BESO System

The **BESO** Manual

Stefan Kimeswenger

version 3.00 (August 15th, 2026)

Rolf Chini Cerro Murphy Observatory (**OCM**)

Contents

1 Introduction and Functionality Philosophy 1

2 Server/Client Structure 3
- 2.1 The Server - functionality and command interface . . . 3
- 2.2 Simple CLI Client: . . . 5

3 The Commands 6
- 3.1 Structure of commands and answers by the server . . . 6
- 3.2 Commands to control the Server: . . . 7
- 3.3 Commands to identify the system: . . . 8
- 3.4 Command query room and spectrograph temperature probes: . . . 9
- 3.5 Commands to set parameters for the exposures: . . . 10
- 3.6 Commands to make and control exposures: . . . 12
- 3.7 Commands selecting storage format: . . . 16
- 3.8 Commands control the calibration unit: . . . 17
- 3.9 Commands controlling the FITS header: . . . 18

4 The FITS header 23

5 Statistics (signal-to-noise, background, etc.) 28

6 Remote Control of Server Computer 29
- 6.1 ssh server_start@jk15-beso.oca.lan . . . 29
- 6.2 ssh server_status@jk15-beso.oca.lan . . . 30
- 6.3 ssh server_kill@jk15-beso.oca.lan . . . 30
- 6.4 ssh server_restart@jk15-beso.oca.lan . . . 30
- 6.5 ssh computer_reboot@jk15-beso.oca.lan . . . 30
- 6.6 ssh camera_restart@jk15-beso.oca.lan . . . 31
- 6.7 ssh shutdown_beso@jk15-beso.oca.lan . . . 32

7 Data Acess 33

8 BESO Room Temperature System 34

A Why do we need our own Interface 36

B FPACK file format - pros and cons 37

C Coordinate Systems from/to ASA AutoSlew 41

D Expert Mode Commands 43

E Document & Software History 46

1 Introduction and Functionality Philosophy

The BESO camera by *Princeton Instruments* was ordered after a call for tender in February 2020. It has several restrictions in its 'operational environment' (see Appendix A). This made even the selection of the LiNUX© operating system (OS) level version special.

THUS, for the moment, no upgrades of OS should be applied on jk15-beso.oca.net!

The computer jk15-beso.oca.net for the camera has a USB3.1 type connection to the camera and is now running the RedHat line distribution Fedora 36. It is a i9-11900 @ 2.50GHz with an intentionally very over-sized CPU cooler to adapt for the elevation of the observatory. Currently, it has 64 GB of DDR4 RAM. To save energy, it intentionally has no dedicated graphics board, a 2 TB NVMe SSD, and two 8-TB HDDs. They are configured in mirror mode and can be used as intermediate storage for large, detailed log files, which the software creates, etc. The computer was also equipped with two PCIe cards. One provided two additional RS232 ports and one additional Ethernet port. The computer uses switches and hardware controls via RS232

1. with the (for our focal ratio dedicated adopted) eShel calibration module[1] by the French company *Shelyak*.
2. with its own brewed shutter control mechanism hardware.

The mainboard Ethernet port is connected to unit eno1 as jk15-beso.ocm.net (192.168.7.127) to the OCM facilities.

At the second Ethernet port (enp3s0, 10.0.2.1) a local network is installed, used for switching an power distribution module to turn on/off the own-brewed shutter unit, the eShell calibration unit containing an Arduino, the ThAr lamp high-voltage power supply and the camera, and a temperature probe and control system (again with an Arduino). For the ThAr power supply, the computer runs an independent watchdog task. This deactivates the ThAr lamp for the last 30 minutes after activation. That's important to prevent it from burning the tube. They are expensive and, because of the radioactive thorium, also hard to get.

The setup to control the system is that an INET socket server, accepting a kind of user-friendly/readable macro commands, translates them to the internal camera interface SDK structures and commands. It is written mostly in ANSI standard C[2] to provide long-term stability. However, requires a few sections of C++[3] type to comply the the external SDK of the camera (called PICam[4] library).

Also, some other hardware besides the camera is attached to this computer. These are shutter control, calibration units, high-voltage supply for the ThAr wavelength calibration lamp, ... They are incorporated into the server as well. For some of those, user commands exist. However, mostly

[1] Calibration module: https://www.shelyak.com/produit/pf0012-etalonnage-eshel/?lang=en
[2] ANSI C: https://blog.ansi.org/ansi/c-programming-language-standard-iso-iec-9899-2024/
[3] C++: https://www.open-std.org/jtc1/sc22/wg21/
[4] PICam: https://www.teledynevisionsolutions.com/products/picam-sdk-amp-driver/

they are controlled by the server independently and without the need for user interaction.
E.g., a client command defines whether the next group of images the shutter should open or close - so it's a light frame or if it is a BIAS/ZERO or DARK. The server then does the action itself, once the exposure starts. This parameter remains valid until it is revoked with another command. This minimizes communication. Then the server selects, based on this parameter, what to do with the hardware when receiving a START EXPOSURE. So the philosophy behind it is that the exposure start will need a series of identical parameters, so they are not repeatedly sent all together with the command to start the exposure as an extended parameter structure.

Generally, despite the power of the PC, the communication should not be excessive. The cadence, e.g., of polling values from the server, would not go beyond the frequency that makes sense physically. E.g., the Princeton Instruments temperature control has only full-center grades, and the changes are slow. A cadence of once a minute is already nearly useless. A cadence on the scale of seconds will prevent the server from doing the things it is designed for and thus will kill the operations.

The spectrograph setup is equipped with 4 temperature probes. Due to the technique and the required small size of at least some of them, the housekeeping sensors, as used elsewhere at OCM, are not applicable (the sensor heads are too big to fit). In addition, it reduces electromagnetic EMC interference by reducing the number of WiFi sending units in the room. Moreover, one of them has to be closely monitored by the server to prevent camera damage in case of failure of the external chiller while the camera Peltier fan is off. Thus, a direct guaranteed hardware connection without an interface to tic or the NITS server (and the potential of network losses) is required. They are attached to an ARDUINO©. This is finally attached via USB to the BESO server computer. The sensors are DS18B20 Digital Temperature Sensor (in the small TO-92 package)[5]. They should have a relative accuracy of ±0.1°C and an absolute accuracy of ±0.5°C. No calibration run was obtained to adjust them to an offset to correct for the latter. The vendor claims extremely low degradation with time as the internal probe is a platinum ceramic-based device. A full description of this system is given in a dedicated document [Kimeswenger, 2025].

[5] DS18B20: https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf

2 Server/Client Structure

2.1 The Server - functionality and command interface

The server is set up at the moment in the following way:

1.1 The server at startup tries to connect to a camera of this vendor and uses the first one the SDK library finds. On failure, it exits at that moment with entries in the logfile. The general logfile is in /var/log/BESO (which is a symbolic link to the larger HDDs). It does not start the communications server (like ASCOM[6] remote/ALPACA[7] does if a driver does not start).

1.2 The server is waiting for INET socket connection on a dedicated port. Currently, at startup, the port is searched for in /etc/services:

```
....
besocam     30020/tcp     # BESO Camera Server - not IANA registered
....
```

If this line is not found in /etc/services, a default port (fallback) is hardcoded in the C code. This makes it very flexible and changeable without touching the code on both sides - server and client.

1.3 The socket connection is normally expected to stay connected.
However, the server is recovering from a broken pipe by the client by going to the "connection wait" condition immediately after a broken pipe is detected. As tcp/ip does not have redundancy, the broken part of the information is lost. So in principle, one can use this server like an API, connecting and disconnecting after each command. I strongly recommend not doing so, as the LiNUX© OS normally does not immediately free a closed port in the kernel.
At the moment, only one client can connect at the same time. Multiple client connections to the same server can be developed, but it is somewhat complicated. In the case of servers like ssh, http etc., this is done by making a copy of the thread (by fork()) which then starts the communicating software (normally a shell). However, as the SDK does not allow multiple connections and asynchronous command interfaces between the server software and the camera, this will not work out here. So an own FIFO pipe would have to be implemented in the server. I did that indeed for the infrared cameras of the DeNIS sky survey Epchtein et al. [1997]. But it was very unsatisfying.
We should go for that workload only if a really strong science case demands it.

1.4 The command interface/socket is dedicated to short commands and to parameters:
It does not transfer the final image data (see description below). As the FITS headers in spectroscopy, anyhow, are fairly extended with other information required by the data pipelines in use nowadays,
Thus, as I had proposed and it was finalized in the vidcon Thursday 21.8.2025 15:00 CEST,

[6] ASCOM: https://ascom-standards.org/
[7] ALPACA:https://www.ascom-alpaca.org/

the server writes a whole, science-ready FITS file on a network-mounted disk (e.g. a network file system NFS mount to perun.oca.lan:/data/fits/jk15/... installed to be accessible in April 2023).

The NFS interface between a pair of LiNUX© machines is a highly optimized kernel-level transfer. Network read() and write() operations do not verify the data integrity. It has internal consistency (checksum) to verify the validity of the data and a sophisticated error correction algorithm (other than the MS Windows©-LiNUX© simulator of SMB Samba, which does not do so). Even if the bit error rate per unit time (BER) in Ethernet is small, it's the large amount of data we intend to transport over a longer period that matters (despite the fact that TCP/IP makes some better corrections than UDP/IP[8]. A comparable quality that we cannot achieve without a great effort.

Another reason is that the BESO data reduction pipeline originating from the Bochum group and based on the FERROS pipeline of ESO, which performs distortion correction, the Hough transform, and wavelength calibration, is still MIDAS-based. Later, it will even require adding even some of ESO HIERARCH[9] extension structures in FITS .

1.5 Additional header information creation:

EITHER the client provides (via parameter commands) the additionally needed information that the spectrograph does not know about, e.g., RA, DEC, OBSERVER, ...

OR we define an API interface where the Server can request this information. This is implemented for many units by using the NITS.io interface at OCM. That reduces the number of requests to the ASCOM/ALPACA interface by not directly asking the telescope interface. Important because it generates an interrupt to the schedule of that MS Windows© 10 computer. This is, as is known, very slow there[10]. Moreover, it is obtained only twice during an exposure - at the start and the end of the exposure. A spectroscopic single exposure can last up to an hour - that is a major difference to photometric observations. It will be the task of the toi, tic or whatever to e.g., move the dome in between (or use FOLLOW mode of the ASA system, which is not used currently in the photometric observations).

The server software is started manually (except in the case of one of the commands below). It requires the camera hardware to be switched ON. Thus, it cannot be part of the system init scripts.

[8] The Cyclic Redundancy Check (CRC-32) implemented in the algorithm of TCP/IP allows mathematically an undetected BER of 10^{-8} - this limit is "obtained" only on a busy network, where the collision rate of the 1500-byte MTUs is high. Thus, in reality, measurements give three orders of magnitude lower numbers in an internal gigabit network. However, as we transfer images of $4096^2 \times 2 \times 8$ bits, then still every 372^{nd} frame is affected. Mostly, we will not see a harmful bit error in the images. In a busy network, the CRC-32 allows the error rate (of undetected errors) to rise even up to 3 bits in every frame (theoretically).

[9] HIERARCH FITS HDU ext.: https://fits.gsfc.nasa.gov/registry/hierarch_keyword.html

[10] MS Windows© scheduler: https://www.javamex.com/tutorials/threads/thread_scheduling_2.shtml

2.2 Simple CLI Client:

The simple client is a text-based command line interface (CLI) even without syntax checks. The latter is done by the server. It can also be used easily for batches, simply by piping command sequences from an ASCII text file via stdin into it. One can type ahead commands. However, as in any kind of such a shell, this will not show up.

It can be started as

```
BESO_cli
```

on the control room computers just in a terminal. It can be used for the moment as a provisional access tool. However, later in its batch mode potentially also be used for background and daytime calibration data taking.

It connects to the INET socket and sends command by command to the socket. Then it waits for a reply. Every command gets a single reply from the server. The server does not send any packages autonomously to the client, nor answer with multiple replies to a single command action. It's on the user to do the interpretation.

When giving a command, it notes first the UT when the command was sent and then the UT when it received an answer. Any command described in this document (also the expert mode commands) can be given there. However, most observers will be fine with the subset described in the current version of the document for observing with BESO with the CLI.

Empty lines and lines starting at the first character with a hashtag ('#') are ignored. That is useful for batch mode (see the document about observing with BESO).

As it uses just the stdin for the input and the stdout for the replies, one can do any kind of UNIX redirection

```
BESO_cli < my_command_in_a_file.txt > the_result_output
```

or pipe applications

```
cat my_command_in_a_file.txt | BESO_cli | my_program_to_analyse_reply
```

The only thing one should be aware, as a redirection or a pipe collapses any program that is permanently waiting for keyboard input, the last command should be the regular BYE when using it that way.

The full source listing was removed from this document. The source is available on the documentation page if an own compilation is required. But most likely just taking a copy of the executable works for all current Linux distributions. Python has the same kind of libraries for opening and closing INET sockets[11]. However, I have never used and thus tested that one yet; I use Python only for quickly making plots, etc. Thus, from my side, only this standard ANSI C version is supplied.

