: high speed when moving to init-switch
        //        |     |      +-- 2: low speed when moving to encoder-ref
        //        |     |      |
        Value (( 0.01,  0.01, 0.001))   // [rad/s]
  END
  ATTRIBUTE Vector      "initTimeout"   (3, rtUINT32)
  BEGIN
        //        +--------------- 0: flip-over timeout
        //        |    +---------- 1: move to limit-switch timeout
        //        |    |    +----- 2: move back to ref-point timeout
        //        |    |    |
        Value (( 720, 720,  30 ))       // [s]
  END

  // Physical parameters of encoder/drive-system:
  // Note: (gearRatio * lineCount) yields [enc-lines per revolution]
  ATTRIBUTE rtUINT32    "gearRatio"     520  // between telescope+encoder axis
  ATTRIBUTE rtUINT32    "lineCount"     1000 // ... of used encoder type

END


// ---------------------------------------------------------------------------
// AMPLIFIER
//
CLASS asmiCLASS asmtelAMPLIFIER
BEGIN
  ATTRIBUTE rtDOUBLE    "powerTimeOut"  20.0    // [s] timeout switch ON/OFF
  ATTRIBUTE rtDOUBLE    "powerOnWait"    0.5    // [s] wait after switch-ON
END


// ---------------------------------------------------------------------------
// VEL_CTLR
//
```

```
CLASS axisSTD_VEL_CTLR  "asmtelVEL_CTLR"
BEGIN
  // Configuration:
  ATTRIBUTE rtDOUBLE    "convFactor"            990.1    // [V / (rad/s)]
  ATTRIBUTE rtDOUBLE    "zero_velocity"         0.0      // [rad/s]
  ATTRIBUTE rtDOUBLE    "zero_velocity_diff"    0.000001 // [rad/s]
END

// ---------------------------------------------------------------------------
// INTERLOCKS
//
CLASS axisINTERLOCKS   "asmtelINTERLOCKS"
BEGIN
END

// ---------------------------------------------------------------------------
// POS_CTLR
//
CLASS axisPOS_CTLR      "asmtelPOS_CTLR"
BEGIN
  // Configuration:
  ATTRIBUTE rtDOUBLE       "Kp"                 3.0
  ATTRIBUTE rtDOUBLE       "Ki"                 1.0
  ATTRIBUTE rtDOUBLE       "Kb"                 0.13
  ATTRIBUTE Vector         "Alg_switch" (2,rtDOUBLE)
  BEGIN
                           Value                        ((0.003 , 0.00005))
//                         Value                        ((0.03 , 0.03))
  END
  ATTRIBUTE rtDOUBLE       "max_velocity"       0.009058        // [rad/s]
  ATTRIBUTE rtDOUBLE       "max_acceleration"   0.01
  ATTRIBUTE rtDOUBLE       "max_jerk"           10.0
# ifdef TEST_USE_IKENC
  ATTRIBUTE ikencENCODER   "ABSENCODER"
# else
  ATTRIBUTE asmtelENCODER  "ABSENCODER"
# endif
  ATTRIBUTE asmtelVEL_CTLR "VELLOOP"
END

// ---------------------------------------------------------------------------
// AXES COMMON DATA
//
CLASS asmiCLASS asmtelAXES
BEGIN
    // common axes data and signals:
    // Encoder device name
    ATTRIBUTE rtBYTES32 "encDevice"     "/ikon0"
    // half-sky status attribute
    ATTRIBUTE rtUINT32   "halfSky"    // ( 0, 1, 2): no, north, south
    // transfer matrix between altaz and eta-theta angles
    ATTRIBUTE Table transferMatrix(3,
                               rtDOUBLE "col1",
                               rtDOUBLE "col2",
                               rtDOUBLE "col3")
    BEGIN
        Value((1,  0,  0),
              (0,  0,  1),
              (0, -1,  0))
    END
    // north software limits for both axes
    ATTRIBUTE Vector    "northLims"     (4,rtDOUBLE)
    BEGIN
        //        +--------+---------------------- eta
        //        |        |      +------+--------- theta
        //        |        |      |      |
        Value (( 0.0 ,  100.0,  -85.0,  85.0))     // [deg]
    END
    // south software limits for both axes
```

```
    ATTRIBUTE Vector    "southLims"     (4,rtDOUBLE)
    BEGIN
        //        +-------+------------------------ eta
        //        |       |       +------+--------- theta
        //        |       |       |      |
        Value (( -100.0,  0.0 ,  95.0 , 265.0))     // [deg]
    END
    // north half-sky digital signal
    ATTRIBUTE asmiSIG_DIGITAL   "northSignal"
    BEGIN
        ALIAS LSNORTHI
    END
    // south half-sky digital signal
    ATTRIBUTE asmiSIG_DIGITAL   "southSignal"
    BEGIN
        ALIAS LSSOUTHI
    END
END

// ---------------------------------------------------------------------------
// AXIS
//
CLASS axisAXIS asmtelAXIS
BEGIN
    // Configuration:
    ATTRIBUTE rtBYTES16 "axisName"              // value to be overloaded
    ATTRIBUTE rtLOGICAL "circularWrap"          0
    ATTRIBUTE rtUINT32  "servoLoopDelay"        5000    // [microsec]
    ATTRIBUTE rtDOUBLE  "trkRadius"             0.0005  // [rad]
    ATTRIBUTE rtINT32   "trkCount"              5
    ATTRIBUTE rtINT32   "maxExtrapolation"      10000000
    ATTRIBUTE Vector    "posLims" (4,rtDOUBLE)  // value to be overloaded
        BEGIN
        // this default values are useful ONLY for simulation
        Value (( 0.0, -10.0 , 10.0 , 0.0))     // [rad]
        END
    ATTRIBUTE rtBYTES32 "acmSendRefs"
    ATTRIBUTE rtBYTES32 "acmRemTrkTime"

    // Components:
    ATTRIBUTE asmtelPOS_CTLR    "POSLOOP"
    ATTRIBUTE asmtelAMPLIFIER   "AMPLIFIER"
    ATTRIBUTE asmtelINTERLOCKS  "INTERLOCKS"
END

// ---------------------------------------------------------------------------
// Tracking
//
CLASS asmiBRANCH "asmtelTRKVTA"
BEGIN
    // Configuration:
    ATTRIBUTE rtBYTES16 "axisName"              // value to be overloaded
    ATTRIBUTE rtLOGICAL "circularWrap"          0
    ATTRIBUTE rtUINT32  "servoLoopDelay"        2000
    ATTRIBUTE rtDOUBLE  "trkRadius"             0.0002
    ATTRIBUTE rtINT32   "trkCount"              5
    ATTRIBUTE rtINT32   "maxExtrapolation"      10000000
    ATTRIBUTE Vector    "posLims" (4,rtDOUBLE)  // value to be overloaded
    ATTRIBUTE rtBYTES32 "acmSendRefs"
    ATTRIBUTE rtBYTES32 "acmRemTrkTime"
END

//
// Tracking class
//
CLASS TRACKING_AXIS "asmtelTRK"
BEGIN
    ATTRIBUTE SETUP              "setup"
    BEGIN
```

```
        ATTRIBUTE bytes256 backupPcLCU "asmtrkPcData.dat"
    END
    ATTRIBUTE T_AXES             "axes"
    BEGIN
        ATTRIBUTE TRK_PRIVATE    "asm"
    END
    ATTRIBUTE VIR_TRACKING_AXIS  "vta"
    BEGIN
        ATTRIBUTE asmtelTRKVTA   "asm"
        BEGIN
          ATTRIBUTE rtBYTES32    "acmSendRefs"    "asmtelTrkSendRefs"
          ATTRIBUTE rtBYTES32    "acmRemTrkTime"  "azRemTrkTime" // dummy
          ATTRIBUTE axisPOS_CTLR "POSLOOP" // this is a dummy for trk!
        END
    END
END

// -----------------------------------------------------------------------------
// The "Telescope" DEVICE branch:
//
CLASS asmiDEVICE asmtelDEVICE
BEGIN
  ATTRIBUTE asmtelAXES "axesData"
  BEGIN
      ALIAS "axesData"
  END

  // tracking:
  ATTRIBUTE asmtelTRK  "Tracking"
  BEGIN
    ALIAS               "asmTracking"
  END

  // eta-axis:
  ATTRIBUTE asmtelAXIS                  "Axis1"
  BEGIN
    ALIAS                               "asmAxis1"
    ATTRIBUTE asmtelPOS_CTLR    "POSLOOP"
    BEGIN
        ATTRIBUTE asmtelENCODER  "ABSENCODER"
        BEGIN
            // Offset of physical to logical zero-point:
            ATTRIBUTE Vector      "offset"      (2, rtINT32)
            BEGIN
                //        +----------------- 0: north offset
                //        |      +---------- 1: south offset
                //        |      |
                Value (( 100 , -100 ))  // [deg]
            END
            ATTRIBUTE asmiSIG_DIGITAL   "initSignal"
            BEGIN
                ALIAS A1VSI
            END
        END
    END
    ATTRIBUTE asmtelAMPLIFIER   "AMPLIFIER"
        BEGIN
            ALIAS asmAmpl1
            ATTRIBUTE asmiSIG_ANALOG    "velocityCommand"
            BEGIN
                ALIAS A1VREFO
            END
            ATTRIBUTE asmiSIG_ANALOG    "currentMonitor"
            BEGIN
                ALIAS A1IMONI
            END
            ATTRIBUTE asmiSIG_DIGITAL   "engPanelSelect"
            BEGIN
                ALIAS A1EPSELI
```

```
          END
      END
  ATTRIBUTE asmtelINTERLOCKS  "INTERLOCKS"
      BEGIN
        ALIAS asmIL1
        ATTRIBUTE axisSTD_IL_CONTROL  "CONTROL"
        BEGIN
          ATTRIBUTE asmiSIG_DIGITAL   "enable"
          BEGIN
              ALIAS A1ILENABO
          END
        END
        ATTRIBUTE axisSTD_IL_STATUS   "STATUS"
        BEGIN
          ATTRIBUTE asmiSIG_DIGITAL   "ampEnable"
          BEGIN
              ALIAS A1ILPAPOWI
          END
          ATTRIBUTE asmiSIG_DIGITAL   "invRange"
          BEGIN
              ALIAS A1ILRANGEI
          END
        END
      END
END

// theta-axis:
ATTRIBUTE asmtelAXIS                  "Axis2"
BEGIN
  ALIAS                               "asmAxis2"
  ATTRIBUTE asmtelPOS_CTLR    "POSLOOP"
  BEGIN
      ATTRIBUTE asmtelENCODER  "ABSENCODER"
      BEGIN
          // Offset of physical to logical zero-point:
          ATTRIBUTE Vector      "offset"      (2, rtINT32)
          BEGIN
              //         +------------------ 0: north offset
              //         |       +---------- 1: south offset
              //         |       |
              Value (( -85.0 , 265.0 ))       // [deg]
          END
          ATTRIBUTE asmiSIG_DIGITAL   "initSignal"
          BEGIN
              ALIAS A2VSI
          END
      END
  END
  ATTRIBUTE asmtelAMPLIFIER   "AMPLIFIER"
      BEGIN
          ALIAS asmAmpl2
          ATTRIBUTE asmiSIG_ANALOG    "velocityCommand"
          BEGIN
              ALIAS A2VREFO
          END
          ATTRIBUTE asmiSIG_ANALOG    "currentMonitor"
          BEGIN
              ALIAS A2IMONI
          END
          ATTRIBUTE asmiSIG_DIGITAL   "engPanelSelect"
          BEGIN
              ALIAS A2EPSELI
          END
      END
  ATTRIBUTE asmtelINTERLOCKS  "INTERLOCKS"
      BEGIN
        ALIAS asmIL2
        ATTRIBUTE axisSTD_IL_CONTROL  "CONTROL"
        BEGIN
```

```
            ATTRIBUTE asmiSIG_DIGITAL   "enable"
            BEGIN
                ALIAS A2ILENABO
            END
          END
          ATTRIBUTE axisSTD_IL_STATUS   "STATUS"
          BEGIN
            ATTRIBUTE asmiSIG_DIGITAL   "ampEnable"
            BEGIN
                ALIAS A2ILPAPOWI
            END
            ATTRIBUTE asmiSIG_DIGITAL   "invRange"
            BEGIN
                ALIAS A2ILRANGEI
            END
          END
        END
  END

END

#endif /*!ASMTEL_CLASS*/
--->END IMPORTED FILE: ../dbl/asmtel.class  (97/6/21  4:56)
```

23.3.7 asmwsBRANCH.class

```
--->BEGIN IMPORTED FILE: ../dbl/../dbl/asmwsBRANCH.class  (97/6/21  4:58)
#ifndef asmwsBRANCH_CLASS
#define asmwsBRANCH_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmwsBRANCH.class,v 1.34 1997/06/20 23:32:38 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ----------------------------------------------
// S.Sandrock 07.11.1996 created
//

// ***********************************************************************
//   NAME
//      asmwsBRANCH - complete database-branch of ASM workstation
//
//   DESCRIPTION
//      The class consists of the complete ASM branch.
//      Following sub-branches are included:
//
//      Control - used by asmwsControl process
//      Devices - for ASM software-devices properties (w/o public data)
//      Public  - contains all provided public data
//
// -----------------------------------------------------------------------

#include "asmi.class"
#include "asmwsCONTROL.class"
#include "asmwsDEVICES.class"
#include "asmwsPUBLIC.class"

CLASS          "asmiBRANCH"    "asmwsBRANCH"
BEGIN
  // branch used by asmwsControl process:
  ATTRIBUTE     "asmwsCONTROL"  "Control"
  BEGIN
                Alias           "asmwsControl"
  END

  // private ASM devices branch:
  ATTRIBUTE     "asmwsDEVICES"  "Devices"
  BEGIN
                Alias           "asmwsDevices"
  END

  // provided ASM data branch:
  ATTRIBUTE     "asmwsPUBLIC"   "Public"
  BEGIN
                Alias           "asmwsPublic"
  END
END

#endif /*!asmwsBRANCH_CLASS*/
--->END IMPORTED FILE: ../dbl/../dbl/asmwsBRANCH.class  (97/6/21  4:58)
```

23.3.8 asmwsCONTROL.class

```
--->BEGIN IMPORTED FILE: ../dbl/../dbl/asmwsCONTROL.class  (97/6/21  4:58)
#ifndef asmwsCONTROL_CLASS
#define asmwsCONTROL_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmwsCONTROL.class,v 1.34 1997/06/20 23:32:39 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ----------------------------------------------
// S.Sandrock 07.11.1996 created
//

// ***********************************************************************
//   NAME
//      asmwsCONTROL - database-branch for asmwsControl process
//
//   SYNOPSIS
//
//   PARENT CLASS
//      evhSTD_COMMANDS
//
//   DESCRIPTION
//
//   CAUTIONS
//
//   EXAMPLES
//
//   SEE ALSO
//
//   BUGS
//
// -----------------------------------------------------------------------

// Defines for standard states.
#include "evhDB_LIST_COMMAND.class"

#include "asmwsCONTROL.h"
#include "asmi.class"           // general definitions

// See asmwsDefines.h for substate #defines

CLASS "asmiTASK" "asmwsCONTROL"
BEGIN
        ATTRIBUTE int32 asmwsCONTROL_busy           0

        ATTRIBUTE int32 asmwsCONTROL_maxAgIteration 10

        // LCU targets that are controlled by asmwsControl process

        ATTRIBUTE evhDB_LIST_COMMAND asmwsCONTROL_stdCmdTargets
        BEGIN
            ATTRIBUTE Vector targets(2, bytes32)
            BEGIN
                Value("@lasm1:lccServer", "@lasm2:lccServer")
            END
        END

        ATTRIBUTE evhDB_COMMAND asmwsCONTROL_asmwsCmdTarget
        BEGIN
            ATTRIBUTE bytes32 destination "@lasm1:asmlcuServer"
        END

        ATTRIBUTE evhDB_COMMAND asmwsCONTROL_trkCmdTarget
        BEGIN
            ATTRIBUTE bytes32 destination "@lasm1:asmtrkServer"
        END
```

```
        ATTRIBUTE evhDB_COMMAND ccdCmdTarget
        BEGIN
            ATTRIBUTE bytes32 destination "ccdconCI_DIMM"
        END
END

#endif /*!asmwsCONTROL_CLASS*/
--->END IMPORTED FILE: ../dbl/../dbl/asmwsCONTROL.class  (97/6/21  4:58)
```

23.3.9 asmwsDEVICES.class

```
--->BEGIN IMPORTED FILE: ../dbl/../dbl/asmwsDEVICES.class  (97/6/21  4:58)
#ifndef asmwsDEVICES_CLASS
#define asmwsDEVICES_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmwsDEVICES.class,v 1.34 1997/06/20 23:32:37 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ----------------------------------------------
// S.Sandrock 12.11.1996 created
//

// ***********************************************************************
//   NAME
//      asmwsDEVICES - private database-branch for all ASM devices
//
//   SYNOPSIS
//
//   PARENT CLASS
//      asmiCLASS
//
//   DESCRIPTION
//      The class consists of the data-branches of all ASM devices.
//
// -----------------------------------------------------------------------

#include "asmwsDEVICES.h"

#include "asmi.class"           // general definitions
#include "asmcat.class"         // Starcat definitions
#include "asmdim.class"         // DIMM definitions
#include "asmmet.class"         // Meteo definitions
#include "asmtelWS.class"       // Telescope definitions

CLASS           "asmiCLASS"     "asmwsDEVICES"
BEGIN
  ATTRIBUTE     "asmcatDEVICE"  "Catalog"
  BEGIN
        Alias   asmwsDEVICES_ALIAS_asmcat
  END

  ATTRIBUTE     "asmdimDEVICE"  "Dimm"
  BEGIN
        Alias   asmwsDEVICES_ALIAS_asmdim
  END

  ATTRIBUTE     "asmmetDEVICE"  "Meteo"
  BEGIN
        Alias   asmwsDEVICES_ALIAS_asmmet
  END

  ATTRIBUTE     "asmtelDEVICE"  "Telescope"
  BEGIN
        Alias   asmwsDEVICES_ALIAS_asmtel
  END
END

#endif /*!asmwsDEVICES_CLASS*/
--->END IMPORTED FILE: ../dbl/../dbl/asmwsDEVICES.class  (97/6/21  4:58)
```

23.3.10 asmwsPUBLIC.class

```
--->BEGIN IMPORTED FILE: ../dbl/../dbl/asmwsPUBLIC.class  (97/6/21  4:58)
#ifndef asmwsPUBLIC_CLASS
#define asmwsPUBLIC_CLASS
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmwsPUBLIC.class,v 1.34 1997/06/20 23:32:37 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ----------------------------------------------
// S.Sandrock 12.11.1996 created
//

// ***********************************************************************
//   NAME
//      asmwsPUBLIC - public database-branch for provided ASM data
//
//   SYNOPSIS
//
//   PARENT CLASS
//      asmiCLASS
//
//   DESCRIPTION
//      The class consists of the public data of all ASM devices.
//      The aliases of the sub-branches are chosen for easy access
//      through the public CCS db interface.
//
// -----------------------------------------------------------------------

#include "asmwsPUBLIC.h"

#include "asmi.class"   // general definitions
#include "asmdim.class" // DIMM definitions
#include "asmmet.class" // Meteo definitions

CLASS           "asmiCLASS"     "asmwsPUBLIC"
BEGIN
  ATTRIBUTE     "asmdimDATA"    "Dimm"
  BEGIN
        Alias   asmwsPUBLIC_ALIAS_Dimm
  END

  ATTRIBUTE     "asmmetDATA"    "Meteo"
  BEGIN
        Alias   asmwsPUBLIC_ALIAS_Meteo
  END
END

#endif /*!asmwsPUBLIC_CLASS*/
--->END IMPORTED FILE: ../dbl/../dbl/asmwsPUBLIC.class  (97/6/21  4:58)
```

23.4 HEADERS

23.4.1 asmcatDefines.h

```
--->BEGIN IMPORTED FILE: ../include/asmcatDefines.h  (97/6/21  4:52)
#ifndef asmcatDefines_H
#define asmcatDefines_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmcatDefines.h,v 1.12 1997/06/12 12:22:08 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- -------------------------------------------------
* S.Sandrock 21.11.1996 created
*/

#include "asmiDefines.h"

/* Module name */
#define asmcatMOD "asmcat"

