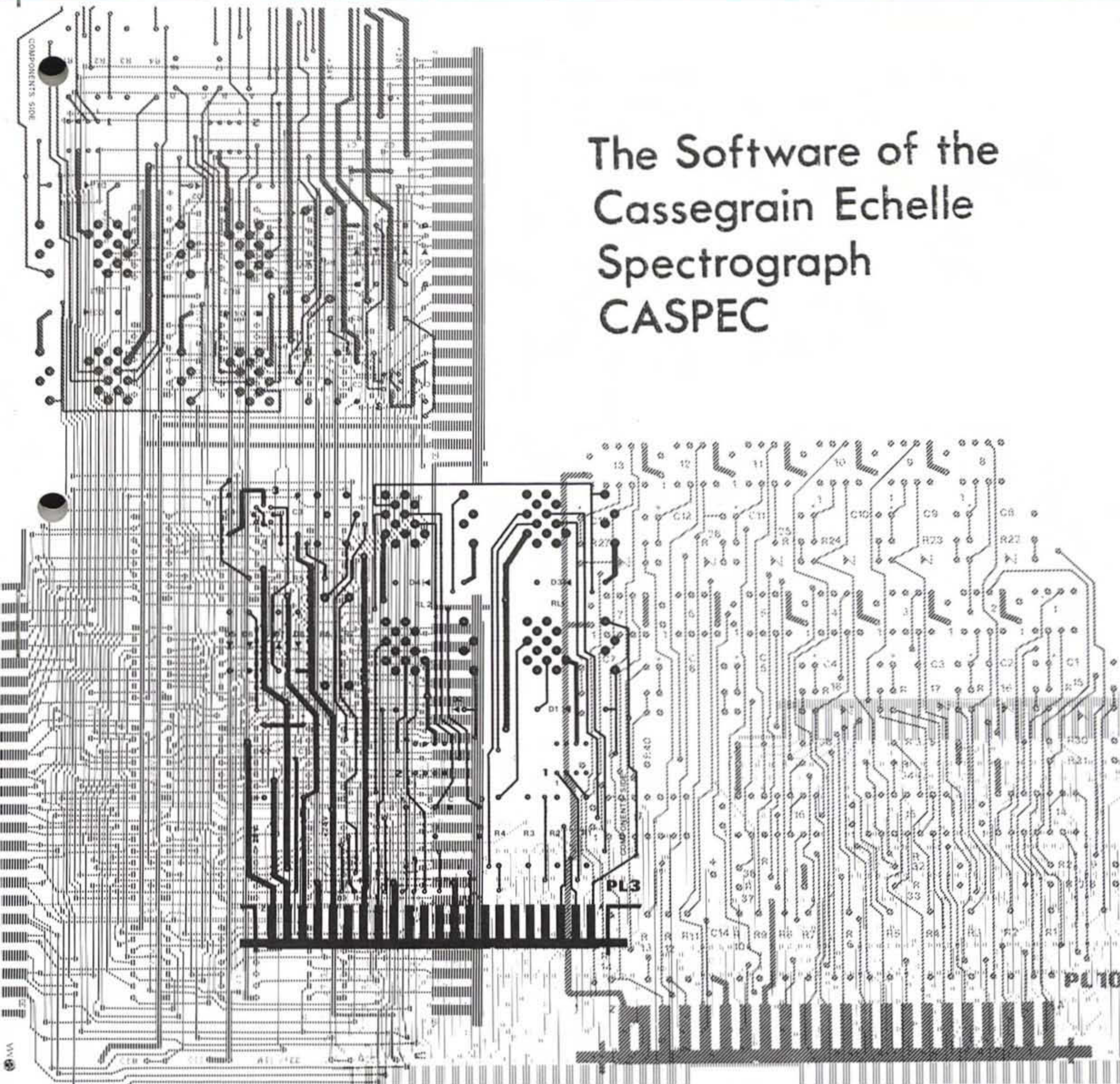
EUROPEAN SOUTHERN OBSERVATORY

MAINTENANCE MANUAL

No. 2 - JULY 1984

The Software of the Cassegrain Echelle Spectrograph CASPEC



M A I N T E N A N C E   M A N U A L

The Software of the CASSEGRAIN ECHELLE SPECTROGRAPH

C A S P E C

G. RAFFI

July 1984

TABLE OF CONTENTS

FIGURES

APPENDICES

ENCLOSED DOCUMENTATION

A. DAQLB library documentation (by G. Raffi)

B. MOTOR library documentation (by G. Raffi)

C. IOLIB library documentation (by G. Raffi)

ADDITIONAL DOCUMENTATION (not enclosed)

D. CCD documentation (by. P. Biereichel)

E. CRT05 interface (by P. Biereichel)

F. COMBI interface (by P. Biereichel)

G. PARMG user guide (by B. Gustafsson)

H. Communication protocol : 4-channel motor driver (by B. Gustafsson)

I. Communication between instrument and CCD program (by P. Biereichel)

Available in form of TPE notes, collected in the Data Acquisition System Folder, from ESO, Garching, TPE group.

INTRODUCTION

The CASPEC control software is installed on a Helwett-Packard 1000 21 MX-E computer with an ESO standard configuration (256 Kw of memory, 50 Mbyte disc, CAMAC crate, 1600 bpi magnetic tape unit) under the RTE-4B operating system.

The CASPEC instrument is at present available at the 3.6 m telescope.

This description refers to features of CASPEC package version 2.0 - Oct. 83 (tapes 831026). Possible following revisions will be notified in the Welcom file displayed when users log-on as CASP.

In June '84 the CASPEC software has been interfaced to the TCINT program at the 3.6 m telescope, which copes with the new telescope control system.

1. THE CASPEC SOFTWARE

The CASPEC on-line software, to control the instrument, acquire data from the CCD detector and to do on-line data reduction, consists of a number of programs cooperating together. It runs under the RTE-4B operating system in an ESO standard HP 1000 configuration (256 Kw of memory, 50 Mbytes disc, CAMAC crate, 1600 bpi magnetic tape unit).

The CASPEC specific control software consists of a main program (CASP), which handles the user interface and the instrument logic and sends commands to a kernel program. This in turn controls and monitors the 15 CASPEC functions (corresponding to 15 DC motors) and operates calibration lamps and shutter.

It is foreseen that the user can define single exposures or sequences of exposures to be executed on the CCD detector. The CASPEC motors are automatically positioned before each exposure. The CASPEC logic is based on a set of parameter tables, where the instrument configuration and installation values are described. These tables can easily be displayed or modified on-line.

The interface with the instrument is via CAMAC and a library of subroutines has been developed to interface with the CAMAC motor controllers.

1.1 The data acquisition system

The CASPEC specific programs rely on what we call data acquisition system : a set of either new or extensively readapted programs and libraries, connected by well defined interfaces.

The main idea behind this is to allow easy portability of detector packages (like CCD) among various instruments and of instrument packages among different telescopes.

The main components of the DAQ system as used by CASPEC are visualized in Fig. 1 and are described below.

- The CCD software sets up and monitors the CCD detector via a microprocessor controller. It executes exposures on demand and stores acquired data on disc and tape (175 kwords per image). The CCD package is now implemented to be completely portable among different instruments and is clearly the kernel of every CCD based instrument.

- The CAMAC/NIM motor controller systems, capable of controlling 4 motors per module, handle the motor-encoder loops for the 15 DC motors of CASPEC in a variety of configurations (circular and linear movements, motors with and without encoders). They are an essential component at the border between software and instrumentation electronics.

- The terminal handler program is instead dealing with the DAQ software at the user end side. It implements and controls access to the user screen by several programs, supporting function keys and forms at a high level.

- A generalized version of the parameter manager program used by CES to access parameter tables is also part of the DAQ system.

Communication among programs is via ASCII messages and to this end a number of interfaces have been defined : Instrument/Detector, DAQ system/telescope control system, DAQ system/IHAP.

The IHAP data processing system is used for on-line data reduction. IHAP has now been extended with new commands to handle echelle data (June 83 version). IHAP runs independently, but within the DAQ system and shares its data base with the CCD program, so that data are written only once to disc.

The CASPEC and CCD programs are implemented in such a way to make remote control feasible, i.e. they are split in two parts, one with the user interface and logic, the second with the CAMAC operations, which can be installed on two different computers and can exchange messages via a link.

CASPEC and the data acquisition system

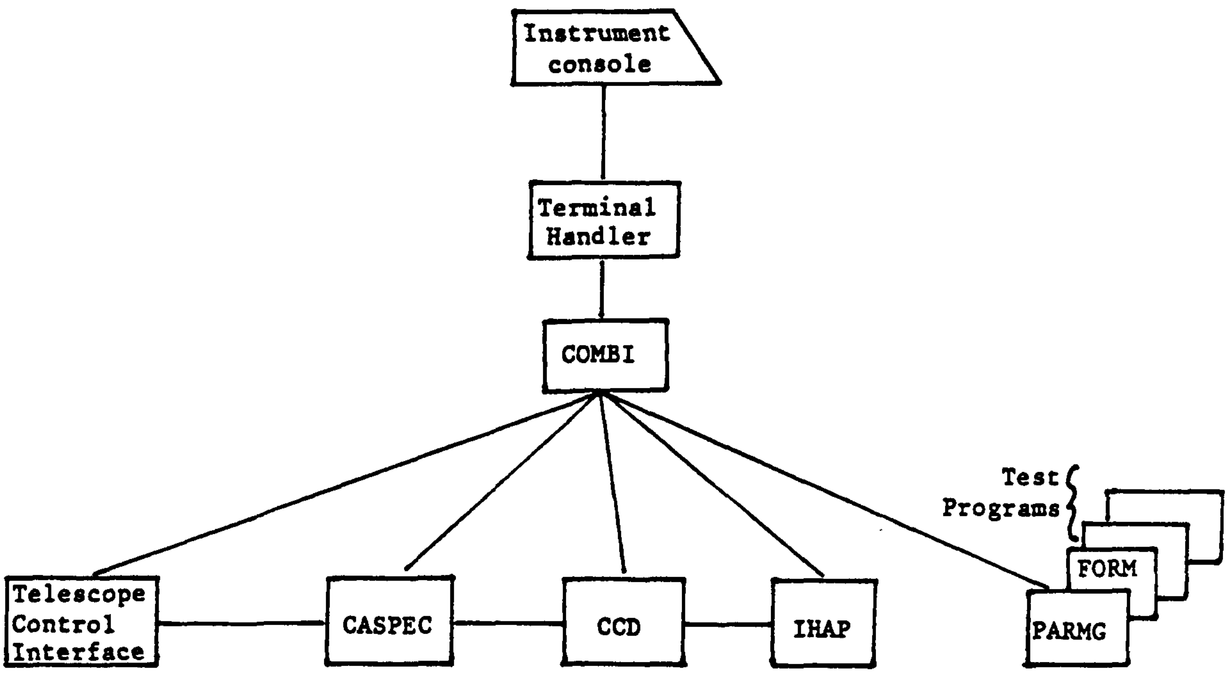


Fig. 1

SOFTKEY MENUS OF CASPEC

OBSERVATION	Telescope Setting	DISPLAY PARAMETERS	Help
Re-Initialize F's	MAINTENANCE	Status	Terminate

MAIN MENU

Define exposure	Display sequence	IHAP	Help
Start single exposure	Start sequence	Status	Previous Menu

OBSERVATION MENU

Change parameters	Lamps, shutter- ...	Test CAMAC	Help
Move Motors	Check Motors	Handset move	Previous Menu

MAINTENANCE MENU

Filter Wheel tab.	Slit deckers tab.	Collim. etc... tab.	Help
Motor vs. Funct's	Functions set-up	Spectral table	Previous Menu

DISPLAY PARAMETERS MENU

Note : Softkey labels are in capital letters when another menu is called.

Fig. 2

1.2 The CASPEC user station

It consists of :

- An instrument console (HP 2645) for user I/O.
- An auxiliary console (HP 2648) used either to display CASPEC additional forms or for on line graphics.
- 2 Ramtek monitors (colour and B & W) and an HP plotter for data reduction.

User input to CASPEC is via function keys (2 levels of menus) and via forms, filled in with appropriate parameters.

The user has also access to CCD programs, IHAP, parameter manager and test programs at the same console.

Typically during long exposures he would work with IHAP to reduce previous data, while the CCD detector and the CASPEC motors are automatically monitored by the corresponding programs.

2. OPERATION

Fig. 2 shows the 2 levels of softkey menus handled by CASP.

All commands to CASP are via softkeys. However the user is also requested to fill-in forms and occasionally to answer questions in the rolling part of the instrument console.

Appendix A gives a description of the softkey logic. The same text can be obtained on-line for all the softkeys in a given Menu, by making use of the Help softkey.

A complete description of CASPEC operation from the user standpoint is given in the CASPEC User's Manual.

3. INSTALLATION

The CASPEC software runs within the ESO Data Acquisition System (DAQ).

The CASPEC programs are listed in Appendix B.

Partial Installation

Normally the CASPEC programs should be already installed on CR = 3 and tables should be on CR = 33. Here are instructions to load the programs on a new system.

If the system contains already IHAP (June 83 or later version) and the DAQ programs installed, only the CASPEC specific programs are needed.

To be sure that you are in this situation, check the following :

1. Existence of following type 6 files :
   IHAP, GR00...GR54 at least for IHAP, CRT05, COMBI, FORM, PARMG for the DAQ system, CCD, DAQ, DAQS0, DAQS1, WCHDG, CCTST for CCD.

2. Make sure that either the DAQ cartridge DAQCC (CR = 75) is mounted or otherwise that the table files needed by CCD and COMBI :
   .CCDSK, .CCDFO, .CCDPA, .CCMSG, .SWTSK are available (check TABLE CR = 33).

3. Mount CASPEC CR = 85 and load CASPEC package by running the transfer file.
   (:) TR, *CASP :: 85
   On a standard system (2308) type 6 files are already available on CR = 85 and so no loading is needed.

   A listing of the loader files *CASP and @CP is given in Appendix C.

   Note : CR reference number needs to be 85 at load time.

   This will load the programs : CASPEC, CPENG, TCINT (for 3.6 m) and optionally the test programs : TMINI, TMOST, TMHND, TCAM, CPCKS.

   The reason for this last option is that test programs (except CPCKS) are general purpose and they might be already available. So possibly only CPCKS needs to be loaded.

4. Make sure that there is an account CASP under which the user can log on.

   If not create one and use *HICP:CP:2 as welcom file for this account. *HICP should already be on CR = 2, but a copy exists on CR = 85. *HICP gives introductory information on CASPEC, its function, and restores (RP) the DAQ and CASPEC programs.

   A copy of it is given in Appendix D. This file is meant to give temporary information and might vary with version and/or installation.

5. Follow then the test and troubleshooting procedure given in the next section.

Complete installation

Like partial installation, but step 1 must include the loading of the DAQ system and possibly IHAP.

IHAP loading is generally done with the transfer file LIHAP :: 100 for the La Silla IHAP. The IHAP version should be June 83 or later. For more information on IHAP refer to file "GIHAP.

The DAQ system and CCD programs can be loaded via :
(:)TR, *DAQCC::75

Then follow the instructions for partial installation.

4. CASPEC TESTS

4.1 Installation tests

The following description shall be followed systematically in the case of a new CASPEC installation.

It might of course be of use also for troubleshooting in particular cases.

Test IHAP installation

From instrument console (SL = 12) logon as IHAP, or (:)RU, IHAP, 12.

Using a test image test graphic terminal, plotter and Ramtek.

```
eg. : DRES, TSTIMG,
      TRAN,#1,S10                            Test graphic terminal
      PLOT                                   Test plotter
      CURS                                   Test graphic cursor
      KDIS,#1                                Test Ramtek
      KCOO
      KTAB, GE, COL01........   COL10
                                              Have a look at colours
```

This is normally enough to test the IHAP peripherals.

A RAMTEK stand-alone test, which you can run without IHAP, is available on CR = 85 (CASPEC) : TRAMT
To load it : RU, LOADR,@CP::85, %TRAMT::85

It displays a number of gradually changing vertical colour bars, according with the last look-up table loaded in the RAMTEK. (:) RU, TRAMT,,n with n number of consecutive displays wanted.

It is assumed that IHAP and all its segments have been restored (RP) by the welcom file at boot-up time.

Test CCD installation

- (:)RU, CCTST
  This allows to test CAMAC Le Croy modules, link with microprocessor and temperature controller.

- Check CCD installation tables (:) RU, PARMG answering
  (form file ?) .CCDFO : PB : 75
  (parameters files ?) .CCDPA : PB : 75

Table 1 contains LU's and CAMAC stations used.