[11] https://www.digitalocean.com/community/tutorials/python-socket-programming-server-client

3 The Commands

3.1 Structure of commands and answers by the server

Commands are in capital letters. An exception are that some expert parameters are filenames and path names, where any kind of upper and lower case can be combined due to their nature in the operating system itself.

The reply always starts with

`READY:`
: for any commands executing some action. The remaining part normally repeats the command. Exceptions are some special cases like `START EXPOSURE` and `GET CAMERA STATUS`.

`ERROR:`
: for an error condition described in the text following the double dot.

`INFO:`
: for commands asking for information – so most of the `GET` commands.

Here are some examples:

```
Command: GET CAMERA STATUS
got reply: INFO: is IDLE; no file was generated since server start

Command: GET SHUTTER
got reply: INFO: SHUTTER STATUS NORMAL (S00000000E)

Command: GET FITS OBJECT
got reply: INFO: GET FITS OBJECT = 'unknown'

Command: GET FITS OBSERVATORY
got reply: ERROR: FITS KEY 'OBSERVATORY' unknown

Command: START EXPOSURE OBJECT='NGC 6025' EXPTIME=10 WAIT
sent 2025-11-29T14:42:42.580
reply received 2025-11-29T14:43:14.719
got reply: READY: START EXPOSURE 10.00 s; file /data/fits/jk15/BESO/1009/jk15s_1009_14008.fits.fz

Command: GET CAMERA STATUS
got reply: INFO: is IDLE; last file created at OCM file server was /data/fits/jk15/BESO/1009/jk15s_1009_14008.fits
```

3.2 Commands to control the Server:

Besides the external commands to control the system as a whole (described then in section 6) this group of commands provides the important controls sent via the INET socket interface.

QUIT or EXIT
: This closes/stops the server on the PC.
It does not reset any camera parameters (e.g., cooling stays ON).
It does not restart the server software. That has to be initiated by an explicit command of a user logged in to the server computer.

DISCONNECT or BYE
: This is not really a server command. It is not sent to it, but only used by the client to break the connection. The latter will bring the server automatically into a state to wait for a (re-)connection.

RECONNECT CAMERA
: This command tries to disconnect and reconnect the camera on the software/interface level. A bit softer than the hard one (next command). Moreover, typical settings in the camera, like temperatures, exposure times, readout speed, etc, are kept as before.

RESTART CAMERA
: This is a remote emergency command. It will switch physically the electricity/power of the camera OFF and then, after 60 seconds ON again. It does not use any firmware or interface tool, but really the power line of electricity. As it waits twice, the total duration is about 2 minutes. As this is a hardware switch, and the cold camera normally should not be forced this way according to the vendor, **this should be used very carefully**.

RESTART PC
: This is a remote emergency command. It will completely reboot the PC. The procedure tries to do a decent shutdown of hardware (except the camera itself). Thus, in operational mode, this is slightly *softer* than the `server_kill@jk15-beso` (see section 6). Thus, connection loss occurs only after about 60 seconds. After that, the boot of the PC needs 35 seconds until, e.g., the `ssh` service is up again. The server software has not been started automatically again. You have to initiate a `server_start@jk15-beso`. Moreover, it does NOT switch the camera (respectively, no RESTART CAMERA is executed).

GET TIME
: Sends the UTC time and date of the server. This is a command to control the integrity of the system time. However, some network delays of a few milliseconds will be normal here.

SET TIME
: will try to restart the system time and synchronize with the international NTP servers. It needs about 10 seconds. Its reply contains the resulting UTC (like in GET TIME) at the end of the execution.

`UPTIME CAMERA`

Returns the connection time of the camera in seconds. This is not necessarily the real uptime, but the time since connected to the server or the last `RESET CAMERA`. The `PICam` interface does not allow reading a kind of firmware wallclock.

`UPTIME SERVER`

returns the uptime of the server in seconds.

3.3 Commands to identify the system:

These commands can be used to get details about the camera system. They are only of `INFO`rmation type and execute no actions. Moreover, all the information here is written anyhow to the headers of the image files.

`IDENTIFY CAMERA`

Returns a text string with the exact model.

`CAMERA SERIAL`

Returns the serial number of the device.

`CAMERA SENSOR`

Returns the exact sensor type in the device.

3.4 Command query room and spectrograph temperature probes:

The spectrograph setup is equipped with 4 temperature probes. They cover

- The room itself (T_OUT). This probe is mounted a few centimeters above the spectrograph, just into the air of the room.
- The spectrograph (T_INSIDE). This probe is mounted inside the spectrograph near the cross-disperser optics.
- The camera body (T_FLUID). This probe is inside the grid of the cooler body. Software tries to control it and switch on the Peltier fan if the external cooling is insufficient.
- The coolant line (T_CAMERA). This sensor is attached to the coolant line of the liquid circuit. It gives additional information about the external cooler/chiller hardware status to prevent camera damage.
- Additionally from the camera firmware: The detector temperature (T_CCD). The temperature reading of the detector in the camera.
- Additionally from the camera firmware: The detector temperature (T_SET). The current temperature setpoint for the detector in the camera.

`GET TEMPERATURE PROBES` or short `GET TEMP`

Replies with the four values of the temperature probes in the spectrograph and in the room, and furthermore, the firmware reading of the detector temperature and its setpoint.
The answer is like:

```
INFO: T_OUT=16.25 T_INSIDE=16.06 T_FLUID=13.94 T_CAMERA=15.13 T_CCD=-60.00 T_SET=-75
```

with the fields having the same meaning as the `FITS` header values of the exposures. To ensure access to these 4 probes (even if the server is inactive, e.g., the camera hardware is OFF), an independent communication (at a different socket) allowing reading them, as long as the PC is running, will be established later.

3.5 Commands to set parameters for the exposures:

Those commands mostly exist as GET and as SET versions. Only for the latter is the parameter required. Thus, the brackets mean it is *optional*.

`SET/GET CAMERA TEMP (value)`

Returns the real camera temperature, respectively sets the setpoint to a new value (units: Celsius). The reading in the firmware allows only full degrees currently. The vendor promised (potentially) a firmware upgrade, as internally better resolution does exist.

Setting a new setpoint does NOT wait until the point is reached!

The `GET CAMERA TEMP` is kept only for backward compatibility. The `GET TEMPERATURE PROBES` should be used now.

`WAIT CAMERA TEMP (timeout)`

This command waits until the real temperature and the setpoint match within tolerance (±1.0). It is intended for, e.g., a batch mode. Optionally, a timeout (in seconds) can be given. If the timeout occurs, an `ERROR` reply is given.

`GET CAMERA SETT`

Returns the currently set temperature setpoint (which was set with `SET CAMERA TEMP value`).

`SET/GET CAMERA GAIN (HIGH/MIDDLE/LOW)`

Returns the current setting or sets the different ADC gain values. The Princeton Instruments library knows 3 types; however, our SOPHIA firmware allows only two different ones currently – HIGH and LOW.

`SET/GET CAMERA SPEED (HIGH/MIDDLE/LOW)`

Sets/gets readout speed (3 MHz, 1 MHz, 100 kHz). Alternatively, the command can also use the speed in kHz (`SET CAMERA SPEED 100/1000/3000`).

`SET/GET CAMERA PORTS (1,2,4)`

Controls the number of output ports used. Tests have shown that most likely we will use (like with the ANDOR) only one port. The Princeton Instruments setup does not allow (like in the ANDOR) to select which one to use. `ONE` always uses the lower left, `TWO` always uses the LOWER LEFT+RIGHT , `FOUR` uses all of them. However, the tests up to now showed we have to use the `ONE` mode as the default due to differences in readout noise. A final decision is pending. However, it has no impact on this interface document.

`SET/GET CAMERA BIASCORR (ON/OFF)`

The firmware knows a kind of automatic BIAS correction. However, this does not work for individual frames.

Useless for our kind of observing; thus, the default is OFF.

`SET/GET CAMERA EXPOTIME [float value in seconds` Sets/gets the exposure time for the next exposure. Alternatively, it can also use the `FITS` entry name `EXPTIME`.

SET/GET SHUTTER (NORMAL/CLOSE/OPEN/H0/H1)

Parameters only for the SET command.

NORMAL ... sets the shutter to be active open during exposure only (science and flat)

CLOSE ... sets the shutter to stay closed during exposure (BIAS/ZERO and DARK)

OPEN ... sets the shutter to be always open - even before and after the exposure.

The latter also means during readout, smearing potential sources - that's thus a test configuration only.

H0 ... deactivates the use of the hardware switch of the HALL sensor on the shutter. This causes no handling of an error condition. However, the status is still returned (see below).

H1 ... activates (current default after start) the sensor use. Thus, if, e.g. the opening command fails after a certain timeout (see details of the new hardware in a dedicated document), it will power down (and thus close) the shutter again. Moreover, it is able to save energy after opening, going from 15 V (0.82 A ≡ 12.2 W) during opening operation to 5 V (0.27 A ≡ 4.1 W) during holding open. This, moreover, reduces heat load from the coil.

The status returned directly from the hardware control unit is a 10-character field starting with an S and ending with an E. Between them, we have 8 single-digit numbers as:

```
digit     name        state
  1       cmd_state   current operation mode
                      0 .. NORMAL operation
                      1 .. (permanent) CLOSE operation
                      2 .. (permanent) OPEN operation
  2       Error       potential error conditions
                      0 .. no error
                      1 .. OPEN operation, but shutter physically closed
                      2 .. CLOSE operation, but shutter open
                      3 .. NORMAL operation, powered shutter but closed
                      4 .. NORMAL operation, camera requests opening, but
                                still closed
  3       HALL_state  physical state of the HALL sensor at shutter
                      0 .. closed
                      1 .. opened
  4       IN1_state   state of the camera line 1 (normally "exposing")
  5       IN2_state   state of the camera line 2 (normally "reading")
  6       shutter_5V  5V line status
                      0 .. OFF
                      1 .. ON
  7       shutter_15V 15V line status
                      0 .. OFF
                      1 .. ON
  8       use_hall    status if the HALL sensor is used
                      0 .. NO - state is ignored when opening the shutter
                      1 .. YES - opening signal is stopped with an error
                                 if failed and shutter is sent to close
                                 on error
```

Thus, these fields can be used for their own more sophisticated condition, state, and error handling by the calling API or GUI.

SET/GET CAMERA FAN (ON/OFF)

Switches the camera's internal FAN. The OFF mode should not be used if liquid chiller cooling is not activated. The firmware does not allow checking the status of the chiller. Thus, careful monitoring is required if the fan will be deactivated (e.g., by looking for a sudden rise in the CCD temperature). The default is ON.

3.6 Commands to make and control exposures:

START EXPOSURE (WAIT) (*options*)

Starts a previously defined exposure.

WAIT (which is NOT the default) does not return with a reply until the exposure and the readout are obtained, and the file is created according to the definitions coming below.

The *options* are an add-on to give certain values of header information, which change potentially from frame to frame more quickly without calling extra SET commands before. Given here, they are overriding and overwriting the values from previously given SET FITS *key value* commands – and in one case, overriding the SET CAMERA EXPOSURE *exptime* command.

The order that they are given is not important.

String values have to be embedded in single quotes ('), like in the FITS standard. Do not use double quotes, as they are allowed themselves in FITS strings in the standard. However, they are allowed to contain empty spaces as well. *Even if allowed here according to the standards, do not use it for the* OBJECT *field as the databases in Warsaw do not handle it properly!*

Only frequently changing options, and options not obtained from the NATS.io server of OCM are implemented here. For a detailed discussion, see the sections about SET FITS and the FITS header below. Other than in the case of the individual SET FITS command, the equal signs are obligatory.

Currently, the implemented options are:

OBJECT='*text string*' ... defines the target's right ascension in decimal degrees [0,360).

RA_OBJ='*HH:MM:SS.s*' ... defines the targets right ascension in the given EQUINOX system (see the section about the FITS header for detailed discussion about Julian and Bessel system). It either contains the double dots, as shown here, the string, or an optional empty space is allowed as well (as given, e.g., in the SIMBAD database).

DEC_OBJ='±*dd:mm:ss.s*' ... defines the targets declination in the EQUINOX system. The sign is optional, and if none is given, a + is assumed.

RA=*double value* ... defines the target's right ascension in decimal degrees [0,360) given in the EQUINOX system (see the section about the FITS header for detailed discussion about

Julian and Bessel system).
CAUTION: only one of the two, RA_OBJ or RA, should be given (respectively, the last given strikes). The other value is then always calculated from the "partner" and, following common conventions, both versions are then given in the header. This convention allows a wider flexibility of the various data pipelines[12].

DEC=*double value* ... defines the targets declination in decimal degrees [-90,+90] given in the EQUINOX system. As above, only DEC_OBJ or DEC should be given.

EXPTIME=*float number* ... defines exposure time in seconds.

RA_TEL=*double value* ... defines the telescope position (in coordinates of the day) at the start of the exposure.

DEC_TEL=*double value* ... defines the telescope position (in coordinates of the day) at the start of the exposure.

OBSTYPE='*text string*' ... defines the intention like TEST, CALIB or SCIENCE.