/*
 * database attribute name for data-table
 */
#define asmcatDB_ALIAS                  "asmcat"

/*
 * database attribute names
 */
#define asmcatATTR_config               "config"
#define asmcatATTR_stars                "stars"
#define asmcatATTR_selected             "selected"

/*
 * index in the site coordinates array
 */
#define asmcatLongitude 0
#define asmcatLatitude  1
#define asmcatHeight    2

/* constant used in time computation */
#define asmcatMDJ_TIME      40587L

/*
 * index in the result array of the getStarData function
 */
#define asmcatGetStarDataLSTrad   0
#define asmcatGetStarDataRArad    1
#define asmcatGetStarDataHAhms    2
#define asmcatGetStarDataHArad    3
#define asmcatGetStarDataMv       4
#define asmcatGetStarDataZdist    5
#define asmcatGetStarDataAirmass  6
#define asmcatGetStarDataZha      7
#define asmcatGetStarDataZmv      8
#define asmcatGetStarDataZ        9
/* dimension of the result array */
#define asmcatGetStarDataDim     10

/*
 * constants used in NOVAS to define astronomical objects
 */
#define asmcatEARTH  3
#define asmcatSUN    0
#define asmcatMOON  -1

#endif  /* !asmcatDefines_H */
--->END IMPORTED FILE: ../include/asmcatDefines.h  (97/6/21  4:52)
```

23.4.2 asmdimDefines.h

```
--->BEGIN IMPORTED FILE: ../include/asmdimDefines.h  (97/6/21  4:54)
#ifndef asmdimDefines_H
#define asmdimDefines_H
/******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmdimDefines.h,v 1.38 1997/06/20 15:24:52 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- --------------------------------------------------
* S.Sandrock 02.12.1996 created
*/

#include "asmiDefines.h"

#define asmdimMOD               "asmdim"

/*
 * database attribute name for data-table
 */
#define asmdimDB_ALIAS          "Dimm"

/*
 * CCD Camara Name (used as point-alias)
 */
#define asmdimCCDNAME           "DIMM"

/*
 * public database attribute names
 */
#define asmdimATTR_data         asmiATTR_data
#define asmdimATTR_History      asmiATTR_History

/*
 * data-table tags for "addressing by content"
 */
#define asmdimTAG_Seeing        "Seeing"
#define asmdimTAG_Scintillation "Scintillation"
#define asmdimTAG_CloudCoverLS  "CloudCoverLS"
#define asmdimTAG_Wavelength    "Wavelength"
#define asmdimTAG_Airmass       "Airmass"
#define asmdimTAG_TimespanDimm  "TimespanDimm"
#define asmdimTAG_Extinction    "Extinction"
#define asmdimTAG_PosAlpha      "PosAlpha"
#define asmdimTAG_PosDelta      "PosDelta"
#define asmdimTAG_TimespanDimm  "TimespanDimm"
#define asmdimTAG_TimespanSky   "TimespanSky"

/*
 * data-table field names: see "asmiDefines.h"
 *
 * Supported in asmdim are:
 *   description, unit, value, timestamp
 */
#define asmdimFIELD_description asmiFIELD_description
#define asmdimFIELD_unit        asmiFIELD_unit
#define asmdimFIELD_value       asmiFIELD_value
#define asmdimFIELD_timestamp   asmiFIELD_timestamp


/*
 * configuration constants
 */

/* LCU identifiers to be given as setup parameters: */
#define asmdim_IPHOOK_NAME asmdimIpHook         /* user IP fcn on LCU */
```

```
/*
 * Max. number of exposures per sequence:
 * Note: DB vector sizes are derived from this value.
 *       In order not to exceed the max. msg size, the value should be <= 1000.
 */
#define asmdimEXP_PER_SEQ_MAX   1000

/*
 * database types:
 */
/*#define asmdimCCDPOS_T        INT32   // DB type for CCD chip coordinates*/
/*#define asmdimCCDCDG_T        DOUBLE  // DB type for CCD centroid coords*/
/*#define asmdimCCDCNT_T        DOUBLE  // DB type for CCD a/d counts*/

#endif  /* !asmdimDefines_H */
--->END IMPORTED FILE: ../include/asmdimDefines.h  (97/6/21  4:54)
```

23.4.3 asmdimInternal.h

```
--->BEGIN IMPORTED FILE: ../include/asmdimInternal.h  (97/6/21  4:54)
#ifndef asmdimInternal_H
#define asmdimInternal_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmdimInternal.h,v 1.38 1997/06/20 15:24:36 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* S.Sandrock 30.10.1996 created
*/

#include "ccd.h"

/*
 * ASM headers
 */
#include "asmi.h"
#include "asmdim.h"

#include "asmdimDB.h"
#include "asmdimCAMERA.h"
#include "asmdimCONFIG.h"
#include "asmdimEXPOS.h"
#include "asmdimErrors.h"

/*
 * Constants
 */
#define asmdimMODULE_ID         "asmdim"
#define asmdimLOG_ID            asmiLOGID_Dimm

/*
 * Data structure
 */
typedef struct
{
    dbSYMADDRESS        dbRoot;
    char               *attrConfigTable;
    char               *attrExposTable;
} asmdimDATA;

/*
 * Functions
 */
ccsCOMPL_STAT asmdimIPHOOK_NAME (ccdIPIMAGE *image,
                                 asmdimEXPOS_spot_values_t *spot,
                                 ccsERROR *error);

ccsCOMPL_STAT asmdimExposEvaluation(asmdimEXPOS_spot_values_t *spot1,
                                    asmdimEXPOS_spot_values_t *spot2,
                                    ccsERROR *error);

ccsCOMPL_STAT asmdimFrameValidation(asmdimEXPOS_spot_values_t *spot1,
                                    asmdimEXPOS_spot_values_t *spot2,
                                    asmdimCONFIG_validation_limits_t *limits,
                                    ccsERROR  *error);

ccsCOMPL_STAT asmdimIpComputation(ccdIPIMAGE *image,
                                  asmdimEXPOS_spot_values_t *spot,
                                  ccsERROR *error);

#endif  /* !asmdimInternal_H */
--->END IMPORTED FILE: ../include/asmdimInternal.h  (97/6/21  4:54)
```

23.4.4 asmdimLCU.h

```
--->BEGIN IMPORTED FILE: ../include/asmdimLCU.h  (97/6/21  4:54)
#ifndef asmdimLCU_H
#define asmdimLCU_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmdimLCU.h,v 1.38 1997/06/20 15:24:36 vltsccm Exp $"
*
* === THIS HEADER IS FOR INTERNAL USE OF ASM-MODULES ONLY ===
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* S.Sandrock 31.10.1996 created
*/

#include "ccs.h"
#include "db.h"
#include "err.h"

/*
 * Exported Functions
 */
const char *asmdimId(void);
ccsCOMPL_STAT asmdimStart(const dbSYMADDRESS modRoot, ccsERROR *error);

extern FUNCPTR asmdim_ccdipIQE;

#endif  /* !asmdimLCU_H */
--->END IMPORTED FILE: ../include/asmdimLCU.h  (97/6/21  4:54)
```

23.4.5 asmiDefines.h

```
--->BEGIN IMPORTED FILE: ../include/asmiDefines.h  (97/6/21  4:52)
#ifndef asmiDefines_H
#define asmiDefines_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmiDefines.h,v 1.22 1997/06/20 23:30:08 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* S.Sandrock 22.11.1996 created
*/

/*
 * module name definition
 */
#define asmiMOD "asmi"

/*
 * used by STATE
 */
#define asmiSTATE_MAX_STATES 10
#define asmiSTATE_MAX_EVENTS 10

/*
 * used by asmiMAIN
 */
#define asmiMAX_RECEIVE_CMD 20

/*
 * used by asmiDEVICE
 */
#define asmiMAX_CMD_DEVICE 20
#define asmiDEV_NAME_LEN   20

/*
 * asmiDATA class: name definitions
 */

/* attribute names: */
#define asmiATTR_data           "data"          /* data-table attribute name */
#define asmiATTR_new_data       "new_data"      /* indicate new data written */
#define asmiATTR_History        "History"       /* history data records      */

/* data-table field names: */
#define asmiFIELD_tag           "tag"           /* for addressing by content */
#define asmiFIELD_loc           "loc"           /* sensor location when appl.*/
#define asmiFIELD_description   "description"   /* data in this row          */
#define asmiFIELD_unit          "unit"          /* unit of the values        */
#define asmiFIELD_value         "value"         /* instant value             */
#define asmiFIELD_timestamp     "timestamp"     /* last update of inst.value */
#define asmiFIELD_timespan_stat "timespan_stat" /* update period of stat's   */
#define asmiFIELD_average       "average"       /* avg over statistics period*/
#define asmiFIELD_minimum       "minimum"       /* min over statistics period*/
#define asmiFIELD_maximum       "maximum"       /* max over statistics period*/
#define asmiFIELD_deviation     "deviation"     /* dev over statistics period*/

/*
 * Defines for asmiCOM class
 */
#define asmiCOM_STATUS_OFF      0
#define asmiCOM_STATUS_ON       1
#define asmiCOM_STATUS_WARN     2
#define asmiCOM_STATUS_ERR      3
#define asmiCOM_STATUS_DOWN     4

/*
```

```
 *# LOG IDs per device
 */
#define asmiLOGID_Monitor       200
#define asmiLOGID_Enclosure     210
#define asmiLOGID_Telescope     220
#define asmiLOGID_Dimm          230
#define asmiLOGID_Targets       240
#define asmiLOGID_Meteo         250

/*
 *# error macros
 */
#define asmiERR_ADD(module, code, arg...) \
{ \
    ErrAdd(#module, module ## ERR_ ## code, __FILE_LINE__, ##arg);\
}

#endif  /* !asmiDefines_H */
--->END IMPORTED FILE: ../include/asmiDefines.h  (97/6/21  4:52)
```

23.4.6 asmiLCU.h

```
--->BEGIN IMPORTED FILE: ../include/asmiLCU.h  (97/6/21  4:52)
#ifndef asmiLCU_H
#define asmiLCU_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmiLCU.h,v 1.22 1997/06/20 23:30:03 vltsccm Exp $"
*
* === THIS HEADER IS FOR INTERNAL USE OF ASM-MODULES ONLY ===
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* S.Sandrock 31.10.1996 created
*/

/*
 *# VxWorks system headers
 */
#include <vxWorks.h>

/* (Posix) */
#include <stdio.h>
#include <stdarg.h>
#include <string.h>
#include <types.h>
#include <time.h>
#include <math.h>

/* (Libraries) */
#include <sysLib.h>
#include <tickLib.h>
#include <msgQLib.h>
#include <memLib.h>
#include <lstLib.h>
#include <semLib.h>
#include <wdLib.h>
#include <taskLib.h>
#include <rebootLib.h>
#include <usrLib.h>
#include <errnoLib.h>
#include <vxLib.h>

/*
 * VLT headers
 */
#include "ccs.h"
#include "cmd.h"
#include "db.h"
#include "err.h"
#include "io.h"
#include "msg.h"
#include "too.h"

#include "lccErrors.h"
#include "asmiErrors.h"


/*
 * Argument for LCC standard functions in asm modules
 */
typedef struct {
    IN     msgMSG     *msg;     /* current message for intermediate replies */
    IN     const char *devName; /* software-device name         */
    IN     vltLOGICAL  verbose; /* verbose enable flag          */
    IN     vltLOGICAL  simulat; /* simulation mode flag         */
    IN     lccDEV_MODE stateNow; /* current device state        */
    IN OUT lccDEV_MODE stateNext;/* coming device state         */
```

```
} asmiCmdArg_t;

/*
 * Prototype for LCC standard functions in asm modules
 */
typedef ccsCOMPL_STAT (*asmiCmdFunc_t)(IN OUT asmiCmdArg_t *cmdArg,
                                       OUT    ccsERROR *error);


/*
 * Exported Functions
 */
const char *asmiId(void);
ccsCOMPL_STAT asmiStart(const dbSYMADDRESS modRoot, ccsERROR *error);

ccsCOMPL_STAT asmiTaskSpawn(ccsCOMPL_STAT (*function)(ccsERROR *error),
                            char *taskName,
                            int taskPriority,
                            int taskOptions,
                            int taskStack,
                            ccsERROR *taskError,
                            int *tid,
                            ccsERROR *error);
ccsCOMPL_STAT asmiTaskDelete(int tid, ccsERROR *error);

/*
 * Command handling macros
 * -----------------------
 * asmiREPLY_ADD: send intermediate reply
 * asmiREPLY_RETURN: format last reply and return from command-handler
 *
 * The identifier `msg' must be defined as pointer to the current message!
 * If `msg' is NULL then the reply is printed to stdout with trailing new-line.
 */
#define asmiREPLY_ADD(format, args...)\
        tooReplyf(msg, format, ##args)

#define asmiREPLY_RETURN(format, args...)\
        tooINTERFACE_REPLY_RETURN(format, ##args)

/*
 * Error handling macros
 * ---------------------
 * asmiERROR_ADD: add error
 * asmiERROR_RETURN: add error and return with FAILURE
 * asmiERROR_CONTINUE: log current error and continue in loop
 * asmiERROR_ADD_CONTINUE: add and log error and continue in loop
 * asmiERROR_BREAK: log current error and break loop
 * asmiERROR_ADD_BREAK: add and log error and break loop
 * asmiERROR_EXIT: log current error, ccsExit, return with FAILURE
 * asmiERROR_ADD_EXIT: add and log error, ccsExit, return with FAILURE
 *
 * The identifier `error' must be defined as pointer to the error-structure!
 */
#define _asmi_tostr(a) #a
#define _asmi_tostr_pass2(a) _asmi_tostr(a)
#define asmiLOC_ID   __FILE__ "(" _asmi_tostr_pass2(__LINE__) ")"

#define asmiERRNOSTR (strerror(errnoGet()))

#define asmiERROR_ADD(module, code, arg...) \
{ \
    errAdd(error, #module, module ## ERR_ ## code, asmiLOC_ID, NULL, ##arg);\
}

#define asmiERROR_RETURN(module, code, arg...) \
{ \
    errAdd(error, #module, module ## ERR_ ## code, asmiLOC_ID, NULL, ##arg);\
    return FAILURE; \
```

```
}

#define asmiERROR_CONTINUE() \
{ \
    errCloseStack(error); /* log this error */  \
    continue; \
}

#define asmiERROR_ADD_CONTINUE(module, code, arg...) \
{ \
    errAdd(error, #module, module ## ERR_ ## code, asmiLOC_ID, NULL, ##arg);\
    asmiERROR_CONTINUE() \
}

#define asmiERROR_BREAK() \
{ \
    errCloseStack(error); /* log this error */  \
    break; \
}

#define asmiERROR_ADD_BREAK(module, code, arg...) \
{ \
    errAdd(error, #module, module ## ERR_ ## code, asmiLOC_ID, NULL, ##arg);\
    asmiERROR_BREAK() \
}

#define asmiERROR_EXIT() \
{ \
    ccsERROR e;\
    errResetStack(&e); errMergeStack(&e, error, errTOP); errCloseStack(&e);\
    if (ccsExit(&e) == FAILURE) errMergeStack(error, &e, errBOTTOM); \
    return FAILURE; \
}

#define asmiERROR_ADD_EXIT(module, code, arg...) \
{ \
    errAdd(error, #module, module ## ERR_ ## code, asmiLOC_ID, NULL, ##arg);\
    asmiERROR_EXIT() \
}

#endif  /* !asmiLCU_H */
--->END IMPORTED FILE: ../include/asmiLCU.h  (97/6/21  4:52)
```

23.4.7 asmlcuInternal.h

```
--->BEGIN IMPORTED FILE: ../include/asmlcuInternal.h  (97/6/21  4:57)
#ifndef asmlcuInternal_H
#define asmlcuInternal_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmlcuInternal.h,v 1.30 1997/06/05 11:44:23 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- -------------------------------------------------
* S.Sandrock 30.10.1996 created
*/

/*
 * VLT headers
 */
#include "acro.h"
#include "iser.h"
#include "too.h"

/*
 * ASM headers
 */
#include "asmlcu.h"
#include "asmi.h"
#include "asmdim.h"
#include "asmmet.h"
#include "asmtel.h"
#include "asmten.h"

/*
 * Local headers
 */
#include "asmlcuCmd.h"
#include "asmlcuErrors.h"
#include "asmlcuMonitor.h"
#include "asmlcuServer.h"

/*
 * Constants
 */
#define asmlcuMODULE_ID         "asmlcu" /* module name                 */
#define asmlcuPATH_NAME_LEN     256      /* max file name length        */

#define NSUP  ((void *)-1)       /* function-pointer "not supported"     */
#define TBD   ((void *)-2)       /* function-pointer "to be implemented" */


/*
 * Log IDs
 */
#define asmlcuLOG_ID            101      /* general ID          */
#define asmlcuSERVER_LOG_ID     102      /* used by asmlcuServer */
#define asmlcuMONITOR_LOG_ID    103      /* used by asmlcuMonitor       */
#define asmlcuCOM_LOG_ID        104      /* used by asmlcuCom tasks     */


/*
 * ------------------------------------
 * Data
 * ------------------------------------
 */
const char *asmlcuId(void);
extern vltLOGICAL asmlcu_verbose;


/*
```

```
 * ------------------------------------
 * Devices
 * ------------------------------------
 */

/* Internal IDs for LCC standard functions (DO NOT CHANGE ORDER!)*/
typedef enum {
    FuncId_low  = -1, /* this must be the first entry! */

    /* mandatory LCC standard commands: */
    Init, Standby, Online, Off, Exit, Stop, Selftst,

    /* optionalLCC semi-standard commands: */
    Test, Status,

    FuncId_high /* this must be the last entry! */
} asmlcuDevicesFuncId_t;

typedef struct {
    const char      *devName;
    vltLOGICAL       verbose;
    asmiCmdFunc_t  devFunc[FuncId_high];
} asmlcuDevices_t;

extern asmlcuDevices_t asmlcuDevices[10];

ccsCOMPL_STAT asmlcuDevicesCheck(IN  const char *devName,
                              OUT int        *index,
                              OUT ccsERROR   *error);


/*
 * ------------------------------------
 * Modules
 * ------------------------------------
 */
typedef struct
{
    const char   *modName;                      /* module name            */
    const char   *modRoot;                      /* relative database root */
    const char *(*modId)(void);                 /* Version ID function    */
                                                /* module Start function  */
    ccsCOMPL_STAT (*modStart)(const dbSYMADDRESS modRoot, ccsERROR *error);
} asmlcuModules_t;

extern asmlcuModules_t asmlcuModules[10];


/*
 * ------------------------------------
 * Functions
 * ------------------------------------
 */
ccsCOMPL_STAT asmlcuStart(dbSYMADDRESS dbRoot, ccsERROR *error);

#endif  /* !asmlcuInternal_H */
--->END IMPORTED FILE: ../include/asmlcuInternal.h  (97/6/21  4:57)
```

23.4.8 asmlcuMonitor.h

```
--->BEGIN IMPORTED FILE: ../include/asmlcuMonitor.h  (97/6/21  4:57)
#ifndef asmlcuMonitor_H
#define asmlcuMonitor_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmlcuMonitor.h,v 1.30 1997/06/05 11:44:21 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* ssandroc   26.01.1996 created
*/

/*
 * task spawn parameters for asmlcuMonitor
 */
#define asmlcuMONITOR_NAME      "asmlcuMonitor"
#define asmlcuMONITOR_PRIORITY  60
#define asmlcuMONITOR_OPTIONS   VX_FP_TASK | VX_STDIO
#define asmlcuMONITOR_STACK     50000

/*
 * constants for monitor configuration table
 */
#define asmlcuMONITOR_CONFIG_FILE       "asmlcuMonitor.config"
#define asmlcuMONITOR_CONFIG_TABLE_SIZE 100 /* max nubmer of entries in table */

/*
 * types for monitor configuration table
 */
typedef enum {
    asmlcuMONITOR_SIGNAL = 1,
    asmlcuMONITOR_ATTR,
    asmlcuMONITOR_TASK
} asmlcuMonitorType_t;

typedef union {
    ioDIRADDRESS sigAddr;
    char *       taskName;
    dbDIRADDRESS dirAddr;
    dbSYMADDRESS attrName;
} asmlcuMonitorId_t;

typedef enum {
    asmlcuMONITOR_EQUAL = 1,
    asmlcuMONITOR_NOTEQUAL
    /* maybe more ... */ /*@TBD@*/
} asmlcuMonitorOperator_t;

typedef vltUINT32 asmlcuMonitorValue_t;

typedef lccDEV_MODE asmlcuMonitorState_t;
#define asmlcuMONITOR_ALL_STATES        -1      /* as extension to above enum type, */
                        /* must be different from values in lccDEV_MODE     */

/*
 * string macros for members of monitor configuration table
 */
#define asmlcuMONITOR_TYPE_STR(type) \
  ( (type==asmlcuMONITOR_SIGNAL) ? "SIGNAL" : \
    (type==asmlcuMONITOR_ATTR)   ? "ATTR"   : \
    (type==asmlcuMONITOR_TASK)   ? "TASK"   : \
    "????")