- (:)RU, CCD
  CCD used in stand-alone mode.
  a) run CCTST (hardware tests softkey) under CCD. Should this fail, after CCTST has succeeded in stand-alone mode, the reason might be due to wrong information in the installation table of CCD.

- Enter debugging commands like

```
(!!)IN
(OK)
(!!) DFRE000010   Define exposure of 10 secs.
(OK)
!!EX              Start exposure
(DONE)
```

During exposure the CCD message with the remaining exp. time should appear and at end data should be recorded in IHAP database.

Have a look at directory etc. with IHAP, eg. :

```
DLIST,   1
WCOMM,   1 ...
```

For more information on CCD installation and troubleshooting see enclosed documentation D.

<u>Test CASPEC installation</u>

- Check CASPEC installation tables
  (:)RU, PARMG answering

```
(form file ?)          .FOCP:CP:33
(parameters file ?)    .PACP:CP:33
```

Files .FOCP, .PACP, .SKCP should be on CR = 33. A copy of them exists on CASPEC CR = 85.

If files are copied from CR = 85 to CR = 33, CASPEC will use the tables given in TABLES CR = 33, under the assumption that this CR is mounted before CR = 85, as mounting order.

CASPEC tables are listed in Appendix F.
Checks under PARMG :

1. Make sure that header form refers to .PACP : CP : 33 or edit it like this.

2. Make sure that LU's and CAMAC modules stations correspond to the description of table 5.

In particular the cartridge CR for auxiliary files should be 33.

- Test CASPEC as a whole, by logging on as CASP or by running : RU, CASP on instrument console.

  Possible messages like :
  "... class deallocated", before the Main Menu Display do not indicate a bad installation.

  CASP can be partially tested without IHAP and without CCD by :
  RU,CASP,,1.
  A message will appear saying that this is a CASPEC stand-alone mode.
  In this case CASP needs only the instrument console.

  If CAMAC is switched off, CASP will not proceed after the Main Menu, leaving to the user only the possibility to terminate.

  However some limited testing can be done with any CAMAC crate connection, without the CASPEC modules. This allows a partial installation check on any ESO HP computer with CAMAC.
  As CASP tries first of all to initialize motors it will fail on all of them. Then it will ask for the CASPEC functions (filter wheels etc.) to be activated. Answering Ø will complete the initialization phase of CASP and will allow to test various parts of CASP.

  CASP can also be tested without CCD. Initialization of CCD will obviously fail, but CASP proceeds and even exposures (of type NO = no CCD) can be done.

CASPEC versions

Caspec version V 2.0 Oct. 83 (tape 831026) described here goes together with DAQ system July 83 (CR = 75, tape either July 83 or 831026).

CASPEC V 2.0 initializes motor controllers in block mode (fast mode ~ 5 secs.). It goes together with motor controller PROMS vers 27.10.83 or later.

4.2 Troubleshooting

The most common problems which might arise are listed here, in the assumption that CASP was properly installed. For problems due to bad installation follow the advice given in the previous section on installation tests.

Appendix E refers to problems which the user himself can fix, like getting the IHAP graphic terminal going or restarting CASPEC from scratch if he gets stuck.

- A problem which might occasionally show up is that an encoder value cannot be reached at a first attempt. Re-initializing CASPEC via the appropriate softkey usually fixes the problem.
- Some encoders locations might be faulty while a location one or two encoder values away is reachable. This might be acceptable on some functions.

In general, for motors troubleshooting, use the advise given in the following table.

SYMPTOM	POSSIBLE DIAGNOSIS	CURE
All motors fail Initializing.	Something wrong with CAMAC installation or cables connected or power switched off on CASPEC cable connections	Check hardware. To modify CAMAC stations use PARMG on table 5 and 6 (for connections)
Some motors fail	Typically encoder problems. Some cables can be disconnected or badly connected. Try to re-initialize all functions or only those functions which fail.	Use softkey STATUS and test program TMOST to get status of controllers and encoder values.
Some motors still fail	Motor cables, encoder or amplifier problem, motor controller problem.	Use programs TMINI and TMHND to know where to go. Be carefull ! you need to know a lot on the functions you move (like valid range, type of axis). This assumes deep knowledge of CASPEC !

Remember also that you can test even one single motor with CASP (softkey re-initialize) and this test repeated on different values is normally a more reasonable and easier way to proceed than to use a test program (e.g. under the maintenance menu). This is because CASP knows the type, valid range and scaling factor for every function. However if you want to move a motor with TMINI get first all the relevant information on the motor to exercise, like :

- Type of axis (linear or circular), upper/lower limits for encoder and offset in case of linear axis. All this is in tables either 2 or 3 or 4 depending on function.

- From table 6 you know to which controller the function is connected and in table 5 you have the controllers slots.

Having done this you can try to move the motor to various positions. Should this fail then you might suspect a bad connection, faulty electronics or faulty function. Changing function on the same controller, using a different controller etc.. helps then in localizing the problem.

Should instead TMINI succeed on a faulty CASPEC function then there might be something wrong with the tables or the particular encoder value wanted cannot be reached for some other reason (e.g. encoder problem).

Use the Status softkey under CASP to get a global view of the functions status.

5. MAINTENANCE

This section deals with routine maintenance operations to do when some instrument components (motors, encoders, controllers functions or simply cables) have been changed or exchanged.

Please note that even the simple dismounting/remounting of a CASPEC function, e.g. a filter wheel, might affect installation parameters as for example the relative encoder values might be different.

The CASPEC installation parameters are kept in the installation tables, which can be changed with PARMG (Parameter Manager).

To run PARMG :

(:)RU, PARMG answering
(form file ?) .FOCP:CP:33
and checking that parameters file is
.PACP:CP:33

The top form of PARMG, when run with the above form file, contains the list of CASPEC tables, as given in Appendix F, where the tables layout with the set-up parameters at CASPEC installation are also listed.

Important note :

Before editing any tables on TABLES CR = 33 take following precautions :

1. List installation tables on printer via PARMG. It is useful to have old values in mind in editing and good to keep a hard copy of previous values in case of troubles later on.

   To get tables listing quicker you can use PARMG on each table or a transfer file like *PFORM::85.

2. Copy .PACP, .FOCP from CR = 33 to temporary files so that you keep a back-up copy during testing of the new tables.

The tables concerned with installation parameters are 2, 3, 4, 5, 6, 13.

Some general points refer to the three installation tables 2, 3 and 4.

Please note that all the encoder values expected are always the absolute encoder values, though internally CASPEC uses relative encoder values for linear functions.

The valid range is 0 - 2047, given that 11 bit serial encoders are used. The timeout value, between 5 and 99 secs, must be long enough to allow for the longest possible function movement (e.g. from min. to max. position), but not much longer than needed, as positioning errors are given only at the end of the timeout time.

Once functions reach the wanted encoder value the motor controllers are operated by CASPEC in two different ways depending on function : either the motor is left connected or it is disconnected (and left disconnected even if position is lost). In the first case the function is kept in the wanted position all the time. The second solution is instead preferred when one does not want that a function moves backwards, forwards even of one bit. A disconnected motor can of course drift due to the weight of the function or simply to a motor amplifier offset, but this (when limited and slow) is sometimes preferable to precise active control.

These effects have been studied at installation and functions are set-up as follows :

Functions 1, 3, 6, 9, 11, 12 - motor disconnected on position.
Functions 2, 7, 8, 10, 13, 14 - motor left connected on position.

Hartmann masks (15, 16) are disconnected once the limit is reached, so that they cannot drift away.

The connect/disconnect alternative is not an installation option, as it is independent of the particular mounting of functions. Subroutine &GTFUN contains anyhow the definition of the above values as a parameter for each function. Another general point concerns the encoder value limits, which cannot be exactly the mechanical limits. They must be some encoder values away from them (2 or 3 bits is enough) so that these values <u>can always be reached under computer control</u>.

Now, let us look in some detail into the various installation tables.

<u>Table 2</u> refers to wheel functions :
Neutral filter, colour filter, calibration filter, photometric filter, calibration switch wheel, calibration source wheel.

They are all circular functions, meaning by this that they are rotating and free to move to any position in either direction.

For the first 4 functions, the offset position corresponds to a no-filter position.

Given that there are 5 other equispaced positions on the filter wheel defined by the sense of rotation, one has to fill in the filter number corresponding to each position. For the meaning of filter numbers refer to table 11, where neutral and colour filter numbers are defined. The calibration switch wheel and the calibration source wheel have instead a number of not predefined positions to go to. The blind position of calibration source wheel is basically any position far away from a lamp. It is used while doing a star exposure to avoid stray light coming in (even if the calibration switch wheel is obviously set to star).

Table 3 refers to functions :

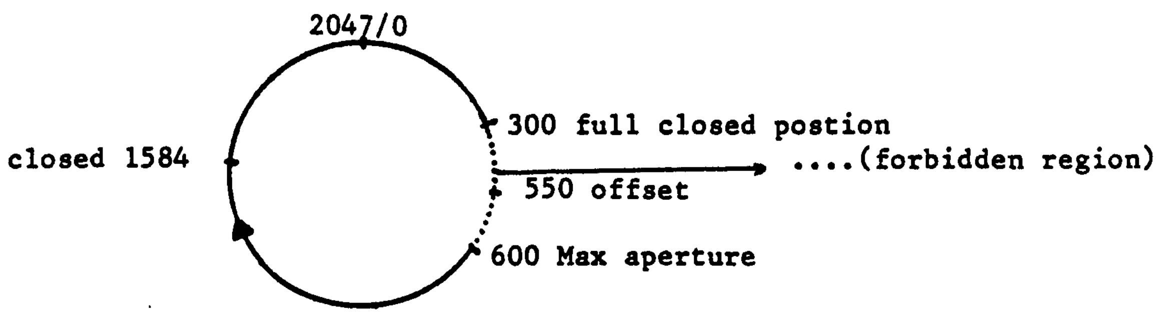
Preslit decker, slit and deckers.
These functions are linear for the motor controller software, as the encoder can only move within a range of values, while mechanical limits prevent it to go into other positions. The meaning of offset in this case is rather special and deserves a detailed explanation.

Offset evaluation (for linear function).

Let us take a decker as an example. It is a function with an eccentric drive, which allows it to move between a minimum and a maximum position. Assuming to work with the handset for the determination of the limits one has to work out the valid range of the function. The limits obtained are the physical/mechanical limits of the function and are not the values to put into the table. In particular for deckers to reach the fully closed position remember that the complementary decker has to be kept open, as deckers are free to push each other around the central position.

Having found the mechanical limits one can then proceed to find the "close" position, i.e. the centre position where the two deckers touch each other. The range of allowed encoder values is now known. The offset is then chosen as an encoder value falling within the forbidden range, so that the relative encoder values in the allowed range are well within the range 0-2047, without crossing 2047.

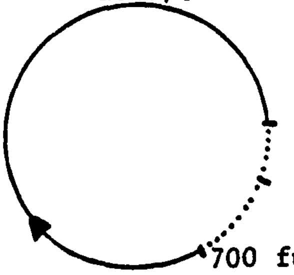
In the case of the deckers, let us assume that fig. 3 contains the results of the handset measurements for decker no.1.



Decker No. 1 - Absolute encoder values - Fig. 3

The continuous line shows the valid range to go from max. aperture to closed position (2 deckers touching in the middle) to full closed position or vice-versa. This allows to choose the offset value. 550 would be appropriate in this case, so that relative values go from 50 to 1797. So, one can say that the offset is connected to the sense of rotation of the motor axis. In this case increasing encoder values lead to movements in the close position direction.

To complete the picture let us consider decker 2 as a further example in Fig. 4, where the offset meaning is less self-evident.



Decker No. 2 - Absolute encoder values - Fig. 4

A value of 600 for the offset is appropriate as it falls in the forbidden region and allows to have relative values well inside the range 0-2047.

However in this case the offset absolute value is higher than the absolute encoder value at max. aperture. This means only, with reference to fig. 5, that the start point of the linear movement is now the full closed position. In other words the mounting of the decker is such that increasing encoder values lead to movements in the max. aperture direction. The arrow indicates the direction of the linear movement, i.e. the direction of movement when relative values increase.

Table 4 refers to functions :

rear slit viewer, collimator, cross disperser, Hartmann masks and echelle.

The rear slit viewer, although physically a wheel, is seen by software as a linear function, as there is a forbidden range in the rotation. Collimator and cross disperser are linear functions. No values have to be filled in for the Hartmann masks as there are no encoders for them and they simply move from one electromechanical limit to the other (upper/lower limits).

The echelle position is adjusted manually and not used by CASPEC, as there is no computer control of this function.

For more information on the mechanical structure of functions please refer to the CASPEC Technical Manual - Part 1.

Complete information on the motor controllers protocol is given in the additional documentation - H.

Table 6 tells instead how the functions are connected (e.g. function 9 connected to Contr. 1 - motor 1). Controller numbers are in turn associated to CAMAC stations by table 5.

Table 6 contains also a code for each function. The code is a switch selectable 5 bits configuration set-up on the function side of CASPEC. This is used to check on-line that functions are connected in the right place. For every function CAMAC is read and the value obtained compared with the one given in this table, to prevent illegal functions exchange, e.g. by a wrong cable connection.

Function codes are unique for each function, but echelle and cross disperser might have more than one valid code. This allows CASPEC to differentiate between echelles and cross-dispersers of different characteristics, so that the correct function characteristics can be recorded on tape in the data headers.

The last installation table, history table 13, gives statistics on the usage of CASPEC. Its purpose is to help to assess when calibration lamps need replacement. Note also that as total exposure times are expressed in hours and minutes, it might happen that lamps which are exposed for very few seconds never get any minutes reported. This is because every time CASPEC is exited, the rounded value of minutes is stored and the fraction is lost. The relevant information in this case is the total number of exposures with that lamp, showing how many times it has been switched on and off.

Once maintenance tables have been created, the following is advisable :

1. list new tables on printer.
2. save them in a secondary location, unknown to user. This prevents that he can accidentally alter installation tables using PARMG.

Only after some time, when it is clear that the new tables are correct the copy of .PACP and .FOCP on CR = 85 should be overwritten with tables from CR = 33 and kept as a reference set-up.

Tables .PACPØ, .FOCPØ are a back-up copy referring to the original CASP installation and should never be overwritten.

6. CASPEC SOFTWARE STRUCTURE

It is assumed that the reader of this section is familiar with the concepts of the ESO software data acquisition environment. The chapter on CASPEC software gives a general overview on the DAQ system components which one should read before.

6.1 Program CASP

It is the main CASPEC program, run by the user when he enters (:)RU, CASP.

Its purpose is to handle the logic concerned with the user interface, menus and tables displays. It interfaces with the user via the terminal handler program CRT05 and with CASPEC via the kernel program CPENG (CASPEC engine) which controls motors, lamps and shutter.

CASP has also links to programs CCD, IHAP, COMBI and TCINT as explained below.

Fig. 5 gives a global view of the scheduling mechanisms of CASPEC at start time. The arrows indicate the programs scheduled, where an arrow passing via COMBI means that CASP schedules programs like IHAP, CCD and TCINT by passing a command to COMBI to do it. The number near the arrow indicates in which order programs are scheduled by CASP at start time.

When CASP schedules COMBI it receives back a sequence of class numbers, which allows later communication with COMBI and CRT05.

Class numbers are always got via COMBI, never directly from the system, so that COMBI can also release them at CASP termination.

If run time parameter 3 is #0 CCD and IHAP are not scheduled. This is a stand-alone mode to run CASP, useful for installation tests. Should COMBI and CRT05 not exist, CASP cannot even start. If CPENG scheduling or initialization fails, this results in a fatal error and CASP can only be terminated.

Should instead IHAP or CCD fail at schedule time or should CCD fail initializing, these are considered non-catastrophic errors and allow to proceed with CASP, although clearly no exposures with CCD can be done.

Should TCINT (TCS - Telescope Control System Interface) fail, this only shows that the link with the TCS computer is not OK. As this condition does not mean that CASPEC cannot be operated properly, full CASPEC operation is possible in any case. The only difference in further operation with CASP is that at start exposure time a form will be displayed asking to enter telescope coordinates and sidereal time, while these values are simply displayed on the rolling screen without pausing CASP operation, when the link with TCS is working properly.

Another important start-up operation is the retrieval of the installation tables by CASP.