IMAGETYP='*text string*' ... defines the mode like BIAS or BIAS, DARK, ARCLAMP, FLATLAMP, SKY, DOMEFLAT, or DOMEFLAT. However, in spectroscopy, the last one does not make sense.

PI='*text string*'

SCIPROG='*text string*'

OBS-ID='*text string*'

UOBI='*text string*' ... these latter four define the link to the various programs of the observatory and thus some metadata used for archives later and are defined by OCM.

The camera firmware would allow starting a series of exposures (e.g., for BIAS/ZERO). However, tests have shown that the exposures following the first one of a series, even if BIASCORR is switched off, have unpredictable shifts. Thus, this option is (intentionally) not offered by this server.
The minimum delay time (at exposure time 0.0) for a WAIT exposure is due to readout and I/O time is:

[12]Be aware that this convention at OCM fixed in the mail on oca-tech by Mikolaj 10/12/25, 14:09 is exactly the reverse of the suggestions on https://heasarc.gsfc.nasa.gov/docs/fcg/common_dict.html where RA/DEC are (optional) *sexagesimal* and RA_OBJ/DEC_OBJ are in decimal degrees.

SPEED LOW (100kHz)	
PORTS	TIME
1	2 min 52.8 sec
2	1 min 28.9 sec
4	46.1 sec
SPEED MEDIUM (1MHz)	
PORTS	TIME
1	21.7 sec
2	13.1 sec
4	7.5 sec

USE those values - they are absolutely reliable!. This includes storage time, however, on the internal NVMe SSD. On a network drive, it might be slightly slower. But normally we speak about a few tens of milliseconds here at the observatory.

The command replies on the success of a `WAIT` mode exposure with a `READY:` (including the filename. A `NOWAIT` mode exposure generates a `INFO:` reply (see example below).

`GET CAMERA STATUS`

Returns the information if the acquisition is still running. This command is required ONLY if one started the exposure without the `WAIT` option. The firmware of the camera itself does not give information back if it is still exposing or already reading (even their own LightField software of the vendor in Windows does that completely wrong in the display of a progress bar).

Once the data is ready, this command creates the file. Thus, a status request is required to move the read data from the camera memory to the PC (and then to its file). Starting a new exposure instead destroys the memory content (in firmware).

While the camera is not ready yet (exposure and readout), the reply starts as `INFO:` giving information about the remaining exposure time and the remaining readout time.

When the exposure is finished, it initiates the file creation and answers with a `READY:` reply and the filename of the generated file on the `OCM` file server (except that operation is disabled – see below).

If this command is used after the exposure is ready, it will reply with an `INFO:` saying that the camera is IDLE. Moreover, this reply contains the name of the last (before) stored file again.

> Thus starting the exposure without `WAIT` requires a loop of this command until the `READY` shows up!

Thus, an example of an exposure sequence is shown here as output from the simple CLI client. The timestamps are NOT generated by the server, but by the client and refer to the times when sending the command to the network and when getting the reply. In this example, first we set the readout time by using the 3MHz mode to a small value. Then I ask for 10-second exposures. The first exposure is in a waiting mode (thus lasting 18.3 seconds). Then an exposure is started with the nowait mode, and the camera temperature is asked for in between. The next status request after more than 18 seconds leads directly to a READY operation. As here the file is generated (and in this setting also is `fpack`ed), it lasts about 1.3 seconds. During the third frame (again in nowait mode), the status is polled more frequently

and thus INFO: answers give the expected remaining time. Finally, I asked 4 minutes after the last exposure again about the status and got the reply that the camera is IDLE and the information about the last obtained file.

```
socket created
connection to server 192.168.7.127 at port 30020 established
Command: SET CAMERA SPEED HIGH
        2025-09-18T13:58:02.542
        2025-09-18T13:58:02.569
got reply: 'READY: SET CAMERA SPEED HIGH'
Command: SET CAMERA EXPOSURE 10.0
        2025-09-18T13:58:16.638
        2025-09-18T13:58:16.649
got reply: 'READY: SET EXPOSURE 10.0'
Command: START EXPOSURE WAIT
        2025-09-18T13:58:30.129
        2025-09-18T13:58:48.471
got reply: 'READY: START EXPOSURE 10.00 s; file /data/fits/jk15/BESO/0937/jk15s_0937_08229.fits.fz'
Command: START EXPOSURE
        2025-09-18T13:59:10.029
        2025-09-18T13:59:10.232
got reply: 'INFO: CAMERA EXPOSURE with 10.00 s STARTED without waiting'
Command: GET CAMERA TEMP
        2025-09-18T13:59:17.004
        2025-09-18T13:59:17.015
got reply: 'READY: CCD_T 11.000000'
Command: GET CAMERA STATUS
        2025-09-18T13:59:31.101
        2025-09-18T13:59:32.549
got reply: 'READY: START EXPOSURE 10.00 s; file /data/fits/jk15/BESO/0937/jk15s_0937_08275.fits.fz'
Command: START EXPOSURE
        2025-09-18T13:59:47.664
        2025-09-18T13:59:47.990
got reply: 'INFO: CAMERA EXPOSURE with 10.00 s STARTED without waiting'
Command: GET CAMERA STATUS
        2025-09-18T13:59:50.772
        2025-09-18T13:59:50.882
got reply: 'INFO: CAMERA exposing; 15 sec. left (7 exp./ 8 read)'
Command: GET CAMERA TEMP
        2025-09-18T13:59:54.836
        2025-09-18T13:59:54.846
got reply: 'READY: CCD_T 11.000000'
Command: GET CAMERA STATUS
        2025-09-18T13:59:59.547
        2025-09-18T13:59:59.657
got reply: 'INFO: CAMERA reading - 6 sec. left (0 exp./ 6 read)'
Command: GET CAMERA STATUS
        2025-09-18T14:00:05.734
        2025-09-18T14:00:07.109
got reply: 'READY: START EXPOSURE 10.00 s; file /data/fits/jk15/BESO/0937/jk15s_0937_08319.fits.fz'
Command: GET CAMERA STATUS
        2025-09-18T14:04:00.812
        2025-09-18T14:04:00.822
got reply: 'INFO: camera is IDLE; last file was /data/fits/jk15/BESO/0937/jk15s_0937_08319.fits.fz'
Command:
```

ABORT CAMERA EXPOSURE

Aborts an exposure started in NOWAIT mode. The firmware does not allow other options. Moreover, it does not allow reading the data accumulated until that point.

3.7 Commands selecting storage format:

`SET/GET STORE FPACK (ON/OFF)`

activates the NASA `fpack`[13] [Pence et al., 2009] at storage (the extension is automatically additionally extended by `.fz`). This option is `ON` by default. See Appendix B for a detailed discussion of this option. If the `fpack` command fails internally (unlikely - never happened during my tests) the option is switched off for the next commands, and the storage and transfer to the current exposure to the `OCM` file server is done without compression

`SET/GET STORE PATH (path)`

Sets/gets the storage path - it has to be an absolute path. The default is currently set to `/data/fits/jk15/BESO`, which is an NFS mount point at `perun.oca.lan:/data/fits`. This command does not give the additional value of the day subdirectory. This is automatically (and always) created by the server if required.

Thus as now the storage will be

e.g. `/data/fits/jk15/BESO/0937/jk15s_0937_08319.fits.fz`

for an exposure started `2025-09-18 13:59:47.917 UT ≡ JD = 2460937.08319334`.

The default is stored in an include file and will be modified after commissioning to `/data/fits/jk15/raw` in regular operations. By this command, the destination can be changed. However, the server does not check the integrity of the mounting system. Pointing to a nonexistent path or to a directory where `poweruser` had no permissions will just cause an error when trying to write to it.

`SET/GET STORE OCM (ON/OFF)`

Selects whether the files are written in fact to the `OCM` file server. The default is `ON`. In fact, this is a kind of test feature and normally will never be `OFF` in regular operations.

`SET/GET STORE EXTENSION (value)`

Setting filename extension (default is `fits`). The new value is given without the dot.

[13] FPACK: https://heasarc.gsfc.nasa.gov/fitsio/fpack/

3.8 Commands control the calibration unit:

The internal calibration unit of the spectrograph consists of a section controlling a mirror (further called M5) in the optical train of the telescope, and the power for the lamps to obtain FLAT images (sometimes also called white light images in Echelle spectroscopy), and the ThAr wavelength calibration lamp.

SET/GET CALIB FLAT (ON/OFF)

This will switch the white light (in fact, a combined blue LED and a red tungsten lamp for the red part of the spectrum) ON/OFF respectively; ask for the information about the state with the GET branch of the command.

As of 2026-05-25 it does not automatically activate the mirror M5 anymore.

It cannot be used (by hardware) simultaneously with the ThAr lamp. Thus, if that is in the ON state at the time here, the FLAT is activated, it will toggle the other lamp.

SET/GET CALIB ThAr (ON/OFF)

This will switch the ThAr wavelength calibration lamp ON/OFF respectively, and ask for the information about the state with the GET branch of the command. As of 2026-05-25 it does not automatically activate the mirror M5 anymore. As above, it may toggle the white light lamp if activated before.

For the security of long-term saving of this delicate lamp, it is switched OFF automatically at the latest after 30 minutes even without a command.

However, the software is unable to see this hardware and will give improper information to headers or for GET commands. Its user responsibility to check the status on the screen.

SET CALIB M5 (ON/OFF)

Switches the mirror in the fiber injection unit. This is required to get light from the calibration fiber into the target fiber.

SET CALIB TOGGLEM5 (ON/OFF)

Caused by the experience of a similar, but 15 years older unit of the fiber injection unit by Shelyak, where the M5 is mounted, this command was added. This mirror, sometimes after a longer period of lack of use, is stuck. One sees that on the slit viewer camera, when activating the FLAT or the ThAr, no light is visible. For this case, a periodic fast switching of M5 is an option. With a 1-second period, this command activates a toggle (forever until switched off). Best to be watched on the slit viewer, with the FLAT being on (or a bright star).

As the state of the mirror after the OFF command is unpredictable, a SET CALIB FLAT ON or SET CALIB FLAT OFF after this emergency/troubleshooting command here is required to force a predictable state.

This is a bit of an emergency troubleshooting command. It adds a lot of stress to the hardware. Thus, manual switching at the calib unit in the control room might be the right option if this fails for too long a time. There is a dedicated document/video tutorial about the repair of this unit in case of failure available.

3.9 Commands controlling the FITS header:

The command interface is able to control many of the FITS header options. However, the system is designed as a standalone, independent controller to produce science-ready files. Thus, many of the default values are already meaningful.
Moreover, due to the nature of such a hardware-controlling server, not all keywords and sections of the FITS header are under user and command-line interface control. Some will be solely controlled by the hardware state.

SET/GET FITS KEYWORD (value)

This section implements a set of FITS header values. It does not (up to now) generate new keys! The latter is implemented in a dedicated #include file of the code. However, it is nice and easy to adopt. As a future extension, such a dynamic generation is foreseen.
Moreover, some KEYWORDs (see the next section) are generated by calculation and communication with the NITS.io server. Many are generated by communication with the instrument (camera and spectrograph calibration unit). Those cannot be controlled by the user commands listed here.
Therefore, only a certain set is available on the user/command interface. Several of them have decent default values and need no setting normally.
Moreover, there is a set of keys, where the values are obtained from, e.g., the NITS.io server, **as long as no setting is done with this command**. Other values (technical settings) (e.g. SATURATE, GAIN, RON, ...) do have default values investigated carefully during commissioning. They should not be overwritten, please, unless you really know what changed indeed. Finally, some of them are normally calculated (most likely more accurate) from other values (e.g., dew temperatures from relative humidity, air pressure, and temperature). The parameters with potential for autosetting by the NITS.io server are marked with an (*). Setting them here (once) with the command (or including it in the START EXPOSURE command as parameters) switches OFF the connection to the NITS.io server[14]. To *'reactivate'* the reading from the NITS.io server, an empty SET FITS KEYWORD has to be given without a value. Values marked with a ‡depend on a "twin", thus only setting one of them makes sense. As setting them immediately calculates the twin and overwrites the previous setting, always, the last call will win the setting. The next section lists all header entries - including those which are not mutable by the user with the command here.

The value parameter is separated either by a space or by an equal sign from the KEYWORD. Text strings can either be embedded in single quotes (') (this would be the FITS standard) or without. If none is given, all the remaining text in the command is interpreted as part of the parameter. According to the FITS standard, they can also contain empty spaces. Only characters between ASCII hex 0x20 and 0xfe are allowed. An exception is the single quote. As it is the delimiter of the string variable itself (again following the FITS standard definition), the handling is complicated. While the standard library makes a doubled single quote out of

[14]The server does not store configuration files. Thus, once the server is reset or newly started, the *'memory'* about such deactivation to query data from the NITS.io or using internal technical defaults is lost.

it (e.g SET FITS OBSERVER it's me) will end up in the header line

```
OBSERVER= 'it"s me' / Observers who acquired the data
```

Most data reduction software I tested reads only until the first single quote (thus, here only using the value in this example as it). Thus, my software marks it as a prohibited character and returns an ERROR message to prevent users from getting into a trap there later. However, in the COMMENT it is allowed.
WARNING for the users: due to the nature of the communication and the system, many values are derived at the start time of the exposure only.