#define asmlcuMONITOR_OPERATOR_STR(operator) \
  ( (operator==asmlcuMONITOR_EQUAL)    ? "==" : \
    (operator==asmlcuMONITOR_NOTEQUAL) ? "!=" : \
```

```
    "??")

#define asmlcuMONITOR_STATE_STR(state) \
  ( (state==asmlcuMONITOR_ALL_STATES) ? "ALL"    : \
    (state==lccDEV_OFF)            ? "OFF"       : \
    (state==lccDEV_LOADED)         ? "LOADED"    : \
    (state==lccDEV_STAND_BY)       ? "STANDBY"   : \
    (state==lccDEV_ON_LINE)        ? "ONLINE"    : \
    "????")


/*
 * monitor configuration table:
 * list of items (signals/attributes/tasks) to be monitored in which states
 */
typedef struct {
  BOOL                    used;          /* flag for used table entry    */
  char                    item[20];      /* item to monitor as string    */
  asmlcuMonitorType_t     type;          /* type of item to monitor      */
  asmlcuMonitorId_t       id;            /* pointer to item              */
  asmlcuMonitorOperator_t operator;      /* what to do with `expected'   */
  asmlcuMonitorValue_t    expected;      /* expected value of item       */
  char                    device[20];    /* software-device to monitor   */
  asmlcuMonitorState_t    monitorState;  /* state in which item shall be monitored */
  asmlcuMonitorState_t    fallbackState; /* new state in case of condition mismatch*/
} asmlcuMonitorTable_t;


/*
 * global data
 */
extern int      asmlcuMonitorTid;       /* asmlcuMonitor task ID         */
extern int      asmlcuMonitorPeriod;    /* asmlcuMonitor loop delay time */
extern ccsERROR asmlcuMonitorError;     /* asmlcuMonitor Error structure */

extern  asmlcuMonitorTable_t \
        asmlcuMonitorTable[asmlcuMONITOR_CONFIG_TABLE_SIZE];
                                        /* monitor configuration table    */

/*
 * functions
 */
ccsCOMPL_STAT asmlcuMonitorInit(asmiCmdArg_t *cmdArg, ccsERROR *error);
ccsCOMPL_STAT asmlcuMonitorOff(asmiCmdArg_t *cmdArg, ccsERROR *error);
ccsCOMPL_STAT asmlcuMonitorConfig(const char *filename, ccsERROR *error);

#endif  /* !asmlcuMonitor_H */
--->END IMPORTED FILE: ../include/asmlcuMonitor.h  (97/6/21  4:57)
```

23.4.9 asmlcuServer.h

```
--->BEGIN IMPORTED FILE: ../include/asmlcuServer.h  (97/6/21  4:57)
#ifndef asmlcuServer_H
#define asmlcuServer_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmlcuServer.h,v 1.30 1997/06/05 11:44:22 vltsccm Exp $"
*
* who        when       what
* ---------- ---------- ------------------------------------------------
* S.Sandrock 30.10.1996 created
*/

/*
 * task spawn parameters for asmlcuServer
 */
#define asmlcuSERVER_NAME       "asmlcuServer"
#define asmlcuSERVER_PRIORITY   80
#define asmlcuSERVER_OPTIONS    VX_FP_TASK | VX_STDIO
#define asmlcuSERVER_STACK      50000

/*
 * functions
 */
ccsCOMPL_STAT asmlcuServerInit(ccsERROR *error);

#endif  /* !asmlcuServer_H */
--->END IMPORTED FILE: ../include/asmlcuServer.h  (97/6/21  4:57)
```

23.4.10 asmtel.h

```
--->BEGIN IMPORTED FILE: ../include/asmtel.h  (97/6/21  4:56)
#ifndef ASMTEL_H
#define ASMTEL_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmtel.h,v 1.35 1997/06/20 15:34:37 vltsccm Exp $"
*
* who       when      what
* --------  --------  ----------------------------------------------
* ramestic  11/11/96  created
*/

/************************************************************************
 *
 *----------------------------------------------------------------------
 */

/*
 *# VxWorks headers
 */
/*
#include <vxWorks.h>
*/

/*
 * VLT headers
 */
#include "ccs.h"
#include "cmd.h"
#include "db.h"
#include "err.h"
#include "acm.h"

/*
 * Exported Functions
 */
const char *asmtelId(void);
ccsCOMPL_STAT asmtelStart(const dbSYMADDRESS modRoot, ccsERROR *error);
ccsCOMPL_STAT asmtelSelftst(asmiCmdArg_t *cmdArg, ccsERROR *error);

#endif /*!ASMTEL_H*/
--->END IMPORTED FILE: ../include/asmtel.h  (97/6/21  4:56)
```

23.4.11 asmtelDefines.h

```
--->BEGIN IMPORTED FILE: ../include/asmtelDefines.h  (97/6/21  4:56)
#ifndef ASMTELDEFINES_H
#define ASMTELDEFINES_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmtelDefines.h,v 1.35 1997/06/20 15:34:38 vltsccm Exp $"
*
* who       when      what
* --------  --------  ----------------------------------------------
* ramestic  06/12/96  created
*/

/*
 * Site monitor main branch
 */
#define ASM_BRANCH "ASM"

/*
 * Axes names
 */
#define ASMTEL_ETA_STR    "eta"
#define ASMTEL_THETA_STR  "theta"

/*
 * Database points
 */
#define asmtelDB_axesData  "axesData"
#define asmtelDB_Tracking  "Tracking"
#define asmtelDB_etaAxis   "Axis1"
#define asmtelDB_thetaAxis "Axis2"

/*
 * CDT and CIT files for the tracking server
 */
#define asmtelTRK_CDT "asmtrkServer.cdt"
#define asmtelTRK_CIT "asmtrkServer.cit"

#endif /*!ASMTELDEFINES_H*/
--->END IMPORTED FILE: ../include/asmtelDefines.h  (97/6/21  4:56)
```

23.4.12 asmtelInternal.h

```
--->BEGIN IMPORTED FILE: ../include/asmtelInternal.h  (97/6/21  4:56)
#ifndef ASMTELINTERNAL_H
#define ASMTELINTERNAL_H
/*******************************************************************************
* E.S.O. - VLT project
*
* "@(#) $Id: asmtelInternal.h,v 1.35 1997/06/20 15:34:36 vltsccm Exp $"
*
* who       when      what
* --------  --------  ----------------------------------------------
* ramestic  03/10/96  created
*/

#define TARGET_TCS VLT_TCS

#define ASM_SERVER_STR    "asmServer"
#define ASMTEL_DEVICE_STR "Telescope"

/*
 * VLT headers
 */
#include "acm.h"
#include "axis.h"
#include "trk.h"
#include "cai.h"
#include "slalib.h"

/*
 * Module headers
 */
#include "asmi.h"
#include "asmtelDefines.h"
#include "asmtelDB.h"
#include "asmtelHwDB.h"
#include "asmtelEncDB.h"
#include "asmtelSignals.h"

#include "asmtelErrors.h"

/*
 * Constants
 */
#define asmtelMODULE_ID    "asmtel"   /* module name                  */
#define ASMTEL_ETA_ID      0
#define ASMTEL_THETA_ID    1
#define ENC_POSITIVE_DIR   1
#define ENC_NEGATIVE_DIR  -1

#define asmtelNO_HALF_SKY 0
#define asmtelNORTH_HALF  1
#define asmtelSOUTH_HALF  2

/*
 * TELOFFS command constants
 */
#define asmtelTELOFFS_TIMEOUT   20      /* 20 * 5 * TEN_MSEC = 1 SEC */
#define asmtelTELOFFS_acc_STR   "acc"
#define asmtelTELOFFS_reset_STR "reset"

/*
 * acm server state strings
 */
#define asmtelACM_STATE_STR     \
    {                           \
    "Undefined",                \
    "Idle",                     \
    "Presetting",               \
```

```
    "Tracking"                  \
    }

/*
 * Log IDs
 */
#define asmtelLOG_ID       105     /* general ID           */

/*
 * Encoders initializiation phases
 */
typedef enum
{
  ENC_PHASE1a,          /* move until init-switch asserted */
  ENC_PHASE1b,          /* PHASE1 ready */

  ENC_PHASE2a,          /* reverse direction of the movement, move until
                           init-switch negated */
  ENC_PHASE2b,          /* move in the same direction until ref-mark found */
  ENC_PHASE2c           /* PHASE2 ready */
} asmtelENC_PHASE;


/*
 * Types
 */
typedef struct
{
    dbSYMADDRESS        dbRoot;
    char                *attrAxesTable;
    vltBYTES32          encDevice;
    vltDOUBLE           gravityMargin;
    vltDOUBLE           northLims[4];
    vltDOUBLE           southLims[4];
    vltDOUBLE           transferMatrix[3][3];
    vltUINT32           halfSky;
    vltDOUBLE           offsetEta, offsetTheta;
    SEM_ID              teloffsSem;
    ioDIRADDRESS        signal[ asmtelNUM_SIGNALS + 1];
}asmtelAXESDATA;

typedef struct
{
  vltLOGICAL                used;
  dbSYMADDRESS              dbRoot;
  vltINT32                  axisId;
  const char               *axisName;
  char                     *attrTable;       /* attribute table for cai
                                                utility */
  vltDOUBLE                 powerTimeOut, powerOnWait;
  acmAXIS_SEND_POS_REFS_P   cbAxisSendRefsP;
  ioDIRADDRESS              signal[ asmetelHW_NUM_SIGNALS + 1];
  int                       subState;
}asmtelHWDATA;

typedef struct
{
    vltLOGICAL                  used;
    dbSYMADDRESS                dbRoot;
    vltINT32                    axisId;
    const char *                axisName;
    const char *                devName;
    int                         devChannel;
    char                        *attrTable;    /* attribute table for cai
                                                  utility */
    acmAXIS_PRESET_POS_P        cbAxisPresetPosP;
    acmAXIS_SEND_VEL_REF_P      cbAxisSendVelRefP;
    acmAXIS_GET_STATE_P         cbAxisGetStateP;
    acmAXIS_UPDATE_POS_LIMS_P   cbAxisUpdatePosLimsP;
```

```
    ioDIRADDRESS                signal[ asmtelENC_NUM_SIGNALS + 1 ];

    int       fd;               /* ikon file-descriptor */
    int       ioctlLatch;       /* set to ikonCMD_LATCH_POSITION_X1/X2 */
    int       ioctlRefMove;     /* set to ikonCMD_REF_MOVE_X1/X2 */
    asmtelENC_PHASE initPhase;  /* phase of init-step */
    vltUINT32 initTimeout[3];   /* timeout for phase of init-step */
    vltDOUBLE initSpeed[3];     /* speed during init-movement */
    vltUINT32 scale;            /* = (gearRatio * lineCount) */
    vltINT32  offset[2];        /* of physical to logical zero-point */
} asmtelENCDATA;

/*
 * asmtel functions
 */
const char *asmtelId(void);

ccsCOMPL_STAT asmtelHwStart(
    char                     *dbRoot,
    acmAXIS_SEND_POS_REFS_P   cbAxisSendRefsP,
    acmHW_P                   acmHw,
    acmAXIS_SIGNAL_CONF_P     acmHwSignalConf,
    ccsERROR                 *error
    );

ccsCOMPL_STAT asmtelEncStart(
    char                      *dbRoot,
    acmAXIS_PRESET_POS_P       cbAxisPresetPosP,
    acmAXIS_SEND_VEL_REF_P     cbAxisSendVelRefP,
    acmAXIS_GET_STATE_P        cbAxisGetStateP,
    acmAXIS_UPDATE_POS_LIMS_P  cbAxisUpdatePosLims,
    acmAXIS_SIGNAL_CONF_P      acmEncSignalConf,
    acmENC_P                   acmEnc,
    ccsERROR                  *error
    );

ccsCOMPL_STAT asmtelTrkSendRefs
    (
    vltINT32                 action,
    ccsTIMEVAL               timeStamp,
    acmSEND_REFS_SET_POINTS  setPoints,
    vltINT32                *state,
    ccsERROR                *error
    );

ccsCOMPL_STAT asmtelSetLims(asmtelENCDATA *data,
                            ccsERROR      *error);

ccsCOMPL_STAT coordinateTransform(vltDOUBLE  R[3][3],
                                  vltDOUBLE  alt,
                                  vltDOUBLE  az,
                                  vltDOUBLE *eta,
                                  vltDOUBLE *theta,
                                  ccsERROR  *error);

#endif /*!ASMTELINTERNAL_H*/
--->END IMPORTED FILE: ../include/asmtelInternal.h  (97/6/21  4:56)
```

23.5 LCU FUNCTIONS

23.5.1 asmdimExposEvaluation(3)

NAME

```
asmdimExposEvaluation - ASM-DIMM LCU end-of-exposure evaluation
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimExposEvaluation(asmdimEXPOS_spot_values_t *spot1,
                                    asmdimEXPOS_spot_values_t *spot2,
                                    ccsERROR *error)
```

DESCRIPTION

```
asmdimExposEvaluation executes on the LCU and performs at the end
of exposure the frame validation, computes the exposure parameters
that are not related to a single spot, and writes the results
to the LCU database.

spot1   IN      image analysis results for spot 1
spot2   IN      image analysis results for spot 2
error   IN/OUT  error stack
```

SEE ALSO

```
asmdimFrameValidation(3),
ASM DIMM Upgrade Plan VLT-SPE-ESO-17410-1174
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.2 asmdimFrameValidation(3)

NAME

```
asmdimFrameValidation - ASM-DIMM LCU Frame Validation
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimFrameValidation(
        asmdimEXPOS_spot_values_t *spot1,
        asmdimEXPOS_spot_values_t *spot2,
        asmdimCONFIG_validation_limits_t *limits,
        ccsERROR  *error)
```

DESCRIPTION

```
asmdimFrameValidation verifies if an exposure can be accepted.
It checks the given exposure parameters against the configuration
limits. In case of failed validation it logs the reason together
with the failed value and returns an error.

spot1   IN      image analysis results for spot 1
spot2   IN      image analysis results for spot 2
limits  IN      configured statistical limits
error   IN/OUT  error stack
```

SEE ALSO

```
ASM DIMM Upgrade Plan VLT-SPE-ESO-17410-1174
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.3 asmdimInit(3)

NAME

```
asmdimInit - DIMM device INIT function
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimInit(asmiCmdArg_t *cmdArg, ccsERROR *error)
```

DESCRIPTION

```
No operation.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.4 asmdimIpComputation(3)

NAME

```
asmdimIpComputation - ASM-DIMM LCU Specific Image Processing
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimIpComputation(ccdIPIMAGE *image,
                                  asmdimEXPOS_spot_values_t *spot,
                                  ccsERROR *error)
```

DESCRIPTION

```
asmdimIpComputation executes on the LCU and calculates those
image analysis parameters that are not provided by the CCD-SW.

image   IN      pointer to image description structure (see ccd.h)
*spot   OUT     parameters of either subarray 1/2 to be set
error   IN/OUT  error stack
```

CAUTIONS

```
The function ccdipIQE is called, so it becomes an external reference.
If asmdim is loaded, then ccdip must have been loaded before.
```

SEE ALSO

```
ccdipIQE(3),
ASM DIMM Upgrade Plan VLT-SPE-ESO-17410-1174
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.5 asmdimIpHook(3)

NAME

```
asmdimIpHook - ASM-DIMM LCU Camera Image Processing Hook
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimIpHook(ccdIPIMAGE *image,
                           void       *dummy,
                           ccsERROR   *error)
```

DESCRIPTION

```
asmdimIpHook executes on the LCU and is configured to be
called indirectly by the CCD-SW at the end of image analysis.
The function is called for each subarray (window 1/2 in order)
with the argument corresponding to subarray 1 or 2.
It computes those parameters not provided by the CCD-SW,
and finally calls the end-of-exposure evaluation.

image   IN      pointer to image description structure (see ccd.h).
error   IN/OUT  error stack
```

RETURN VALUES

```
SUCEESS or FAILURE
```

CAUTIONS

```
When the function returns FAILURE, the CCDSW regards the exposure
as failed, and the loop will be stopped (ExpStatus = Failure).
```

SEE ALSO

```
asmdimIpComputation(3), asmdimExposEvaluation(3)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.6 asmdimOff(3)

NAME

```
asmdimOff - DIMM device OFF function
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimOff(asmiCmdArg_t *cmdArg, ccsERROR *error)
```

DESCRIPTION

```
No operation.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.7 asmdimStart(3)

NAME

```
asmdimStart - starts the asmdim module
```

SYNOPSIS

```
ccsCOMPL_STAT asmdimStart(IN dbSYMADDRESS *dbRoot, OUT ccsERROR *error)
```

DESCRIPTION

```
asmdimStart initializes the asmdim module.
It determines the direct addresses of all used database attributes.
The function is indirectly called by asmlcuStart, or from the shell.
All arguments can be zero to use default values.

The function can be called with or w/o prior ccsInit.
If not already registered, it will call ccsInit/Exit itself.
```

DATABASE

```
asmdimDEVICE_ALIAS_CONFIG
asmdimDEVICE_ALIAS_EXPOS
```

RETURN VALUES

```
SUCCESS or FAILURE
```

CAUTIONS

```
None
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:54
```

23.5.8 asmi(3)

NAME

```
asmi - common LCU functionality module of ASM
```

SYNOPSIS

```
const char *asmiId(void)
```

DESCRIPTION

```
asmiId - returns the version-ID of the module
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:52
```

23.5.9 asmiTask(3)

NAME

```
asmiTask,
asmiTaskSpawn, asmiTaskDelete
- task library
```

SYNOPSIS

```
ccsCOMPL_STAT asmiTaskSpawn(ccsCOMPL_STAT (*function)(ccsERROR *),
                            const char *taskName,
                            int taskPriority,
                            int taskStack,
                            ccsERROR *taskError,
                            int *tid,
                            ccsERROR *error)

ccsCOMPL_STAT asmiTaskDelete(int tid, ccsERROR *error)
```

DESCRIPTION

```
asmiTaskSpawn     - spawn a task

  function        - function to spawn
  taskName        - name of the task
  taskPriority    - priority of the task  default: same as this
  taskStack       - stack of the task     default: 50000
  taskError       - pointer to static (!) error-stack (may be NULL)
  tid             - pointer to TID to be returned (may be NULL)
  error           - returned error

  `taskError' must be static and is forwarded to the spawed task.
  It may signal an error in the first first of the start-up
  (ie. before it is blocked the first time) which will be merged
  to the the returned `error' then.

asmiTaskDelete    - terminate task
```

CAUTIONS

```
The priority of the calling task is temporarily changed below
those of the spawned task.
The spawned task should not close its error-stack before exiting.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:52
```

23.5.10 asmlcuCmdLCC(3)

NAME

```
asmlcuCmdLCC,
asmlcuCmdInit, asmlcuCmdStandby, asmlcuCmdOnline, asmlcuCmdOff,
asmlcuCmdExit, asmlcuCmdStop, asmlcuCmdSelftst,
asmlcuCmdSimulat, asmlcuCmdStopsim,
asmlcuCmdTest, asmlcuCmdStatus
- commands-handlers for LCC standard commands
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuCmd...(const char *devName,
                           msgMSG *msg, ccsERROR *error)
```

DESCRIPTION

```
Command-handlers for all LCC standard commands:

    - INIT STANDBY ONLINE OFF EXIT STOP SELFTST SIMULAT STOPSIM -

Not used by LCC, but handled equally here:

    - TEST STATUS -

Includes indirect calls to handlers of sub-modules in case of:

    - INIT STANDBY ONLINE OFF EXIT STOP SELFTST TEST STATUS -

If any specific actions are needed for these, the module of
the software-device has to provide a corresponding function.