SCHEDULING MECHANISM OF CASPEC

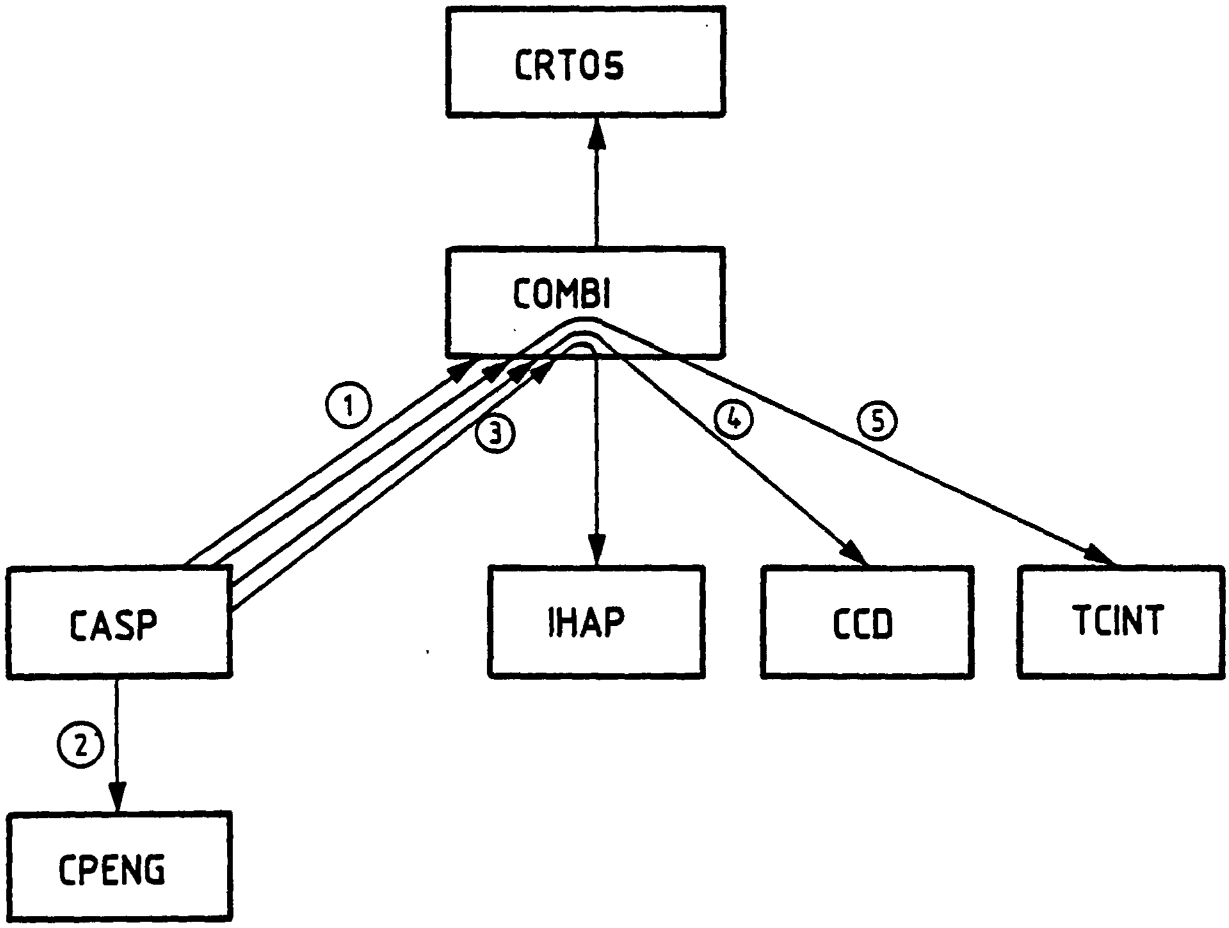


Fig. 5

First of all CASP looks for the LU's installation table 5 on file .PACP on the first mounted CR containing .PACP (it should be the TABLES CR = 33). Table 5 contains in turn a CR reference for auxiliary files (should be 33 again) to be used in further parameter file operations. Table 5 is retrieved with subroutine GETPR, all other tables with subroutine GETAL.

A third function of the start-up section of CASP is to initialize motors to a given "last run" setting. This is a way of checking that all functions are working properly. The actual initialization is done via the CPENG program (see later). Should some of the F's fail, re-initialization is attempted again but interactively, allowing the user to select or exclude certain functions.

After this, whatever the result, the main menu is displayed. Re-initialization can be done over and over from the main menu if needed.

The menus displayed by CASPEC and the actions corresponding to their softkeys are already described in the Operations chapter and Appendix A. This corresponds to the handling of user input. However CASP deals also with messages coming from CCD and COMBI and replies to commands sent to IHAP and CPENG.

The logic corresponding to the exposure definition and to the start exposure softkeys is explained here in greater detail.

Logic of Exposure Definition

Table 9 is filled-in with the wanted exposure parameters, while table 11 is displayed on the IHAP graphic terminal, used as an auxiliary display screen.

Table 7 and 1 are then filled in with the appropriate parameters from table 1, where table 1 is used to give a status display on request.

Parameters are then converted into table 8 values (initial set-up or engineering set-up table). This last table is the one actually used to drive CASPEC.

The reason for this is that table 9 does not contain all functions (e.g. Hartmann mask), being a higher level table for the user. Table 8 is instead filled in with suitably converted table 9 values, taking default values when necessary (e.g. Hartmann masks off) and gives a one to one correspondence with the 15 functions driven by CASP. The values of table 8 are the ones converted into encoder values and used to control the CASPEC motors.

If the sequence number defined is greater than Ø the definition of table 9 is also inserted in table 10 in the location corresponding to the sequence number specified.

Logic of Start exposure and Start sequence

Motors are put in position via CPENG, checking first if any of the present positions have to be changed on the basis of table 9 values.

Then one exposure (single or in a sequence) is started via a sequence of commands to CCD described under CCD operation. Switching on and off calibration lamps and shutter operation via CPENG (ON, OFF) follow respectively to the shutter open and shutter close commands from CCD (this last is given even for dark exposure).

After data have been written by CCD into the IHAP database, CASP writes instrument parameters into the IHAP header (see later under IHAP operation).

Program PARMG is scheduled by CASP in the Maintenance Menu to allow changes in installation tables. The recommended way is however to use PARMG off-line to this end.

CASP is not in control of the instrument console during exposures and IHAP usage. During exposures CCD gets terminal input. In a sequence of exposures, as CCD is not aware that this is a sequence, control is passed back to CASP via COMBI at every exposure end. However CASP disregards terminal input in this phase and starts immediately the next exposure passing then control again to CCD.

The only way to abort a sequence is therefore to abort an exposure in a sequence under CCD control.

The CASPEC logic dealing with CCD, IHAP, TCINT communication is explained better in the next sections.

All CASPEC tables are saved at termination in file .PACP. The original idea was to create different sets of tables where users could save their configuration for next run. This was tested but it proved simpler and better to use only one set of tables. Next time CASP is started the tables are read from disc as they were left last time CASP was used, unless of course they were modified via PARMG.

6.2 Program CPENG

Control of CASPEC functions, lamps and shutter is all done from the CPENG program via CAMAC. The CASPEC functions, associated with 15 DC motors and 13 encoders (Hartmann masks have no encoders) are listed in table 8 (functions set-up) (see Appendix F).

The control actions are normally started by a request coming via class I/o from CASP, but autonomous checks are also performed, like in the case of a periodic check on the motor positions every 60 secs during long exposures to verify that they are in position.

All communication between CPENG and CASPEC is via CAMAC.

The modules used are :

- 4 ESO motor controllers (capable of driving 4 motors each).
- One I/o register to drive the shutter.
- One I/o register to control calibration lamps and function codes readout.

The following table summarizes the commands and replies accepted and returned by CPENG via class I/o.

CASP/CPENG PROTOCOL

CASP command	CPENG reply
	On reply to every command A 3 words buffer is returned with IBUF (1) = 2HOK or 2HER IBUF (2) = IZDON = 0 if CAMAC Z was not done and/or needed again. = 1 if Z was done - Furthermore a status line message is output on the instrument console with status of motors, lamps and shutter (except for IN command).
IN - Initialize - IBUF (1) = 2HIN - Buffer of 40 words passed to CEPNG with &CPCOM common - Must be first command	IBUF(3) contains echelle and cross disperser function codes in upper/lower byte Z is done on CAMAC to reset motor controllers. (IBUF(2) should say so on reply)
MO - Move IBUF(1) = 2HMO IBUF(2) = INDON If = 0 initialization needs to be done. From IBUF(5) on the parameters of table 8 are given (wanted encoder values).	IBUF(3) = INMSK mask with functions successfully moved.
SH - Shutter IBUF(1) = 2HSH IBUF(2)= 2HOP - Open or 2HCL - close	IBUF(3) = ISTSH Status of shutter
LI - Calibration lamps IBUF(1) = 2HLI IBUF(2) = 2HON or 2HOF IBUF(3) = lamp no. (if = 0 all lamps).	IBUF(3) = ISTLA bit pattern with lamps status.

CASP/CPENG PROTOCOL

CASP command	CPENG reply
ST - Status of CASPEC All functions are checked to see if they are at wanted position. Shutter and lamp status are checked as well.	IBUF(3) = ICKMSK, bit pattern with functions in correct position. Messages to describe status of motors, lamps and shutter are displayed to terminal status line.
CK - Check CASPEC like ST, but only OK/ER reply wanted.	IBUF(3) = ICKMSK OK/ER reply not given when CK command is internally generated by CPENG (selftest mode during long exposures). Message to terminal status line.
MK - Motor check on/off IBUF(1) = ZHMK IBUF(2) = 2HON / 2HOF Enables/disables periodic check of motors.	No reply given to this command.
On all class I/O commands parameters 1 and 2 (IP1, IP2) are used as follows IP1 = class number for reply to CASP. (if < Ø CPENG terminates without reply) IP2 = source + destination numbers of CASP and CPENG. programs in upper/lower bytes (COMBI conventions).	On all class I/O replies parameters IP1 and IP2 are used as follows IP1 = 0 IP2 = source + destination of message. (CPENG + CASP number according to COMBI conventions).

In addition to the protocol described one should add that CPENG gets the installation tables of CASPEC by obtaining the parameter file name at IN init time and reading then the tables from disc. As a consequence every time installation tables are changed, e.g. by usage of PARMG, a new IN command is sent to CPENG.

This does not apply to user tables changed dynamically under CASP. The only relevant part of information for CPENG in this case is the table 8 content, saying where to position the CASPEC functions, and this is transmitted along with every MOVE command to CPENG.

Note that motor controllers need a Z on CAMAC for resetting. So Z to CAMAC is given by CPENG at INitialization time.

Messages from CPENG go either to the rolling part of the instrument console or/and to the instrument status line. CASPEC messages are not tagged to say if they come from CASP or CPENG, as this has no meaning for the user. A compound status line with status of functions, lamps and shutter appears after every CPENG command (except IN).

As said before, during long exposure a self-check on motors position is enabled every 60 secs. Motors must be at wanted encoder value +/- 1 in order to be considered in position. If an error occurs this is made clear to the user in the compound status line on the instrument console, but no active action is taken, i.e. the exposure continues.
CASP itself does not know of this at this stage. CASP is notified of occurred failures only at exposure end, and then it will warn the user and ask that functions are re-initialized. The exposure done is not lost.

When a user sees in the status line that an error occurred during an exposure he can simply press the status softkey under CCD to know more. CCD sends a status request to CASP and this in turn passes it to CPENG. Full information on function positions is given on status request.

6.3 Interface with CCD

The CCD program is described in the CCD additional documentation (D), by P. Biereichel.

The interface instrument/CCD is fully described in the additional documentation (I). CASPEC makes use of the protocol defined there to handle commands to and from CCD.

The CCD program takes control of the instrument terminal and displays its own menu once it receives the start exposure command EX. It gives back terminal control at exposure end.

Fig. 6 shows how the dialog between CASP , CCD (and IHAP) at exposure time is implemented.

CASP/CCD DIALOG

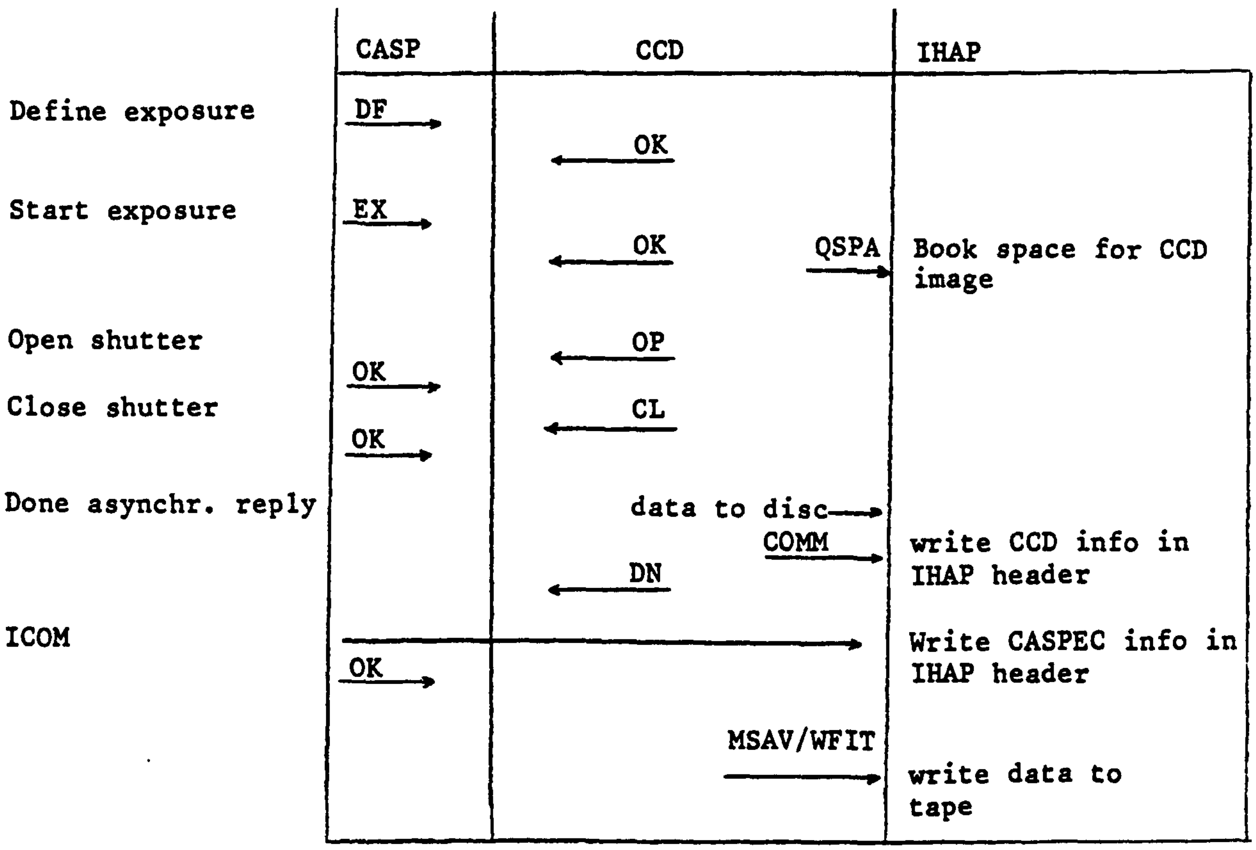


Fig. 6

6.4 Interface with IHAP

IHAP (see references) is scheduled by COMBI in the Data Acquisition System environment on CASP request. COMBI displays also the IHAP softkey menu. All commands to IHAP are via COMBI (terminal→CRT05→COMBI→IHAP).

When the user presses the softkey IHAP either under CASP or CCD, these programs pass terminal control to COMBI, which delivers user commands to IHAP via class I/o and displays replies on the rolling part of the instrument terminal.

IHAP is terminated only when CASP exits, not with the TERM command.

IHAP shares its database with CCD (see QSPA command to understand how space is reserved for CCD images).

The WCOM command is used by CCD to record CCD information in the IHAP comment area. The ICOM command is used by CASP to record CASPEC information in the extended directory.

The directory itself is filled in both by CCD and CASP. The format in which the information is recorded is by using a structure of FITS keywords as listed in Appendix G.

Subroutine WKEYS of CASP implements the dialog with IHAP to store this information.

The default format for tape recording is IHAP, but FITS is available as well.

In short, IHAP receives commands either from CCD, CASP or from the user (via COMBI).

A program called IHAP2 can also be installed for test purposes. The relevant files for IHAP2 are available on CR = 85. It can be run e.g. from the system console and passes commands to IHAP, once this has been scheduled by CASP, via COMBI. So two users can in principle work on the same database at the same time. Note however that while the main IHAP is suspended during CCD data readout from CAMAC, as this is timing critical, the same does not apply to program IHAP2. So its usage is recommended as a debugging tool but not for normal operation.