Currently implemented KEYWORDs for the SET FITS command are:

OBJECT
: The object identifier - other than in toi - here empty spaces are allowed (that's FITS standard). Current default is 'unknown'.

IMAGETYP
: It defines the mode like BIAS or ZERO, DARK, ARCLAMP, FLATLAMP or SKY.

OBSTYPE
: It defines the intention, such as TEST, CALIB, or SCIENCE for a high-level separation in the archive (e.g., at ESO, ALMA, etc., used also for proprietary period selection).

GAIN (*)
: This option is to override internally already stored (a setup file on the computer) defaults. However, I worked hard on creating good default values for the various modes. Moreover, one might discuss whether we add several ones depending on the number of ports used (e.g. GAIN_01, GAIN_02,...), as the values slightly differ from amplifier to amplifier[15]

RON (*)
: same as for GAIN.

[15] Extension desired to GAIN_nn and/or RON_nn:
I already added two digits in this proposal, as the STA camera at the comming 2.5m will have 16 amplifiers.

RA (‡)

The right ascension of the (desired) target coordinate as a decimal float number in degrees [0,360.0)[16] for the EQUINOX - a negative number indicates an invalid setting.

DEC (‡)

as above for the declination in degrees as float number from [-90,+90]. A number below -90 indicates an invalid setting.

RA_OBJ (‡)

The same as information as given in RA, but as a string in either the ISO 8601-1:2019 standard format (other than RA in hours) as HH:MM:SS.ss or as used in the SIMBAD astronomical database with empty spaces as HH MM SS.ss. The settings of RA and RA_OBJ are complementary. Any setting command calculates the other one internally. Thus, when setting both, the last one takes precedence.

DEC_OBJ (‡)

As above for the declination, in degrees according to the ISO 6709:2022 standard as ±dd:mm:ss.ss or again like in SIMBAD as ±dd mm ss.ss. A positive sign is optional. There must not be a space between a sign (if given) and the first digit. Again, DEC and DEC_OBJ are coupled internally.

EQUINOX

The equinox value of the target coordinates. Default is 2000.0[17] — in fact it is Julian Year based J2000.0 (≡ 2000 January 1, 11:58:55.816 UTC), which slightly differs from the rarely used B2000.0. However, when asking for 1950.0 it switches to the system of Besselian Years B1950.0 (≡ 1949 December 31, 22:09 UT).

RA_TEL (*)

The real pointing reading of the telescope in the system of the day (EPOCH) in decimal degrees (not hours!). From it, the user-friendly, readable pointing value RA_PNT is calculated (see below). Currently, there is a bug in the ASA software causing inaccuracies of more than 30 arcseconds. Thus, this represents the reading you see on the TOI GUI or at the remote desktop at the AutoSlew of ASA. But it does not represent the physical sky position according to IAU.

DEC_TEL (*)

Same as for RA_TEL.

FOCUS (*)

Focus value of the telescope secondary mirror.

T_DOME (*)

[16] RA should be expressed in standard FITS in decimal degrees (not in hours like defined in the ALPACA interface document at https://ascom-standards.org/api/ and thus returned by the ASA software) - see: https://heasarc.gsfc.nasa.gov/docs/fcg/common_dict.html

[17] EQUINOX should be a float number - see: https://heasarc.gsfc.nasa.gov/docs/fcg/standard_dict.html

Temperature in the dome of jk15. Default if not obtained when using autosetting -999.

`RH_DOME (*)`
Relative humidity in the dome of jk15. Default if not obtained -999.

`DEW_DOME (*)`
Dewpoint temperature in the dome of jk15. Normally calculated from theory [see: Alduchov and Eskridge, 1996] using the the August-Roche-Magnus equation as follows:

```
RH = 100*(EXP((17.625*TD)/(243.04+TD))/EXP((17.625*T)/(243.04+T)))
```

where, `RH` is and `T` is air temperature (Celsius)[18]. It is highly recommended not to override these calculations from the international standard, and thus only use the settings of `T_DOME` and `RH_DOME`.

`P-WS (*)`
The air pressure at the weather station is in units of hPa. This is the real on-site value, not corrected by using a standard atmosphere to sea level counterpart.

`RHUM-WS (*)`
Relative humidity at the weather station (calculated as above).

`T-WS (*)`
The temperature at the weather station.

`DEW-WS (*)`
Ambient dew temperature - normally calculated the same way as `DEW_DOME`. Again, it is not recommended to override the calculation following the international standard.

`WIND-DIR (*)`
Wind direction in degrees by the weather station; 0.0° is N.

`WIND-AVG (*)` Wind average speed at the weather station in `m/s`.

`WIND-GUS`
Wind gusts at the weather station in `m/s`..

`TRACKING (*)`
Boolean `TRUE` or `FALSE` indicating if the tracking is activated. Default is `TRUE`.

`GUIDING`
Boolean `TRUE` or `FALSE` indicating if a (automatic) guiding system is activated. Default is currently `FALSE`.

`DOME-AZ (*)`
Dome azimuth (0 is North) at start of the exposure – I propose to remove that entry as it does not indicate the situation throughout a long spectroscopic observation well.

[18] For more details, see also the page of the U.S. Environmental Protection Agency https://qed.epa.gov/hms/meteorology/humidity/algorithms

DOME-SHT (*)
: Status of the dome shutter: 0,2=open(ing) 1,3=close(ing) 4&5=error.

MIRR-COV (*)
: Status of the mirror cover: 0&4=unknown, 1=closed, 2=move, 3=open.

FAN-MIRR
: Status of the ventilators at the mirror: Boolean TRUE or FALSE; Default is FALSE.

FAN-DOME
: Status of the ventilators in the dome: Boolean TRUE or FALSE; Default is FALSE.

LAMP-FLT
: Status of the dome flat lamp: 0=unknown, 1=off, 2=busy, 3=on. Default is 0.

PI
: A text string identifying the name of the Principal Investigator of the science program. Current default is 'unknown'.

OBSERVER
: The observer's name. Current default is 'unknown'.

OBSMODE
: A text typically gives something like "service", "visitor", etc. Current default is 'service'.

SCIPROG
: A text string identifying the name of the science program. Current default is 'commissioning'.

OBS-ID
: A text string identifying the BLOCK-ID of the observer's program. Current default is 'unknown'.

UOBI
: A text string identifying the unique observing block id. Current default is 'unknown'.

OCASTD (*)
: By default, an internal standard is used, as this is defined by the code section generating the header, in fact. However, I added that to override this setting by command (not desired).

4 The FITS header

In addition to the entries shown above in the SET FITS KEYWORD command, a set of parameters that have to be controlled by the server is applied. They can not be changed or overridden by user-given SET FITS KEYWORD interface commands. However, they often depend on other (hardware configuration commands) of the system. They are given here and described in detail (how they are obtained, etc.). Some of them are generated, even containing redundant information, and do comply with often-used software like (e.g. DS9, MIDAS, IRAF, molecfit) to allow direct further data reduction without wrapping/modifying headers or getting error messages.

Basics about the image and the HDU type:

Those need no further explanation.

```
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                   32 / number of bits per data pixel
NAXIS   =                    2 / number of data axes
NAXIS1  =                 4096 / length of data axis 1
NAXIS2  =                 4096 / length of data axis 2
EXTEND  =                    T / FITS dataset may contain extensions
```

Basics about world coordinate system:

World coordinates in spectroscopy are defined as wavelength against spatial position along the slit. However, for multi-order Echelle data, that does not apply. In the FITS standard, it would thus be allowed to be omitted. However, especially older software packages like MIDAS and IRAF create a whole set of (nasty) error messages. Thus, the world coordinates are defined here to exist and to be identical to the pixel coordinate system. As the original FITS standard is FORTRAN- based, they start all with 1 (not with 0 as it would be for C or Python).

```
CRPIX1  =                    1 / reference pixel
CDELT1  =                   1. / pixel stepsize
CRVAL1  =                   1. / reference pixel value
CRPIX2  =                    1 / reference pixel
CDELT2  =                   1. / pixel stepsize
CRVAL2  =                   1. / reference pixel value
```

Basics about the observatory:

```
OBSERVAT= 'OCA (Cerro Murphy)' / Observatory name
OBSGEO-B=            -24.59875 / [deg] geographic latitude of the observatory
OBSGEO-L=            -70.20171 / [deg] geographic longitude of the observatory (
OBSGEO-H=                2817. / [m] elavation of the observatory
ORIGIN  = 'CAMK PAN'           / Institution created this FITS file
TELESCOP= 'jk15    '           / Telescope name
```

Basics about the instrument:

Here, only the instrument name is hardcoded on the server side. The others are read from the firmware of the camera itself.

```
INSTRUME= 'BESO    '           / Instrument name
CAMERA  = 'SOPHIA: 4096B-2'    / Camera model
SERIAL  = '12733520'           / Camera Serial Number
SENSOR  = 'E2V 4096 x 4096 (CCD 230-84)(B)(MP)' / Camera sensor
```

Basics about the instrument state:

While T_CCD gives the real measured CCD temperature of the sensor, T_SET gives the desired value. The control is (sadly) very poorly implemented by *teledyne*. It neither returns sub-degree accuracy nor is it stable to ± a full degree. However, 2 or 3 degrees were also observed sometimes.

The camera has two GAIN-MOD modes. To follow the convention of OCM, these were implemented now as integer values, like for the Andor cameras (they have more modes). First commissioning data until May 2025 are here; the strings LOW and HIGH are used.

The camera can use either 1 (default), 2, or 4 readout ports CCD-PORT. Other than the Andor, the firmware does not allow for which ports are used. It always uses either the *lower-left* alone, the *lower-left* + *lower-right*, or all 4 of them.

The vendor allows a mode in which the camera executes an automatic bias correction. The results are horrible and unstable. Thus, this is disabled by default.

The allowed readout speeds CCD-SPD are 100kHz, 1MHz and 3 MHz.

Finally, the internal fan of the Peltier can be switched off, indicated in CAM-FAN, which slightly reduces the readout noise. However, if the external chiller fails, the camera might get damaged. Thus, an isolated background thread in the server checks the camera temperature probe and the chiller liquid temperature. However, one should be aware that this only works as long as the computer itself is active and operational.

```
T_CCD   =                -75. / [C] CCD actual temperature temperature
T_SET   =                -75. / [C] CCD set temperature
GAIN-MOD=                   1 / camera gain setting mode (1=LOW, 2=HIGH)
CCD-PORT=                   4 / number of used readout ports (amplifiers)
CCD-BCOR=                   F / bias autocorrection of vendor
CCD-SPD =                3000 / camera readout speed in kHz
CAM-FAN =                   T / status of camera Peltier system fan
```

Basics about exposure timing:

Besides the exposure time itself and the typical astronomical time information in civil (UT) time and date, the Julian Date (since days since beginning at noon at Greenwich 1st of January 4713 B.C. Julian Calender which is then in fact the 24th of November 4714 BC, at 12:00 PM in our nowadays Gregorian Calendar) and the Modified Julian Date (17th of November

1858 midnight at Greenwich and thus MJD = JD - 2400000.5) the mean sideral times at Greenwich reference and the local one are given. The airmass values are given at the start and at the end of the exposure. All values (except AIRM_E) are at the moment when the exposure start was sent to the camera (not when the command arrives at the server). However, there is some internal hardware delay for e.g., for cleaning the detector. A future extension will try to read out the camera's external trigger by adding some hardware to make it even more accurate. The telescope azimuth, altitude, and zenith distance are calculated from the coordinates and cannot be overridden by user command as well.
Finally, the user-readable versions of the actual pointing coordinates are given in RA_PNT and DEC_PNT. They are calculated from the hardware coordinate pointings RA_TEL and DEC_TEL. However, to make it user-readable, they are calculated from EPOCH to EQUINOX. This way, the user can directly compare the real pointing to the desired object pointing RA_OBJ/DEC_OBJ. However, due to a non standard handling in the RA_TEL/DEC_TEL calculation by ASA AutoSlew, I discovered 24.3.2026 for the EPOCH the header currently also gives a RA_EPOC/DEC_EPOC pair, which is the precession corrected RA_OBJ/DEC_OBJ for the observers EPOCH[19]. The offset between them can be used to calculate fiber centering differences. This offset depends on the sky position. For details, see the appendix C

```
 EXPTIME =                  300. / [s] exposure time
 DATE-OBS= '2025-08-10T14:30:37.292' / Start of exposure (UT)
 UTC     =              52237.29 / UT time (seconds)
 JD      =     2460898.10459829 / Julian date of observation start
 MJD     =      60897.604598287 / Modified Julian date
 GMST    =     42474.9420875706 / [sec] Greenwich Mean Sideral time
 LMST    =     25626.5308326878 / [sec] Local Mean Sideral time
 EQUINOX =                2000. / [yr] equinox for RA/DEC and RA_OBJ/DEC_OBJ
RA       =     78.6344666666667 / [deg] field RA  (at EQUINOX)
DEC      =    -8.20163611111111 / [deg] field DEC (at EQUINOX)
RA_OBJ   = '05:14:32.272'       / [HH:MM:SS.SS]   Program object RA (at EQUINOX)
DEC_OBJ  = ' -08:12:05.89'      / [+/-dd:mm:ss.s] Program object DEC (at EQUINOX)
EPOCH    =             2026.616 / [yr] Epoch of observation (of RA_TEL, DEC_TEL)
RA_EPOC  = '05:15:49.041'       / [HH:MM:SS.SS]   apparent pointing RA (at EPOCH)
DEC_EPOC= '-08:10:04.83'        / [+/-dd:mm:ss.s] apparent pointing DEC(at EPOCH)
RA_TEL   =     270.320693872876 / [deg] Telescope mount RA  (at EPOCH)
DEC_TEL  =     10.0006310914454 / [deg] Telescope mount DEC (at EPOCH)
RA_PNT   = '17:59:59.984'       / [HH:MM:SS.SS]   pointing RA at EQUINOX
DEC_PNT  = '+09:59:56.95'       / [+/-dd:mm:ss.s] pointing DEC at EQUINOX
ALT_TEL  =                32.4 / [deg] Telescope mount ALT at exposure start
AZ_TEL   =              301.12 / [deg] Telescope mount AZ at exposure start
AZ_TEL_E=              320.45 / [deg] Telescope mount AZ at exposure end
ZD_TEL   =                57.6 / [deg] Zenith distance at exposure start
```