Simulation mode: all requests will be intercepted, and simply
OK is returned, except for the STATUS command.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

ERRORS

```
asmlcuERR_INTERNAL
asmlcuERR_PARAMETER
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.11 asmlcuCmdState(3)

NAME

```
asmlcuCmdState - STATE command-handler
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuCmdState(const char *devName,
                             msgMSG *msg, ccsERROR *error)
```

DESCRIPTION

```
Returns the state of a specific or all software-devices.

The reply is in the form "<state>,<simul>" for one device,
or "<device1>,<state1>,<simul1>,..." and repeated for each
device in case of "all".

In verbose mode all individual states are also returned as
intermediate replies.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.12 asmlcuCmdVerbose(3)

NAME

```
asmlcuCmdVerbose - VERBOSE command-handler
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuCmdVerbose(const char *devName, const char *what,
                               msgMSG *msg, ccsERROR *error)
```

DESCRIPTION

```
Switches verbose mode for one specific or all software-devices.
In verbose mode intermediate replies to commands are enabled.
Each device can be individually put into verbose mode,
or all devices at once. Both setting are independent, i.e.
when the verbose mode for _all_ devices is switched off,
the individual settings are restored.

`devName' is the name of the device, or "all" for all devices.

`what' can be "get" to query the current verbose setting,
or "on"/"off" to toggle the setting.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.13 asmlcuCmdVersion(3)

NAME

```
asmlcuCmdVersion - VERSION command-handler
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuCmdVersion(msgMSG *msg, ccsERROR *error)
```

DESCRIPTION

```
Returns the version-IDs of all loaded ASM modules as last reply,
separated by new-lines.

In verbose mode all individual IDs are also returned as
intermediate replies.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.14 asmlcuDevices(3)

NAME

```
asmlcuDevices - definitions for software-devices handled by asmlcuServer
```

SYNOPSIS

```
asmlcuDevices_t asmlcuDevices[]
int asmlcuDevicesShow(void)
ccsCOMPL_STAT asmlcuDevicesCheck(IN  const char *devName,
                                 OUT int        *index,
                                 OUT ccsERROR   *error)
```

DESCRIPTION

```
asmlcuDevices:
  This table contains the software-devices handled in the ASM:

  - pointers to LCC standard functions contained in lower level modules
  - one verbose-flag per device

  Special values for function-pointers:

  NULL:   to be used when a function is not defined, and the
          invocation shall bahave as if the function had returned OK.
  NSUP:   to be used when a function does not exist,
          and the invocation shall trigger a "not supported" error,
          even if the device is in simulation.
  TBD:    to be used when a function must exist but is not yet
          implemented; invocation shall trigger an internal error,
          unless the device is in simulation.

  The order of functions must match the enum-type asmlcuDevicesFuncId_t!


asmlcuDevicesShow: shell utility to print the devices-table

asmlcuDevicesCheck: finds a device in the table and returns its index
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.15 asmlcuModules(3)

NAME

```
asmlcuModules - definitions for software-modules included in ASM
```

SYNOPSIS

```
asmlcuModules_t asmlcuModules[]
int asmlcuModulesShow(void)
```

DESCRIPTION

```
asmlcuModules - Table with pointers to standard functions for each module

  Fields:
    modName:      name of the module
    modRoot:      database root point relative to global ASM root
    modId:        function that returns the module version ID
    modStart:     function that starts up the module

  Values:
    NULL: to be used when no function is defined, function is skipped.


asmlcuModulesShow - shell utility to print the modules-table
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.16 asmlcuMonitor(3)

NAME

```
asmlcuMonitor - ASM Monitor Supervisor task
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuMonitorTask(ccsERROR *error)
ccsCOMPL_STAT asmlcuMonitorInit(ccsERROR *error)
ccsCOMPL_STAT asmlcuMonitorKill(ccsERROR *error)
```

DESCRIPTION

```
asmlcuMonitor supervises all asm facilities and switches back to
        a lower state when a problem is encountered.

asmlcuMonitorInit spawns asmlcuMonitor as a task.

asmlcuMonitorOff deletes the asmlcuMonitor task.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.17 asmlcuServer(3)

NAME

```
asmlcuServer,
asmlcuServerInit, asmlcuServerTask, asmlcuServerBreak, asmlcuServerKill
- Main Command Interpreter task of the asm module
```

SYNOPSIS

```
void asmlcuServer(ccsERROR *error)
ccsCOMPL_STAT asmlcuServerInit(ccsERROR *error)
void asmlcuServerBreak(int)
void asmlcuServerKill(int)
```

DESCRIPTION

```
asmlcuServer enters the LCC command interpreter using the corresponding
        CDT and CIT. Dedicated KILL and BREAK handlers are installed.

asmlcuServerInit spawns asmlcuServer as a task and checks for successful
        start-up.

asmlcuServerBreak is the BREAK handler. It sends STOP to all devices.

asmlcuServerKill is the KILL handler. It calls first asmlcuServerBreak,
        and then sends EXIT to all devices.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.18 asmlcuStart(3)

NAME

```
asmlcuStart - starts the asm modules
```

SYNOPSIS

```
ccsCOMPL_STAT asmlcuStart(IN dbSYMADDRESS *dbRoot, OUT ccsERROR *error)
```

DESCRIPTION

```
asmlcuStart initializes the asm module by the following steps:

- Create application semaphores
- Configure standard LCC database
- Configure I/O (including serial signals)
- Start all sub-modules
- Spawn asmlcuServer task (commmand interpreter)
- Spawn asmlcuMonitor task (supervisor daemon)

The function should be called from the LCU boot script.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

CAUTIONS

```
None
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:57
```

23.5.19 asmtelEnc(3)

NAME

```
asmtelEnc,
asmtelEncStart,
asmtelEncInit, asmtelEncOff, asmtelEncRead, asmtelEncVersion
- encoder-dependent interface to the axis module
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelEncStart(char                    *dbRoot,
                             acmAXIS_PRESET_POS_P    cbAxisPresetPosP,
                             acmAXIS_SEND_VEL_REF_P  cbAxisSendVelRefP,
                             acmAXIS_GET_STATE_P     cbAxisGetStateP,
                             acmAXIS_SIGNAL_CONF_P   acmEncSignalConf,
                             acmENC_P                acmEnc,
                             ccsERROR                *error)
ccsCOMPL_STAT asmtelEncInit(void *encData, ccsERROR *error)
ccsCOMPL_STAT asmtelEncOff(void *encData, ccsERROR *error)
ccsCOMPL_STAT asmtelEncRead(void        *data,
                            vltDOUBLE   *positionInRad,
                            vltLOGICAL  tracking,
                            ccsERROR    *error)
ccsCOMPL_STAT asmtelEncVersion(char *buffer, ccsERROR *error)
```

DESCRIPTION

```
asmtelEncStart is the entry point to the encoder-dependent part of
the asmtel module. This function is passed as parameter to the
axisInitAll, from which it will be called to initialize internal
database direct address array, init signals and set the pointers
to the rest of the encoder-dependent functions contained in this
file:

asmtelEncInit - attach and initialize encoder
asmtelEncOff - detach encoder
asmtelEncRead - read position from encoder
asmtelEncVersion - return the version-ID string of asmtel module
```

RETURN VALUES

```
SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.5.20 asmtelHw(3)

NAME

```
asmtelHw,
asmtelHwStart,
asmtelHwVersion, asmtelHwOn, asmtelHwOff,
asmtelWriteSpeed, asmtelCheckInterlock
- amplifier dependent interface to the axis module
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelHwStart(char                    *dbRoot,
                            acmAXIS_SEND_POS_REFS_P  cbAxisSendRefsP,
                            acmHW_P                  acmHw,
                            acmAXIS_SIGNAL_CONF_P    acmHwSignalConf,
                            ccsERROR                *error)

ccsCOMPL_STAT asmtelHwVersion(char *buffer, ccsERROR  *error)

ccsCOMPL_STAT asmtelHwOn(void *asmtelData, ccsERROR *error)

ccsCOMPL_STAT asmtelHwOff(void *asmtelData, ccsERROR *error)

ccsCOMPL_STAT asmtelWriteSpeed(void      *asmtelData,
                               vltDOUBLE  voltage,
                               vltDOUBLE *voltageWritten,
                               ccsERROR  *error)

ccsCOMPL_STAT asmtelCheckInterlock(void       *asmtelData,
                                   vltLOGICAL *interlockActive,
                                   ccsERROR   *error)

ccsCOMPL_STAT asmtelHwInit(void      *asmtelData,
                           ccsERROR  *error)

ccsCOMPL_STAT asmtelHwExit(void      *asmtelData,
                           ccsERROR  *error)
```

DESCRIPTION

```
asmtelHwStart - is the entry point to the aplifier dependent part of
the asmtel module. This function is passed as parameter to the
axisInitAll, from which it will be called to initialize internal
database direct address array, init signals and set the pointers
to the rest of the amplifier dependent functions contained in this
file:

asmtelHwVersion - return the version-ID string of asmtel module

asmtelHwOn - write zero-speed and enable the interlock chain.
If after powerOnWait seconds the amplifier does not power on
(ILPAPOWI set) then a problem exist in the hardware and the function
exit with an error.

asmtelHwOff - switch off the amplifier clearing the ILENABO signal.

asmtelHwWriteSpeed - write velocity-reference to amplifier signal.

asmtelHwCheckInterlock - check all interlock signals.
If an interlock is asserted, the enable signal ILENABO is negated.

asmtelHwInit - read configuration variables from the database,
               powerTimeOut and powerOnWait.

asmtelHwExit - empty.
```

FILES

```
asmtelHwDigital.config - digital signal configuration
asmtelHwAnalog.config - analog signal configuration
```

ENVIRONMENT

```
CONFIGPATH - search-path for config files
```

RETURN VALUES

```
SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.5.21 asmtelSelftst(3)

NAME

```
asmtelSelftst - perform self-test without movement
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelSelftst(asmiCmdArg_t *cmdArg, ccsERROR *error)
```

DESCRIPTION

```
Items to be checked:
 - Error substate asserted?
 - two limit-switches at the same time asserted?
 - control-loop tasks alive?
 - encoder I/F in consistent state?
 - if velocity reference <> 0: is motor current zero? (only if
   supported by amplifier)

 Allowed in any state.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.5.22 asmtelSetLims(3)

NAME

```
asmtelSetLims - set the software limits corresponding to the actual
                tracking zone.
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelSetLims(int         axisId,
                            ccsERROR   *error)
```

DESCRIPTION

```
The software limits define the valid tracking zone where the telescope
can point without refusing incoming commands. If the valid zone is
not keeped by the hardware the tracking is stopped. The checking duty
is performed by the tracking device.

 halfSky = ccsTRUE means the south half-sky
 halfSky = ccsFALSE means the north half-sky

The new software limits are updated in the database and in the
internal structure of axis using the axisUpdatePosLims function.
```

RETURN VALUES

```
 SUCCESS or FAILURE
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.5.23 asmtelStart(3)

NAME

```
asmtelStart - start up function of the asmtel module
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelStart(const dbSYMADDRESS dbRoot, ccsERROR *error)
```

DESCRIPTION

```
The start-up of the asmtel module consists of calls to axisInitAll
in order to spawn the axis-servers and control-loops.
This will also setup the database and configure the signals.
First is initialized the eta axis and then the theta one.
This means that the correspondence of the asmtelHwDataGlobal and
asmtelEncDataGlobal array entries is eta-theta in that order.
Using the defines ASMTEL_ETA_ID and ASMTEL_THETA_ID the following
can be asure in the hardware, encoder and SendRefs functions:

    asmtelHwDataGlobal[ASMTEL_ETA_ID]    : eta-hardware part data
    asmtelEncDataGlobal[ASMTEL_ETA_ID]   : eta-encoder part data

    asmtelHwDataGlobal[ASMTEL_THETA_ID] : theta-hardware part data
    asmtelEncDataGlobal[ASMTEL_THETA_ID]: theta-encoder part data

asmtelStart also read from the database the north and south limits.
This values are not read again during the INIT LCC-standard command.
```

RETURN VALUES

```
SUCCESS or FAILURE
```

CAUTIONS

```
The configuration of all the Telescope signals is performed from
the asmtelStart function, but the direct addres find out is done
by asmtelStart and the respective interface functions of the HW and
ENC parts. If the axisInitSignals function is change in some way
the compatibility with this module should be checked.
axisInitSignals configure the signals only if the file string
is valid and find out the DIRADDR if the numer of signals
(acmAxisSignalNumber) is not zero.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.5.24 asmtelTrk(3)

NAME

```
asmtelTrk,
asmtelTrkSendRefs, coordinateTransform
- interface between trk and axis modules
```

SYNOPSIS

```
ccsCOMPL_STAT asmtelTrkSendRefs(vltINT32                action,
                                ccsTIMEVAL              timeStamp,
                                acmSEND_REFS_SET_POINTS  setPoints,
                                vltINT32               *state,
                                ccsERROR               *error)
ccsCOMPL_STATE coordinateTransform(vltDOUBLE  alt,
                                   vltDOUBLE  az,
                                   vltDOUBLE *eta,
                                   vltDOUBLE *theta,
                                   ccsERROR  *error)
ccsCOMPL_STAT asmtelTrkOffs(vltDOUBLE offsetEta, vltDOUBLE offsetTheta,
                            msgMSG *msg, ccsERROR *error)
```

DESCRIPTION

```
asmtelTrkSendRefs receives new position set-points from the
tracking module in alt/az coordinates, converts them into the
eta/theta system, and forwards each component to the respective
axis servo-loop.

action:    IN  - target mode
timeStamp: IN  - UTC when to reach the specified reference position
setPoints: IN  - structure containing :
           posAlt:    IN  - altitude axis reference position in rad
           posAz:     IN  - azimuth axis reference position in rad
           posRot:    IN  - rotator axis reference (ignored)
           parallacticAngle:  IN  - parallactic angle (ignored)
state:     OUT - current substate
error:     OUT - error structure

coordinateTransform resolve altazimuthal coordinates into eta-theta
components. For the convertion a matrix operation is performed and
for that purpose the slaLib functions are used.

  R:   IN  - rotation matrix
alt:   IN  - altitud in radians
az:    IN  - azimuth in radians
eta:   OUT - eta DIMM angle in radians
theta: OUT - theta DIMM angle in radians

asmtelTrkOffs is the TELOFFS command code. It adds the offsetEta and
offsetTheta values to the offset accumulator of each axis.
The offset accumulator is added to the setpoints forwarded to the
respective axis server:

ref.pos = <axis>RefPos + asmtelAxesData.offset<Axis>;

in this way autoguiding corrections could be performed directly without
making a convertion to the altaz system neither sending them to the
trk server through the OFFSAA command.

A semaphore is used to be sure that both offsets are added at the same
time to the respective setpoint.

The trk function trkGetStatus is used to get the current status of
the trk server. This is necesary to check the successful or failful
completion of the command.

offsetEta:      IN  - eta offset in arcsecond
offsetTheta:    IN  - theta offset in arcsecond
```

```
msg:            OUT - msgMSG structure of the received command
error:          OUT - error structure
```

RETURN VALUES

```
SUCCESS or FAILURE
```

CAUTIONS

```
Note that only the last reference is kept when this routine is called
several times during the current reference interval is processed.
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:55
```

23.6 COMMANDS AND PROCESSES

23.6.1 asmwsControl

```
--->BEGIN IMPORTED FILE: ../CDT/asmwsControl.cdt  (97/6/21  4:58)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmwsControl.cdt,v 1.34 1997/06/20 23:32:39 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- ----------------------------------------------
// S.Sandrock 07.11.1996 created
//

//***********************************************************************
PUBLIC_COMMANDS

//=======================================================================
// ASM specific commands
//=======================================================================

COMMAND=  START

FORMAT= A

PARAMETERS=
    PAR_NAME=    mode
    PAR_UNIT=
    PAR_TYPE=    STRING
    PAR_RANGE=   ENUM "night" , "day" , "auto"
    PAR_OPTIONAL= YES
    PAR_DEF_VAL= "auto"

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"


HELP_TEXT=
Starts the automatic operation of the Astronomical Site Monitor.

Depending on time of day and external conditions it brings the system to
either Online or Standby state, which is furtheron automatically updated.

The automatic operation can be interrupted by an explicit ONLINE or STANDBY
command, and resumed after that by another START command.
@
//-----------------------------------------------------------------------

COMMAND= CENTER

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    alpha
    PAR_UNIT=    HMS
    PAR_TYPE=    REAL
    PAR_RANGE=   INTERVAL MIN=0.0;MAX=240000.0
    PAR_DEF_VAL=

    PAR_NAME=    delta
    PAR_UNIT=    DMS
    PAR_TYPE=    REAL
    PAR_RANGE=   INTERVAL MIN=-900000.0;MAX=900000.0
```

```
    PAR_DEF_VAL=

    PAR_NAME=    num
    PAR_TYPE=    INTEGER
    PAR_RANGE=   INTERVAL MIN=1;MAX=100
    PAR_DEF_VAL=

REPLY_FORMAT= A

HELP_TEXT=
Centers the given star in the field of vision of the camera. First send a
OBJSTAR command to the tracking server, waits until the telescope is tracking
and then initiate a finit loop of at maximum `num' iterations, giving in each
one offset corrections to the telescope. If the calculated error vector is
`small' enough then the loop it's stopped and an OK reply is send back. In any
other case and error reply is sent back.
@

//=======================================================================
// Standard commands
//=======================================================================

COMMAND=  PING

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Get lifesign from ASM.
@
//-----------------------------------------------------------------------


COMMAND=  STATE

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "ONLINE"

HELP_TEXT=
Get state of ASM.
@
//-----------------------------------------------------------------------


COMMAND=  STATUS

SYNONYMS= getStatus, GSTS

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
```

```
    PAR_TYPE=    STRING

HELP_TEXT=
Get complete status of ASM.
@
//-----------------------------------------------------------------------



//***********************************************************************
MAINTENANCE_COMMANDS

//-----------------------------------------------------------------------
COMMAND= INIT

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Initializes the ASM system.

It aborts all operation and sets the system to Loaded state,
and reads configuration parameters from the database.
@
//-----------------------------------------------------------------------

COMMAND= EXIT

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Perform a shutdown by terminating the process.
Brings the system to non-operational state.
@
//-----------------------------------------------------------------------

COMMAND= STOP

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"
```

```
HELP_TEXT=
Stop any hardware movement. Any other pending command is canceled
sending to the originator an error reply. After this the LCCSTOP
command is send to the LCU environments.
The final state of the asmwsControl process depends on its original state:
- Loaded and Stand-by: no change of the asmwsControl state
- Online: Idle
@
//----------------------------------------------------------------------

COMMAND= ONLINE

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Set the system explcitly into Online state.

This interrupts the automatic operation of the ASM, and stays in
Online state as long as no other state is explcitly requested.

The automatic operation can be resumed by the START command.
@
//----------------------------------------------------------------------

COMMAND= STANDBY

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Set the system explcitly into Standby state.

This interrupts the automatic operation of the ASM, and stays in
Standby state as long as no other state is explcitly requested.

The automatic operation can be resumed by the START command.
@
//----------------------------------------------------------------------

COMMAND= OFF

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"
```

```
HELP_TEXT=
Set the system explcitly into Loaded state.
@
//----------------------------------------------------------------------

COMMAND=  VERBOSE

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    onOff
    PAR_UNIT=
    PAR_TYPE=    STRING
    PAR_RANGE=   ENUM "On" , "Off"
    PAR_DEF_VAL= "Off"

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Switch on or off verbose mode.
@
//----------------------------------------------------------------------

COMMAND=  VERSION

SYNONYMS= GetVersion

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "@(#) $Id: asmwsControl.cdt,v 1.34 1997/06/20 23:32:39 vltsccm Exp $"

HELP_TEXT=
Get version of asm module.
@

//----------------------------------------------------------------------

COMMAND= BREAK

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Stop action on the ASM.
@
//----------------------------------------------------------------------
```

```
COMMAND= KILL

SYNONYMS=

FORMAT= A

PARAMETERS=

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Kill the process. Identical to EXIT.
@

//*********************************************************************
TEST_COMMANDS

//---------------------------------------------------------------------

COMMAND= SELFTST

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    detId
    PAR_TYPE=    INTEGER
    PAR_OPTIONAL=YES

    PAR_NAME=    function
    PAR_TYPE=    STRING
    PAR_OPTIONAL= YES
    PAR_MAX_REPETITION=999


REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Perform a self-test of the software module
@
//---------------------------------------------------------------------

COMMAND= SIMULAT

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    detId
    PAR_TYPE=    INTEGER
    PAR_OPTIONAL=YES

    PAR_NAME=    function
    PAR_TYPE=    STRING
    PAR_OPTIONAL= YES
    PAR_MAX_REPETITION=999

REPLY_FORMAT= A
REPLY_PARAMETERS=
```

```
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Switch on simulation mode
@
//-----------------------------------------------------------------------

COMMAND= STOPSIM

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    detId
    PAR_TYPE=    INTEGER
    PAR_OPTIONAL=YES

    PAR_NAME=    function
    PAR_TYPE=    STRING
    PAR_OPTIONAL= YES
    PAR_MAX_REPETITION=999

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Switch off simulation mode
@

//-----------------------------------------------------------------------

COMMAND= ASMTEST

SYNONYMS=

FORMAT= A

PARAMETERS=
    PAR_NAME=    function
    PAR_TYPE=    STRING
    PAR_RANGE=   ENUM "selectStar"
    PAR_OPTIONAL= NO

    PAR_NAME=    funcArg
    PAR_TYPE=    STRING
    PAR_OPTIONAL= YES
    PAR_MAX_REPETITION=999

REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME=    done
    PAR_TYPE=    STRING
    PAR_DEF_VAL= "OK"

HELP_TEXT=
Perform a specific test of the software module
@

//-----------------------------------------------------------------------

COMMAND= DUMMY

SYNONYMS=
```

```
FORMAT= A

PARAMETERS=
        PAR_NAME=    detId
        PAR_TYPE=    INTEGER
        PAR_OPTIONAL=YES

REPLY_FORMAT= A

HELP_TEXT=
Dummy command, send the DUMMY command to asmlcuServer.
@
// ___oOo___
--->END IMPORTED FILE: ../CDT/asmwsControl.cdt  (97/6/21  4:58)
```