6.5 Interface with CRT05 and COMBI

The protocols with these programs are described both in computer readable documentation and in the additional documentation (E) and (F).

When COMBI is scheduled at start-up time by CASP it does not take control of terminal. This happens though when CASP is terminated via the Main Menu Terminate softkey.

When this is pressed CASP does not terminate yet, but simply gives terminal control to COMBI and keeps waiting for commands in class I/o.

COMBI at this point displays its own top menu containing the softkeys : CASP, CCD, IHAP and Terminate.

If Terminate is pressed then COMBI informs all active programs in the environment that they have to terminate. CASP in particular terminates CPENG before exiting, as for COMBI there is only one instrument program (CASP) and it does not know of the existence of CPENG.

The Top Menu of CASP allows to use once more IHAP. So IHAP can be called from 3 different Menus of 3 different programs : CASP, CCD and COMBI.

In all three cases the same IHAP menus are displayed (this is the reason to manage the whole interface with IHAP in COMBI).

If the CASP softkey is pressed CASP takes again terminal control and displays its top Menu. The previous initialization is still valid and the user can carry on with CASP operations.

However the main reason, apart from termination, to go to the COMBI menu with the Terminate key is probably to use then the CCD softkey. This gives terminal control to the CCD program, which will display its top menu. This is equivalent to use CCD in stand-alone mode and makes available to the user all the Menus and softkeys of CCD. This way of working might be practical for example to change CCD tables (with PARMG under CCD) while working with CASPEC. This occurs for example when a different binning factor on CCD is wanted.

Once CASP has given control to COMBI the only commands it expects back is either to terminate (IP1 = -1) or to continue. In this last case the same menu which was on the screen when terminal control was given up by CASP is displayed again.

This situation occurs not only when the Terminate softkey was pressed, but every time an exposure ends.

As explained in the section about CCD interaction, CCD receives terminal control when an exposure is started (and displays its own "Pause Exposure Menu"). At exposure end CCD gives back terminal control to CASP.

However this passing and receiving terminal control mechanism is handled by COMBI, i.e. CASP tells COMBI to give control to CCD and at exposure end CASP receives by COMBI a command to proceed and display again its previous menu.

The purpose of COMBI should now be clearer ; it allows and controls terminal access to the instrument and detector programs and handles common data acquisition functions.

6.6 TCINT program

TCINT handles the interface with the telescope control system. In the present installation of CASPEC at 3.6 m telescope, TCINT deals with the 3.6 m telescope control system based on RTE-4E via the Suter link.

CASPEC does not have any active control over the telescope. The link is only used to retrieve right ascension, declination and sidereal time for every exposure.

Should CASPEC be installed on a different telescope, the TCINT program will have to be the appropriate one for that telescope. However CASPEC will not have to be changed at all, as the interface instrumentation software-TCINT is instrument and telescope independent.

TCINT has never control of the instrument terminal, nor displays any message on it.

7. TEST PROGRAMS

The test programs used by CASPEC are stand-alone programs which can be run directly e.g. from system console or under CASP via the Maintenance Menu function keys.

The maintenance Menu is password protected by password CP.

In particular programs TMOST, TMINI, TMHND are general purpose test programs for motors (though specifically written for CASPEC). They are fully documented in the MOTOR library, (see enclosed documentation - B).

To get a general description of what they mean and how they work refer to CASPEC User interface Appendix A, or use on-line the Help softkey in the maintenance menu.

All these programs can be simply run without parameters in stand-alone mode and will work interactively. Using them under CASP one has the advantage that some parameters, like terminal and CAMAC LU's, are directly passed to test programs with the run string and the user has not to know all this.

TCAM is a CAMAC general purpose test program.
CPCKS and TMOH are instead tests, which are CASPEC specific. Note that TMOH, used to test Hartmann mask, is so rarely used that it is not available under CASP in the Maintenance Menu. It is not even loaded normally with CASPEC. To load it : RU, LOADR, @ CP::85,%TMOH::85

Program TMOST does not interfere with CASP work and can be run in parallel e.g. from system console, while TMHND might interfere and TMINI certainly does, as it resets all the motor controllers with Z. Be aware of this when running tests from system console, while CASPEC is running.

8. CASPEC AND REMOTE CONTROL

Although the CASPEC package is not implemented to be directly remotely controlled (RC) some concepts dealing with RC have been kept in mind in its implementation.

This should allow an easier transition to a possible remote controllable version of the instrument software.

In our case the remote control environment is defined as a connection between two computers, one near the user (La Serena, Garching) and one near the instrument in La Silla.

The CASPEC and the CCD packages have been implemented in a way that makes possible the separation into two parts, one near the user and one near the instrumentation (in practice near CAMAC).

CASPEC in fact consists basically of two packages CASP and CPENG. The interface between the two programs is simple and very few words are exchanged. In a remote control environment the class I/o interface will be substituted by a link interface.

CASP should run together with CRT05, PARMG etc at the user end, CPENG at the instrument end.

The same applies to CCD where CCD deals with the user interface and DAQ + WCHDG deal with the instrument.

This implementation shall allow Remote Control experiments with lines at a fairly low baud rate. At the same time the same "high level" interface can be offered to the remote user, thanks to the fact that Menus and forms handling can be dealt with entirely by the user end computer. However further work has to be done both on CASPEC and CCD packages to make them remotely controllable.

9. CASPEC FORMS and SOURCES

CASPEC forms can be listed with command file :

```
(:)TR,*PFORM::85
```

CASPEC sources (not libraries) can be listed via :

```
(:)TR,*ELIST::85
```

10. COMPUTER READABLE DOCUMENTATION

To list these files use utility PR :
(:)RU,PR,Name::CR:-1

- CASPEC user interface (Appendix A) : file "CPUSR
- CASPEC software troubleshooting (Appendix E) : File "CPHLP
- CAMAC documentation and tests : Files "CAMAC and "TCAM
- CCD : file "CCD
- CRT05 : file "CRT05
- COMBI : file "COMBI
- DAQLB : file "DAQLB
- Motor library : file "MOTOR
- IOLIB : file "IOLIB
- "GIHAP : Installation of IHAP
- "IMG01 : IHAP manual text

11. REFERENCES

- Cassegrain Echelle Spectrograph
  ESO - User's Manual

- Cassegrain Echelle Spectrograph
  ESO - Technical Report - Part 1

- The new Data Acquisition System for ESO instrumentation
  The Messenger, Dec. 83

- IHAP manual, by F. Middelburg
  ESO - June 83

CASPEC USER INTERFACE

The CASP program interfaces with the user via a 2 levels structure of Menus. While the screen layout is that foreseen by the Terminal Handler interface, the various menus are displayed at the top of the screen and the 8 white fields correspond to the 8 function keys of an HP2645 terminal.

MAIN MENU:

OBSERVATION	Telescope setting	DISPLAY PARAMETERS	Help
f1	f2	f3	f4
Re-initialise F's	MAINTENANCE	Status	Terminate
f5	f6	f7	f8

f1-Observation:
Displays the Observation menu (second level).

f2-Telescope setting:
Displays telescope coordinates. Should the link with the TCS system not work, those are the last coordinates got.
Editing is not possible here, but will be possible when an exposure is started, should the link still fail.

f3-Display parameters:
Displays the Display parameters menu (second level).

f4-Help:
Provides explanatory text for softkeys in a given menu

f5-Re-initialise F's:
All or a subset of the functions are initialised (Z on CAMAC).
Worth using if something goes wrong with motors, instead of terminating CASPEC and restarting it again.

f6-Maintenance:
Activates the Maintenance Menu(second level).

f7-Status:
Checks all CASPEC motors against the wanted positions and displays status of calibration lamps and shutter.
It takes some time to get the encoder values from CAMAC, particularly for those motors which are not initialised, as encoder values are then read directly from the encoder NIM module.
During CCD exposure the Status key in the CCD menu has to be pressed twice to get full-status display, so by making sure that there are at least 30 secs before exp. end one is sure not to introduce delays in the closure of shutter.

f8-Terminate:
Control is passed to program COMBI, while CASP is in a suspended state.
If CASP is re-started from COMBI initialisation does not need to be done again.

MAINTENANCE MENU: (activated by Maintenance function key)

Change parameters	Lamps, shutter...	Test CAMAC	Help
f1	f2	f3	f4
Move motors	Check motors	Handset move	Previous menu
f5	f6	f7	f8

f1-Change parameters:
Starts program PARMG to display, list or change tables.
PARMG (parameter manager) can also be used in stand -alone mode(on 2645).

f2-Lamps, shutter...:

Starts CPCKS program,which can control shutter,calibration lamps and give a listing of function codes(identifiers for funct. connectors). CPCKS can also be run in stand-alone mode.

f3-Test CAMAC

Starts program TCAM,which can be run also as a stand-alone CAMAC test.

f5-Move motors:

Starts TMINI program to initialise up to 4 motors in a controller. Note that after this all other motor controllers will be reset (by Z). TMINI can also be run as a stand-alone motor controller test.

f6-Check motors:

Starts TMOST program,which gives encoder readouts and status of all motors connected to a motor controller. This program,which can also be run in stand-alone mode,does not affect the positions of motors. Encoder values are read also if the correspond. motor controller is not initialised; in this case the value is not read from the controller but directly from the NIM encoder module. TMOST is a program detached from CASP and does not know whether the encoder values got are the ones wanted or not. For a global test on all motors including wanted position check use the Status key.

f7-Handset move:

Allows to use handset on motors and to revert to computer control. Really an extreme tool!.

DISPLAY PARAMETERS MENU: (activated by Display parameters function key)

Filter wheel tab.	Slitdeckers tab.	Collim. etc.. tab.	Help
f1---------------	f2--------------	f3---------------	f4---------------
Motors vs funct's	Functions set-up	Spectral table	Previous menu
f5---------------	f6--------------	f7---------------	f8---------------

OBSERVATION MENU: (activated by Observation function key)

Define exposure	Display sequence	IHAP	Help
f1---------------	f2--------------	f3---------------	f4---------------
Start single exp.	Start sequence	Status	Previous menu
f5---------------	f6--------------	f7---------------	f8---------------

Note: The function keys Start single exposure and Start sequence give control of the terminal to the CCD program which in turn displays a Menu allowing various options during exposure(IHAP, STATUS, ABORT, . . .)
IHAP also can be operated by using function keys. It is a sequence of menus at same level linked by the ETC. function key. The IHAP menus are the same if IHAP is run under CASP or CCD

f1-Define exposure:

Definition of either single exposure or sequence of exposures. Note: When starting CASP the definitions contained in the parameters file are read. However the single exposure definition has to be validated before starting exposure by pressing this key. Otherwise CASP will not accept to start exposure.

f2-Display sequence:

When a sequence of exposures has been defined by the previous softkey and the user has declared that he has no more exposures in the sequence, all the existing definitions above the max no.

of exposures wanted will be cancelled.

f3-IHAP:

Calls IHAP (already started). IHAP cmds can be passed either via rolling screen or using function keys.

f4-Start single exp.:

Sequence of actions:

Motors are put in position (according to definition)
(possibly) calibration lamp is switched on
Control is passed to CCD program, which displays its own menu.
Complete info. on CASPEC can be got any time by pressing Status (twice)
Motors are automatically checked against positions every 60 secs (for exposures>90 secs)
Calibration prism is put back to star position after exposure.
(Possibly) cal. lamp is switched off at shutter closing time.

f5-Start sequence:

The same sequence of actions as for single exposure, but the calib. switch is moved back to star only at the end of the sequence.
The current exposure definition is updated on the 2648 screen while the execution of the sequence proceeds.

"CPACK 83/ 8/29 16:19: 5

THE CASPEC PROGRAMS

THE CASPEC package consists in a set of on-line programs, which cooperate and form the so called data acquisition environment (DAQ).

They are:

CASP	handles user end and logic of CASPEC
CPENG	CP engine : acts on CAMAC to control motors, lamps, shutter
TCINT	Interface program to TCS
CCD	CCD user end
DAQ	The CCD data acquisition program
WCHDG	A watch dog program to prevent CCD hang-ups
COMBI	A program to connect together instr. and detector pack.
CRT05	Terminal handler, to cope with softkeys and forms
PARMG	Parameter manager, to handle CASPEC and CCD tables
IHAP	

Note: CCD, IHAP, PARMG, can be used as stand alone programs. For PARMG operation remember that the form file name for CASPEC is .FOCP:CP:33

Off-line programs: A number of off-line programs are also part of the CASPEC package and can also be used in stand alone mode to do various functions.

They are:

CCTST	Test of CCD CAMAC and link
CPCKS	Checks lamps, shutter, function codes for CASPEC
TMINI	Motor controller initialization.
TMOST	Motor controller positions
TCAM	Test CAMAC in read/write and DMA
TMOH	Move Hartmann mask
TMHND	Use handset on motors

```
*CASP  T=00004 IS ON CR00001 USING 00005 BLKS R=0000

:* Transfer file to load CASPEC package (assumes DAQ and CCD loaded)
:* (to load DAQ +CCD use *DAQCC::75)
:OF,CASP
:OF,CPENG
:RU,LOADR,@CP::85,%CASP::85
:PU,CASP:CP:3
:SP,CASP:CP:3
:RU,LOADR,@CP::85,%CPENG::85
:PU,CPENG:CP:3
:SP,CPENG:CP:3
:* following is TCINT program for 3.6m (@TC360)
:OF,TCINT
:RU,LOADR,@TC360::85
:PU,TCINT:CP:3
:SP,TCINT:CP:3
:* RU,LOADR,@PARMG::85
:* PU,PARMG:CP:3
:* SP,PARMG:CP:3
:* type TR to (re)load all of them
:TR,1
:* copy tables to tables CR=33
:* (assumes tables are updated,else use ST instead of DU cmd)
:* DU,.PACP::85,.PACP:CP:33
:* DU,.FOCP::85,.FOCP:CP:33
:* DU,.SKCP::85,.SKCP:CP:33
:OF,CPCKS
:RU,LOADR,@CP::85,%CPCKS::85
:PU,CPCKS:CP:3
:SP,CPCKS:CP:3
:OF,TMINI
:RU,LOADR,@CP::85,%TMINI::85
:PU,TMINI:CP:3
:SP,TMINI:CP:3
:OF,TMOST
:RU,LOADR,@CP::85,%TMOST::85
:PU,TMOST:CP:3
:SP,TMOST:CP:3
:OF,TMHND
:RU,LOADR,@CP::85,%TMHND::85
:PU,TMHND:CP:3
:SP,TMHND:CP:3
::
```

```
OP,LB
OP,DC    * prevent multiple copies
OP,CP    * current page linking
LI,%IOLIB:CP:85
LI,%FMLIB:CP:85
LI,%MOTOR:CP:85
LI,ZRCAM::32766
SL
* following modules use (local named) common
** RE,%CASP:CP:85
** RE,%CPENG:CP:85
SE,%MOVMS:CP:85
SE,%CAMDF:CP:85
SE,%ZMOTS:CP:85
SE,%GETAL:CP:85
* SE,%CKSEQ:CP:85
* follow.modules do not use common
SE,%LINKP:CP:85
SE,%SCPRG:CP:85
SE,%ASKMT:CP:85
SE,%CKLIM:CP:85
SE,%EVALS:CP:85
SE,%NEWPS:CP:85
SE,%CKCAM:CP:85
SE,%TCST:CP:85
SE,%CKLNK:CP:85
SE,%LKTST:CP:85
SE,%GETEQ:CP:85
* SE,%PARFG:CP:85
SE,%IACTF:CP:85
SE,%RUPRG:CP:85
SE,%GETSK:CP:85
SE,%MTHWT:CP:85
SE,%CKIOL:CP:85
SE,%MOTGT:CP:85
SE,%WTMOT:CP:85
SE,%PMSG:CP:85
SE,%CEXP:CP:85
SE,%CMDHS:CP:85
SE,%DFORM:CP:85
SE,%PRTST:CP:85
* following modules use (local named) common
SE,%GTFUN:CP:85
SE,%MOVM:CP:85
SE,%FILTW:CP:85
SE,%CODCK:CP:85
SE,%TLCOM:CP:85
SE,%IIPT7:CP:85
SE,%CVF78:CP:85
SE,%UPD13:CP:85
SE,%WKEYS:CP:85
SE,%GTGRV:CP:85
SE,%PRTPS:CP:85
* follow.modules do not use common
SE,%CFMSK:CP:85
SE,%MOVE:CP:85
SE,%ASKFN:CP:85
SE,%SHUTT:CP:85
```

```
SE, %LIGHT:CP:85
SE, %GTMPS:CP:85
SE, %CKWPS:CP:85
SE, %CNARC:CP:85
SE, %IHPCM:CP:85
SE, %IACTF:CP:85
SE, %TIMCV:CP:85
*
SE, %FORM4::75
SL
SE
END
```

*HICP 83/ 8/29 16:15:?

```
:SV,4,,IH
:DP,mhJ&a+15C&dBWelcom to use the CASPEC spectrograph&d@
:DP,
:DP,The DAQ system works with IHAP (June 83)-ESO standard 2301 system.
:DP,This has to be run from the instrument console (Logical unit 12).
:DP,
:DP,A connection to CAMAC helps to get the CASP main menu displaied,
:DP,but having got CAMAC you do not need CASPEC to get a feeling of how
:DP,the program works (the computer though will be frustrated...).
:DP,
:DP,An on line HELP key on every menu level tells you how CASP works.
:DP,
:DP,To try CASPEC without CCD and without IHAP use (:)RU,CASP,,,1
:DP,To do partial tests try TCAM,TMINI,TMOST,TMHND,CPCKS,CCTST,CCD,IHAP
:DP,  in order of increasing complexity.
:DP,
:DP,For off line LU changes (e.g.initial error on LU's and CASP aborted)
:DP,run (:)PARMG on form file .FOCP:CP:33...,but cuidado !.
:DP,
:DP,To abort type (:)TR,*ZCP from system console and logon again as CASP.
:DP,
:DP,To continue with a normal CASPEC session type (:)TR
:TR,1
:SYTO,13,0,,,,    TO=0 on IHAP graphic terminal
:SYTO,12,0,,,,,   TO=0 on instrument console
:TR,RPDAQ::75
:TR,RPCASP::2
:RU,CASP
:SYOF,TCINT,1
:TR,OFDAQ::75
:TR,OFCASP::2
:OF,IHSPL
:DP,m
:EX
::
::
```

TROUBLESHOOTING PROCEDURE FOR CASPEC

LU assignement errors at start-up time

A frequent problem is the occurrence of the message: Ready LU 17, meaning that the graphic terminal is stuck. Hit the RETURN key on the graphic terminal to unlock it.
Should this not work, hit the RESET TERMINAL key consecutively twice on same terminal and then the RETURN key: this will cure the problem.
This point is relevant because at initialisation time CASP does not proceed if the graphic terminal is not available.