[19]The official IAU approved routines for precession/nutation implemented in https://www.iausofa.org/ are used.

```
AIRM_S  =                1.86 / Airmass at exposure start
AIRM_E  =                1.86 / Airmass at exposure end
DOME-AZ =            179.9983 / [deg] Dome azimuth, at start , N=0 E=90
DOME-AZE=            179.9983 / [deg] Dome azimuth, at end,    N=0 E=90
```

Temperature probes inside of the experiment:

The experiment has some internal temperature probes supporting control of the cooling. The latter is also required to prevent system damage in case of failures. Only the T_CHILL currently cannot be obtained by a hardware reading and thus is set to the standard operation value, which has to be set at the front panel of the chiller system.

```
T_OUT   =                16.63 / [deg C] Temperature spectrograph room probe
T_INSIDE=                16.44 / [deg C] Temperature inside spectrograph
T_CAMERA=                16.19 / [deg C] Temperature inside camera body
T_FLUID =                14.38 / [deg C] Temperature of cooling fluid pipe
T_CHILL =                 13.5 / [deg C] Temperature of cooling fluid setting
```

State of hardware outside the camera but inside the experiment:

The calibration unit is able to switch the M5 mirror in and activate either a ThAr wavelength calibration lamp or a white light internal FLAT flat-field lamp to detect the positions of the orders. Thus, the CALIB is set accordingly. As of 2026-05-25 an additional flag for the status of the M5 mirror is added. The shutter status can be NORMAL (opening in case of non-zero exposure times during the exposure), CLOSE always keep closed (e.g., for DARKs), and for tests, always OPEN.

```
CALIB   = 'OFF     '           / Status of the internal calibration unit
M5      = 'OFF     '           / Position of the M5 mirror in the fiber unit
SHUTTER = 'NORMAL  '           / Status of the spectrograph shutter
```

Observatory provided probes:

Here a longer list of probes added from the observatory databases is provided. Namely, the meteo information might be important for data reduction. It contains both information inside the dome (and thus the mirror temperature) and also the information about the conditions outside.

```
T_DOME  =                7.57 / [deg C] Temperature of OCM JK15 dome probe
RH_DOME =                75.1 / [%] Relative humidity of OCM JK15 dome probe
DEW_DOME=             3.44075 / [deg C] Dome dewpoint (August-Roche-Magnus)
P-WS    =            726.9561 / [hPa] Pressure - weather station (WS)
RHUM-WS =                 78. / [%] Relative humidity - weather station
T-WS    =            5.666667 / [deg C] Temperature - weather station
DEW-WS  =            2.129899 / [deg C] Dome dewpoint (August-Roche-Magnus)
```

```
WIND-DIR=                338. / [deg] Wind direction - WS, N=0 E=90
WIND-AVG=            16.98752 / [m/s] Wind average speed - weather station
WIND    =            16.98752 / [m/s] current Wind - weather station
TRACKING=                   T / Mount tracking:  T=on F=off
GUIDING =                   F / Guiding status:  T=on F=off
DOME-SHT=                   0 / Dome: 0,2=open(ing) 1,3=close(ing) 4&5=err
MIRR-COV=                   1 / Mirror cover: 0&4=unkn 1=closed 2=move 3=open
FAN-MIRR=                   F / Ventilator mirror status:  T=on, F=off
FAN-DOME=                   F / Ventilator dome status:  T=on, F=off
LAMP-FLT=                   0 / Dome flat light: 0=unknown 1=off 2=busy 3=on
```

Project and observing block (OB) descriptions:

Here the interface can add various information about the observer, observing block, and project. These entries should be really used to make follow-up possible.

```
PI      = 'Przybilla/Kimeswenger' / Name of the Principal Investigator
SCIPROG = 'Innsbruck Stars'    / Name of the science program
OBS-ID  = 'unknown '           / BLOCK-ID.SEQ-ID.LOOP-ID (8.3.2) base64.dec.dec
UOBI    = 'unknown '           / Unique observing block id
```

File integrity check entries:

The real creation date of the file and the checksum of the header unit, as well as that of the data, are given

```
CREATION= '2025-08-10T14:35:39.949' / FITS file creation time (UT)
CHECKSUM= 'Uc2WXc1TUc1TUc1T'   / HDU checksum updated 2025-08-10T....
DATASUM = '1147122986'         / data unit checksum updated 2025-08-10T...
```

Software version control:

The two fields give information about the software versions. While the one is a string defined in the main include file of the code, the second looks up at server start the age of the compiled executable and thus the real build version.

```
SW_VERS = '2.00 2025-09-16' / Main software version of the server code
SW_BUILD= '20251117_132406' / Software build date and time
SW_IAU  = '19 (2023-10-11)' / Version of IAU SOFI library
```

When using the fpack this adds further lines (which are removed after funpack) about its compression tool and version.

5 Statistics (signal-to-noise, background, etc.)

A special section added 14.8.2026, not mentioned before, contains some first-guess statistics. They are only intended as a quick-look analysis and do not replace a real analysis in a pipeline.

The first 2 values for the readout noise and the gain originate from detailed lab experiments, when I was able to illuminate the whole chip. Something hard to derive with the small orders of an Echelle image. They are listed in a table in the software and switched depending on the selected mode (readout speed, gain setting high or low, ...).
The flipping is user selectable by a `SET FITS` command. However, the version given here is recommended, having the red ones in the upper part, and the blue ones low. Physically, the chip is reversed in this camera. However, most pipelines prefer it this way.
The saturation value is where real saturation starts. Nonlinearity starts a bit before. However, due to charge transfer, heavily overexposed frames (tested with FLATs) can reach ADU values up to 180 000. But that's from electrons from other rows. Thus, this value was chosen as an indicator for the physical limit of the data.
The minimum and maximum are 'only' taken in the inner part from row 1000 to row 3096. However, there is no single-pixel filtering (cosmic removal) which certainly dominate the maximum value. The background is derived in a region between row 600 and 800, but only ±500 from the central line. There is no regular illumination. Thus, this is BIAS + DARK current combined.
Finally, the signal-to-noise ratio (SNR). It is also displayed now in the exposure reply. It is derived from the brightest order in FLAT images at the image position of the rectangle `1970,2200:2130,2215` integrating 16 rows, compared to the inter-order flux just below and taking into account, in a Gaussian error propagation, the readout noise. This value ignores (the at -75°C hardly measurable) dark noise. Moreover, it assumes the position of the order. Thus, the correct temperature setting in the room. It will be an issue of adoption if, e.g., mechanical changes are done at the optical drain. It is intended for the user to decide ad hoc about correction factors in the exposure time, for e.g. bad seeing and high airmass, and thus slit losses, or cirrus clouds, were spectroscopic observations still make sense, while imaging is stopped already. This SNR assumes a more or less nearly continuum source over that region. Emission line objects or the ThAr lamp give much smaller numbers.

```
GAIN    =                 0.99 / [e-/ADU] Gain
RON     =                  8.2 / [e-/read] Readout noise
Y-FLIP  =                    T / Camera raw data is flipped in Y direction
SATURATE=               130000 / [ADU] Data value at which saturation occurs
DATAMIN =                 1655 / [ADU] minimum data value
DATAMAX =                41346 / [ADU] maximum data value
BACKGR  =             1705.876 / [ADU] background (bias+dark) estimator
SNR     =             234.4228 / estimated peak signal-to-noise ratio (SNR)
```

An example of the communication taking 40 seconds FLAT images.

```
got reply: 'READY: SET CAMERA EXPOSURE 40.0'
Command: SET CALIB FLAT ON
2026-08-15T04:19:17.861
2026-08-15T04:19:19.172
got reply: 'READY: SET CALIB FLAT ON'
Command: START EXPOSURE WAIT
2026-08-15T04:19:19.172
2026-08-15T04:20:18.870
got reply: 'READY: START EXPOSURE 40.00 s; peak SNR=233.4; file /data/fits/jk15/BESO/1267/jk15s_1267_68008.fits.fz'
```

6 Remote Control of Server Computer

The computer has (besides the normal `poweruser`) a set of special users for interaction with normal observers. It is **NOT** intended that normal users/observers log in to this computer. These users are called with a normal `ssh` *user*`@jk15-beso` command. The password is the normal one for the `observer` at the computers in the control room[20]. However, it will not start a shell on login; instead, it runs a set of special scripts.

The *user*(name) selects the type of desired operation.

This option/command does, more or less, what the observer does when logging in to the `ASA` computers via remote desktop and starting the individual facilities there. However, here is a simple single package without requiring knowledge of details behind the internal structure.

6.1 ssh server_start@jk15-beso.oca.lan

This command will start the server software. Intentionally, it is not included in the start scripts of the system after reboot. It might cause conflicts with the hardware of the camera, pumps, etc. Thus, it at least has to be run after any reboot of the computer. An example shows that it starts and echoes success (or, less likely, an error message).

```
[stefan@lap1-c706 BESO]$ ssh server_start@jk15-beso.oca.lan
Last login: Tue Apr  7 17:39:48 2026 from 192.168.7.83
INFO: server output at /data1/LOG/Server_2026_04_07_17_42_Tue.out

SUCCESS: BESOCam_server started in the background as process 3320

Sleeping 10 seconds to start Server ...
Try to connect to Server with a client and read temperatures ...
INFO: T_OUT=16.31  T_INSIDE=16.50  T_FLUID=14.19  T_CAMERA=16.50  T_CCD=-69.00  T_SET=-75.00

End of Sterver start script ...
Connection to jk15-beso closed.
```

If called twice (or on top of an already started server), an error condition arrives.

```
[stefan@lap1-c706 ~]$ ssh server_start@jk15-beso.oca.lan
server_start@jk15-beso's password:
Last login: Tue Nov 18 13:17:45 2025 from 192.168.7.83

ERROR: Server running already as process 4574
       Command IGNORED !

Connection to jk15-beso closed.
```

[20] For some computers in the control room a `ssh-copy-id` was executed, saving from even asking a password. However, now, as no shell is started anymore, it does not work that way. If one intends to add further computers, one has to edit the files in the `.ssh` directory in that user's home directory - it is in fact only one with several user names. That requires root access.

6.2 ssh server_status@jk15-beso.oca.lan

This command gives information about whether the server executable is, in fact, running. It cannot check whether the socket is alive properly or if some misaligned command in the interface caused some rampage condition.

```
[stefan@lap1-c706 ~]$ ssh server_status@jk15-beso.oca.lan
server_status@jk15-beso's password:
Last login: Tue Nov 18 13:17:59 2025 from 192.168.7.83

INFO: Server is running as process (PID) 4574

Connection to jk15-beso closed.
```

6.3 ssh server_kill@jk15-beso.oca.lan

This command will use the UNIX/LiNUX system type kill commands to stop the executable (if existing). Normally, the server should be stopped with the command interface (see EXIT command above). It is thus a bypass in case the command interface fails.
It first tries a soft version via an EXIT in the user interface, then a soft UNIX/LiNUX kill (with -QUIT signal). This allows the server to do normally, if still alive, a regular shutdown in the same way, as if it got the EXIT command through the command interface. It also tries to close hardware connections, such as to the calibration unit, and switch off the ThAr calibration lamp (if running) .... Especially the latter is important due to the limited lifetime of this device.
If that was not successful after a small grace period of a few seconds, it does it the hard way with a -KILL signal. This then does not allow (like for any other UNIX/LiNUX thread) soft closing of devices, handles, etc. However, it powers off at least the ThAr lamp to save it.

```
[stefan@lap1-c706 BESO]$ ssh server_kill@jk15-beso.oca.lan
Last login: Tue Apr  7 17:33:50 2026 from 192.168.7.83

Try a SOFT software based exit  ...
Sleep 10 seconds ...
SOFT exit successfull  ...
Connection to besocam closed.
```

6.4 ssh server_restart@jk15-beso.oca.lan

This combines in one step the server_kill and the server_start operations.