23.6.2 asmlcuServer

```
--->BEGIN IMPORTED FILE: ../CDT/asmlcuServer.cdt  (97/6/21  4:57)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmlcuServer.cdt,v 1.30 1997/06/05 11:44:24 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 31.10.1996 created
//

PUBLIC_COMMANDS

//===========================================================================
// LCC standard commands
//===========================================================================

COMMAND=  INIT
SYNONYMS= InitDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
INIT initializes the specified device and sets its state to Loaded.
@

COMMAND=  STANDBY
SYNONYMS= StandbyDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
STANDBY brings the specified device to state Stand-By.
@

COMMAND=  ONLINE
SYNONYMS= OnlineDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
ONLINE brings the specified device to state On-Line.
@

COMMAND=  OFF
SYNONYMS= OffDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
```

```
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
OFF brings the specified device to state Loaded.
@

COMMAND=  EXIT
SYNONYMS= ExitDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
EXIT brings the specified device to state Off.
@

COMMAND=  STOP
SYNONYMS= StopDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
STOP terminates any movement of the specified device.
@

COMMAND=  SELFTST
SYNONYMS= SelftestDevice
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
SELFTST verifies the specified device without moving anything.
@

COMMAND=  SIMULAT
SYNONYMS= Simulation
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
SIMULAT enables simulation-mode for the specified device,
in which all standard commands are intercepted.
@

COMMAND=  STOPSIM
SYNONYMS= StopSimulation
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
```

```
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
//  PAR_DEF_VAL= ""
REPLY_FORMAT= A
HELP_TEXT=
STOPSIM disables simulation-mode for the specified device,
@

COMMAND=  BREAK
SYNONYMS= BreakServer
FORMAT= A
PARAMETERS=
REPLY_FORMAT= A
HELP_TEXT=
Breaks the asmlcuServer.
The STOP command is forwarded to all devices controlled by asmlcuServer.
@

COMMAND=  KILL
SYNONYMS= KillServer
FORMAT= A
PARAMETERS=
REPLY_FORMAT= A
HELP_TEXT=
Kills the asmlcuServer.
After implicit call of BREAK, the EXIT command is forwarded to all
devices controlled by asmlcuServer. Then asmlcuServer itself exits.
@

COMMAND=  PING
SYNONYMS= PingServer
FORMAT= A
PARAMETERS=
REPLY_FORMAT= A
HELP_TEXT=
PING returns an OK reply if the asmlcuServer is alive.
@

//===========================================================================
// General asmlcuServer commands
//===========================================================================

COMMAND=  STATE
SYNONYMS= GetState
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
    PAR_DEF_VAL= "all"
REPLY_FORMAT= A
REPLY_PARAMETERS=
    PAR_NAME= device
    PAR_UNIT=
    PAR_TYPE= STRING
    PAR_NAME= state
    PAR_UNIT=
    PAR_TYPE= STRING
    PAR_NAME= mode
    PAR_UNIT=
    PAR_TYPE= STRING
HELP_TEXT=
Returns the state and simulation-mode setting for the
specified device, or all devices controlled by asmlcuServer.

    Off:        the software device is not operational.

    Loaded:     the software is loaded and initialized,
```

```
                the hardware is ready to be initialized.

    Stand-By:   the software is loaded and initialized,
                the hardware is initialized, but in
                standby (applicable parts switched off,
                brakes clamped, etc.)

    On-Line:    the software is loaded and initialized,
                the hardware is initialized and fully
                operational.
@

COMMAND=  VERBOSE
SYNONYMS= Verbose
FORMAT= A
PARAMETERS=
    PAR_NAME=    device
    PAR_UNIT=
    PAR_TYPE=    STRING
//  PAR_RANGE=   ENUM
    PAR_DEF_VAL= "all"

    PAR_NAME=    what
    PAR_UNIT=
    PAR_TYPE=    STRING
    PAR_RANGE=   ENUM "get","on","off"
    PAR_DEF_VAL= "get"
REPLY_FORMAT= A
HELP_TEXT=
Enables or diables verbose-mode for the specified device,
or all devices controlled by asmlcuServer.
Verbose-mode gives multiple replies to commands.
@

COMMAND=  VERSION
SYNONYMS= GetVersion
FORMAT= A
PARAMETERS=
REPLY_FORMAT= A
HELP_TEXT=
Return the version-IDs of all loaded asm modules.
@

//=============================================================================
// Public ASM specific commands
//=============================================================================


//=============================================================================

MAINTENANCE_COMMANDS

//=============================================================================
// Protected ASM specific commands
//=============================================================================


//=============================================================================

TEST_COMMANDS

//=============================================================================
COMMAND=  DUMM
SYNONYMS=
FORMAT= A
PARAMETERS=
REPLY_FORMAT= A
HELP_TEXT=
Dummy command. DO NOT USE THIS COMMAND.
```

```
@

//_oOo_
--->END IMPORTED FILE: ../CDT/asmlcuServer.cdt  (97/6/21  4:57)





--->BEGIN IMPORTED FILE: ../CIT/asmlcuServer.cit  (97/6/21  4:57)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: asmlcuServer.cit,v 1.30 1997/06/05 11:44:24 vltsccm Exp $"
//
// who        when       what
// ---------- ---------- -------------------------------------------------
// S.Sandrock 31.10.1996 created
//

//===========================================================================
// LCC standard commands
//===========================================================================

INIT,   IasmlcuCmdInit,         FUNCTION
STANDBY,IasmlcuCmdStandby,      FUNCTION
ONLINE, IasmlcuCmdOnline,       FUNCTION
OFF,    IasmlcuCmdOff,          FUNCTION
EXIT,   IasmlcuCmdExit,         FUNCTION
STOP,   IasmlcuCmdStop,         FUNCTION
SELFTST,IasmlcuCmdSelftst,      FUNCTION
SIMULAT,IasmlcuCmdSimulat,      FUNCTION
STOPSIM,IasmlcuCmdStopsim,      FUNCTION

//===========================================================================
// General
//===========================================================================

STATE,  IasmlcuCmdState,        FUNCTION
VERBOSE,IasmlcuCmdVerbose,      FUNCTION
VERSION,IasmlcuCmdVersion,      FUNCTION

//===========================================================================
// Test (remove at the end of the development)
//===========================================================================
DUMM,IasmlcuCmdDummy,           TASK

//___oOo___
--->END IMPORTED FILE: ../CIT/asmlcuServer.cit  (97/6/21  4:57)
```

23.6.3 asmtrkServer

Basically identical to standard trkServer.

23.6.4 etaServer

Basically identical to standard axisServer.

23.6.5 thetaServer

Basically identical to standard axisServer.

23.7 ERRORS

23.7.1 asmcat_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmcat_ERRORS  (97/6/21  4:52)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  May  4 22:13:40 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################



10 F {}
asmcatERR_INTERNAL : Failure when calling %.40s.
noHelp
20 F {}
asmcatERR_GETVALUE : GetValue fails when reading the database (%.20s).
noHelp
30 F {}
asmcatERR_CANNOT_SELECT : Cannot select a suitable guide star.
noHelp
40 W {}
asmcatERR_FORMAT : The stars list entry %.20s contains an ilegal format (%.20s).
noHelp
50 F {}
asmcatERR_NOVAS : Unknow fail on Novas call: %20.
noHelp
#
# "@(#) $Id: asmcat_ERRORS,v 1.12 1997/06/12 12:22:07 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmcat_ERRORS  (97/6/21  4:52)
```

23.7.2 asmdim_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmdim_ERRORS  (97/6/21  4:54)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  Jun 20 17:19:28 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################



10 F {}
asmdimERR_INTERNAL : internal error after: %80s
asmdimERR_INTERNAL.hlp
30 F {}
asmdimERR_DB_SIZE_MISMATCH : %.20s: internal size mismatch of struct to corresponding DB attribute
asmdimERR_DB_SIZE_MISMATCH.hlp
40 F {}
asmdimERR_CCDIP_WINNUM : Got unspected sub-array index (%d) from CCDIP; expected: %.20s
asmdimERR_CCDIP_WINNUM.hlp
41 S {}
asmdimERR_CCDIP_IQE : image quality estimation in CCDIP failed; FWHM may be trash: x/y %g %g
asmdimERR_CCDIP_IQE.hlp
42 W {}
asmdimERR_CCDIP_FOR_WINDOW : image analysis failed for %.20s window
asmdimERR_CCDIP_FOR_WINDOW.hlp
43 W {}
asmdimERR_CCDIP_FOR_EXPOSURE : exposure evaluation failed (%.20s)
asmdimERR_CCDIP_FOR_EXPOSURE.hlp
51 S {}
asmdimERR_FRAME_FWHM_INVALID : FWHM provided by CCDIP is invalid: spot #%d fx=%g fy=%g
asmdimERR_FRAME_FWHM_INVALID.hlp
52 W {}
asmdimERR_FRAME_FWHM_ELONGATED : FWHM is elongated in x/y: spot #%d ratio=%3.3f%% limit=%3.3f%% fx=%g
fy=%g
asmdimERR_FRAME_FWHM_ELONGATED.hlp
53 W {}
asmdimERR_FRAME_DECENTERED : image too close to edge of frame: distance x=%g y=%g, limit x=%g y=%g
[pixels]
asmdimERR_FRAME_DECENTERED.hlp
54 W {}
asmdimERR_FRAME_LOW_SNR : signal/noise ratio too low: spot #%d actual=%g limit=%g
asmdimERR_FRAME_LOW_SNR.hlp
55 W {}
asmdimERR_FRAME_VALIDATION : validation of exposure #%d failed: x1=%g y1=%g x2=%g y2=%g A1=%g A2=%g
xg=%g yg=%g
asmdimERR_FRAME_VALIDATION.hlp
59 S {}
asmdimERR_EXPOS_FAILED : exposure sequence failed
asmdimERR_EXPOS_FAILED.hlp
60 S {}
asmdimERR_CALC_PARAM : invalid parameter during calculation: name=%.20s value=%g
asmdimERR_CALC_PARAM.hlp
66 S {}
asmdimERR_CALC_STAT_VALIDATION : [ref 6.6] too many rejected exposures: total=%d validated=%d reject-
ed=%g%% limit=%g%%
asmdimERR_CALC_STAT_VALIDATION.hlp
68 S {}
asmdimERR_CALC_SEEING_VALIDATION : [ref 6.8] two FWHMs are not equal within limit: FWHMpara=%g FWHM-
perp=%g actual=%g limit=%g
asmdimERR_CALC_SEEING_VALIDATION.hlp
81 F {}
asmdimERR_CONFIG_Nd : illegal configuration of Nd: %d, maxmimum: %d
asmdimERR_CONFIG_Nd.hlp
```

```
110 W {}
asmdimERR_AG_NO_STAR : autoguider found no valid star in the frame (got from CCD: ipCenVal=%g ipXCen=%g
ipYCen=%g)
asmdimERR_AG_NO_STAR.hlp
#
# "@(#) $Id: asmdim_ERRORS,v 1.38 1997/06/20 15:25:05 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmdim_ERRORS  (97/6/21  4:54)
```

23.7.3 asmi_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmi_ERRORS  (97/6/21  4:52)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  Jun 13 10:01:28 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################


define asmiErrOffset 0

1 F asmiErrOffset
asmiERR_INTERNAL : Software error, contact SW support, SPR required (%.70s)
noHelp
2 S asmiErrOffset
asmiERR_VXWORKS : system call failed: %.20s (%.20s)
noHelp
3 F asmiErrOffset
asmiERR_MEMORY : cannot allocate memory (%d bytes)
noHelp
4 S asmiErrOffset
asmiERR_DRIVER : driver call to %.20s failed (%.20s)
noHelp
5 W asmiErrOffset
asmiERR_PARAMETER : invalid parameter %.20s (value = %.20s)
noHelp
6 W asmiErrOffset
asmiERR_FILE : failed to %.20s file "%.20s" (%.20s)
noHelp
10 S asmiErrOffset
asmiERR_TASK_SPAWN : failed to spawn task %.20s
noHelp
11 S asmiErrOffset
asmiERR_TASK_DELETE : failed to delete task %.20s (TID=%x)
noHelp
12 S asmiErrOffset
asmiERR_TASK_FAILED : failed to start up task %.20s (task's error-stack attached)
noHelp
13 S asmiErrOffset
asmiERR_TASK_TERMINATED : task %.20s terminated immediately after being spawned
noHelp
14 S asmiErrOffset
asmiERR_SEND_CMD : Cannot send command (env: %.20s, proc: %.20s, cmd: %.20s).
noHelp
15 W asmiErrOffset
asmiERR_CMD_CANCELED : Command %.20s canceled.
noHelp
16 W asmiErrOffset
asmiERR_CMD_ACTIVE : Command %.20s rejected because the previous one it's still active.
noHelp
17 S asmiErrOffset
asmiERR_SEND_REPLY : Failed to send %.20s reply to %.20s command.
noHelp
18 W asmiErrOffset
asmiERR_RECOVER : At least one error took place but still alive (last call: %40).
noHelp
19 S asmiErrOffset
asmiERR_CMD_ATTACH : Cannot attach the %.20s command.
noHelp
20 S asmiErrOffset
asmiERR_CMD_DETACH : Cannot detach the %.20s command.
noHelp
21 W asmiErrOffset
```

```
asmiERR_CMD_NOT_ACTIVE : Command %.20s not active.
noHelp
22 W asmiErrOffset
asmiERR_CMD_NOT_ATTACHED : Command %.20s not attached.
noHelp
23 W asmiErrOffset
asmiERR_DEV_CMD : Device command fails.
noHelp
24 W asmiErrOffset
asmiERR_DEV_RETURN : Return method fails.
noHelp
25 W asmiErrOffset
asmiERR_CMDS_ACTIVE : This asmiCMD_SEND is pending in the %.20s command.
noHelp
26 S asmiErrOffset
asmiERR_SWITCH_STATE : Failed to switch state to %.20s.
noHelp
27 S asmiErrOffset
asmiERR_SWITCH_SUBSTATE : Failed to switch substate to %.20s.
noHelp
28 W asmiErrOffset
asmiERR_SEM_TIMEOUT : Semaphore %.20s timed out.
noHelp
#
# "@(#) $Id: asmi_ERRORS,v 1.22 1997/06/20 23:30:05 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmi_ERRORS  (97/6/21  4:52)
```

23.7.4 asmlcu_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmlcu_ERRORS  (97/6/21  4:57)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  Jun  4 19:05:20 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################


define asmlcuErrOffset 0

10 W asmlcuErrOffset
asmlcuERR_NOT_SUPPORTED : This function is not supported for device %.20s
noHelp
20 F asmlcuErrOffset
asmlcuERR_SERVER_START : start-up of server task %.20s failed with error code %d
noHelp
30 F asmlcuErrOffset
asmlcuERR_MODULE_START : start-up of module %.20s failed
noHelp
40 F asmlcuErrOffset
asmlcuERR_FILE_PARSE : file %.20s, line %d: invalid %.20s (value: %.20s)
noHelp
41 F asmlcuErrOffset
asmlcuERR_FATAL : Fatal internal error: %.40s
noHelp
#
# "@(#) $Id: asmlcu_ERRORS,v 1.30 1997/06/05 11:44:32 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmlcu_ERRORS  (97/6/21  4:57)
```

23.7.5 asmtel_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmtel_ERRORS  (97/6/21  4:56)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  Jun 13 11:24:41 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################


define asmtelErrOffset 0

10 W {}
asmtelERR_INSTANCES : More than two asmtel instances is not allowed (%.20s).
noHelp
11 W {}
asmtelERR_IOCONFIG : Failed to config %.20s I/O signals.
noHelp
12 W {}
asmtelERR_IOGETDADDR : Failed to get direct address of signals in %.20s.
noHelp
20 W {}
asmtelERR_DBACCESS : Error when accessing database: %.20s.
noHelp
30 W {}
asmtelERR_INTERNAL : Internal error return in call to %.20s.
noHelp
40 W {}
asmtelERR_SENDREFS : asmtel send references failure: %.20s.
noHelp
50 W {}
asmtelERR_NOBIND : No binding function defined: %.20s.
noHelp
60 W {}
asmtelERR_COORD : Unresolved coordinates transform %.20s.
noHelp
70 W {}
asmtelERR_TASK_TIMEOUT : Timeout while waiting for task %.20s.
noHelp
80 S {}
asmtelERR_ENC_READ : encoder read error (ikon status: %d, IK320 position status: 0x%x)
noHelp
90 W {}
asmtelERR_ENC_VEL : Cannot move %.20s axis at velocity reference (%.20s)
noHelp
100 S asmtelErrOffset
asmtelERR_ENC_INIT : encoder init failed (%.20s, %.20s)
noHelp
110 S {}
asmtelERR_ENC_REF_MOVE : Failed to move to encoder reference point (ikon status: %d)
noHelp
120 W {}
asmtelERR_ENC_INIT_STOP : Servo task switched to axisServoStateHwOff (%.20s axis).
noHelp
130 W {}
asmtelERR_ENC_UPDATE_LIMS : axisUpdatePosLims(%.20s) return FAILURE.
noHelp
150 W {}
asmtelERR_HW_EP_SELECTED : while the engineer panel is selected the amplifier refuse the power-on
(%.20s)
noHelp
160 W {}
asmtelERR_HW_TIMEOUT : Time out while waiting for the power-%.20s of the amplifier.
noHelp
```

```
200 W {}
asmtelERR_NO_HALFSKY : Was not possible to determine the half-sky position.
noHelp
210 W {}
asmtelERR_SET_POSLIMS : Cannot set in the database the posLims for the half-sky.
noHelp
220 W {}
asmtelERR_TRK_TELOFFS : Command %.20s only allowed when TRACKING.
asmtelERR_TRK_TELOFFS.hlp
230 W {}
asmtelERR_TRK_TELOFFS_ERR : Error while waiting for completion of TELOFFS command (%.20s).
noHelp
500 W {}
asmtelERR_TST_FAIL : test of function %.20s fail.
noHelp
#
# "@(#) $Id: asmtel_ERRORS,v 1.35 1997/06/20 15:34:44 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmtel_ERRORS  (97/6/21  4:56)
```

23.7.6 asmws_ERRORS