CASP or CCD stuck

In case of CASP or CCD programs being stuck
do the following:

(from system console) (:):*ZCP::85

This is an abort and restart procedure file that will allow to run then (:)CASP again from CASPEC console.
Let this procedure file run up to completion before starting CASP again.
Should it be stuck somewhere apply comments which appear on screen and remember that to further proceed with a procedure file one has to type (:)TR

HOWEVER it is not always easy to get the system console FMGR prompt :, in order to enter :*ZCP::85.
If this is the case then a number of programs have to be aborted "by hand" and then the prompt will come up again.
Here are some hints on how to proceed:

```
Hit the RETURN key to get the * prompt
(*)OF,DAQ,1
(*)OF,CCD,1
(*)OF,CPENG,1
(*)OF,CASP,1
(*)OF,CRTO5,1
```

At this point either the FMGR prompt comes up on its own or you can get it up by: (*)ON,FMGR.
Should the system reply with an INVALID STATUS error message, try the following: (*)OF,FMGR,1
(*)ON,FMGR

Once you get the FMGR prompt : type :*ZCP::85, as explained before.

WHAT ELSE ?

Should all the above remedies fail do not despair.
If having done all this and restarted CASP, this still misbehaves, e.g giving LU errors, terminate it and type (:)EX, exiting your session.
Logging in as CASP reassigns all logical units and might fix the problem.

...and LAST

Rebooting the computer is the extreme remedy. Enter date on system console and wait up to end of bootup procedure before logging on at CASPEC console as CASP.
However this will not help fixing problems with CAMAC or peripherals.

CASPEC TABLES

INSTALLATION TABLES:	2	Filter wheels, calibration lamps
(read-only for user)	3	Slit and deckers
	4	Rear slit viewer, collimator, cross disperser
	5	CASPEC LU's and CAMAC stations
	6	Cabling and function codes
	11	Spectral table
	13	History table
USER DEFINED TABLES:	1	Instrument status
(set-up under CASP)	7	Instrument set-up
	8	Instrument engineering set-up
	9	Exposure definition
	10	Summary of exposures
	12	Telescope set-up

CASSEGRAIN ECHELLE SPECTROGRAPH * Instrument status * FORM 1 *

CASPEC 3.6 m telescope installation

Detector mounted CCD RCA

Camera SHORT focal length 281 mm

Gratings cross disperser: grooves/mm = 300 blaze angle = 4.3

echelle : grooves/mm = 0 blaze angle = .0

INSTRUMENT SETTING:

Light source 5 0:STAR, 1..5:calibration lamps

1:quartz, 2:neon, 3:Hg, 4:Fe-Ne, 5:thorium

Slit width 300 microns or 2.0 arcseconds

Slit length 700 microns or 4.8 arcseconds

Central wavelength (Lambda 0) 500 nm (300,1100)

Collimator focus - encoder abs.value 1496 (0,2047)

Neutral filter 0 (0,5)

Colour filter 0 (0,5)

Calibration filter 0 (0,5)

Photometric filter 0 (0,5)

Rear slit viewer 0 (0=off, 1=on, 2=Zeeman)

* CASSEGRAIN ECHELLE SPECTROGRAPH * Filter wheels & calib. lights * FORM 02
There are 6 equispaced positions (where pos. O=no filter)
Offset for posit. O is adjustable (0...2047)(11 bit ser. encoder)

	OFFSET	Filter No. s	Timeout(5-99 secs)
Neutral filter	979	1 2 3 4 5	10
Colour filter	396	1 2 3 4 5	10
Calibration filter	2013	1 2 3 4 5	10
Photometric filter	580	1 2 3 4 5	5

Calibration switch wheel-Offset for star 1582

- Offset for lamp 197

- Timeout(5-99 secs) 10

Calibration source wheel
Offsets for :0:blind pos., 1:quartz, 2:neon, 3:Hg, 4=Fe-Ne, 5:thorium

0	1	2	3	4	5
150	698	1402	1202	417	964

- Timeout(5-99 secs) 10

* CASSEGRAIN ECHELLE SPECTROGRAPH * Slit and deckers table * FORM 03 *

PRESLIT DECKER

Positions:	0=all out	1	2	3	4	5	6=all over
	789	616	541	466	394	323	250

- Offset (0,2047) 0
- Timeout (5-99 secs) 10

SLIT (both sides are driven by same motor)

- Position when closed 947
- Max aperture positon(~2mm) 399
- Offset (0,2047) 0
- Timeout (5-99 secs) 10

Total width coeff.=2.817 micron/encod. unit
Seeing coeff. =144 micron/arcsec

DECKERS No.1(up) and No.2(down)

	Closed	Max aperture	Offset
Decker no. 1 -	1584	600	550
Decker no. 2 -	1567	330	600

- Timeout (5-99 secs) 10

One decker width coeff.=13.34 micron/encod. unit

```
* CASPEC              * Rear slit viewer,collimator,cross disp.  * FORM  04 *
REAR SLIT VIEWER:
    ON  position            1344  (0,2047)
                            ----
    OFF position              16  (0,2047)
                            ----
    Zeeman position         1750  (0,2047)
                            ----
    Offset                  1200  (0,2047)
                            ----
    Timeout                   10  (5,99)
                              --
COLLIMATOR:
    Low end position          55  (0,2047)
                            ----
    Top end position        1507  (0,2047)
                            ----
    Offset                     0  (0,2047)
                            ----
    Timeout                   30  (5,99)
                              --
CROSS DISPERSER:
    Position at blue end 1663  Position at red end  123  (0,2047)
                         ----                       ----
    Offset   (0,2047)       0  Timeout (5-99)        90
                         ----                        --
  Coeff.A,B,C: TL=A+B*LO+C*LO**2  (TL=cr.disp.tilt,LO=central wavel.)
    (with short camera)  A 1369.87304  B    -1.27083  C       .00000 (F10.5)
                           ----------     ----------    ----------
    (with long camera)   A 1369.87304  B    -1.27083  C       .00000 (F10.5)
                           ----------     ----------    ----------
HARTMAN MASKS:
    Timeout                   30  (5,99)
                              --
ECHELLE:
    Position                 550  (0,2047)(manual adjustement)
                            ----
```

* CASSEGRAIN ECHELLE SPECTROGRAPH * LU'S and CAMAC table * FORM 05 *

General installation parameters :

Telescope identifier 3.6 (3.6,2.2)

Detector mounted 1 1:CCD,2:Photographic plate,0:none

Camera 1 1:short(281mm) 2:long(560mm)

LOGICAL UNITS :

System LU for CASPEC I/O terminal	LUSCP	0	0=any LU ok
CR ref.number for auxiliary files	ICPCR	33	
CAMAC LU for CASPEC crate	LCAMC	46	
LU for link with TCS (0=no link)	LUTCS	26	

CAMAC STATIONS :

Status register station	NRSTS	13
Shutter register station	NSTSH	12
Motor controller 1 station	NMOT1	14
Motor controller 2 station	NMOT2	15
Motor controller 3 station	NMOT3	16
Motor controller 4 station	NMOT4	17

* CASSEGRAIN ECHELLE SPECTROGRAPH * Cabling and F. codes table * FORM 06 *

CABLING:

	FUNCTIONS (1,16)					FUNCTIONS (1,16)			
Contr. 1(motors 1,4)	9	2	1	3	Contr. 2(motors 5,8)	4	6	7	8
Contr. 3(motors 9,12)	5	10	12	13	Contr. 4(motors 13,16)	16	14	15	11

FUNCTION CODES (1,31):

F. 1 Neutral filter	3	F. 2 Calib. switch	2
F. 3 Colour filter	16		
F. 5 Preslit decker	9	F. 6 Slit width	6
F. 7 Decker no. 1	7	F. 8 Decker no. 2	8
F. 9 Calib. filter	1	F. 10 Slit viewer	10
F. 11 (Not used)	25	F. 12 Calib. sources	14
		F. 14 Collimator	4
F. 15 Hartmann msk1	24	F. 16 Hartmann msk2	23

CODES (0=no more, 1...31) (for F's with multiple codes):

F. 4 Echelle	- Codes	20	0	0	0
	grooves/mm	32	0	0	0
	blaze angle	63.4	.0	.0	.0
F. 13 Cross. disp.	-Codes	22	0	0	0
	grooves/mm	300	0	0	0
	blaze angle	4.3	.0	.0	.0

INSTRUMENT SETTING:

Light source 5 0:STAR,1..5:calibration lamps

1:quartz,2:neon,3:Hg,4:Fe-Ne,5:thorium

Slit width 300 microns or 2.0 arcseconds

Slit length 700 microns or 4.8 arcseconds

Central wavelength (Lambda 0) 500 nm (300,1100)

Collimator focus - encoder abs.value 1496 (0,2047)

Neutral filter 0 (0,5)

Colour filter 0 (0,5)

Calibration filter 0 (0,5)

Photometric filter 0 (0,5)

Rear slit viewer 0 (0=off,1=on,2=Zeeman)

* CASSEGRAIN ECHELLE SPECTROGRAPH * Functions set-up * FORM 8 *

FUNC.	Description	Value
F.1	Neutral filter	0 (0,5)
F.3	Colour filter	0 (0,5)
F.9	Calib. filter	0 (0,5)
F.11	Photometric filter	0 (0,5)
F.2	Calib. switch	OF ON/OF=star
F.5	Pre-slit decker	0 0=all out ...6=max width
F.6	Slit width	300 microns or 2.0 arcseconds
F.7	Docker 1 half width	350 microns
F.8	Docker 2 half width	350 microns
F.10	Rear slit viewer	0 0=off, 1=on, 2=Zeeman
F.12	Calib. source	5 1: quartz, 2: neon, 3: Hg, 4: Fe-Ne, 5: thorium, 0: no lamp
F.13	Cross disp. tilt	734 encoder value(0,2047) (+)
F.14	Collimator position	1496 encoder value(0,2047) (+)
F.15	Hartman mask 1	OF ON/OF
F.16	Hartman mask 2	OF ON/OF

(+): Must be within limits of appropriate function table

```
* CASSEGRAIN ECHELLE SPECTROGRAPH * Exposure definition  * FORM  09 *

Rel. sequence # 0 0=single exp., 1...8=sequence
                -
Exposure type  RE RE=regular,FF=CCD flat f.,DK=dark c.,NO=no CCD,$$=No F's
               --                                                     mode
Exposure time  0  0  1 hours(0-8),mins(0-59),secs(0-59)
               - -- --
Tape recording 1       0=Off,1=IHAP format,2=FITS format
               -
Sequence number for IHAP   38
                         -----
Identifier     CL 223311  500
               ---------------

INSTRUMENT SETTING:

Light source 5 0:STAR,1..5:calibration lamps
             -
               1:quartz,2:neon,3:Hg,4:Fe-Ne,5:thorium
Slit width    300 microns  or  2.0 arcseconds
             ----              -- -
Slit length   700 microns or   4.8 arcseconds
             -----            --- -
Central wavelength (Lambda 0)  500  nm (300,1100)
                               ----
Collimator focus - encoder abs. value 1496 (0,2047)
                                      ----

Neutral filter       0  (0,5)
                     -
Colour  filter       0  (0,5)
                     -
Calibration filter   0  (0,5)
                     -
Photometric filter   0  (0,5)
                     -
Rear slit viewer     0 (0=off,1=on,2=Zeeman)
                     -
```

* CASSEGRAIN ECHELLE SPECTROGRAPH * Summary of exposures * FORM 10 *

Exposure definitions							Instrument setting									
#	TP	H.	mm.	ss	MT	Seq#	Identifier	S	Wdt	Hgth	Lam.	Foc.	N	C	L	P	V
1	RE	0.	0.	1	1	35	CL EPS ERI 500	5	300	700	500	1496	0	0	0	0	0
2	RE	0.	0.	8	1	36	FF EPS ERI 500	1	300	700	500	1496	4	0	0	0	0
3	RE	0.	0.	8	1	37	FF EPS ERI 500	1	300	700	500	1496	4	0	0	0	0
4	DK	0.	0.	8	1	38	DK EPS ERI 500	0	300	700	500	1496	0	0	0	0	0
0		0.	0.	0	0	0		0	0	0	0	0	0	0	0	0	0
0		0.	0.	0	0	0		0	0	0	0	0	0	0	0	0	0
0		0.	0.	0	0	0		0	0	0	0	0	0	0	0	0	0
0		0.	0.	0	0	0		0	0	0	0	0	0	0	0	0	0

#=rel. seq. (1,8) TP=type H.mm.ss=expos. time MT=mag. tape(0=no)
Seq#=sequence for IHAP
S=light source Wdt=slit width Hgth=slit height Foc.=collim. focus
Filters: N=neutral C=colour L=calib. l. P=photometric V=slit viewer

* CASSEGRAIN ECHELLE SPECTROGRAPH * Spectral table * FORM 11 *

Characteristic wavelengths of filters and CASPEC

NEUTRAL FILTER:

Number	0	1	2	3	4	5
Density	free	2.6	1.33	0.90	0.33	0.33

COLOUR FILTERS:

Number	Wavelengths	Recommended central wavelength
0	free	
1	from 610 to 999	0 nm
2	from 580 to 999	0 nm
3	from 520 to 639	580 nm
4	from 400 to 560	480 nm
5	from 426 to 502	466 nm

OVERALL BANDWIDTH of CASPEC:

Centre	yields from	to
400 nm	340	462 nm
450 nm	390	556 nm
500 nm	449	550 nm
550 nm	500	600 nm
600 nm	550	650 nm
650 nm	600	695 nm
700 nm	650	740 nm

TELESCOPE SETTING:

	HH mm ss.t		SDD mm ss.t
Right ascension	23 47 45.5	Declination	-6 24 42.0
			HH mm ss.t
Airmass	.000	Sidereal time	1 3 40.8

CASSEGRAIN ECHELLE SPECTROGRAPH * History tables * FORM 13 *

Total exposure time with CCD 31 hours 3 mins

Total number of exposures 486

CALIBRATION LAMPS:

Lamp	No. of exp's	Total exp. time (Hours	mins)	Install. date (YY	MM	DD)	MTBF (Hours)
1-quartz	61	0	9	83	6	15	30
2-Neon	0	0	0	83	6	15	30
3-Hg	5	0	0	83	6	15	30
4-Fe-Ne	0	0	0	83	6	15	30
5-thorium	33	0	0	83	6	15	30

April 27, 1983

FITS Keywords

The following list of keywords has been suggested for ESO instruments and detectors.