6.5 ssh computer_reboot@jk15-beso.oca.lan

This is the hardest of all those operations. It really reboots the computer. For security reasons, it has to be entered twice within 120 seconds to work.

When giving it for the first time, it will give a warning only and initiate the timeout server:

```
[stefan@lap1-c706 ]$ ssh computer_reboot@jk15-beso.oca.lan
Last login: Tue Apr  7 17:33:58 2026 from 192.168.7.83

WARNING: This will fully reboot the computer.
WARNING: However, it is unable to handle the camera hardware.
```

```
         Is that your intention?
WARNING: Otherwise, contact the BESO shutdown procedures
         in the documentation.
INFO: If this is indeed your intention, give the command AGAIN
      within 2 minutes - it will finally then execute the reboot!

Connection to besocam closed.
```

Given the second time is below 120 seconds, it will finally do it:

```
[stefan@lap1-c706 ]$ ssh computer_reboot@jk15-beso
Last login: Tue Apr  7 17:39:45 2026 from 192.168.7.83

SUCCESS: second time called (within 2 minutes) - REBOOT

NOW: Tue Apr 7 17:39:48 UTC 2026
LAST: Tue Apr 7 17:39:45 UTC 2026

socket 1 - off
socket 2 - off
socket 3 - off
socket 4 - on
Connection to besocam closed.
```

6.6 ssh camera_restart@jk15-beso.oca.lan

While the previous actions used only the computer and kept the camera untouched, this step now restarts the camera physically (and the Server software).

```
[stefan@lap1-c706 ]$ ssh camera_restart@jk15-beso.oca.lan
Hard restart of BESO camera!

WARNING: This command physically cuts the electricity of
WARNING: the camera. Then restarts it after 10 seconds. Thus
WARNING: the camera is back to firmware setting.

WARNING: Moreover it stops the Server software and try to
WARNING: restart it. But that causes all settings there lost
WARNING: as well. However the quick restart allows most likely
WARNING: to restore the cooling to previous state.

HINT: Check camera temperature after a few minutes.

QUESTION: do you really want to continue (Yy/Nn) ? y
Proceeding ...
Deactivate Server if running ...

Try a SOFT software based exit  ...
Sleep 10 seconds ...
SOFT exit successfull  ...
Camera power OFF ...
Sleeping 10 seconds ...
Camera power ON ...
```

```
Sleeping 10 seconds ...
Starting Server ...
Sleeping 10 seconds to allow Server to start all hardware ...
T_OUT=16.56  T_INSIDE=16.50  T_FLUID=14.31  T_CAMERA=16.56  T_CCD=-71.00  T_SET=-75.00
```

6.7 ssh shutdown_beso@jk15-beso.oca.lan

This is the complete shutdown (PC + Camera + heaters described in the next section). There is no external control available on the chiller in front of the BESO room. That has to be deactivated manually. If you have time, it's better to warm up the camera by command and wait a bit. However, sometimes that's not possible.

Be aware that you have to switch off the power lines (see dedicated shutdown and restart manual) to avoid the BESO PC starting automatically after a power failure/cut.

In this example, I answered with *no*!

```
[stefan@lap1-c706 ]$ ssh shutdown_beso@jk15-beso.oca.lan
Hard shutdown BESO PC including switching off the camera!
Did you try first to warmup the camera e.g. with the
command SET CAMERA TEMPERATURE +10 in the command line tool BESO_cli?
If you are not in emergency please try this and wait until you see
with the command GET TEMP, that it warmed up.

WARNING: Server is NOT running !
HINT:    Did you check first if that's the reasion for the problem ?


WARNING: This command physically cuts the electricity of
WARNING: the camera. Then it shutdown the PC.

QUESTION: do you really want to continue (Yy/Nn) ? n
Exiting...
Connection to jk15-beso.oca.lan closed.
```

7 Data Acess

The files are regularly written for normal access to perun.oca.lan. The directory for BESO data currently is

/data/fits/jk15/BESO

As this directory is remotely mounted on most computers here, it is the same path on most machines. There, the directory structure is the same as for the other telescopes. Daily, a directory is created with a 4 digit number representing the last four digits of the Julian Date of the time of exposure start before the comma. Within these directories the files start with jk15s_ (s for spectrograph) followed by the same 4 digits as before, a second underscore, and then the first 5 digits of the Julian Date after the comma. The extension is either fits or fits.fz. The latter means that it was run through the NASA fpack reducing the file size typically by a factor of 5, from 67 121 280 bytes to something around 13 000 000. Thus, the file taken with JD = 2461266.4629161 (which is DATE-OBS= '2026-08-13T23:06:36.222') got the path and name:

/data/fits/jk15/BESO/1266/jk15s_1266_46291.fits.fz

This should be the regular data access point for the user.

In rare cases, data transport due to timeouts on the network transfer failed. Then the user may access the data by going to ssh observer@jk15-beso.oca.lan (normal password) directly on the computer. There, in the directory formed by the date /data1/fits/YYYY/YYYY_MM_DD, the files can be found for some time of a few months. The date *"rollover"* is approximately at noon local time. Thus, data from the second half of a night and morning calibrations go to the same directory. However, the files are named in ESO convention. Thus, the same file as above is now named with a prefix for the instrument, and then the UT date and time of writing the file (so a bit later than DATE-OBS). Thus, in the case before, it is the file BESO_2026_08_13T23_06_54.716.fits.fz. However, when retrieved in that emergency way from this backup, it is a very simple script that everyone can create to rename it from the JD entry in the header if required for the own pipeline structure.

8 BESO Room Temperature System

The BESO room has a dedicated AC system. This is completely controlled and configured in firmware by the climate technician. He gave the sweet spot of 16°C from his ESO experience, a point between summer and winter, for a bit more than two years. The whole centering of optics during the last 2 years was thus obtained using this setting. It is also not changeable by me (neither in software nor in the room). Analysis of data showed that

1. Already temperature changes of 1°C change completely the optics. The fibers are not centered on the slicer anymore; the spectra move on the chip and get unfocused at the edges, ...
2. Getting BESO in thermal equilibrium needs more than a day, even if room (air) reading looks fine quickly. This is certainly due to the enormous mass and the low thermal conductivity that the up to 60cm wide and 10 cm thick ceramic mirrors do have.
3. The consumption of the AC is fairly high. However, cooling down costs more than heating up when it is cold outside.

So, as of early August 2026, to partly save the energy of the AC, which was needed at night with Δ_T of only 2.5°C (6.8.2026, 01:24 local time) already 1.5 kW, a heater system was installed in the room. It allows heating with less energy than the AC system requires. There is a dedicated document about these tests. The current (as of 14.8.2025) installed version is described here now. It contains two normal heaters from the market. They are positioned diagonally in the room and are connected via a black distributor to one common socket of the computer-controlled power distribution unit (PDU). There is a piece of software installed in the BESO PC, which switches then ON and OFF when going below 15.9°C and above 16.0°C on the room sensor. However, as the fans do not have very high turbulence, the hysteresis is higher than 0.1°C in the room. But measurements show that the inside BESO temperature probe shows very stable behavior.
If one does not like the unit to be active (e.g. due to energy crisis), just switch it off at the black distributor (Fig. 1). However, be aware

- **BESO data will be crap if the temperature is not stable.**
- **BESO needs a day or so in advance of use to stabilize.**
- **This system is NOT able to cool.** Thus, even in winter, once the outside temperature reaches 14 to 15 degrees during daytime and the sun shines on the building, the inside temperature easily reaches up to 18°C. Data will be crap again. **Thus, the AC is required !**
  And again - also in cooling well in advance of the observations to obtain useful data.

The measurements of the first full night in service are shown below (Fig. 2). The Energy requirement (now still without additional insulation) is very constant until sunrise adds load from outside, lowering the required energy. Currently, the load reaches, in these very cold nights with Δ_T of 11°C at most 800 W, and this unit can go to 1500 W. Thus, the estimated minimum is -6°C. The heaters are at the half-power position each. However, do not switch them to full. That will overload the fuse.



Figure 1: The BESO room heater system and the location for complete de/re–activation.

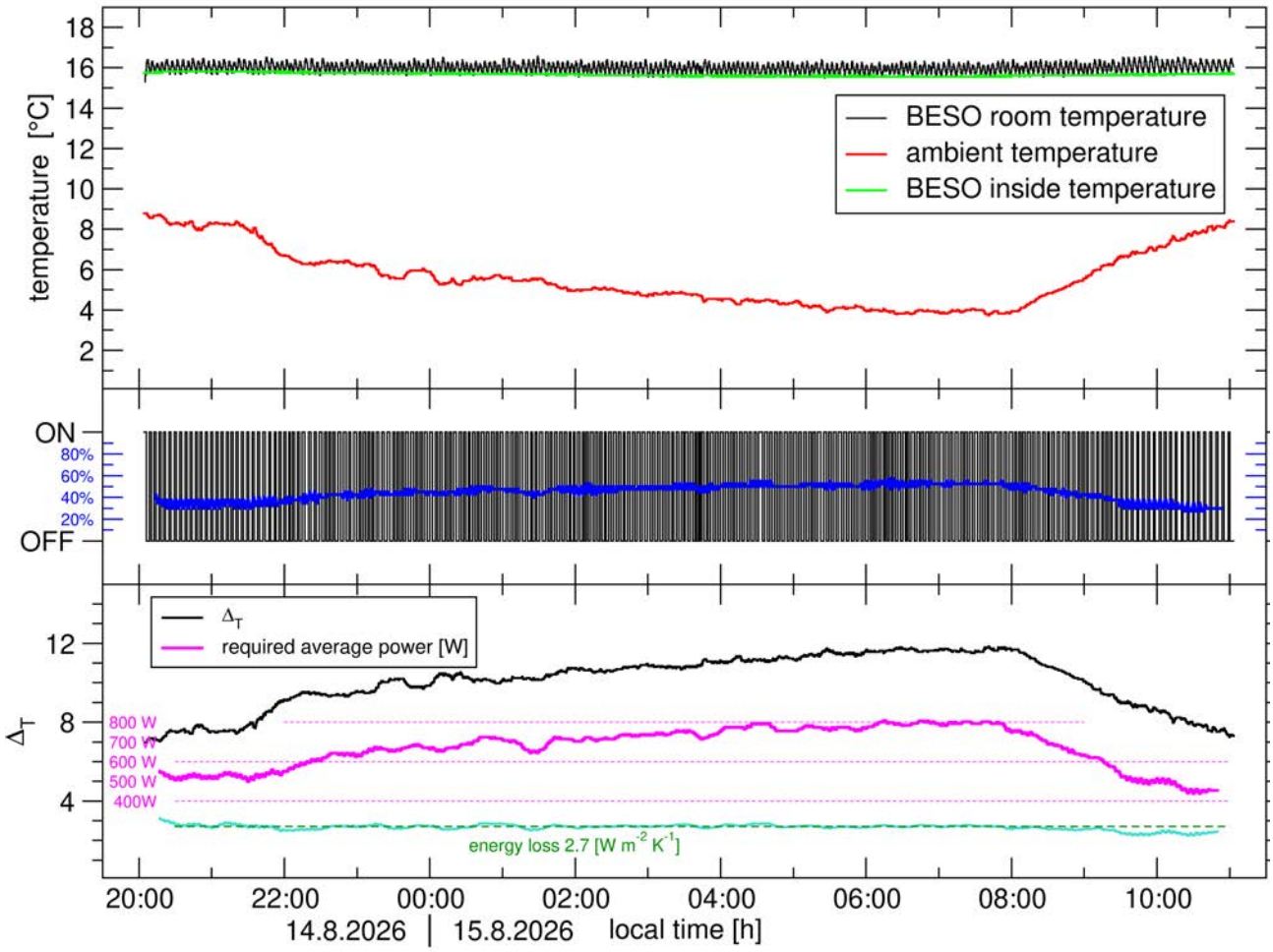


Figure 2: The measured temperatures on the fairly cold night 14./15.8.2026 (upper panel). The ON/OFF cycles of the heating system and the running average in percent of time (middle panel). The Temperature difference, the required power, and the energy loss per degree temperature difference and area (lower panel).

A Why do we need our own Interface

The BESO Camera is a system by Princeton Instruments (in the meantime a sub-company of *teledyne*). It provides a strange MS Windows© Software without remote interfacing capabilities in some standards like ASCOM, INDY, . . . , a very strange GUI with all settings hidden in sub-menus or even below them. It is called LightField[21]. It is proprietary and allows us to run only with a USB license dongle we obtained. The data format is also proprietary and thus unknown to regular astronomical data reduction software. Moreover, the offered FITS export writes FITS files are making trouble. E.g., the key names for the variables are longer than a full 80-character line, and the data follows on the second line; key names contain empty spaces - this is a bit like ESO's HIERARCH architecture. However, extension beyond a single line is not a decent definition in any other system. Interfacing (with extra costs) is provided by them only to LabView and MATLAB. Moreover, LightField and the libraries backing LabView and MATLAB run only with an outdated version of the Microsoft C++ Redistributable Library in versions before 2011. One has to downgrade a modern MS Windows© PC for that. This causes troubles with more modern software on the same PC.
I had in May 2024 to downgrade my brand new MS Windows© 11 Laptop during their remote tests; otherwise, the Lightfield started with the error message "This computer has too low resources - stop other programs and retry thereafter"
The laptop was completely free of other stuff - a 16-core i9-13900HX with 128 GB RAM and a 2 TB NVMe SSD. Their technician then logged in via TeamViewer and did the downgrades. No more comments about their software competence.