```
--->BEGIN IMPORTED FILE: ../ERRORS/asmws_ERRORS  (97/6/21  4:58)
#
# Version Id. is specified at the end of the file
#
# Error Definition File    Created on  May 29 12:41:21 1997
#
# This file has been generated by a utility
#
#  !!!!!!!!!!!  DO NOT MANUALLY EDIT THIS FILE  !!!!!!!!!!!
#
###########################################################


define asmwsErrOffset 0

10 W asmwsErrOffset
asmwsERR_NOT_SUPPORTED : This function is not supported for device %.20s
noHelp
20 F asmwsErrOffset
asmwsERR_SERVER_START : start-up of server task %.20s failed with error code %d
noHelp
30 F asmwsErrOffset
asmwsERR_MODULE_START : start-up of module %.20s failed
noHelp
40 F asmwsErrOffset
asmwsERR_FILE_PARSE : file %.20s, line %d: invalid %.20s (value: %.20s)
noHelp
41 F asmwsErrOffset
asmwsERR_FATAL : Fatal internal error: %.40s
noHelp
42 S asmwsErrOffset
asmwsERR_LCU : At least one of the LCU's failed to %.20s
noHelp
43 S asmwsErrOffset
asmwsERR_SWITCH_STATE : Failed to switch state to %.20s
noHelp
44 S asmwsErrOffset
asmwsERR_TIMEOUT : Timeout expired for %.20s command.
noHelp
45 S asmwsErrOffset
asmwsERR_SEND_CMD : Cannot send %.20s command to LCU's
nohelp
46 W asmwsErrOffset
asmwsERR_INVALID_STATE : Command %.20s not allowed in current state %.20s
noHelp
47 W asmwsErrOffset
asmwsERR_SEND_REPLY : Failed to send %.20s reply to %.20s command to %.20s
noHelp
48 W asmwsErrOffset
asmwsERR_BUSY : Cannot execute %.20s cmd, still busy with %.20s cmd
noHelp
49 W asmwsErrOffset
asmwsERR_CMD_STOPPED : Command %.20s canceled because %.20s command received.
noHelp
50 W asmwsErrOffset
asmwsERR_ERROR : And error condition occurs when processing the %.20s command.
noHelp
51 W asmwsErrOffset
asmwsERR_TEST : Internal error while executing test (%.20s).
noHelp
52 S asmwsErrOffset
asmwsERR_ERROR_REPLY : Command %.20s got an error reply.
noHelp
53 W asmwsErrOffset
asmwsERR_ARGS : Cannot parse arguments to the %.20s command.
noHelp
54 W asmwsErrOffset
```

```
asmwsERR_CENTER : Cannot center the star after %.20s autoguiding corrections.
noHelp
55 W asmwsErrOffset
asmwsERR_ERROR_DISCARD : Error reply to the command %.20s discarded.
noHelp
56 S asmwsErrOffset
asmwsERR_DEV_CALL : A call to the %.20s method of the object %.20s fails.
noHelp
#
# "@(#) $Id: asmws_ERRORS,v 1.34 1997/06/20 23:32:39 vltsccm Exp $"

--->END IMPORTED FILE: ../ERRORS/asmws_ERRORS  (97/6/21  4:58)
```

23.8 DATABASE CONFIGURATION

23.8.1 wasm

23.8.1.1 DATABASE.db

```
--->BEGIN IMPORTED FILE: wasm/dbl/DATABASE.db  (97/6/24 10:48)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: DATABASE.db,v 1.31+ 1997/06/23 13:14:38 vltsccm Exp vltsccm $"
//
// who       when      what
// --------  --------  ----------------------------------------------
//

//*****************************************************************
//    CCS DEFINITIONS
//*****************************************************************

//
// Loads classes definition from standard file
//
#include "CCS.db"

//
// Loads points for CCS
//
#include "USER.db"


//*****************************************************************
//    APPLICATION DEFINITIONS
//*****************************************************************

#include "ASM.db"       // all ASM branch

#include "lcustat.db"   // LCU status display

// ___oOo___
--->END IMPORTED FILE: wasm/dbl/DATABASE.db  (97/6/24 10:48)
```

23.8.1.2 USER.db

```
--->BEGIN IMPORTED FILE: wasm/dbl/USER.db  (97/6/24 10:48)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: USER.db,v 1.31+ 1997/06/23 13:14:38 vltsccm Exp vltsccm $"
//
// who       when      what
// --------  --------  ----------------------------------------------
//

#include "CCS.class"

//***************************************************************************
//   SCAN SYSTEM points
//***************************************************************************

#include "asmiSCANDEV.class"

// lasm0: only used for test in simulation
POINT "asmiSCANDEV" "ccs_config:scan config:LAN:lasm0"
BEGIN
        Alias   "lasm0"
```

```
END

// lasm1: ASM general control
POINT "asmiSCANDEV" "ccs_config:scan config:LAN:lasm1"
BEGIN
        Alias   "lasm1"
END

// lasm2: DIMM CCD control
POINT "asmiSCANDEV" "ccs_config:scan config:LAN:lasm2"
BEGIN
        Alias   "lasm2"
END

//
//   SAMPLING PLOTTING  points
//
// If you need more plots, just duplicate this point, giving:
// - new point name
// - new alias
//

POINT PLOT  "ccs_config:plot config:samplePlot"
BEGIN
        Alias "samplePlot"
END

// ___oOo___
--->END IMPORTED FILE: wasm/dbl/USER.db  (97/6/24 10:48)
```

23.8.1.3 ASM.db

```
--->BEGIN IMPORTED FILE: wasm/dbl/ASM.db  (97/6/24 10:48)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: ASM.db,v 1.31+ 1997/06/23 13:14:39 vltsccm Exp vltsccm $"
//
// who       when      what
// --------  --------  ----------------------------------------------
//

#include "asmws.db"

// ___oOo___
--->END IMPORTED FILE: wasm/dbl/ASM.db  (97/6/24 10:48)
```

23.8.1.4 DB.tree

```
--->BEGIN IMPORTED FILE: ../doc/wasm_DB.tree  (97/6/24 10:48)
  |--users config
  |   |--rtap
  |   |--ccsuser
  |--ccs_config
  |   |--historian config
  |   |--history point
  |   |--schedule config
  |   |--alarm class
  |   |   |--alrmccs_alarm
  |   |--scan config
  |   |   |--LAN
  |   |   |   |--lasm0
  |   |   |   |   |--SCAN_ALARM
  |   |   |   |--lasm1
```

```
|  |  |  |  |--SCAN_ALARM
|  |  |  |--lasm2
|  |  |  |  |--SCAN_ALARM
|  |--cmd_table
|  |--xntpd
|  |--plot config
|  |  |--plot_table
|  |  |--samplePlot
|  |  |  |--plot_table
|  |--class config
|  |--BASE_CLASS
|  |  |--alrmSTATE_ALARM
|  |  |  |--alrmCCS_ALARM
|  |  |--evhDB_TASK
|  |  |  |--evhSTD_COMMANDS
|  |  |  |  |--asmiTASK
|  |  |  |  |  |--asmwsCONTROL
|  |  |--asmiCLASS
|  |  |  |--asmiDEVICE
|  |  |  |  |--asmcatDEVICE
|  |  |  |  |--asmdimDEVICE
|  |  |  |  |--asmmetDEVICE
|  |  |  |  |--asmtelDEVICE
|  |  |  |--asmiBRANCH
|  |  |  |  |--asmwsBRANCH
|  |  |  |--asmiDATA
|  |  |  |  |--asmdimDATA
|  |  |  |  |--asmmetDATA
|  |  |  |--asmiCOMWS
|  |  |  |  |--asmmetCOM
|  |  |  |--asmdimCONFIG
|  |  |  |--asmdimAG
|  |  |  |--asmdimEXPOS
|  |  |  |--asmdimSEQ
|  |  |  |--asmwsDEVICES
|  |  |  |--asmwsPUBLIC
|  |  |--evhDB_COMMAND
|  |  |  |--evhDB_LIST_COMMAND
|  |  |--ccdAPPLICATION
|  |  |  |--ccdLCUWSAPPLICATION
|  |  |  |  |--ccdCAMERADCS
|  |  |  |  |  |--ccdCAMERA
|  |  |  |  |  |  |--asmdimCAMERA
|  |  |  |--ccdSUBSYSTEM
|  |  |  |  |--ccdIMAGESUBSYSTEM
|  |  |  |  |  |--ccdIMAGEOPTSUBSYS
|  |  |  |  |  |  |--ccdIMAGEPROCESS
|  |  |  |  |  |--ccdIMAGETRANSFER
|  |  |  |  |  |--ccdIMAGEHWSUBSYSTEM
|  |  |  |  |  |  |--ccdREADOUT
|  |  |  |  |--ccdHWSUBSYSTEM
|  |  |  |  |  |--ccdOPTHWSUBSYSTEM
|  |  |  |  |  |  |--ccdSHUTTER
|  |  |  |  |  |  |--ccdTEMPERATURE
|  |  |  |  |  |  |--ccdTELEMETRY
|  |  |--ccdEXPGENERAL_SET
|  |  |  |--ccdEXPOSURE_SETUP
|  |  |  |--ccdEXPGENERAL
|  |  |  |  |--ccdEXPOSURE
|  |  |--ccdEXPRDTGEN_SET
|  |  |  |--ccdEXPREADOUT_SET
|  |  |  |--ccdEXPRDTGEN
|  |  |  |  |--ccdEXPOSUREREADOUT
|  |  |--ccdEXPWINRDT_SET
|  |  |--ccdEXPTRANSFER_SET
|  |  |  |--ccdEXPOSURETRANSFER
|  |  |--ccdEXPPRCGEN_SET
|  |  |  |--ccdEXPPROCESS_SET
|  |  |  |  |--ccdEXPOSUREPROCESS
```

```
|   |   |   |--ccdEXPPRCGEN
|   |   |--ccdEXPWINPRC_SET
|   |   |--ccdEXPDISPLAY_SET
|   |   |   |--ccdEXPOSUREDISPLAY
|   |   |--ccdEXPOSURETEMP
|   |   |--ccdEXPOSURETELE
|   |   |--ccdEXPOSURES
|   |   |--ccdPROCESS_WINDOW
|   |   |--ccdIMAGES
|   |   |--ccdDEVICE
|   |   |   |--ccdDEVICE_OPT
|   |   |   |   |--ccdOUTPUT_AMPLIFIER
|   |   |   |   |--ccdDETECTOR_COMP
|   |   |   |   |--ccdCRYOSTAT
|   |   |--ccdCHIP
|   |   |   |--ccdDETECTOR
|   |   |--ccdCLOCK_PATTERN
|   |   |--ccdACE
|   |   |--ccdWORLDCOORD
|   |   |--ccdOBSERVATIONS
|   |   |--lcustat
|--Appl_data
|   |--lcustat
|--PARAMS
|   |--SCALARS
|   |--TABLES
|   |--VECTORS
|--ASM
|   |--Control
|   |   |--stdCmdTargets
|   |   |--lcuCmdTarget
|   |   |--trkCmdTarget
|   |   |--ccdCmdTarget
|   |--Devices
|   |   |--Catalog
|   |   |--Dimm
|   |   |   |--Configuration
|   |   |   |--Autoguider
|   |   |   |--Sequence
|   |   |   |   |--History
|   |   |   |--Camera
|   |   |   |   |--exposures
|   |   |   |   |   |--exposure_next
|   |   |   |   |   |   |--readout
|   |   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |   |--window_2
|   |   |   |   |   |   |--transfer
|   |   |   |   |   |   |--process
|   |   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |   |--window_2
|   |   |   |   |   |   |--display
|   |   |   |   |   |--exposure_1
|   |   |   |   |   |   |--readout
|   |   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |   |--window_2
|   |   |   |   |   |   |--transfer
|   |   |   |   |   |   |--process
|   |   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |   |--window_2
|   |   |   |   |   |   |--display
|   |   |   |   |   |   |--temperature
|   |   |   |   |   |   |--telemetry
|   |   |   |   |   |--exposure_2
|   |   |   |   |   |   |--readout
|   |   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |   |--window_2
|   |   |   |   |   |   |--transfer
|   |   |   |   |   |   |--process
|   |   |   |   |   |   |   |--window_1
```

```
   |   |   |   |   |   |   |   |--window_2
   |   |   |   |   |   |   |--display
   |   |   |   |   |   |   |--temperature
   |   |   |   |   |   |   |--telemetry
   |   |   |   |   |--images
   |   |   |   |   |   |--process
   |   |   |   |   |   |   |--window_1
   |   |   |   |   |   |   |--window_2
   |   |   |   |   |   |--transfer
   |   |   |   |   |--detector
   |   |   |   |   |   |--output_1
   |   |   |   |   |   |--output_2
   |   |   |   |   |   |--output_3
   |   |   |   |   |   |--output_4
   |   |   |   |   |   |--chip_1
   |   |   |   |   |--readout
   |   |   |   |   |   |--clock_1
   |   |   |   |   |   |--clock_2
   |   |   |   |   |   |--clock_3
   |   |   |   |   |   |--clock_4
   |   |   |   |   |   |--clock_5
   |   |   |   |   |   |--clock_6
   |   |   |   |   |   |--clock_7
   |   |   |   |   |   |--clock_8
   |   |   |   |   |   |--clock_9
   |   |   |   |   |   |--clock_10
   |   |   |   |   |--shutter
   |   |   |   |   |--temperature
   |   |   |   |   |--telemetry
   |   |   |   |   |--cryostat
   |   |   |   |   |--ace
   |   |   |   |   |--wcs
   |   |   |   |   |--observations
   |   |   |--Meteo
   |   |   |   |--Com
   |   |   |   |   |--Alarm
   |   |   |--Telescope
   |   |   |   |--History
   |   |--Public
   |   |   |--Dimm
   |   |   |   |--History
   |   |   |--Meteo
   |   |   |   |--History
--->END IMPORTED FILE: ../doc/wasm_DB.tree  (97/6/24 10:48)
```

23.8.2 lasm1

23.8.2.1 DATABASE.db

```
--->BEGIN IMPORTED FILE: lasm1/dbl/DATABASE.db  (97/6/24 10:31)
//*************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: DATABASE.db,v 1.31+ 1997/06/23 13:14:34 vltsccm Exp vltsccm $"
//*
//* who       when      what
//* --------  --------  ----------------------------------------------
//* NNNNNNNN  dd/mm/yy  created
//*/

//************************************************************************
// VLT COMMON SOFTWARE
//************************************************************************/

// Loads classes definition from standard file
#include "CCS.db"

//
// USER.db contains necessary points for CCS, but requiring editing to match
// installed configuration.
#include "USER.db"


//************************************************************************/
// APPLICATION SOFTWARE
//
#include "ASM.db"

// ___oOo___
--->END IMPORTED FILE: lasm1/dbl/DATABASE.db  (97/6/24 10:31)
```

23.8.2.2 USER.db

```
--->BEGIN IMPORTED FILE: lasm1/dbl/USER.db  (97/6/24 10:31)
//*************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: USER.db,v 1.31+ 1997/06/23 13:14:34 vltsccm Exp vltsccm $"
//*
//* who       when      what
//* --------  --------  -----------------------------------------------
//*

//
// ***********************************************************************
// Points needed by LCC, but requiring editing to match user installed config.
// ***********************************************************************
//

//**************************************************************************
//   SCAN SYSTEM points
//**************************************************************************

#include "asmiSCANDEV.class"

POINT "asmiSCANDEV" "LAN:wasm"
BEGIN
        Alias   "wasm"
END

// ___oOo___
--->END IMPORTED FILE: lasm1/dbl/USER.db  (97/6/24 10:31)
```

23.8.2.3 ASM.db

```
--->BEGIN IMPORTED FILE: lasm1/dbl/ASM.db  (97/6/24 10:31)
// ****************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: ASM.db,v 1.31+ 1997/06/23 13:14:35 vltsccm Exp vltsccm $"
//
// who      when     what
// -------- -------- ----------------------------------------------
// ssandroc 17.12.96 created
//

#define TARGET_TCS VLT_TCS

#include "asmi.class"
#include "asmtel.class"
#include "asmten.class"
#include "asmmet.class"

POINT asmiCLASS         "ASM"
BEGIN
 Alias                  "asm"

 ATTRIBUTE asmtelDEVICE "Telescope"
 BEGIN
        Alias           "asmTelescope"
 END

 ATTRIBUTE asmtenDEVICE "Enclosure"
 BEGIN
        Alias           "asmEnclosure"
 END

 ATTRIBUTE asmmetDEVICE "Meteo"
 BEGIN
        Alias           "asmMeteo"
 END

END

// ___oOo___
--->END IMPORTED FILE: lasm1/dbl/ASM.db  (97/6/24 10:31)
```

23.8.2.4 DB.tree

```
--->BEGIN IMPORTED FILE: ../doc/lasm1_DB.tree  (97/6/21  5: 7)
  |--ccs_config
  |  |--class config
  |  |--BASE_CLASS
  |  |  |--alrmSTATE_ALARM
  |  |  |  |--alrmCCS_ALARM
  |  |  |--evhDB_TASK
  |  |  |  |--evhSTD_COMMANDS
  |  |  |  |  |--asmiTASK
  |  |  |--asmiCLASS
  |  |  |  |--asmiDEVICE
  |  |  |  |  |--asmtelDEVICE
  |  |  |  |  |--asmtenDEVICE
  |  |  |  |  |--asmmetDEVICE
  |  |  |  |--asmiBRANCH
  |  |  |  |  |--asmtelTRKVTA
  |  |  |  |--asmiDATA
  |  |  |  |  |--asmmetDATA
  |  |  |  |--asmiSIGNAL
  |  |  |  |  |--asmiSIG_DIGITAL
  |  |  |  |  |--asmiSIG_ANALOG
```

```
|   |   |   |--asmtelAMPLIFIER
|   |   |   |--asmtelAXES
|   |   |--tooCOM_PARAMS
|   |   |   |--asmiCOM
|   |   |   |   |--asmtenCOM
|   |   |   |   |--asmmetCOM
|   |   |--axisDIGITAL_SIGNAL
|   |   |--axisANALOG_SIGNAL
|   |   |--axisSTD_VEL_CTLR
|   |   |   |--asmtelVEL_CTLR
|   |   |--axisSTD_ENCODER
|   |   |   |--asmtelENCODER
|   |   |--axisSTD_IL_CONTROL
|   |   |--axisSTD_IL_STATUS
|   |   |--axisSTD_LIMITSWITCH
|   |   |--axisBACKOFF_SETUP
|   |   |--axisPOS_CTLR
|   |   |   |--asmtelPOS_CTLR
|   |   |--axisINTERLOCKS
|   |   |   |--asmtelINTERLOCKS
|   |   |--axisDB_STD
|   |   |--axisNEW_REF
|   |   |--axisAXIS
|   |   |   |--asmtelAXIS
|   |   |--SETUP
|   |   |--MONITORED_DATA
|   |   |--ENVIRONMENT
|   |   |--TIMES
|   |   |--TRK_COORDINATES
|   |   |--TRK_INTERN_COORD
|   |   |--TRK_ACT_COORD
|   |   |--TRK_STATUS
|   |   |--SITE_CONSTANTS
|   |   |--POLAR_MOTION
|   |   |--POINTING_CORRECTION
|   |   |--POS_NAMES
|   |   |--VIR_TRACKING_AXIS
|   |   |--TRK_PRIVATE
|   |   |--T_AXES
|   |   |--TRACKING_AXIS
|   |   |   |--asmtelTRK
|--LAN
|   |--wasm
|--PARAMS
|   |--SCALARS
|   |--TABLES
|   |--VECTORS
|--LCC
|   |--ALARMS
|   |--DEVICES
|   |--STATUS
|--ASM
|   |--Telescope
|   |   |--axesData
|   |   |   |--northSignal
|   |   |   |--southSignal
|   |   |--Tracking
|   |   |   |--environment
|   |   |   |--times
|   |   |   |--siteConstants
|   |   |   |--polarMotion
|   |   |   |--posNames
|   |   |   |--setup
|   |   |   |--axes
|   |   |   |   |--asm
|   |   |   |   |   |--trkStatus
|   |   |   |   |   |--trkCoordinates
|   |   |   |   |   |--trkInternCoord
|   |   |   |   |   |--trkActCoord
```