INSTRUMENTS

Keyword	Description	Type	Unit
INSTRUME	Name of instrument	char.	-
BLAZANGL	Grating blaze angle	real	deg.
GRATFREQ	Grating groove period	real	m
CAMRLEN	Camera focal length	real	m
FLTRNR	Color filter number	int.	-
NDFLTR	Neutral density filter	real	D.Dens
SLITWDT	Slit width	real	m
SLITLEN	Slit length	real	m
COLLPOS	Collimator focus encoder position	real	-
WAVELENG	Central wavelength	real	m
HARTPOS	Hartmann mask position	int.	-
LAMPNR	Calibration lamp number	int.	-

DETECTORS

Keyword	Description	Type	Unit
DETTYPE	Name of detector type	char.	-
DETNAME	Name of detector	char.	-
EXPTYPE	Type of exposure (dark curr.,flat field,..)	char.	-
DETMTEMP	Mean temperature of detector	real	deg.K
DETDTEMP	Temperature drift during exposure	real	deg.K
DETBLTIM	Time of bias light exposure	real	sec
DETDKTIM	Total dark current time	real	sec
DETSTAT	Status of detector and data acquisition	char.	-

"DAQLB 83/ 8/28 15:22:

TECHNICAL NOTE TPE/GR/830327/19

Edited by G. Raffi ESO-Garching

```
-------------------------
!                       !
!                       !
!     DAQLB library     !
!                       !
!                       !
!                       !
-------------------------
```

DAGLB library

830827

G. Raffi ESO-Garching

Update history: Preliminary collection GR 830827

Library of subroutines in the Data Acquisition System Environment.

INDEX:

Introduction

This library is intended as an expandable library of subroutines, with the aim to help and make faster the development of programs within the ESO Data Acquisition System.
All programmers are strongly encouraged to contribute their "portable" subroutines to this library.
The initials on the header line are not necessarily the ones of the author of the original version: "stealing" and readaptation of subroutines is in fact encouraged with the aim of saving time and making as many subroutines as possible available to the programmers community.

You are also invited to use some form of standard header, describing the subroutine calls as standard programming practice. This will also make easier the documentation, as you can simply merge the top part of your subroutines into this document.

1. List of subroutines

1.1 Program to program communication

SCPRG	Check PRG status and schedule	GR 830614
CKPRG	Check PRG status	GR 830614
LINKP	V1.0 Link program to COMBI	PB 830216
IHPCM	Send msg to IHAP	GR 830804

1.2 Equipment handling

CKCAM	Check CAMAC	GR 830615
GETEQ	Get EQT numb. for given session LU	GR 830614
CKIOL	Check I/O LU's(direct-or class-I/O)	GR 830614

1.3 Others

TIMCV	Convert secs into hh,mm,ss,tens	GR 830609

2. Subroutine calls

2.1 Program to program communication

```
      SUBROUTINE SCPRG(IPNAM,IER,LU,IP1,IP2,IP3,IP4,IP5)
     .,Check PRG status and schedule           GR 830614
C
C parameters: IPNAM(3)=6 chars program name
C              IER     =0 program OK(ready to be scheduled)
C              LU      =LU or class # for messages
C                       (like Restoring program)
C
C adapted from CCD subroutine SCRT5 by P.B.
C
C -----------------------------------------------------------------------
C
      SUBROUTINE CKPRG(IPNAM,IER,LU)
     .,Check PRG status                          GR 830614
C
C parameters: IPNAM(3)=6 chars program name
C              IER     =0 program OK(ready to be scheduled)
C              LU      =LU or class # for messages
C                       (like Restoring program)
C
C adapted from CCD subroutine SCRT5 by P.B.
C
      SUBROUTINE LINKP (PARAMS,ICLMY,IPRGNR,ICLFTH,IPRFTH)
     .    ,V1.0 Link program to COMBI                   PB 830216
C
C
C     SUMMARY : Link program to COMBI's enviroment
C
C
C     BY       : Peter Biereichel ESO/GAR
C
C     DATE     : 16  FEB., 1983
C
C     KEYWORDS:
C
C     PACKAGE : Detector and instrument programs
C
C---
C
C REQUIREMENTS:  Program COMBI
C
C INSTALLATION:
C
C     USAGE    :  Must be the first executable statement in the program
C
C DESCRIPTION :
C
C---
C
```

```
C INPUT          : --
C
C OUTPUT         :     PARAMS : 20 word buffer
C                          (IPARM( 1),ICRASH),   ! CRASH FLAG (#O: CRASH OCCURED)
C                          (IPARM( 2),ICLOT ),   ! INPUT MAILBOX OF COMBI
C                          (IPARM( 3),ICLSTS),   ! CRTOS STATUS MAILBOX
C                          (IPARM( 4),ICAPOS),   ! CRTOS INPUT MAILBOX
C                          (IPARM( 5),ICOSAP),   ! CRTOS OUTPUT MAILBOX
C                          (IPARM( 6),ICLIHI),   ! IHAP INPUT MAILBOX
C                          (IPARM( 7),ISTS1 ),   ! STATUS LINE 1
C                          (IPARM( 8),ISTS2 )    ! STATUS LINE 2
C                      ICLMY  : Input class# for calling program
C                      IPRGNR : Program# for calling program
C                               (=1 if father program)
C                      ICLFTH : Class# of father program
C                      IPRFTH : Program# of father program
C
C CALLING SEQUENCE       : CALL LINKP (PARAMS,ICLMY,IPRGNR,ICLFTH,IPRFTH)
C
C ---------------------------------------------------------------------------
C
      SUBROUTINE IHPCM (ICMD,NB,ICLASS,ICLIHI,ICLOU)
     .,Send msg to IHAP                         GR 830827
C
C
C     DESCRIPTION:   The IHAP command contained in ICMD (Length=NB bytes)
C                    is sent to the mailbox ICLIHI. The program waits
C                    until it gets a prompt from IHAP via the mailbox
C                    ICLASS.
```

2.2 Equipment handling subroutines

```
      SUBROUTINE CKCAM(LUCAM,IER,LU)
     .,Check CAMAC                                    GR 830615
C
C
C      NAME     :  CKCAM
C
C      SUMMARY  :  Check CAMAC driver and CAMAC crate (on-line)
C
C      BY       :  Peter Biereichel  ESO/GAR
C
C      DATE     :  11:41 AM  THU.,  3  JUNE, 1982
C
C      PACKAGE  :  CCD program
C
C---
C
C      REQUIREMENTS:  CAMAC
C
C      USAGE      : Should be called to check if CAMAC is on-line
C
C      DESCRIPTION:  Checks the driver status and puts the driver
C                    up if it was down.
C                    The CAMAC controller status will be checked
C                    and if it is off-line a warning message will
C                    be given (without putting down the driver).
C
C
C---
C
C      ! 821025 ! 1.0   ! CCD common suppressed.                GR         !
C
C- -------------------------------------------------------------------------
C
      SUBROUTINE GETEQ(LUSES,IEQT,IER,LU)
     .,Get EQT numb. for given session LU         GR 830614
C
C -------------------------------------------------------------------------
C
      SUBROUTINE CKIOL(LUOUT,LUIN,IER)
     .,Check I/O LU's(direct-or class-I/O)        GR 830614
C
C Parameters: LUOUT=Output LU(for direct I/O is also input LU)
C             LUIN =Input  LU(0 for direct I/O)
C                   (for direct I/O set to LUOUT on output)
```

2.3 Other subroutines

```
      SUBROUTINE TIMCV (ZZ,IH,IM,IS,IT)
     .,convert secs into hh,mm,ss,tens           GR 830609
C
C   converts real into HH mm ss.t(hours,mins,secs,tens of sec)
C  original version La Silla (DH,FG)
C
```

3. Installation

Once the relocatable modules are available the library %DAQLB:GR:75 is created by running the MERGE utility with the *DAQLB::75 commands file:

```
,Command file for MERGE to create DAQLB::75 (purge before)
,Program to program communication
%SCPRG::75
%LINKP::75
%IHPCM::75
,Equipment handling
%CKCAM::75
%GETEQ::75
%CKIOL::75
,Others
%TIMCV::75
```

4. Maintenance

To maintain the library by modification of existing modules or addition of new ones, follow these steps:

1- Compile appropriate modules
2- Modify *DAQLB:GR:75 for MERGE
3- Build with MERGE new %DAQLB
4- Edit all changed sections of this document("DAQLB:GR:75)
5- To list this :RU,PR,"DAQLB::75:-1

Important NOTE:

Given that existing programs use DAQLB, backwards compatibility must be maintained. Furthermore DAQLB must be the same on all installations.

For this reason it is advisable that when you have subroutines to contribute, you do it in agreement with either G.Raffi in Garching or E.Allaert in La Silla.

"MOTOR 83/ 8/28 15:33:

TECHNICAL NOTE TPE/GR/830827/20

G. Raffi ESO-Garching

```
--------------------------
!                        !
!     MOTOR library      !
!          +             !
! motor test programs    !
!                        !
!                        !
--------------------------
```

MOTOR library + motor test programs 830827

G.Raffi ESO-Garching

Update history: Original version GR 830827

Library of HP 1000 RTE-4B FTN4X subroutines to interface with the ESO 4 channel motor controllers.

The motor controllers interfaced by these subroutines are the ones developed by B.Gustafsson,ESO-Garching.

The protocol HP-CAMAC motor controller is described in:
"Communications protocol between 4 channel motor driver and H-P computer"-B.Gustafsson-Technical Note TPE 1/81 Rev.2 May 24,83.

INDEX:

Introduction

This library is a by-product of the development of CASPEC.
So while on one hand is quite suitable to the software Data Acquisition Sytem of ESO and tested under the CASPEC conditions, it does not intend to be good for all applications.
In particular the initialisation subroutines do refer to:

DC motors with absolute, serial, binary encoders.

They can easily be adapted to other cases.
Initialisation subroutines do exist in two versions:

normal initialisation MOTIN
block initialisation (faster) MBLIN.

The subroutine MOVLM moves DC motors without encoders against limits.

The motor test programs TMOST, TMINI, TMHND are used by CASPEC, but they are stand-alone test programs, which can be used whenever the motor controllers are used. In particular TMOST and TMHND do not refer to any particular type of motor or encoder and so should be completely general.

1. List of subroutines

MCNST	Check motor controller status	GR 830614
MBLST	Get block status from motor controller	GR 830614
MBLIN	Move motor to initial position	GR 830614
MOTZ	Initialise motor controller module	GR 830614
MOTIN	Move motor to initial position	GR 830717
MOVLM	Move motor (no encoder) to hardw. limit	GR 830308
MTCOM	Send command to motor	GR 830614
MTRED	Read msg from motor controller	GR 830319
MIWT	Wait for msg. from motor controller	GR 830614
MTSKP	Skip any msgs from motor contr.	GR 830614
MTPAR	Ask info. on motor	GR 830319
MOTCK	Check motor position	GR 830319
MOTMV	Move motor to position	GR 830309
MBLCM	Send block or single cmd to motor	GR 830614
MBLRD	Read single or block msg from motor	GR 830614
RDENC	Read encoder value	GR 830614
MTERR	Print motor commnds errors	GR 830614
WTMOT	Wait for active motors positions	GR 830615
ASKMT	Ask parameters for motor	GR 830103

2. Subroutine calls
- ---------------

```
      SUBROUTINE MCNST(IADR, ICSTAT, MSTAT, LSTAT, IRPOS, LU)
     .,Check motor controller status             GR
C
C Parameters:
C               IADR      =coded CAMAC addresses
C               NMOT      =motor number(1..4)  .
C                          (any if interested only in cotroller status)
C               ICSTAT    =-2 controller in messy state(better to give a Z)
C                         =-1 if motor did not yet receive a Z
C                         =0  if motor was reset by Z or power dip
C                         =1 if motor controller(!) is initialised
C               MCODE     = status reply for given motor
C                          (makes sense only for ICSTAT=1)
C               LU        = LU for error messages
C
C ---------------------------------------------------------------------------
C
      SUBROUTINE MBLST(IADR, MSTAT, LSTAT, IPOS, IER, LU)
     .,Get block status from motor controller   GR
C
C ------------- -------------------------------------------------------------
C
C
      SUBROUTINE MBLIN(IADR, MOTN, IENP, ICF, ISOLO, ISOFU, IAXS, IPAFG, IER, LU)
     .,Move motor to initial position             GR
C
C adapted from B.Gustafsson RTCAM
C
C MOTN contains a motor number.In principle it can be >4(4 mot.per
C module) so that in case of error the motor number is correct
C (e.g.motor 13 failed).However internally it is transformed into
C NMOT=motor no. between 1-4(modulo 4)
C
C ---------------------------------------------------------------------
C
C
      SUBROUTINE MOTZ(IADR, NMOD, IER, LU)
     .,Initialise motor controller module       GR
C
C Initializes motor controller
C CAMAC Z not assumed before,as this subr.will be repeated for every
C motor controller module (but hopefully not Z)
C
C ASSUMES THAT A Z ON CAMAC HAS BEEN DONE
C
C ------------------------------------------------------------------------
C
      SUBROUTINE MOTIN(IADR, MOTN, IENP, ICF, ISOLO, ISOFU, IAXS, IPAFG, IER, LU)
     .,Move motor to initial position             GR
C
C adapted from B.Gustafsson RTCAM
C
C MOTN contains a motor number.In principle it can be >4(4 mot.per
```

```
C module) so that in case of error the motor number is correct
C (e.g. motor 13 failed). However internally it is transformed into
C NMOT=motor no. between 1-4(modulo 4)
C
C ------------------------------------------------------------------------
C
      SUBROUTINE MOVLM(IADR, MOTN, INITY, ISPED, IER, LU)
     .,Move motor (no encoder) to hardw. limit   GR
C
C adapted from B.Gustafsson RTCAM
C
C
C  -----------------------------------------------------------
C
      SUBROUTINE MTCOM(MSBUF, IADR, IER, LU)
     .,Send command to motor                     GR
C
C adapted from B.Gustafsson SCOM
C
C
C --------------- ------------------------------------------
C
      SUBROUTINE MTRED(MSBUF, IADR, IMSGF)
     .,Read msg from motor controller           GR
C
C adapted from B.Gustafsson RMES
C
C ----------------------------------------------------------------
C
      SUBROUTINE MTWT(MSBUF, IADR, ITO, IER, LU)
     .,Wait for msg. from motor controller      GR
C
C Note: msg takes some 10-20 msec to return, so the wait time
C       before first and second attempt should not be bigger
C       than 50 msec
C ITO: timeout in seconds
C
C  ------------------------------------------------------------------
C
      SUBROUTINE MTSKP(MSBUF, IADR, IMSGF, IER, LU)
     .,Skip any msgs from motor contr.           GR
C
C
C -------------------------------------------------------------
C
      SUBROUTINE MTPAR(MSBUF, IADR, NMOT, ICODE, IER, LU)
     .,Ask info. on motor                       GR
C
C
C
C ----------------------------------------------------------
C
      SUBROUTINE MOTCK(IADR, NMOT, IPOS, MCODE, IER, LU)
     .,Check motor position                     GR
C
C
C ----------------------------------------------------
C
      SUBROUTINE MOTMV(IADR, NMOT, IPOS, IER, LU)
     .,Move motor to position                    GR
```

```
C
C
C connect motor first(in case it was disconnected)
C
C ------------------------------------------------------------
C
      SUBROUTINE MBLCM(MSBUF,IADR,IER,LU)
     .,Send block or single cmd to motor         GR
C
C adapted from B.Gustafsson SCOM
C NBYTS considered only for block parameters write
C
C  -----------------------------------------------------------
C
      SUBROUTINE MBLRD(MSBUF,IADR,IMSGF,IER,LU)
     .,Read single or block msg from motor     GR
C
C copes with block read
C
C  -----------------------------------------------------------
C
      SUBROUTINE RDENC(IADR,NMOT,IEABS,IER,LU)
     .,Read encoder value                      GR
C
C Reads from hardware registers in two bytes encoder value
C 3 bits in upper byte for 11 bit encoder.
C
C Parameters:
C            IADR     = CAMAC def's
C            NMOT     = # of motor (1...4)
C            IEABS    = abs.encoder value(read via hardw.registers)
C            IER      = err.if #0
C            LU       = LU FOR error msgs
C
C ----------------------------------------------------------------
C
      SUBROUTINE MTERR(IADR,NMOT,IERCD,LU)
     .,Print motor commnds errors              GR
C
C adapted from B.Gustaffson PRERR subroutine
C
C used by MBLCM to give immediate error messages due to the
C structure of commands,as they are detected by motor controllers
C
C ----------------------------------------------------------------
C
      SUBROUTINE WTMOT(MOTD,IPOSS,MMSK,ITMS,IDSCS,IER,LU,ICLST)
     .,Wait for active motors positions        GR
C
C PARAMETERS:
C            MOTD(16,4)=CAMAC definitions for 4 motor controller
C            IPOSS(16)=positions for 16 motors
C                      (motors 1..16 corresp.to mot.contr.1...4)
C            MMSK     =motors mask(bits 0...15<=>motors 1...16)
C            ITMS(16)=timeouts for motors(corresp.to motors 1...16)
C                      says which motors should be waited for
C            IDSCS(16)=flags to leave motors connected or disconn.
C            IER      =error flag(err.if#0,negative motor no.1...16)
C            LU       =output LU for errors
C
```

```
C NOTE:Subroutine useful to wait for up to 16 motors to reach position,
C       once they have been initialized    or moved in parallel.
C       Used by CASPEC
C
C
C ---------------------------------------------------------------------
C
      SUBROUTINE ASKMT(LCAMC,NMOD,NMOT,IENP,IAXS,ISW,IILU,IOLU)
     .,Ask parameters for motor                          GR
C
C Works with direct I/O or CRTOS I/O
C isw=0 for all params
C isw=1 for all params,but no encoder and no axis value
C isw=2 for all params,but no axis value
C
C Note:Refers to subroutines of library %IOLIB.
C       Used in test programs TMINI,TMOST,TMHND.
```

3. Installation

Once the relocatable modules are available the library %MOTOR:GR:75 is created by running the MERGE utility with the *MOTOR::75 commands file:

```
,Transfer file for MERGE to create %MOTOR:GR:75 (purge beforehand)
%MCNST::75
%M8LST::75
%M8LIN::75
%MOTZ  ::75
%MOTIN::75
%MOVLM::75
%MTCOM::75
%MTRED::75
%MTWT  ::75
%MTSKP::75
%MTPAR::75
%MOTCK::75
%MOTMV::75
%M8LCM::75
%M8LRD::75
%RDENC::75
%MTERR::75
%WTMOT::75
%ASKMT::75
```

When %ASKMT is used the library %IOLIB is needed as well at load time.