However, they provide a C/C++ SDK library. Again, has, as I found, severe restrictions. In Ubuntu systems (they recommend), it runs with the libraries only until version 16.04 LTS. Tried on my computer back in my lab online a downgrade of libraries with 18.04 and 20.04. Then I gave up and tested the RedHat line distros. I made it work (their example code - not mine) at that time with systems up to Fedora 36. As this is significantly more advanced than Ubuntu 16.04, this OS was selected. Another bug is that the whole interface runs only as root.

This issue is not in our hands. We have to rely on the camera firmware vendor. However, as the PC is very isolated, not allowing attacks from outside, and as it will not be used as a workstation running browsers, etc, and thus no infection by malware through web serving can be injected, the risk of the outdated OS is very marginal.

[21] LightField: https://www.teledynevisionsolutions.com/products/lightfield/

B FPACK file format - pros and cons

The FITS packing fpack[22] was developed to specially compact data arrays of integer data. It is a lossless compression like compress, gzip, bzip2, xz, zstd and lz4 used as well on UNIX/LiNUX© computers. It is very efficient for our typical unsigned integer format data. The default compression by the NASA program is the so-called Rice compression. Normally, on our integer data, the other modes are less efficient according to the original authors [Pence et al., 2009], or, in the case of Hcompress, not lossless. Be careful: with float data, it is in the default configuration, NOT a lossless compression in the so-called rice algorithm. However, with float data, any compression is very inefficient.

The main difference from the standard compressions listed before and included in the systems it knows about the file type. Thus, it is able to keep the FITS header units (HDU) decompressed.
This is certainly the main **pro**:

We are able to read metadata without decompressing the file.

However, that's potentially also the main **con**:

It works only with FITS files.

E.g. data of an image field as produced by ASCOM drivers cannot be treated that way (moreover, most ASCOM drivers on the MS Windows© side do not support any compression at all). Thus, the computer obtaining the data / doing the data acquisition has to build at least rudimentary FITS files to allow this compression. Moreover, one should mention the added value of CHECKSUM handling in fpack. Control sums for the data and for the header unit in the header itself are created directly (and checked when opening/unpacking the files). The handling of external checks such as md5sum is not required.

Thus, the different compressions were compared with the real data frames of BESO. This camera has an 18 bit analog digital A/D converter, and thus the frames are stored as 32-bit unsigned integers. The files currently have 67 118 400 bytes each. As a test bench, the observations taken 2025-08-05 were chosen. The two time values correspond to user-space CPU time (real processing) and to total time (including I/O on the NVME internal disk of the besocam computer, aka jk15-beso.oca.net). In the case of fast algorithms, the I/O of the larger result file when decompressing makes a significant fraction of the total time. Some algorithms have a fairly large standard deviation at their compression rate. Thus, it is not guaranteed to obtain a certain compression rate.

method	compression [sec/frame]	decompression [sec/frame]	file size [bytes]	result size [%]	stdev [%]	compression ratio
fpack	0.089/0.104	0.124/0.181	15 004 800	22.35	2.03	4.47
compress	0.529/0.552	0.131/0.164	19 206 097	28.61	7.73	3.49
gzip	4.091/4.106	0.257/0.270	21 484 260	32.01	5.65	3.12
bzip2	1.639/1.653	1.163/1.184	14 440 401	21.51	2.95	4.64
xz	25.341/25.420	0.539/0.556	14 997 364	22.34	2.61	4.47
zstd	0.248/0.252	0.057/0.058	20 135 648	30.00	6.34	3.33
lz4	0.097/0.112	0.025/0.042	30 439 304	45.35	8.69	2.20
lz4 -best	12.878/12.855	0.022/0.038	23 116 741	34.44	6.98	2.90

[22] FPACK: https://heasarc.gsfc.nasa.gov/fitsio/fpack/

Concluding that fpack beats all other algorithms (except decompressing time). In compression efficiency bzip2 (as best) and xz are also good. However, their computational times are 18 and 285 (!) times higher, respectively. Taking into account the additional value of the decompressed header, which allows searching for data information without decompressing the file, for the BESO data acquisition system, the selection was obvious. Moreover, nowadays most FITS reading libraries do not require decompression. They can deal directly with the Rice-compressed data field (e.g. DS9, MIDAS, saoimage or the astroPy FITS library).

The authors [Pence et al., 2009] also mention that the algorithm is even more efficient on "source-free" data. Those are the typical calibration frames like BIAS/ZERO, DARK, FLAT .... One has to take into account that, especially in spectroscopy, the calibration sets outnumber the target frames by orders of magnitude. Here is an example of a set of 275 BIAS and 88 dark frames (covering temperatures from -30 to -78°C):

```
23G     2025_05_26/HIGH_0100
4.8G    2025_05_26_fz/HIGH_0100
```

To fpack those 363 files (23 GByte ≡ to 5 DVDs) required only 38.61 seconds (0.11 seconds per frame on average) on the besocam computer (funpack required 50.42 seconds) and ended up with a content size of one DVD. The compression rate was 1:4.75 (21.0% size remains)[23]. The data transfer with scp to Innsbruck then needed 54 minutes - about 8.4 seconds per compressed frame - without compression before transfer, it would be about 5 times longer. My test needed even more time. Namely, 291.83 minutes ≡ 5.4 times longer or 45 seconds per frame. However, the latter type of tests varies due to network loads on the site.

Finally, the transfer efficiency of 1000 files via the NFS mount to was tested, while the observatory was under full load of operations (even the jk15 with the other instrument, the Andor XL). During this test, to ensure the data is being physically written to the destination, and not to a kernel cache only, after each individual transfer, the sync is called for the mounted file system. It shows(see Fig. 3), that due to the lower TCP/IP packet collisions, the transfer is even slightly faster than the factor gained by the compression itself. Even taking into account the 0.1 seconds of additional time for compressing the file, the benefits are enormous.

Thus, this mode is picked as the default in the BESO server. This default setting can be overridden by a user/interface command.

[23] To gzip all files needed 1505 (!) seconds and led to a rate of 1:3.26 (30.7% remains - decompressing needed 94.14 seconds). Using bzip2 needed 657.18 seconds and led to a rate of 1:5.02 (19.8% remains - decompressing needed 419.42 seconds). Classical UNIX compress (which is still used by the ESO archive) needed 214.46 seconds and led to 1:3.61 (27.7% remains - decompression needed 57.06 seconds)

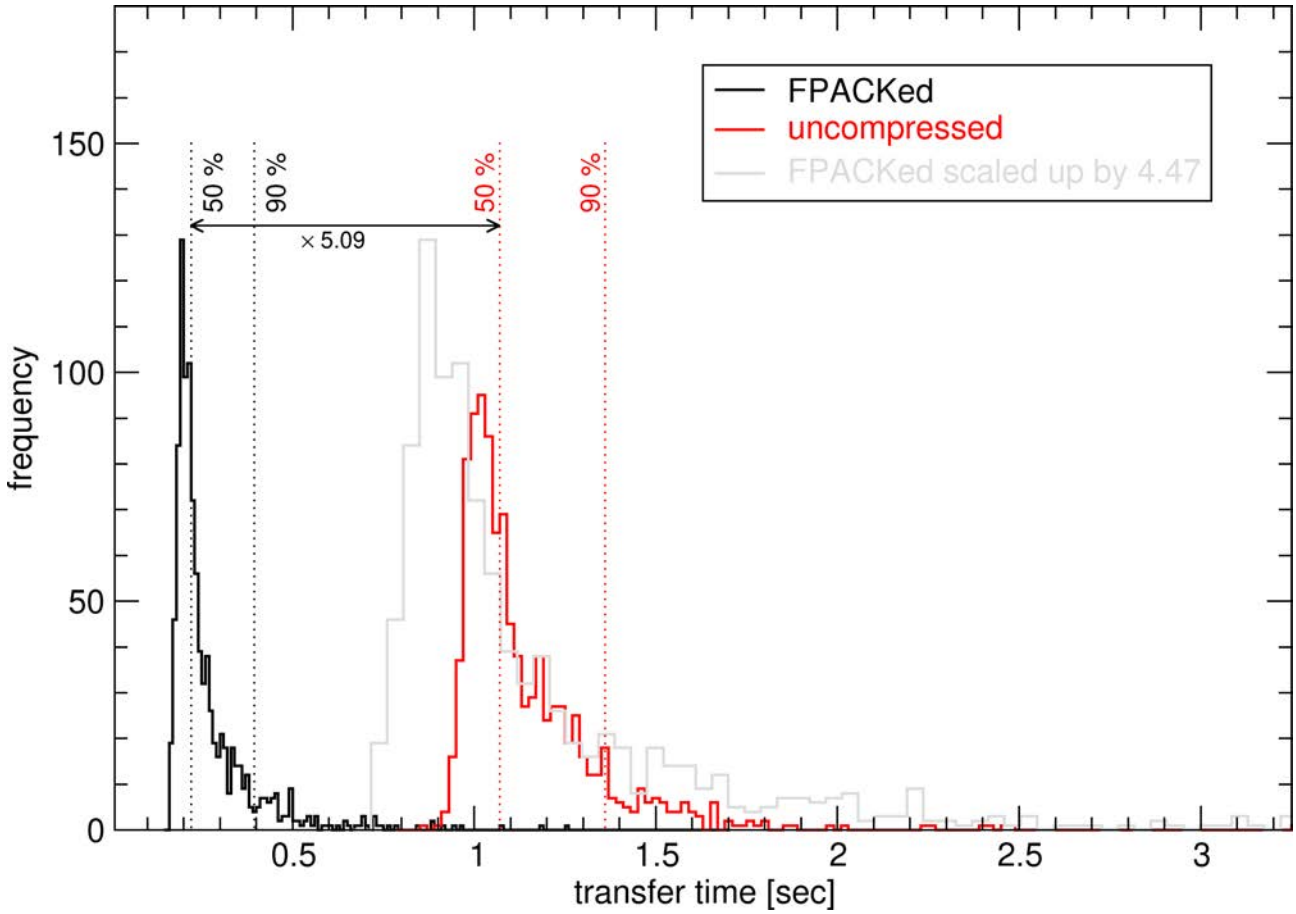


Figure 3: The timing of the `NFS` transfer within `OCM` of 1000 files in decompressed and in `fpack` compressed mode. To compare the distributions in timing, the packed distribution was also plotted, scaled up by the average compression factor of 4.47. The times required to transfer half and 90% of all files are indicated for each mode as well.

Finally, some remarks on float data in FITS images of our data reduction:
After BIAS/ZERO subtraction and FLATfield, our data has a float format. IEEE float has 4 bytes. The format is led by a sign bit. The exponent has 8 bits, ranging thus from -126 to 127. The mantissa then has 24 significant bits to cover the desired range. However, only 23 bits are saved, indeed. The generation of the exponent is always changed to have the highest significant bit (HSB) set to 1. Due to this generation of the exponent, even values in neighboring cells, which do not differ strongly in real images, have very different bit patterns. Thus, compression is more or less impossible. A widely used compression in astrophysics, originally designed for the digitized sky surveys, is the Hcompress. It smooths out blocks of cells changing by less than a predefined noise level. It is thus not lossless. However, it makes decent and accurate photometry of stellar fields still possible, as we showed already 20 years ago [Bacher et al., 2005].
However, one should take a more sophisticated view of the data. A float has an accuracy of 24 bits. This thus is gives an accuracy of $1 : 1 - [(2^{24} - 1)/2^{24}]$ and thus $1 : 5.9\,10^{-8}$. Our data originates from detectors with 16- to 18-bit. Assuming 18 bit we obtain $1 : 3.8\,10^{-6}$ for a single frame. This is thus 66 times deeper. Calibration and averaging/stacking will improve accuracy. However, assuming a Poisson-like improvement as the best possible case, it goes with the square. Thus, we need a stack of at least 4300 frames.
If we now change to an integer number space by rescaling the value v between minimum a and maximum b to a stored integer value s, and assuming an accuracy of 32 bit by

$$s = \frac{(v-a)}{(b-a)} \times (2^{32} - 1)$$

we obtain perfect unsigned integers. CAUTION: as the accuracy of float is less than that of a 32-bit integer, this operation itself has to be executed in *double precision* (thus in Python as a numpy vector with dtype=64 as Python otherwise will select the inaccurate float mode automatically).
Then we store into the FITS header the KEYWORDs

BZERO = a
BSCALE = $(b-a)$

for reconstruction.
Such images have integer values very close together in values of neighboring fields, and thus compression gets very efficient. Even better is if we do not use the whole number space of $[0, (2^{32} - 1)]$, but only a fraction, which is significant for our data accuracy. E.g., if we get, with the estimate above, sufficient accuracy in the float type (or had it in float in an intermediate step anyhow), we use only, for example

$$s = \frac{(v-a)}{(b-a)} \times (2^{24} - 1)$$

but the BSCALE as before. This scaling can be reduced even lower to some factors adjusted to the real accuracy instead. The lower the number, the higher the compression efficiency will get.