```
   |  |  |  |  |  |--pointing
   |  |  |  |  |  |--trkMonitoredData
   |  |  |  |--vta
   |  |  |  |  |--asm
   |  |  |  |  |  |--POSLOOP
   |  |  |  |  |  |  |--ABSENCODER
   |  |  |  |  |  |  |--VELLOOP
   |  |  |--Axis1
   |  |  |  |--axisStd
   |  |  |  |--lastRef
   |  |  |  |--nextRef
   |  |  |  |--LIMITSWITCH
   |  |  |  |--BACKOFF
   |  |  |  |--POSLOOP
   |  |  |  |  |--VELLOOP
   |  |  |  |  |--ABSENCODER
   |  |  |  |  |  |--initSignal
   |  |  |  |--AMPLIFIER
   |  |  |  |  |--velocityCommand
   |  |  |  |  |--currentMonitor
   |  |  |  |  |--engPanelSelect
   |  |  |  |--INTERLOCKS
   |  |  |  |  |--CONTROL
   |  |  |  |  |  |--enable
   |  |  |  |  |--STATUS
   |  |  |  |  |  |--ampEnable
   |  |  |  |  |  |--invRange
   |  |  |--Axis2
   |  |  |  |--axisStd
   |  |  |  |--lastRef
   |  |  |  |--nextRef
   |  |  |  |--LIMITSWITCH
   |  |  |  |--BACKOFF
   |  |  |  |--POSLOOP
   |  |  |  |  |--VELLOOP
   |  |  |  |  |--ABSENCODER
   |  |  |  |  |  |--initSignal
   |  |  |  |--AMPLIFIER
   |  |  |  |  |--velocityCommand
   |  |  |  |  |--currentMonitor
   |  |  |  |  |--engPanelSelect
   |  |  |  |--INTERLOCKS
   |  |  |  |  |--CONTROL
   |  |  |  |  |  |--enable
   |  |  |  |  |--STATUS
   |  |  |  |  |  |--ampEnable
   |  |  |  |  |  |--invRange
   |  |--Enclosure
   |  |  |--Com
   |  |--Meteo
   |  |  |--Com
   |  |  |--Data
--->END IMPORTED FILE: ../doc/lasm1_DB.tree  (97/6/21  5: 7)
```

23.8.3 lasm2

23.8.3.1 DATABASE.db

```
--->BEGIN IMPORTED FILE: lasm2/dbl/DATABASE.db  (97/6/24 10:47)
//*****************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: DATABASE.db,v 1.31+ 1997/06/23 13:14:41 vltsccm Exp vltsccm $"
//*
//* who       when      what
//* --------  --------  ----------------------------------------------
//* NNNNNNNN  dd/mm/yy  created
//*/

//************************************************************************
// VLT COMMON SOFTWARE
//************************************************************************/

// Loads classes definition from standard file
#include "CCS.db"

//
// USER.db contains necessary points for CCS, but requiring editing to match
// installed configuration.
#include "USER.db"


//************************************************************************/
// APPLICATION SOFTWARE
//
#include "ASM.db"

// ___oOo___
--->END IMPORTED FILE: lasm2/dbl/DATABASE.db  (97/6/24 10:47)
```

23.8.3.2 USER.db

```
--->BEGIN IMPORTED FILE: lasm2/dbl/USER.db  (97/6/24 10:47)
//*****************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: USER.db,v 1.31+ 1997/06/23 13:14:41 vltsccm Exp vltsccm $"
//*
//* who       when      what
//* --------  --------  -----------------------------------------------
//*

//
// *************************************************************************
// Points needed by LCC, but requiring editing to match user installed config.
// *************************************************************************
//

//**************************************************************************
//   SCAN SYSTEM points
//**************************************************************************

#include "asmiSCANDEV.class"

POINT "asmiSCANDEV" "LAN:wasm"
BEGIN
        Alias   "wasm"
END

// ___oOo___
--->END IMPORTED FILE: lasm2/dbl/USER.db  (97/6/24 10:47)
```

23.8.3.3 ASM.db

```
--->BEGIN IMPORTED FILE: lasm2/dbl/ASM.db  (97/6/24 10:47)
// ***************************************************************************
// E.S.O. - VLT project
//
// "@(#) $Id: ASM.db,v 1.31+ 1997/06/23 13:14:42 vltsccm Exp vltsccm $"
//
// who      when     what
// -------- -------- ----------------------------------------------
// ssandroc 17.12.96 created
//

#define TARGET_TCS VLT_TCS

#include "asmi.class"
#include "asmdim.class"

POINT asmiCLASS         "ASM"
BEGIN
 Alias                  "asm"

 ATTRIBUTE asmdimDEVICE "Dimm"
 BEGIN
        Alias           "asmdim"
 END

END

// ___oOo___
--->END IMPORTED FILE: lasm2/dbl/ASM.db  (97/6/24 10:47)
```

23.8.3.4 DB.tree

```
--->BEGIN IMPORTED FILE: ../doc/lasm2_DB.tree  (97/6/21  5:10)
  |--ccs_config
  |  |--class config
  |  |--BASE_CLASS
  |  |  |--alrmSTATE_ALARM
  |  |  |  |--alrmCCS_ALARM
  |  |  |--evhDB_TASK
  |  |  |  |--evhSTD_COMMANDS
  |  |  |  |  |--asmiTASK
  |  |  |--asmiCLASS
  |  |  |  |--asmiDEVICE
  |  |  |  |  |--asmdimDEVICE
  |  |  |  |--asmiBRANCH
  |  |  |  |--asmiDATA
  |  |  |  |--asmiSIGNAL
  |  |  |  |  |--asmiSIG_DIGITAL
  |  |  |  |  |--asmiSIG_ANALOG
  |  |  |  |--asmdimCONFIG
  |  |  |  |--asmdimEXPOS
  |  |  |--tooCOM_PARAMS
  |  |  |  |--asmiCOM
  |  |  |--ccdAPPLICATION
  |  |  |  |--ccdLCUWSAPPLICATION
  |  |  |  |  |--ccdCAMERADCS
  |  |  |  |  |  |--ccdCAMERA
  |  |  |  |  |  |  |--asmdimCAMERA
  |  |  |  |--ccdSUBSYSTEM
  |  |  |  |  |--ccdIMAGESUBSYSTEM
  |  |  |  |  |  |--ccdIMAGEOPTSUBSYS
  |  |  |  |  |  |  |--ccdIMAGEPROCESS
  |  |  |  |  |  |--ccdIMAGETRANSFER
  |  |  |  |  |  |--ccdIMAGEHWSUBSYSTEM
  |  |  |  |  |  |  |--ccdREADOUT
```

```
|   |   |   |   |--ccdHWSUBSYSTEM
|   |   |   |   |   |--ccdOPTHWSUBSYSTEM
|   |   |   |   |   |   |--ccdSHUTTER
|   |   |   |   |   |   |--ccdTEMPERATURE
|   |   |   |   |   |   |--ccdTELEMETRY
|   |   |--ccdEXPGENERAL_SET
|   |   |   |--ccdEXPOSURE_SETUP
|   |   |   |--ccdEXPGENERAL
|   |   |   |   |--ccdEXPOSURE
|   |   |--ccdEXPRDTGEN_SET
|   |   |   |--ccdEXPREADOUT_SET
|   |   |   |--ccdEXPRDTGEN
|   |   |   |   |--ccdEXPOSUREREADOUT
|   |   |--ccdEXPWINRDT_SET
|   |   |--ccdEXPTRANSFER_SET
|   |   |   |--ccdEXPOSURETRANSFER
|   |   |--ccdEXPPRCGEN_SET
|   |   |   |--ccdEXPPROCESS_SET
|   |   |   |   |--ccdEXPOSUREPROCESS
|   |   |   |--ccdEXPPRCGEN
|   |   |--ccdEXPWINPRC_SET
|   |   |--ccdEXPDISPLAY_SET
|   |   |   |--ccdEXPOSUREDISPLAY
|   |   |--ccdEXPOSURETEMP
|   |   |--ccdEXPOSURETELE
|   |   |--ccdEXPOSURES
|   |   |--ccdPROCESS_WINDOW
|   |   |--ccdIMAGES
|   |   |--ccdDEVICE
|   |   |   |--ccdDEVICE_OPT
|   |   |   |   |--ccdOUTPUT_AMPLIFIER
|   |   |   |   |--ccdDETECTOR_COMP
|   |   |   |   |--ccdCRYOSTAT
|   |   |--ccdCHIP
|   |   |   |--ccdDETECTOR
|   |   |--ccdCLOCK_PATTERN
|   |   |--ccdACE
|   |   |--ccdWORLDCOORD
|   |   |--ccdOBSERVATIONS
|--LAN
|   |--wasm
|--PARAMS
|   |--SCALARS
|   |--TABLES
|   |--VECTORS
|--LCC
|   |--ALARMS
|   |--DEVICES
|   |--STATUS
|--ASM
|   |--Dimm
|   |   |--Configuration
|   |   |--Exposures
|   |   |--Camera
|   |   |   |--exposures
|   |   |   |   |--exposure_next
|   |   |   |   |   |--readout
|   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |--window_2
|   |   |   |   |   |--transfer
|   |   |   |   |   |--process
|   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |--window_2
|   |   |   |   |   |--display
|   |   |   |   |--exposure_1
|   |   |   |   |   |--readout
|   |   |   |   |   |   |--window_1
|   |   |   |   |   |   |--window_2
|   |   |   |   |   |--transfer
```

```
   |   |   |   |   |   |--process
   |   |   |   |   |   |   |--window_1
   |   |   |   |   |   |   |--window_2
   |   |   |   |   |   |--display
   |   |   |   |   |   |--temperature
   |   |   |   |   |   |--telemetry
   |   |   |   |   |--exposure_2
   |   |   |   |   |   |--readout
   |   |   |   |   |   |   |--window_1
   |   |   |   |   |   |   |--window_2
   |   |   |   |   |   |--transfer
   |   |   |   |   |   |--process
   |   |   |   |   |   |   |--window_1
   |   |   |   |   |   |   |--window_2
   |   |   |   |   |   |--display
   |   |   |   |   |   |--temperature
   |   |   |   |   |   |--telemetry
   |   |   |   |--images
   |   |   |   |   |--process
   |   |   |   |   |   |--window_1
   |   |   |   |   |   |--window_2
   |   |   |   |   |--transfer
   |   |   |   |--detector
   |   |   |   |   |--output_1
   |   |   |   |   |--output_2
   |   |   |   |   |--output_3
   |   |   |   |   |--output_4
   |   |   |   |   |--chip_1
   |   |   |   |--readout
   |   |   |   |   |--clock_1
   |   |   |   |   |--clock_2
   |   |   |   |   |--clock_3
   |   |   |   |   |--clock_4
   |   |   |   |   |--clock_5
   |   |   |   |   |--clock_6
   |   |   |   |   |--clock_7
   |   |   |   |   |--clock_8
   |   |   |   |   |--clock_9
   |   |   |   |   |--clock_10
   |   |   |   |--shutter
   |   |   |   |--temperature
   |   |   |   |--telemetry
   |   |   |   |--cryostat
   |   |   |   |--ace
   |   |   |   |--wcs
   |   |   |   |--observations
--->END IMPORTED FILE: ../doc/lasm2_DB.tree  (97/6/21  5:10)
```

23.9 ENVIRONMENT CONFIG FILES

23.9.1 wasm

23.9.1.1 PROCESSES

```
--->BEGIN IMPORTED FILE: wasm/PROCESSES  (97/6/24 10:48)
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: PROCESSES,v 1.31+ 1997/06/23 13:14:39 vltsccm Exp vltsccm $"
#
asmwsControl
ccdconCI_DIMM
msgServer
cmdManager
logManager
--->END IMPORTED FILE: wasm/PROCESSES  (97/6/24 10:48)
```

23.9.1.2 RtapEnvTable.normal

```
--->BEGIN IMPORTED FILE: wasm/RtapEnvTable.normal  (97/6/24 10:48)
#******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: RtapEnvTable.normal,v 1.31+ 1997/06/23 13:14:40 vltsccm Exp vltsccm $"
#
# who       when      what
# --------  --------  ----------------------------------------------
# ssandroc  07.03.97  created for ASM from envs 1.13

#-------------------------------------------------------------------------
#
#
#   P        R  M  C  G    S   P    C
#   R        E  U  L  E    H   R    O
#   O  P  U  S  L  E  T    U   I    M
#   C  H  S  T  T  A  O    T   O    M
#   N  A  E  A  I  N  B    D   R    A
#   U  S  R  R  P  U  I    W   T    N
#___M__E__?__T__L__P__T____N___Y____D_______________________________________
#
#  These are the core processes which I will always want to run.
#
    1  0  N  N  N  N  N  100  128   RtapScheduler
    2  1  N  P  N  N  Y   99  128   RtapMonitor -a -s > rtap_log 2>&1
    3  1  N  P  N  A  Y  100  128   RtapQServer
    4  1  N  N  N  A  N   80  128   RtapEventTrig
    5  1  N  N  N  A  Y   80  128   RtapEventConfg
    6  1  N  E  N  A  Y   80  128   RtapTimeKeeper
# First thing for CCS : setup of the shared memory area.
# imunoz  28.10.1996  ccsSHManager default to 50.000 to cope with increased CDTs
    0  1  Y  N  N  A  N  100  128   ccsSHManager -s 50000
#
    7  1  N  N  N  A  N    1  128   RtapDbStartup
#
#RtapDbStartup requests to proceed to next phase. i.e. phase 2
#
   58  0  N  E  N  A  N   80  128   RtapMQDBM
    9  2  N  Y  N  A  N    1  128   RtapDbCfServer
#
# Miscellaneous Tasks which may have to be performed.
#
    0  2  N  P  N  N  E  100  128   logManager
#
# WARNING: on empty databases the RtapScanMngr MUST be started with -f 300
#          Later on, once scan configuration has been performed, a
#          factor -f 4 is enough. When more scan links will be used, this
```

```
#          might have to be reduced again.

   10  2  Y  N  N  A  N   20  128   RtapScanMngr -f 300
# To be tried after confirmation of necessary space
#   10  2  Y  N  N  A  N   20  128   RtapScanMngr -m 100000
#   10  2  Y  N  N  A  N   20  128   RtapScanMngr -f 4
# CCS tasks
   11  0  Y  N  N  Y  Y    1  128   ccsScan
    0  0  Y  N  N  Y  Y    1  128   scanOut
    0  0  N  N  N  N  N    1  128   scanConfig
    0  2  Y  E  N  A  N  100  128   envsKill
   13  2  N  N  N  A  N  100  128   msgServer
   14  2  N  P  N  N  N  100  128   cmdManager
    0  0  N  N  Y  A  N  100  128   cmdSetup
    0  0  N  N  N  Y  N  100  128   accServer
    0  0  N  N  N  N  N  100  128   bookServer
#
#  Rtap utilities
#
# New Rtap processes for handling of alarms have been added hereafter.
# Once the environment is started, an alarm display can be invoked with
# one of the run strings:
#   history mode: RtapASDisplay -c -H 40 -F All &
#   summary mode: RtapASDisplay &
#**********************************************************************
   15  2  Y  P  N  A  N  100  128   RtapASServer
    0  0  Y  N  N  A  N  100  128   RtapASDisplay
# the next entry is for logging of alarms.
# REMARK:
# rtap must be able to create/write the file "alarmsLog" in the environment
# directory.
#
   16  2  N  N  N  A  N  100  128   RtapASLogger -f alarmsLog -F All
    0  0  N  N  N  A  N    1  128   RtapDbConfig
    0  0  N  Y  Y  A  N    1  128   RtapPtDisplay
    0  0  N  Y  Y  A  N    1  128   RtapPtSelector
    0  0  N  N  Y  A  N    1  128   RtapDbLoader
    0  0  N  N  Y  A  N    1  128   RtapDbUnloader
    0  0  N  Y  Y  A  N    1  128   RtapPlotDisp -s -p -q

   17  2  N  P  N  A  N   50  128   RtapDHMngr -h 1000000 -L p
    0  0  N  N  N  A  N    1  128   RtapDHConfig
#
# CCS Applications
#
    0  0  N  N  N  N  N    1  128   dbRead
    0  0  N  N  N  N  N    1  128   dbWrite
   20  2  N  P  N  N  N   50  128   ntpLog -t 60

#
# CCD Application Processes
#
    90  0  N  N  N  Y  N  100  128   ccdconCI DIMM
    91  0  N  N  N  Y  N  100  128   ccditWs DIMM

#
# ASM Application Processes
#
  100  0  N  N  N  Y  N  100  128   asmwsControl

#
# Global environment start script
#
    0  2  N  N  N  Y  N  100  128   wasmStart.sh
#
#___oOo___
--->END IMPORTED FILE: wasm/RtapEnvTable.normal  (97/6/24 10:48)
```

23.9.2 lasm1

23.9.2.1 PROCESSES

```
--->BEGIN IMPORTED FILE: lasm1/PROCESSES  (97/6/24 10:31)
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: PROCESSES,v 1.31+ 1997/06/23 13:14:36 vltsccm Exp vltsccm $"
#
asmlcuServer
asmtrkServer
etaServer
thetaServer
lccServer
msgServer
rdbServer
inducerServer
--->END IMPORTED FILE: lasm1/PROCESSES  (97/6/24 10:31)
```

23.9.2.2 devicesFile

```
--->BEGIN IMPORTED FILE: lasm1/devicesFile  (97/6/24 10:31)
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: devicesFile,v 1.31+ 1997/06/23 13:14:36 vltsccm Exp vltsccm $"
#
<CWP>: :LCC:DEVICES
<ATTRIBUTE>: deviceTable <TYPE>: Table <REC>: 0 - 6 <FIELDS>: 0 - 5
        "trk"        "asmtrkServer" 1 0 0 1
        "etaAxis"    "etaServer"    1 0 0 1
        "thetaAxis" "thetaServer"  1 0 0 1
        "Monitor"    "asmlcuServer" 1 0 0 1
        "Enclosure" "asmlcuServer" 1 0 0 1
        "Telescope" "asmlcuServer" 1 0 0 1
        "Meteo"      "asmlcuServer" 1 0 0 1
#_oOo_
--->END IMPORTED FILE: lasm1/devicesFile  (97/6/24 10:31)
```

23.9.2.3 bootScript

```
--->BEGIN IMPORTED FILE: lasm1/bootScript  (97/6/24 10:31)
#! VxWorks
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: bootScript,v 1.31+ 1997/06/23 13:14:35 vltsccm Exp vltsccm $"
#
# THIS IS A GENERATED FILE - DO NOT EDIT (unless you know how)
#
# Created by
#   User: ramestic
#   Host: te67
#   Date: Thu Apr 10 15:50:19 MESZ 1997
#   Tool: vccConfigLcu
#   Proc: vccConfigLcuGenerateBootScript 1.86
#

#:::   VCCDATA   :::
# vLcuEnv:      lasm1
# vLcuHost:     te78
# vLcuIPaddr:   134.171.12.121
# vLcuTCPport:  2160
```

```
# vLcuCpu:      MC68040
# vLcuType:     68k
# vLcuBsp:      mv167
# vBwsEnv:      wasm
# vBwsHost:     te67
# vBwsIPaddr:   134.171.12.87
# vBwsTCPport:  2250
# vBwsCpu:      hppa
# vBwsType:     hp9700
# BOOTHOME:     /diskb/vlt/data/ENVIRONMENTS/lasm1
# VXROOT:       /diskb/vw5.2
# vVltHost:     te67
# vVltIPaddr:   134.171.12.87
# VLTROOT:      /diskb/vlt/MAY97
# vIntHost:     te67
# vIntIPaddr:   134.171.12.87
# INTROOT:      /diskb/introot/asm
# vModHost:     te67
# vModIPaddr:   134.171.12.87
# MODROOT:
# vSysmodlist:  lcudrv lculog lqs acro aio iser ikon tim lcc cai scan
# vUsrmodlist:  inducer too tcsVcc sla trk axis asmlcu
# vDevlist:     {1 /acro} {1 /aio} {1 /ikon} {10 /iser} {1 /tim}
#

#:::   ROOT   :::
putenv "BOOTHOME=/diskb/vlt/data/ENVIRONMENTS/lasm1"
putenv "VLTROOT=/VLTROOT"
putenv "INTROOT=/INTROOT"
#putenv "MODROOT=/MODROOT"

#:::   NETWORK   :::
nfsAuthUnixSet getenv("HOSTNAME"),138,300 /* vx,vlt */
hostAdd "te67","134.171.12.87"
nfsMount "te67","/diskb/vlt/MAY97","/VLTROOT"
hostAdd "te67","134.171.12.87"
nfsMount "te67","/diskb/introot/asm","/INTROOT"

#:::   LCUBOOT   :::
cd getenv("VLTROOT")
ld 0,1,"vw/bin/MC68040/lcuboot"
lcubootLogInit