4. Maintenance

To maintain the library by modification of existing modules or addition of new ones, follow these steps:

1- Compile appropriate modules
2- Modify *MOTOR:GR:75 for MERGE
3- Build with MERGE new %MOTOR
4- Edit all changed sections of this document("MOTOR:GR:75)
5- To list this :RU,PR,"MOTOR::75:-1

Important NOTE:

Given that existing programs use MOTOR, backwards compatibility must be maintained. Furthermore MOTOR must be the same on all installations.

For this reason it is advisable that when you have subroutines to contribute, you do it in agreement with either G.Raffi in Garching or E.Allaert in La Silla.

5. Motor test programs

The motor test programs TMOST, TMINI, TMOST can be run in stand-alone mode directly from a terminal or in the DAQ environment under CRTO5. Furthermore they can be run with run time parameters (from console or from another program) or interactively, when no parameters are specified.

They make use of the libraries %MOTOR, %IOLIB, %DAQLB.
To load or reload them the transfer file:*TMOTR::75 should be used, but if CASP is installed on your system these programs are already available.

5.1 Motor status TMOST

```
C
      PROGRAM TMOST()
     .,Check status of DC motor                    GR 830309
C
C PARAMETERS : PAR(1)=LU (for direct I/O),Class for output(via CRTO5)
C               (default LOGLU)
C               PAR(2)=CAMAC LU (default 60)
C               PAR(3)=NMOD          where NMOD=CAMAC N for mot.controll.
C               PAR(4)=NMOT=number of motor in this motor contr.(1...4)
C               PAR(5)=0 for direct I/O,Class # for input(CRTO5)
C
```

"IOLIB 83/ 8/28 15:24: 8

TECHNICAL NOTE TPE/GR/830826/21

G. Raffi ESO-Garching

```
-------------------------
!                       !
!                       !
!     IOLIB library     !
!                       !
!                       !
!                       !
-------------------------
```

5.2 Motor initialisation TMINI

```
C
      PROGRAM TMINI ()
     .,Init. DC motor with abs.serial encoder  GR 830530
C
C works with dc motor,11 bit ser.encoder
C PARAMETERS : PAR(1)=LU (for direct ItO),Class for output(via CRTOS)
C              (default LOGLU)
C              PAR(2)=CAMAC LU (default 60)
C              PAR(3)=NMOD CAMAC N for motor controller
C              PAR(5)=0 for direct I/O,Class # for input(CRTOS)
C
```

5.3 Motor handset control TMHND

```
C
      PROGRAM TMHND ()
     .,Motor handset control                     GR 830311
C
C PARAMETERS : PAR(1)=LU (for direct ItO),Class for output(via CRTOS)
C              (default LOGLU)
C              PAR(2)=CAMAC LU (default 60)
C              PAR(3)=NMOD          where NMOD=CAMAC N for mot.controll.
C              PAR(4)=NMOT=number of motor in this motor contr.(1...4)
C              PAR(5)=0 for direct I/O,Class # for input(CRTOS)
C
```

IOLIB library

830826

G. Raffi   ESO-Garching

Library of subroutines for direct-, class-I/O and tables handling.

Update history: Original version   GR 830826

INDEX:

Introduction

The purpose and need of a library of I/O subroutines is that programs developed in the ESO Data Acquisition System environment do not talk directly to the terminal,but via the Terminal Handler CRTO5.
While the usage of an I/O library instead of formatted I/O gives always advantages in terms of maintenability of I/O software,this has become a necessity with the usage of the Terminal Handler CRTO5 package written by P.Biereichel.
An extra advantage of this is "device-handler independence",as the same calls are used both for direct I/O and for class I/O to a terminal LU via CRTO5.Test programs for example can benefit from this,as they can be run either in stand-alone mode directly from a terminal or be scheduled within the DAQ environment.

The way the subroutines make the distinction between direct output and CRTO5 output is by looking at the LU received:if it is >255 then class output via CRTO5 is wanted.
Output can also be sent to the status lines of CRTO5,by passing the corresponding class number.However in this case an additional info. is needed by CRTO5,namely the line number.This is passed by an initialization subroutine INISL.

The set of class subroutines are used by the I/O package,but are available to class I/O communicate with programs.They are recommended in place of EXEC calls,for the degree of extra portability that a program has (e.g.changing machine would mean substituting the subroutines,rather than changing the whole code).

A group of subroutines deal with tables (forms+parameters),providing a simple interface to CRTO5,which supports these functions.
In this group the DFORM subroutine is an exception,as it provides direct table display.The reason for putting it in this library is that it seems general enough to be used by e.g.Instrument programs to display auxiliary tables on the IHAP graphic terminal,when IHAP is not used.

1. List of subroutines

The IOLIB subroutines are grouped in 4 sections, where they are listed according to the source files structure, with modules ordered from higher to lower level subroutines.
The list does not include internally used subroutines, which are not meant for the user.

1.1 I/O subroutines

INISL	Set-up status line number	GR 830617
PREFX	Prefix all output with special character	GR 830617
PRMPT	Gives a default prompt for ASCII input	GR 830617
RDIN	Read with prompt integer in a range	GR 830617
ASKYN	Ask for Yes/No reply	GR 830617
PRTTX	Print an immediate string	GR 830617
PRTBF	Print a string from buffer	GR 830617
PRTTN	Print text+number	GR 830617
PRTNM	Print an integer with given radix	GR 830613
PRTEX	Output string with *** prefix	GR 830613
PRTEN	Output string + number with *** prefix	GR 830613
INP	Prompt for input string	GR 830614
REDDG	Read number from ASCII buffer	GR 830614

1.2 Class I/O subroutines

CLGET	Get a class number.	V1.0	GR 830210
CLRLS	Release a class number. PB/GAR	V1.1	GR 830210
CLEAN	Clean class numbers	V1.0	GR 830210
CLRED	Start class input from term.		GR 830214
MGET	Class get line		GR 830214
PUTM	Class put message		GR 830214
MWAIT	Class wait for line		GR 830214
PTPAR	Pass ip1, ip2 params via class I/O		GR 830214
CKCLS	Checks for valid class number		GR 830214

1.3 Tables handling subroutines

DISPA	Display form & params from memory	GR 830513
REDPA	Read updated param's in buffer	GR 830513
PUTPA	Put out form&params(no wait)	GR 830513
PACHK	Read param's to be checked	GR 830513
GETPR	Get from disc form parameters	GR 830614
PUTPR	Put form parameters into param's file	GR 830614
PINFO	Get info. on form	GR 830614
PARFG	Parameter table configurator	BG 830519
DFORM	Direct display of form+params	GR 830715

1.4 Other subroutines

```
TIME
TIMA   RS/810217 Convert time or angle to ASCII
NDSEC  Gives time difference in secs          GR 830128
DAT    RS/810216 Convert year,day to ASCII
MOVE   RS/810423 Move words
MYNAM  CCD - Get programs name                pb 820401
CONV
```

2. Subroutine calls

2.1 I/O subroutines

```
      SUBROUTINE INISL(LINST,ICLST)
     .,Set-up status line number                    GR
C
C Used to inform the output subroutines of a given status line
C number associated to the class number for status ICLST
C If ICLST<255 this has no effect
C Following calls to PRTTX,PRTBF,PRTNM,PRTTN,PRTDG,OUT will
C direct output to the line LINST if the LU they indicate
C =ICLST,otherwise output will go to the rolling part of the
C screen or to LU directly(if LU<255).
C
C Parameters:LINST = line number for status line (Must be within
C                    upper locked part of screen)
C            ICLST = status class number for terminal handler
C
C -----------------------------------------------------------
C
      SUBROUTINE PREFX(IPCHR)
     .,Prefix all output with special character GR
C
C  PARAMETER:IPCHR= character(in upper byte,e.g.1H&)
C                   to be prefixed to any following output
C                 =0 (binary 0) deletes prefix for following output
C
C
C  ----------------------------------------------------------
C
      SUBROUTINE PRMPT(IPCHR)
     .,Gives a default prompt for ASCII input   GR
C
C Note: to be used with func.call INP for following input
C  PARAMETER:IPCHR= character(in upper byte,e.g.1H&)
C                   to be prompted as default for input
C                 =0 (binary 0) deletes prompt for following input
C
C
C  ----------------------------------------------------------
C
      SUBROUTINE RDIN(IPBUF,INUM,IRDX,IMIN,IMAX,IILU,IOLU)
     .,Read with prompt integer in a range       GR
C
C The text (IMIN,IMAX) is added to the prompt string
C The prompt is repeated up to when valid input is got
C The (IMIN,IMAX) text is not added if IMIN=IMAX=0
C The validity check is not performed on input if IMIN>=IMAX
C Printout of min,max and check are skipped also for non dec.input
C Parameters:  IPBUF = prompt buffer (_! to get inp.on same line)
C              INUM    integer value
C              IRDX    input radix(8,10,16...)
C              IMIN  = min value accepted for INUM
C              IMAX  = max value accepted for INUM
C              IILU  = input LU or class for CRT05
```

```
C              IOLU  = output LU or class for CRT05
C
C
C -------------------------------------------------------
C
      SUBROUTINE ASKYN(IPBUF,IANS,IILU,IOLU)
     .,Ask for Yes/No reply                        GR
C
C The text (YE/NO) is added to the prompt string
C The prompt is repeated up to when a valid input is received
C Parameters: IPBUF = prompt buffer (no _ ,input is from same line)
C             IANS  = first to letters or reply (YE or NO)
C             IILU  = input LU or class for CRT05
C             IOLU  = output LU or class for CRT05
C
C Only valid YE or NO reply acceted,else prompt is repeated
C
C
C  ----------------------------------------------------------
C
      SUBROUTINE PRTTX (IBUF,IOLU)
     .,Print an immediate string                   GR
C
C IBUF = THE BUFFER TO PRINT
C IOLU = THE LU TO PRINT ON ,-VE=>NO BUFFER FLUSHING(NO OUTPUT NOW)
C
C
C
C -----------------------------------------------------------
C
      SUBROUTINE PRTBF(IBUF,NCHAR,LU)
     .,Print a string from buffer                  GR
C
C Parameters* IBUF = buffer with string
C             NCHAR= number of char's to print
C             LU= output lu or class
C                if negative output is not flushed now
C                (this allows to continue on same line)
C
C ---------------------------------------------------------
C
      SUBROUTINE PRTTN(IPBUF,INUM,IRDX,IOLU)
     .,Print text+number                           GR
C
C Printing of text + number on same line done with one EXEC call
C Number printed with 6 char's,including sign
C Pararmeters: IPBUF = text buffer (no _ ,but ! needed for string)
C              INUM  = integer number to print on same line
C              IOLU  = output LU or class for CRT05
C             IRDX    input radix(8,10,16...)
C                      if negative no flushing at end
C                     (this allows to stay on same line)
C
C
C ------------------------------------------------------
C
      SUBROUTINE PRTNM(INUM,IRDX,LU)
     .,Print an integer with given radix           GR
C
C The number of digits chosen for decimal conversion depends on
```

```
C the value to print (sign is printed only for dec.negative numb.)
C
C Parameters: INUM = integer
C             IRDX    input radix(8,10,16...)
C             LU    = output LU or class
C
C -------------------------------------------------------
C
      SUBROUTINE PRTEX(IBUF,LU)
     ..Output string with *** prefix           GR
C
C Otherwise identical to PRTTX
C Normally used to prefix error messages
C
C --------------------------------------------------------
C
      SUBROUTINE PRTEN(IBUF,IN,IB,LU)
     ..Output string + number with *** prefix  GR
C
C Otherwise identical to PRTTN
C Normally used to prefix error messages
C
C -------------------------------------------
C
      FUNCTION INP(IPBUF,IBUF,ILEN,IILU,IOLU)
     ..Prompt for input string                  GR
C
C Parameters: IPBUF prompt buffer(if input wanted on same line
C                  terminate with _!,else with !)
C             IBUF input buffer
C             ILEN no.of chars in input buffer for input
C                  actual no.of chars read on return
C             IILU input LU or class (via CRT05)
C             IOLU output LU or class (via CRT05)
C
C             INP return the actual no.of input char's
C
C
C -----------------------------------------------------------
C
      SUBROUTINE REDDG (IVAL,IRDX,IERR,IBUF,IDGS)
     ..Read number from ASCII buffer            GR
C
C IVAL = THE RESULTING INPUT VALUE
C IRDX = THE INPUT RADIX
C IERR = #0 for error(e.g.non numeric character read)
C IBUF = THE INPUT BUFFER
C IDGS = THE NUMBER OF DIGITS IN input buffer
C
C
```