Finally, **no FITS generator having real integer data should write BSCALE (even not 1.0) into the header**, as then the libraries (AstroPy as well as CFITSIO) assume that those data should be float when reading, and thus reducing accuracy and doing statistical calculations differently as well. This will cause really erroneous results.

C Coordinate Systems from/to ASA AutoSlew

While the ASCOM + ALPACA interface allows sending back various coordinate systems from a telescope[24] It is not adjustable (according to the standard) by the user/client computer. Which method the telescope driver vendor prefers can be queried by the interface element *equatorialsystem* (https://ascom-standards.org/api/#/Telescope%20Specific%20Methods/get_telescope__device_number__equatorialsystem). This knows, according to the standard, the values 1 = *topocentric*, 2 = *J2000*, 3 = *J2050*, 4 = *B1950*, and finally 0 = *Custom or unknown equinox and/or reference frame*. There is already the first flaw in this standard. IAU has decided, starting from the 2006 standard (currently the standard, but with an addendum to a so-called 2013 motion model), that the J2050 will never be applied, but all catalogs will stay on J2000.

Now the ASA software returns the information 1 = *topocentric*. That is a coordinate system, according to the official definition, including (in this order) Precession, Nutation, Annual parallax, Annual aberration, for a virtual observatory at the position of the Earth's center (called apparent C), and then additionally the correction towards the Earth's surface and the effects of the atmosphere. The latter additional points are diurnal aberration due to the velocity of the observer with the planet's rotation, Diurnal parallax due to the change of the observer's position relative to the center of the Earth, polar motion, and atmospheric refraction.

However, extensive tests clearly show that the JK15, despite a different statement in a mail from ASA Sat, 28 Mar 2026 07:21:40 UTC, does not return in reality the topocentric coordinates, but those of the virtual observer at the Earth's center. Marginal (sub arcsecond) differences are most likely caused by the use of own brewed software which most likely, as they refer to a document used by amateurs dated 2004 [25] does not correct for the IAU standard 2006 and the new motion model 2013 which would be automatically included, if the would compile the software against the IAU library SOFA. These geocentric coordinates are indeed those displayed in TOI and added to the headers as RA_TEL/DEC_TEL.

Here is an observation of a pointing done with jk15 (for one of our photometric targets, namely NGC 4590 center). TOI sends the first coordinate J2000 to the telescope and gets back the 3rd line geocentric apparent in fact.

```
ICRS, epoch J2000.0: 12 39 28.0000000 -26 44 38.600000
    catalog -> CIRS: 12 39 32.2093086 -26 53 26.544474
geocentric apparent: 12 40 53.2471547 -26 53 26.544474
CIRS -> topocentric: 12 39 32.2368094 -26 53 26.774632
   CIRS -> observed: 12 39 30.8026560 -26 53 27.114257
   ICRS -> observed: 12 39 30.8026562 -26 53 27.114260
```

However, the definition of topoentric is the last line. The output comes from the official code given as Chapter 6 in the IAU library cookbook. Test the quality; also, the example from the Wallace paper was run through my code implementation (just a copy from the IAU homepage):

[24] ALPACA API Telescope Specific Methods

[25] Wallace (2004) https://syrte.obspm.fr/iau/iauWGnfa/ExPW04.pdf

```
      ICRS, epoch J2000.0: 23 32 55.1706170 +52 16 38.288892
          catalog -> CIRS: 23 32 55.9204920 +52 17 43.950216
          catalog -> CIRS: 353.2330020500  52.2955417266
      geocentric apparent: 23 33 06.3632389 +52 17 43.950216
      CIRS -> topocentric: 23 32 55.9416624 +52 17 43.926501
         CIRS -> observed: 23 32 55.9221887 +52 17 44.016185
         ICRS -> observed: 23 32 55.9221882 +52 17 44.016185
         ICRS -> observed: 353.2330091176  52.2955600514
ICRS -> observed (AZ/ALT): 116.4234929475  89.7984040078
```

This result does not mean that ASA does not apply the topocentric corrections when calculating Alt/Az. They have to, to obtain a right pointing. However, it means that they do not return this data, but one of the intermediate steps of the conversion to the user at the ALPACA interface.
As the imaging observers run anyhow an astrometric software, that does not make so much of an issue. However, in spectroscopy, that does not work out. Thus, my code currently calculates backwards the coordinates with this finding. However, the code segment, if at a future point, really topocentric coordinates arrive, is already implemented and can be switched with a compiler flag.

D Expert Mode Commands

These commands are kept in the code for internal testing and special storage. They are not intended for regular use on normal user interfaces. They are quasi-prohibited commands. The only effect is internal storage and alternatives to the OCM facility standards. Moreover, they do not affect the storage of the OCM file server. They are only used for some parallel world. The storage of the OCM file server will not be changed by them et al.

SET/GET STORE

This set of settings is to switch various other standards, like the older MIDAS/IRAF way of prefix_nnnn.extension (like e.g. bias_0001.fits) or the ESO standard like e.g. BESO_2025_05_03T19_43_10.613.fits containing the instrument name and the time down to milliseconds. The default can be changed very quickly in the code setup.

... NUMBER (ON/OFF)
use a running number in filename (default is OFF)

... NEXT (number)
setting next running number (default is 1 at start of the server and running during the night until changed by this user command)

... NSIZE (digits)
setting the number of digits for running number (default is 4 digits)

... DATETIME (ON/OFF)
use date/time in filename (default is currently ON)

... PREFIX (file_prefix)
setting filename prefix (default is BESO)

SET/GET STORE INTERNALPATH (path)

Sets/gets the storage path; it has to be an absolute path. The default is currently set to /home/data/YYYY-MM-DD at server start. Later, it will be modified to the NFS mount position.

CHECK STORE INTERNALPATH

Checks that the storage path exists, as the server runs as root It will always be able to write to it. This command is required, as the client may be working over the network and thus does not have access to the local file system, and/or if an NFS mount is active and correctly mounted, to check it for itself.

CREATE STORE INTERNALPATH

If it does not exist, it creates it (including potential subdirectories and the full tree).

SET/GET STORE USER/GROUP (value)

This section is meant to set the final level permissions. The Server runs as root. Thus, the files would be under that permission. Locally, the set is fully implemented. This command only affects the internal and archive storage. It does not touch the OCM file server permissions. There, the user is defined by the server. At the NFS mount, the permissions of the root would convert by default to user and group nobody. Thus, OCM has defined poweruser/powergroup for that, independent of the setting of this command here.

... USER (username:to_store_with)
setting user for the UNIX/LiNUX© chown command (default is data)

... GROUP (group:to_store_with)
setting group for UNIX/LiNUX© chgrp command (default is data)

SET/GET STORE ARCHIVE (ON/OFF)

selects whether the files are written to the internal archive. Default is ON. In fact, this is a kind of test feature and normally will never be OFF in regular operations.

References

O. A. Alduchov and R. E. Eskridge. Improved Magnus Form Approximation of Saturation Vapor Pressure. *Journal of Applied Meteorology*, 35(4):601–609, Apr. 1996. doi: 10.1175/1520-0450(1996)035<0601:IMFAOS>2.0.CO;2.

A. Bacher, S. Kimeswenger, and P. Teutsch. Photometry from online Digitized Sky Survey plates. *MNRAS*, 362(2):542–548, Sept. 2005. doi: 10.1111/j.1365-2966.2005.09329.x.

N. Epchtein, B. de Batz, L. Capoani, L. Chevallier, E. Copet, P. Fouqué, P. Lacombe, T. Le Bertre, S. Pau, D. Rouan, S. Ruphy, G. Simon, D. Tiphène, W. B. Burton, E. Bertin, E. Deul, H. Habing, J. Borsenberger, M. Dennefeld, F. Guglielmo, C. Loup, G. Mamon, Y. Ng, A. Omont, L. Provost, J. C. Renault, F. Tanguy, S. Kimeswenger, C. Kienel, F. Garzon, P. Persi, M. Ferrari-Toniolo, A. Robin, G. Paturel, I. Vauglin, T. Forveille, X. Delfosse, J. Hron, M. Schultheis, I. Appenzeller, S. Wagner, L. Balazs, A. Holl, J. Lépine, P. Boscolo, E. Picazzio, P. A. Duc, and M. O. Mennessier. The deep near-infrared southern sky survey (DENIS). *The Messenger*, 87:27–34, Mar. 1997.

S. Kimeswenger. *Technical Note*: BESO Spectrograph Temperature Controller, 2025.

W. D. Pence, R. Seaman, and R. L. White. Lossless Astronomical Image Compression and the Effects of Noise. *PASP*, 121(878):414, Apr. 2009. doi: 10.1086/599023.

E Document & Software History

The main version number of the software (see the FITS header entry SW_VERS) also always corresponds to the version of this document. Thus, some changes are not causing changes in the software, but only in the document. The others mentioned here might be bug fixes in the software without changes in the document.

Vers.	Date		submitted to oca-tech @camk.edu.pl
2.00	26th Sep. 2025	Initial version after the original draft stage. Created after a VidCon discussion about final policies on Thursday, 21.8.2025 15:00 CEST	2025-09-26
2.01	29th Sep. 2025	minor correction and adding SW_IAU to FITS header	*not submitted*
2.10	6th Oct. 2025	Corrected EXPTIME/EXPOTIME to seconds. Corrected typos in document. Corrected errors in archive user handling, causing the wrong user after the fpack command. Corrected reply of some erroneous GET commands. Added special handling for the single quote character in FITS strings and blocked other illegal characters. Implemented RESTART CAMERA Implemented RESTART PC Implemented GET/SET TIME Implemented GET TEMPERATURE PROBES Implemented UPTIME SERVER Implemented UPTIME CAMERA Added parameter handling to START EXPOSURE	2025-10-09
2.11	11th Oct. 2025	added GET FITS KEY command	*not submitted*
2.12	17th Nov. 2025	Implemented and added support for new shutter hardware and thus SET SHUTTER H0 and H1 commands Added the previously forgotten documentation for the already existing and working GET/SET CALIB commands.	*not submitted*
2.20	19th Nov. 2025	Implemented and added documentation for user space start, stop, and reboot operations - whole section 6. Changed the document structure to create a fast hop to the command groups from the table of contents. Text and language style polishing. Implemented the SET CALIB M4_TOGGLE command. Bug fix that allowed starting a second exposure while one was still pending (in NOWAIT mode).	2025-11-19

Vers.	Date	*(continued form previous page)*	submitted to oca-tech @camk.edu.pl
2.21	20th Nov. 2025	Some tests of SET commands during exposures (in NOWAIT state) led to problems. Thus, after looking through the list of commands and identifying no need for any of the SET commands during exposures, they are now all blocked as long as an exposure state is pending (unread). Added for security operations like deactivating ThAr lamp, FLAT lamp, and shutter hardware, when using server_kill@jk15-beso and server_reboot@jk15-beso.	*not submitted*
2.22	22th Nov. 2025	Reconfiguring internal FITS handling in code to make future extensions easier. Extending this document for clarification. Added pointing coordinates in user readable form as RA_PNT/DEC_PNT (see mail on oca-tech by Mikolaj 10/12/25, 14:09).	*not submitted*
2.23	29th Nov. 2025	full implementation of the DAVIES meteo station data.	*not submitted*
2.24	26th March 2026	Adding precession corrections from IAU - as discovered error in ASA AutoSlew, with those two new FITS header entries created to give the real (correct) pointing position.	*not submitted*
2.25	6th April 2026	Some minor bugfixes in handling the optional parameters in START EXPOSURE. Modified the client to handle lower-case characters in options fields for FITS strings properly.	*not submitted*
2.30	4th April 2026	Release of new video tutorials. Added a detailed description of the coordinate system and the improper handling by ASA. Added full example headers (both with FPACK and without) Changed and extended commands for restart options.	2026-04-07
2.31	29th April 2026	Some minor bugfixes in handling the optional parameters RA= and RA_OBJ= in START EXPOSURE.	*not submitted*
2.40	25th May 2026	Renaming in this document and in the commands M4 to M5 as in fact the real 4th mirror is a fixed one. Removing automatic M5 handling in SET CALIB FLAT/THAR ON/OFF and introducing instead a SET CALIB M5 ON/OFF. Renaming SET CALIB M4_TOGGLE to SET CALIB TOGGLEM5	2026-05-25
2.50	14th August 2026	Adding statistical values to FITS header and user output Fixing some minor bugs in the calculation of the airmass at the end of the exposure. Added complete shutdown of the system in that part controllable by software for fast (emergency) case with shutdown_beso@jk15-beso	*not submitted*

Vers.	Date	*(continued form previous page)*	submitted to oca-tech @camk.edu.pl & Paulina
		Added section about the BESO room temperature control	
3.00	15th August 2026	Renamed document to `BESO Manual` as it already contains much more than only the command line interfaces, but also many other user instructions.	2026-08-15