#:::   ENVIRONMENT   :::
putenv "CPU=MC68040"
putenv "LOCALENV=lasm1"
putenv "LOCALTCPPORT=2160"
putenv "HOSTENV=wasm"
putenv "HOSTTCPPORT=2250"
lcubootAutoEnvInit /* set rest of variables automatically */

#:::   MODULES   :::
lcubootAutoLoad 1,"lcudrv",0
lcubootAutoLoad 1,"lculog",0
lcubootAutoLoad 1,"lqs",0
lcubootAutoLoad 1,"acro",0
lcubootAutoLoad 1,"aio",0
lcubootAutoLoad 1,"iser",0
lcubootAutoLoad 1,"ikon",0
lcubootAutoLoad 1,"tim",0
lcubootAutoLoad 1,"lcc",0
lcubootAutoLoad 1,"cai",0
lcubootAutoLoad 1,"scan",0
lcubootAutoLoad 1,"inducer",0
lcubootAutoLoad 1,"too",0
lcubootAutoLoad 1,"tcsVcc",0
lcubootAutoLoad 1,"sla",0
lcubootAutoLoad 1,"trk",0
lcubootAutoLoad 1,"axis",0
```

```
lcubootAutoLoad 1,"asm",0
# Initialize System-Modules:
lcubootAutoCdBoot 1,"lcudrv.boot"
< lcudrv.boot
lcubootAutoCdBoot 1,"lculog.boot"
< lculog.boot
lcubootAutoCdBoot 1,"lqs.boot"
< lqs.boot
lcubootAutoCdBoot 1,"acro.boot"
< acro.boot
lcubootAutoCdBoot 1,"aio.boot"
< aio.boot
lcubootAutoCdBoot 1,"iser.boot"
< iser.boot
lcubootAutoCdBoot 1,"ikon.boot"
< ikon.boot
lcubootAutoCdBoot 1,"tim.boot"
< tim.boot
lcubootAutoCdBoot 1,"lcc.boot"
< lcc.boot
lcubootAutoCdBoot 1,"cai.boot"
< cai.boot
lcubootAutoCdBoot 1,"scan.boot"
< scan.boot

#:::   DEVICES   :::
lcubootAutoDevCheck "/acro",1
lcubootAutoDevCheck "/aio",1
lcubootAutoDevCheck "/ikon",1
lcubootAutoDevCheck "/iser",10
lcubootAutoDevCheck "/tim",1

#:::   USER   :::
# Initialize User-Modules:
lcubootAutoCdBoot 1,"inducer.boot"
< inducer.boot
lcubootAutoCdBoot 1,"too.boot"
< too.boot
lcubootAutoCdBoot 1,"tcsVcc.boot"
< tcsVcc.boot
lcubootAutoCdBoot 1,"sla.boot"
< sla.boot
lcubootAutoCdBoot 1,"trk.boot"
< trk.boot
lcubootAutoCdBoot 1,"axis.boot"
< axis.boot
lcubootAutoCdBoot 1,"asmlcu.boot"
< asmlcu.boot
# Execute User-Script:
cd getenv("BOOTHOME")
< userScript

#:::   TERMINATION   :::
taskDelay 100
lcubootAutoSpawn 1,"tLccResumeInit",80,0x18,40000,"lccResumeInit"
lcubootAutoExec 1,"lccWaitInit"
lcubootLogFinish

# ___END_OF_BOOT_SCRIPT___

--->END IMPORTED FILE: lasm1/bootScript  (97/6/24 10:31)
```

23.9.2.4 userScript

```
--->BEGIN IMPORTED FILE: lasm1/userScript  (97/6/24 10:31)
#! VxWorks
#*******************************************************************************
```

```
# E.S.O. - VLT project
#
# "@(#) $Id: userScript,v 1.31+ 1997/06/23 13:14:37 vltsccm Exp vltsccm $"
#
# lasm1 user-script
#

tooBootTime "update" /* record time of reboot in file */
tooSetLCUTime /* set time from WS clock; not needed with NTP or TimeBus */
tooDate /* show current time */

# ___oOo___
--->END IMPORTED FILE: lasm1/userScript  (97/6/24 10:31)
```

23.9.2.5 ikon0.dat

```
--->BEGIN IMPORTED FILE: lasm1/ikon0.dat  (97/6/24 10:31)
#*************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: ikon0.dat,v 1.31+ 1997/06/23 13:14:37 vltsccm Exp vltsccm $"
#
# ASM-specific /ikon0 device parameters
#
# THIS IS A DRIVER-GENERATED FILE - EDIT WITH CARE AND KEEP THE FORMAT

#=====================================================
# Parameters
#
# spec  value   comment
# ----------------------------------------------------
P 1.1   0       #          directionX1: normal
P 1.2   0       #          directionX2: normal
P 1.3   0       #      *directionCombi: normal
P 2.1   3       #           axisTypeX1: circular (+/- inf)
P 2.2   3       #           axisTypeX2: circular (+/- inf)
P 2.3   1       #        axisTypeCombi: linear
P 3.0   12      #            validBits:
P 4.1   0       #        refDistanceX1: (not used)
P 4.2   0       #        refDistanceX2: (not used)
P 5.1   1000    #         sigPeriodsX1:
P 5.2   1000    #         sigPeriodsX2:
P 5.3   0       #      sigPeriodsCombi: (not used)
P 6.1   0       #         corrEnableX1: off
P 6.2   0       #         corrEnableX2: off
P 7.1   0       #          corrRangeX1: (not used)
P 7.2   0       #          corrRangeX2: (not used)
P 8.1   1000    #          corrCountX1: (not used)
P 8.2   1000    #          corrCountX2: (not used)
P 9.1   1       #          corrWidthX1: (not used)
P 9.2   1       #          corrWidthX2: (not used)
P 10.0  0       #         latchDisable: none
P 19.1  0       #          refOffsetX1: (not used)
P 19.2  0       #          refOffsetX2: (not used)
P 21.0  0       #            combiMode: none
P 30.1  3       #        corrParamX1X2: 80-5Hz,upwards (not used)
P 30.2  0       #          corrParamX2: upwards (not used)
P 70.1  0       #          extPresetX1: (not used)
P 70.2  0       #          extPresetX2: (not used)
P 70.3  0       #       extPresetCombi: (not used)
P 71.1  0       #          vmePresetX1: (not used)
P 71.2  0       #          vmePresetX2: (not used)
P 71.3  0       #       vmePresetCombi: (not used)
#P 72.1 0       #         axisOffsetX1: (set from database)
#P 72.2 0       #         axisOffsetX2: (set from database)
#P 72.3 0       #      axisOffsetCombi: (not used)
P 80.1  0       #         extFunction1: none (not used)
```

```
P 80.2  0       #         extFunction2: none (not used)
U # update parameters
#======================================================

# Correction X1 disabled, no data saved

# Correction X2 disabled, no data saved

# ___oOo___
--->END IMPORTED FILE: lasm1/ikon0.dat  (97/6/24 10:31)
```

23.9.2.6 ikon.boot

```
--->BEGIN IMPORTED FILE: lasm1/ikon.boot  (97/6/24 10:31)
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: ikon.boot,v 1.31+ 1997/06/23 13:14:37 vltsccm Exp vltsccm $"
#
# ASM-specific ikon driver installation

ikonN = lcubootAutoDevRegister("/ikon0", 0x3d,0xcfc000,1,1,1)
ikonN = lcubootAutoDevRegister("/ikon1", 0x3d,0xcf8000,1,1,1)
lcubootAutoDrvInstall "ikon"
lcubootAutoDevCreate "/ikon0",0xcfc000,0x8000,143,3,3,0x0000,0,"ikon0.dat"
lcubootAutoDevCreate "/ikon1",0xcf8000,0x8000,142,3,3,0x0000,0,"ikon0.dat"
ikonInterruptVerbose=1 /* for verbose interrupt logs */
# ___oOo___
--->END IMPORTED FILE: lasm1/ikon.boot  (97/6/24 10:31)
```

23.9.2.7 trk.boot

```
--->BEGIN IMPORTED FILE: lasm1/trk.boot  (97/6/24 10:31)
#*******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: trk.boot,v 1.31+ 1997/06/23 13:14:36 vltsccm Exp vltsccm $"
#
# ASM-specific trk installation

# dependent modules:
lcubootAutoLoad 1,"tcsVcc"
lcubootAutoLoad 1,"sla"
lcubootAutoLoad 1,"trk"

lcubootAutoSpawn 1,"trkTimTask",2,0x8,10000,"trkTimTask"
# trkEntryPoint will be called with proper DB-root by asmtelStart!
# ___oOo___
--->END IMPORTED FILE: lasm1/trk.boot  (97/6/24 10:31)
```

23.9.3 lasm2

23.9.3.1 PROCESSES

```
--->BEGIN IMPORTED FILE: lasm2/PROCESSES  (97/6/21  5:10)
#******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: PROCESSES,v 1.30 1997/06/19 16:14:58 vltsccm Exp $"
#
lccServer
msgServer
rdbServer
inducerServer
ccdcon_DIMM
ccdit_DIMM
ccdip_DIMM
ccdrdt_DIMM
ccdtel_DIMM
ccdtmp_DIMM
--->END IMPORTED FILE: lasm2/PROCESSES  (97/6/21  5:10)
```

23.9.3.2 devicesFile

```
--->BEGIN IMPORTED FILE: lasm2/devicesFile  (97/6/24 10:47)
#******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: devicesFile,v 1.31+ 1997/06/23 13:14:43 vltsccm Exp vltsccm $"
#
<CWP>: :LCC:DEVICES
--->END IMPORTED FILE: lasm2/devicesFile  (97/6/24 10:47)
```

23.9.3.3 bootScript

```
--->BEGIN IMPORTED FILE: lasm2/bootScript  (97/6/24 10:47)
#! VxWorks
#******************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: bootScript,v 1.31+ 1997/06/23 13:14:42 vltsccm Exp vltsccm $"
#
# THIS IS A GENERATED FILE - DO NOT EDIT (unless you know how)
#
# Created by
#   User: ssandroc
#   Host: te67
#   Date: Wed Mar 26 14:16:36 MEZ 1997
#   Tool: vccConfigLcu
#   Proc: vccConfigLcuGenerateBootScript 1.86
#

#:::   VCCDATA   :::
# vLcuEnv:      lasm2
# vLcuHost:     te79
# vLcuIPaddr:   134.171.12.122
# vLcuTCPport:  2160
# vLcuCpu:      MC68040
# vLcuType:     68k
# vLcuBsp:      mv167
# vBwsEnv:      wasm
# vBwsHost:     te67
# vBwsIPaddr:   134.171.12.87
# vBwsTCPport:  2250
```

```
# vBwsCpu:       hppa
# vBwsType:      hp9700
# BOOTHOME:      /diskb/vlt/data/ENVIRONMENTS/lasm2
# VXROOT:        /diskb/vw5.2
# vVltHost:      te67
# vVltIPaddr:    134.171.12.87
# VLTROOT:       /diskb/vlt/MAY97
# vIntHost:      te67
# vIntIPaddr:    134.171.12.87
# INTROOT:       /diskb/introot/asm
# vModHost:      te67
# vModIPaddr:    134.171.12.87
# MODROOT:
# vSysmodlist:  lcudrv lculog lqs tim lcc cai scan b016
# vUsrmodlist:  inducer too ccd asmdim
# vDevlist:      {1 /tim}
#

#:::   ROOT   :::
putenv "BOOTHOME=/diskb/vlt/data/ENVIRONMENTS/lasm2"
putenv "VLTROOT=/VLTROOT"
putenv "INTROOT=/INTROOT"
#putenv "MODROOT=/MODROOT"

#:::   NETWORK   :::
nfsAuthUnixSet getenv("HOSTNAME"),138,300 /* vx,vlt */
hostAdd "te67","134.171.12.87"
nfsMount "te67","/diskb/vlt/MAY97","/VLTROOT"
hostAdd "te67","134.171.12.87"
nfsMount "te67","/diskb/introot/asm","/INTROOT"

#:::   LCUBOOT   :::
cd getenv("VLTROOT")
ld 0,1,"vw/bin/MC68040/lcuboot"
lcubootLogInit

#:::   ENVIRONMENT   :::
putenv "CPU=MC68040"
putenv "LOCALENV=lasm2"
putenv "LOCALTCPPORT=2160"
putenv "HOSTENV=wasm"
putenv "HOSTTCPPORT=2250"
lcubootAutoEnvInit /* set rest of variables automatically */

#:::   MODULES   :::
lcubootAutoLoad 1,"lcudrv",0
lcubootAutoLoad 1,"lculog",0
lcubootAutoLoad 1,"lqs",0
lcubootAutoLoad 1,"tim",0
lcubootAutoLoad 1,"lcc",0
lcubootAutoLoad 1,"cai",0
lcubootAutoLoad 1,"scan",0
lcubootAutoLoad 1,"b016",0
lcubootAutoLoad 1,"inducer",0
lcubootAutoLoad 1,"too",0
lcubootAutoLoad 1,"ccd",0
lcubootAutoLoad 1,"asmdim",0
# Initialize System-Modules:
lcubootAutoCdBoot 1,"lcudrv.boot"
< lcudrv.boot
lcubootAutoCdBoot 1,"lculog.boot"
< lculog.boot
lcubootAutoCdBoot 1,"lqs.boot"
< lqs.boot
lcubootAutoCdBoot 1,"tim.boot"
< tim.boot
lcubootAutoCdBoot 1,"lcc.boot"
< lcc.boot
lcubootAutoCdBoot 1,"cai.boot"
```

```
< cai.boot
lcubootAutoCdBoot 1,"scan.boot"
< scan.boot
lcubootAutoCdBoot 1,"b016.boot"
< b016.boot

#:::   DEVICES   :::
lcubootAutoDevCheck "/tim",1

#:::   USER   :::
# Initialize User-Modules:
lcubootAutoCdBoot 1,"inducer.boot"
< inducer.boot
lcubootAutoCdBoot 1,"too.boot"
< too.boot
lcubootAutoCdBoot 1,"ccd.boot"
< ccd.boot
lcubootAutoCdBoot 1,"asmdim.boot"
< asmdim.boot
# Execute User-Script:
cd getenv("BOOTHOME")
< userScript

#:::   TERMINATION   :::
taskDelay 100
lcubootAutoSpawn 1,"tLccResumeInit",80,0x18,40000,"lccResumeInit"
lcubootAutoExec 1,"lccWaitInit"
lcubootLogFinish

# ___END_OF_BOOT_SCRIPT___

--->END IMPORTED FILE: lasm2/bootScript  (97/6/24 10:47)
```

23.9.3.4 userScript

```
--->BEGIN IMPORTED FILE: lasm2/userScript  (97/6/24 10:47)
#! VxWorks
#*************************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: userScript,v 1.31+ 1997/06/23 13:14:44 vltsccm Exp vltsccm $"
#
# lasm2 user-script
#

tooBootTime "update" /* record time of reboot in file */
tooSetLCUTime /* set time from WS clock; not needed with NTP or TimeBus */
tooDate /* show current time */

# ___oOo___
--->END IMPORTED FILE: lasm2/userScript  (97/6/24 10:47)
```

23.10 GLOBAL CONFIGURATION FILES

Imported listing of config files.

TBD

23.11 INSTRUMENTATION FILES

23.11.1 dimmAG.det(5)

NAME

```
dimmAG.det - DIMM autoguider setup file
```

DESCRIPTION

```
This file contains the constant setup parameters to be used
for the DIMM autoguider exposure mode.
Several further parameters will be set at run-time.
```

FILES

```
The file must be located in:
$INS_ROOT/SYSTEM/COMMON/SETUPFILES/DET/
```

ENVIRONMENT

```
INS_ROOT - path to INS_ROOT structure
```

CAUTIONS

```
This file is installed by the asmINS module.

=====   D O   N O T   E D I T   =====
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:59
```

23.11.2 dimmBG.det(5)

NAME

```
dimmBG.det - DIMM background determination setup file
```

DESCRIPTION

```
This file contains the constant setup parameters to be used
for the DIMM background determination mode.
Several further parameters will be set at run-time.
```

FILES

```
The file must be located in:
$INS_ROOT/SYSTEM/COMMON/SETUPFILES/DET/
```

ENVIRONMENT

```
INS_ROOT - path to INS_ROOT structure
```

CAUTIONS

```
This file is installed by the asmINS module.

=====   D O   N O T   E D I T   =====
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:59
```

23.11.3 dimmSEQ.det(5)

NAME

```
dimmSEQ.det - DIMM sequence setup file
```

DESCRIPTION

```
This file contains the constant setup parameters to be used
for the DIMM seeing measurement sequence.
Several further parameters will be set at run-time.
```

FILES

```
The file must be located in:
$INS_ROOT/SYSTEM/COMMON/SETUPFILES/DET/
```

ENVIRONMENT

```
INS_ROOT - path to INS_ROOT structure
```

CAUTIONS

```
This file is installed by the asmINS module.

=====   D O   N O T   E D I T   =====
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:59
```

23.12 SPECIAL MODULES

23.12.1 asmBUILD(1)

NAME

```
asmBUILD - Astronomical Site Monitor Build Module
```

SYNOPSIS

```
cd asmBUILD/src
make man install
cd ../../INSTALL; ./build
```

DESCRIPTION

```
This modules builds the complete ASM software and its environments
from scratch. This includes the required TCS modules, but not the
standard VLTSW and CCD-SW releases.

The list of modules and their version is taken from:

    asmBUILD/config/Modules.config

The following parts of the ASM system are built:
- the code under INTROOT
- the environments under VLTDATA
- the essential files under INS_ROOT

Here is how to use it:

Assume that you put the module under your home directory:

    cd ~
    cmmCopy asmBUILD

The build-procedure is then started by:

    cd ~/asmBUILD/src
    make install
    cd ../../INSTALL; ./build

Alternatively, a sample job-file to rebuild everything during each
night can be put into the file ~/job. This example also forwards the
job's output to the users `goethe' and `schiller' on machine `weimar',
assuming that you don't have a .forward file. The contents of the
file is then:

    cd ~/asmBUILD/src
    make install
    cd ../../INSTALL; ./build
    at -f ~/job 0300 tomorrow

    echo 'goethe@weimar, schiller@weimar' > ~/.forward
    echo "rm ~/.forward" | at now + 1 minute

The job will then delete and create the directory ~/INSTALL (or more
exactly "../../INSTALL" relative to the Makefile), where all the code
is installed. It will also clean (!) your entire INTROOT which must be
defined, and delete (!) and re-create all ASM environments and restart
them. One should logout the LCU shell before running the job.
```

FILES

```
An INSTALL directory will be created on the same level as asmBUILD.
All further files will the put there.
```

ENVIRONMENT

```
Needed are: INTROOT, VLTDATA, INS_ROOT, etc.
```

SEE ALSO

```
asmENV(1), asmINS(1)
```

```
- - - - - -
Last change:  22/06/97-19:00
```

23.12.2 asmENV(1)

NAME

```
asmENV - Astronomical Site Monitor Environments Module
```

SYNOPSIS

```
cd asmENV/src
make clean wasm lasm1 lasm2 lasm0
```

DESCRIPTION

```
This module contains specific templates from which the
environments of the ASM system are built.
```

ENVIRONMENT

```
VLTDATA - points to the VLTDATA area
ACC_HOST - host where ACC database server is running
```

SEE ALSO

```
vccEnvCreate(1)
```

```
- - - - - -
Last change:  22/06/97-19:04
```

23.12.3 asmINS(1)

NAME

```
asmINS - Astronomical Site Monitor Instrumentation Module
```

SYNOPSIS

```
cd asmINS/src
make install
```

DESCRIPTION

```
This module contains files to be installed in $INS_ROOT, ie:
- CCD configuration
- CCD reference files (dark field FITS image for bias substraction)
- CCD setup files

INS_ROOT structure must already exist, use eg. oslxInsManger
to create it. The install sequence is:

- make a tar backup of the current INS_ROOT
- install all std CCD files for CCD head number 80 (this is our CCD!)
- copy FITS reference and setup files (should be gzip'ed in the module)
- gunzip all compressed files in INS_ROOT (forced overwrite)
```

ENVIRONMENT

```
INS_ROOT - points to the INT_ROOT structure that must already exist
        (it shall be the _physical_ path w/o links!)
```

CAUTIONS

```
The module shall contain FITS files in gzip'ed format.
```

SEE ALSO

```
ccdInstall.sh(1)
```

```
- - - - - -
Last change: ASM-XXX 21/06/97-04:59
```

___oOo___