2.2 Class I/O subroutines

```
      SUBROUTINE CLGET (ICLAS,IER)
     .,Get a class number.              V1.0     GR
C
C original version RS 810901
C
C  CLGET Allocates a class number and logs it to a disc file.
C  (adapted from HP Communicator Vol II, isuue 6, page 5.)
C
C  Up to 31 8-word entries are stored in file CL.NO.:SY::1:4, after
C  a copy of the most recent entry in which the first word is set to
C  the number of entries. The format for the (other) entries is:
C
C       class number          : word 1
C       program name          :      2-4
C       time of entry, year   :      5
C                      day    :      6
C                  seconds    :      7,8
C
C -----------------------------------------------------------------
C
      SUBROUTINE CLRLS (ICLASO,IER)
     .,Release a class number. PB/GAR V1.1     GR
C
C Original version by RS/810901
C  CLRLS does the reverse of CLGET. It de-allocates the class
C  number and removes its entry from the log file CL.NO.:SY.
C  For comments see HP Communicator Vol II, issue 6, page 9.
C
C  ----------------------------------------------------------------
C
      SUBROUTINE CLEAN(NUMCL,IER)
     .,Clean class numbers              V1.0     GR
C
C adapted from CCRLS by P.Biereichel,used in CCD
C     output parameter:NUMCL =number of class numbers actually
C                             released(0 for none)
C

      SUBROUTINE CLRED(IILU,ICLAS,LLBUF,IP1,IP2)
     .,Start class input from term.                  GR
C  STARTS CLASS READ FROM IILU TERMINAL
C  USING CLASS NUMBER ICLAS
C
C IP1 TYPICALLY USED TO MARK ORIGIN OF MESSAGE
C THIS IS USEFUL WHEN WAITING FOR MESSAGES FORM VARIOUS SOURCES ON
C A SINGLE CLASS # (IP1 COULD GIVE CLASS # IDENTIFYING SOURCE)
C IP2 TYPICALLY USED TO SAY IF HANDSHAKE IS WANTED (CLASS # FOR
C THIS PASSED THEN) OR NOT WANTED (=0 THEN)
C READ WITH ECHO ON
C
C
C ----------------------------------------------------------------
C
      FUNCTION MGET(NCLAS,IBUF,LLBUF,IP1,IP2)
     .,Class get line                                GR
```

```
C   DOES CLASS GET WITHOUT WAIT ON IILU TERMINAL
C
C   MGET=NUMBER OF CHARACTERS GOT
C        =NEGATIVE NUMBER OF REQUESTS TO CLASS FOR NO MESSAGE
C
C   ------------------------------------------------------------
C
      SUBROUTINE PUTM(NCLAS,IBUF,LLOUT,IP1,IP2)
     ..Class put message                              GR
C
C   STRTS CLASS I/O WRITE/READ OF A MESSAGE TO ANOTHER PROGRAM
C
C   LLOUT=NUMBER OF CHARACTERS TO OUTPUT
C
C
C ---------------------------------------------------------------
C
      FUNCTION MWAIT(NCLAS,IBUF,LLBUF,IP1,IP2)
     ..Class wait for line                            GR
C   DOES CLASS GET WITH WAIT
C
C   MWAIT=NUMBER OF CHARACTERS GOT
C
C -------------------------------------------------------
C
      SUBROUTINE PTPAR(NCLAS,IP1,IP2)
     ..Pass ip1,ip2 params via class I/O             GR
C
C Passes a dummy buffer and IP1,IP2 params in class I/O
C
C --------------------------------------------------
C
      SUBROUTINE CKCLS(ICLAS,IER)
     ..Checks for valid class number                 GR
C
C Useful when a subroutine executes only under CRTOS as
C a preliminary check
C e.g.when ICLAS=0(typing error resulting in non initialised
C variable) a new class number might be allocated and the
C error would possibly go unnoticed in this section of program
C
```

2.3 Table handling subroutines

```
      SUBROUTINE DISPA(NSBSET,IBUF,IFNAM,IER,ICLIN,ICLOU)
     .,Display form & params from memory         GR
C
C      Parameters: NSBSET= subset number
C                  IBUF  = buffer with params
C                  IFNAM  = name of form file
C                  IER    = error flag(error if #0)
C                  ICLOU =class for output(via CRT05)
C
C Note:THIS SUB.CAN BE USED ONLY WITH CRT05(NO DIRECT I/O)
C
C It is assumed here that params are already in CASPEC mem.tables
C having been got previously from disc via GETPR
C
C
C  -------------------------------------------------------------
C
      SUBROUTINE REDPA(NSBSET,IBUF,IFNAM,IER,ICLIN,ICLOU)
     .,Read updated param's in buffer           GR
C
C Displays a form with the parameters contained in IBUF
C and then reads back the (possibly) updated param's in same buffer
C
C      Parameters: NSBSET= subset number
C                  IBUF  = buffer for params
C                  IER   = error flag(error if #0)
C                  ICLIN =class for input(via CRT05)
C                  ICLOU =class for output(via CRT05)
C
C Note:THIS SUB.CAN BE USED ONLY WITH CRT05(NO DIRECT I/O)
C
C
C ------------------------------------------------------------------
C
      SUBROUTINE PUTPA(NSBSET,IBUF,IFNAM,IER,ICLOU)
     .,Put out form&params(no wait)              GR
C
C subroutine used internally by DISPA & REDPA
C not meant for external usage,because CRT05 is not prompted
C for input and the sequence of actions on CRT05 must be completed
C outside
C
C
C ------------------------------------------------------
C
      SUBROUTINE PACHK(NSBSET,IBUF,IFNAM,IPTR,IER,ICLIN,ICLOU)
     .,Read param's to be checked                GR
C
C Meant to get parameters and then pass then back for check
C If sthg is wrong an error should be printed (INISL(23,ICLOU)+
C CALL PRTTX( H Value out of range valid range is 1,1024!,ICLOU))
C and then PACHK should be called again with IPTR pointing to
C invalid field to read again.Should the params be correct then
C PACHK has to be called with IPTR=0 to exit form mode
```

```
C The first time PACHK is called with IPTR=-1
C Displays a form with the parameters contained in IBUF
C and then reads back the (possibly) updated param's in same buffer
C After PACHK INISL(line no.,ICLST) should be called again to
C restore status line initialisation for IOLIB
C      Parameters: NSBSET= subset number
C                  IBUF  = buffer for params
C                  IPTR  = field where cursor should point to
C                          (it is assumed that form and params are
C                           already on screen)
C                           if=0 then exit form mode (all OK)
C                           if<0 it is assumed that form and params
C                           are not yet on screen (initialisation call)
C                           negated value of field to point to
C                  IER   = error flag(error if #0)
C                  ICLIN =class for input(via CRT05)
C                  ICLOU =class for output(via CRT05)
C
C Note:THIS SUB.CAN BE USED ONLY WITH CRT05(NO DIRECT I/O)


      SUBROUTINE GETPR(FPNAM,IFNAM,NSBSET,IPBUF,IER,LU)
     .,Get from disc form parameters              GR
C
C
C       Get parameters associated with subset(see PARFG code
C                  for a definition of subset)
C
C      REQUIREMENTS: Parameters data file(corresponding to formss.)
C                    file) must have been built with PARMG-parameters
C                    manager off-line program.
C                    Every form has an associated data record in the
C                    parameters file,identified by a subset number.
C                    Subroutine PARFG of PARMG and PINFO allow to
C                    retrieve from a subset number the corresponding
C                    form no.,location and size of data record in the
C                    parameters file
C
C
C      INPUT        : FPNAM= name of params file (6 CHARS)
C                            +one word with SC
C                            + one word with CR
C                     IFNAM = name of form file
C                     NSBSET=number of subset(according with PARFG)
C                     IPBUF =buffer for expected parameters
C                     IPBLL =length of parameters buffer
C                             (must correspond exactly to CASFG info)
C      OUTPUT       : IER    =0 no error
C                            =1 no subset found
C                            =2 mismatch between params buffer length
C                               (length of CASFG given in IPBLL)
C                            <0 FMP error
C                     IPBLL =actual length read (possibly different
C                             from input if error detected)
C
C
C ------------------------------------------------------------
C
      SUBROUTINE PUTPR(FPNAM,IFNAM,NSBSET,IPBUF,IER,LU)
     .,Put form parameters into param's file     GR
```

```
C
C Parameters:
C            FPNAM(5)=1..3 ASCII file name of param's file
C                     FPNAM(4)=security code
C                     FPNAM(5)=CR
C            NSBSET   =subset number(according to PARFG)
C            IPBUF    =input buffer (of right size)
C            IER      = error return flag(err.if #0)
C            LU       =lu for errors
C
C
C --------------------------------------------------------
C
      SUBROUTINE PINFO(NSBSET,NOFRM,NOREC,LGREC,IFNAM,IER,LU)
     .,Get info. on form                          GR
C
C
C IFNAM (5) contains form file name+sec.code +CR
C file is open once,first time PINFO is called
C
```

```
      SUBROUTINE PARFG (INAM,LGPAR,PDIC,DATE,PNUM,IERR)
     .,Parameter table configurator BG
C
CDOC
C*****************************************************************************
C
C     NAME    :  PARFG
C     SUMMARY :  Returns information about  parameter file format
C
C     BY  :  Birger Gustafsson  ESO/GAR
C
C     DATE    :  11:00 AM WED., 4 MAY, 1983
C
C     KEYWORDS:
C
C     PACKAGE :  I/O and Class I/O subroutines
C
C---
C
C     REQUIREMENTS:  disc
C
C     DESCRIPTION:  The program returns record number and word count
C                   of the parameters for each form in the form file.
C                   Number of forms are also returned.
C                   It also returns the record with creation date and
C                   last update date.
C
C---
C
C     INPUT   :  INAM     ! File name of forms file
C                           Format: word 1 - 3 = file name
C                                   word 4 = security code
C                                   word 5 = cartridge number
C                LGBUF     ! Length of parameter  dictionary  buffer =
C                           2 times the number of forms requested
C
C     OUTPUT  :  PDIC     ! Parameter file  dictionary.
C                           Output format: word 1 = record number for
C                                                   parameters for form
C                                                   number 1
C                                          word 2 = number of parameter
C                                                   words in form 1
C                                          word 3 = record number for
C                                                   parameters for form
C                                                   number 2
C                                          word 4 = number of parameter
C                                                   words in form 2
C                                          word 5 = record number for
C                                                   parameters for form
C                                                   number 3
C                                          word 6 = number of parameter
C                                                   words in form 3
C                                                .
C                                                .
C                                                .
C                         A nonexisting form gives record number -1
C                         and number of words = 0
```

```
C
C
C               PNUM     ! Number of forms
C               DATE     ! Record number for creation and update date
C                          Record format :
C                                   word 1 = creation date, year
C                                   word 2 = creation date, day
C                                   word 3 = creation date, hour
C                                   word 4 = creation date, min
C                                   word 5 = last update date, year
C                                   word 6 = last update date, day
C                                   word 7 = last update date, hour
C                                   word 8 = last update date, min
C               IERR     ! Error flag.
C                           Flag = negative : file manager error on foms
C                                            file.
C                           Flag = 0 : no error
C
C      CALLING SEQUENCE: CALL PARFG(INAM,LGPAR,PDIC,DATE,PNUM,IERR)
C
C---
C
C ---------------------------------------------------------------
C
      SUBROUTINE DFORM(IFNAM,NSBSET,IBUF,IMOD,IER,LU)
     .,Direct display of form+params             GR
C
C Parameters: IFNAM(5)=filename +sec.code + CR
C             NSBSET  =params.subset no.
C             IBUF    =buffer with params
C             IMOD    =0 normal display,opendclose form file
C                     =-1 first display,leave form file open
C                     =1 assume form is on screen,fill in new params
C                     =2 close form file and clean up
C             IER     =orror flag(err.if#0)
C             LU      =direct I/O lu(not via CRT05,no class # allowed)
C
C The form is simply left on screen all time
```

2.4 Other subroutines

```
      SUBROUTINE TIME (IYR,IDY,SEC),RS/810509 Get year, day and seconds.
C
C  IYR <--  Year
C  IDY <--  Julian day of year
C  SEC <--  Time of day in seconds (floating)
C
C ---------------------------------------------------------

 NAM TIMA,7 RS/810217 Convert time or angle to ASCII
*
TIME  --> Time or angle in seconds (floating)
BUF   <-- 12-char. ASCII string like: "-275:08:59.3"
*
TIMA NOP       CALL TIMA (TIME,BUF)
*
C  --------------------------------------------------------

      FUNCTION  NDSEC(INIT)
     .,Gives time difference in secs          GR
C
C Gives integer number of seconds elapsed from
C reference time
C Parameters:
C                INIT= 0 (count delta from now)
C                      pass back in NDSEC elapsed secs
C
C --------------------------------------------------------------
C
 NAM DAT,7 RS/810216 Convert year,day to ASCII
*
YEAR       --> year
JDAY       --> julian day
BUF        <-- 10-character ASCII string like: "16 FEB '81"
*
DAT NOP      CALL DAT (IYEAR,IDAY,IBUF)
*
C ------------ -------------------------------------------------
C
 NAM MOVE RS/810423 Move words
*
A   'From' address
B   ' To ' address
N   Number of 16-bit words
*
MOVE NOP  CALL MOVE (A,B,N)
*
C ------------ -------------------------------------------------
C
      SUBROUTINE MYNAM (NAME)
     .    ,CCD - Get programs name          pb
C
```

```
      HED CONVERSION * GENERAL SUBROUTINES
      NAM CONV,7
*
*****************************************************************
*                                                               *
*    C O N V E R S I O N    S U B R O U T I N E S               *
*                                                               *
*---------------------------------------------------------------*
*                                                               *
*  ENTRY POINTS  :                                              *
*  ============                                                 *
*  -  CI1AS  :  INTEGER SIMPLE PRECISION TO ASCII               *
*  -  CI2AS  :  INTEGER DOUBLE PRECISION TO ASCII               *
*  -  CF1AS  :  FLOATING SIMPLE PRECISION TO ASCII              *
*  -  CF2AS  :  FLOATING DOUBLE PRECISION TO ASCII              *
*  -  CI2F1  :  INTEGER DOUBLE PRECISION TO FLOATING ( .DFLT )  *
*  -  CF1I2  :  FLOATING TO INTEGER DOUBLE PRECISION ( .DRND )  *
*                                                               *
*---------------------------------------------------------------*
*                                                               *
*  AUTHOR  :  PH. ROSSIGNOL                                     *
*  DATE    :  JANUARY 1979                                      *
*                                                               *
*****************************************************************
*
*****************************************************************
*
*    C O N V E R S I O N    I N T E G E R   ====>   A S C I I
*    ========================================================
*
*  MODES D'APPEL  :  JSB CI1AS ( S.P. )   JSB CI2AS ( D.P. )
*                    DEF *+4        RETURN ADDRESS
*                    DEF VALUE      INTEGER VALUE
*                    DEF BUFAS      BUFFER ASCII ( RIGHT JUSTIFIED )
*                    DEF FORMT      DISPLAY FORMAT ( # CHARACTERS )
*
*****************************************************************
*
*
*****************************************************************
*
*    C O N V E R S I O N    F L O A T I N G   ====>   A S C I I
*    ==========================================================
*
*  MODES D'APPEL  :  JSB CF1AS ( S.P. )   JSB CF2AS ( D.P. )
*                    DEF *+4        RETURN ADDRESS
*                    DEF VALUE      FLOATING POINT VALUE
*                    DEF BUFAS      BUFFER ASCII ( RIGHT JUSTIFIED )
*                    DEF FORMT      DISPLAY FORMAT
*                                   - BITS 0 A 7 : # TOTAL OF CHARACTERS
*                                   - BITS 8 A 15 : # DECIMAL CHARACTERS
*
*****************************************************************
*
*
******************************************************************
*
```

```
*  C O N V E R S I O N   D . P . I N T E G E R  <====>  F L O A T I N G
*  ======================================================================
*
*  MODES D'APPEL  :   JSB CI2F1  ( INTEGER D.P.  ==> FLOATING )
*                     DEF *+3      RETURN ADDRESS
*                     DEF INTGR    DOUBLE PRECISION INTEGER
*                     DEF FLOAT    FLOATING POINT
*
*                     JSB CF1I2  ( FLOATING ==> INTEGER D.P.  )
*                     DEF *+3      RETURN ADDRESS
*                     DEF FLOAT    FLOATING POINT
*                     DEF INTGR    DOUBLE PRECISION INTEGER
*
*                     DLD INTGR    (A,B): NOMBRE DOUBLE PRECISION
*                     JSB .DFLT    (A,B): NOMBRE FORMAT FLOTTANT
*
*                     DLD FLOAT    (A,B): NOMBRE FORMAT FLOTTANT
*                     JSB .DRND    (A,B): NOMBRE DOUBLE PRECISION
*
***********************************************************************
*
```

3. Installation

Once the relocatable modules are avilable the library %IOLIB:GR:75 is created by running the MERGE utility with the *IOLIB::75 commands file:

```
,Instructions for MERGE to create library %IOLIB
,I/O subroutines
%HILIO::75
%PRTEX::75
%INP::75
%PROUT::75
ZBYTE::75
ZBITS::75
,Class I/O subroutines
%CLGET::75
%CLRED::75
,Tables handling subroutines
%DISPA::75
%GETPR::75
%PARFG::75
%DFORM::75
,Other subroutines
%TIME::75
%TIMA::75
%NOSEC::75
%DAT::75
%MOVE::75
%MYNAM::75
%CONV::75
```

4. Maintenance

To maintain the library by modification of existing modules or addition of new ones, follow these steps:

1- Compile appropriate modules
2- Modify *IOLIB:GR:75 for MERGE
3- Build with MERGE new %IOLIB
4- Edit all changed sections of this document("IOLIB:GR:75)
5- To list this :RU,PR,"IOLIB::75:-1

Important NOTE:

As existing programs use IOLIB, backwards compatibility must be maintained. Furthermore IOLIB must be the same on all installations.

For this reason it is advisable that when you have subroutines to contribute, you do it in agreement with either G. Raffi in Garching or E. Allaert in La Silla.