The control system of the MAGIC Telescope
MAGIC-TDAS 00-07
Version 11.3

J. Cortina - Central Control, General Coordination
G. Giavitto - Central Control
T. Toyama - Camera Control
E. Carmona - M1 Data Acquisition
D. Tescaro, J. Aleksic - Data Acquisition
S. Covino - Burst Alarm
R. Paoletti - M1 trigger, sumtrigger control
Th. Bretz - Telescope Drives, Star guiders
D. Mazin - Readout control, trigger, calibration, L3 trigger
A. Biland - Active Mirror Controls
J. Hose - Pyrometer, lidar
Ll. Font - Weather Station

August 7, 2012

Contents

1 Introduction **11**
1.1 Design characteristics 11
1.2 Distributed development 13

I The subsystems **15**

2 MAGIC-1 and MAGIC-2 Data Acquisition **19**
2.1 General tasks 19
2.1.1 Startup/Selftest 20
2.1.2 Configuration 20
2.1.3 Taking data 20
2.1.4 DAQ monitor 21
2.1.5 Online analysis 21

3 MAGIC-2 Data Acquisition **23**
3.1 General tasks 23
3.2 Controlled Parameters 23
3.3 States and State Transitions 23
3.4 Implementation 23

4 Camera oscillation monitors **25**

5 MAGIC-1 and MAGIC-2 Telescope drive and star guider **27**
5.1 General tasks 27
5.1.1 Startup/Self-Test 27
5.1.2 Coordinate transformations 27
5.1.3 Positioning 28
5.1.4 Positioning error monitoring 28
5.1.5 Interaction with the tracking monitor 29
5.1.6 Bending correction 29
5.1.7 Safety cross-checks 29
5.1.8 Stopping 29

5.1.9 Shutdown 30
5.1.10 Parameter data base 30
5.1.11 Interaction with CC 30
5.2 Controlled Parameters 30
5.3 States and State Transitions 30
5.4 Implementation 30

6 Pyrometer **31**
6.1 General tasks 31
6.1.1 Startup/Shutdown 31

7 MAGIC-1 Camera and Calibration Control **33**
7.0.2 Power supplies 33
7.0.3 Setting/reading of parameters in the camera 34
7.0.4 Motors 34
7.1 General tasks 34
7.1.1 Startup/Selftest 35
7.1.2 Configuration 35
7.1.3 Normal operation 35
7.1.4 Shutdown 35
7.1.5 Interaction with CC 35
7.2 Controlled Parameters 35
7.3 States and State Transitions 35
7.4 Implementation 36
7.4.1 Communication with the calibration system 36

8 MAGIC-2 camera control **37**
8.1 General tasks 37
8.2 Controlled Parameters 37
8.3 States and State Transitions 37
8.4 Implementation 37

9 MAGIC-2 calibration control **39**
9.1 General tasks 39
9.2 Controlled Parameters 39
9.3 States and State Transitions 39
9.4 Implementation 39

10 MAGIC-1 and MAGIC-2 active mirror controls **41**
10.1 General tasks 41
10.1.1 Startup/Selftest 41
10.1.2 Mirror adjustment 42
10.1.3 Mirror heating 42

10.1.4 Shutdown 42
10.1.5 Interaction with CC 43
10.2 Controlled Parameters 43
10.3 States and State Transitions 43
10.4 Implementation 43

11 Lidar **45**

12 M1 Level 2 trigger, i.e. prescaler **47**
12.1 General tasks 47
12.1.1 Startup/Selftest 47
12.1.2 Trigger Tables 47
12.1.3 Trigger Rates Monitor 48
12.1.4 Stopping 48
12.1.5 Shutdown 48
12.1.6 Interaction with CC 48
12.2 Controlled Parameters 49
12.3 States and State Transitions 49
12.3.1 State “Error” 49
12.3.2 State “Idle” 49
12.3.3 State “Loading” 49
12.3.4 State “Ready” 49
12.3.5 State “Active” 49
12.3.6 State “Stopped” 49
12.4 Implementation 50

13 M2 trigger: prescaler **51**
13.1 General tasks 51
13.2 Controlled Parameters 51
13.3 States and State Transitions 51
13.4 Implementation 51

14 Level 3 or stereo trigger **53**
14.1 General tasks 53
14.2 Controlled Parameters 53
14.3 States and State Transitions 53
14.4 Implementation 53

15 Burst Alarm **55**
15.1 General Tasks 55
15.1.1 Startup/Selftest 55
15.1.2 Normal operations 56
15.1.3 Alert 57

15.2 Controlled Parameters 57
15.3 States and State Transitions 57
15.3.1 GCN status 57
15.3.2 CC *client* status 58
15.3.3 CC *server* status 58
15.3.4 Alarm status 58
15.4 Implementation 59

16 Auxiliary systems **61**
16.1 General tasks 61
16.1.1 Startup/Selftest 61
16.1.2 Weather 61
16.1.3 NSB 62
16.1.4 Smoke sensors 62
16.2 Controlled Parameters 62
16.3 States and State Transitions 62
16.4 Implementation 62

17 Central Control **63**
17.1 General tasks 63
17.1.1 Startup/Selftest 63
17.1.2 Communication with the other subsystems 63
17.2 Auxiliary elements 63
17.3 Controlled Parameters 64
17.4 Implementation 69
17.4.1 The main loop event 69

II Communication protocols **71**

18 Generic subsystem-CC interaction **73**
18.1 Guidelines regarding interaction with CC 73
18.1.1 Connection/Control commands 74
18.1.2 Data from subsystems to be kept: send to CC 74
18.2 Implementation 74
18.2.1 Communication flows 74

19 MAGIC-1 and MAGIC-2 DAQ - CC **77**
19.1 Commands to be received from CC 77
19.1.1 Commands concerning the DAQ 77
19.2 Data sent to CC 80
19.2.1 Data concerning the DAQ 80
19.2.2 Data concerning the communication 81

19.2.3 Format definition 81
19.2.4 Special Report 82
19.2.5 Run Reports stored in .rep file 84

20 Camera oscillation monitor - CC **87**

21 MAGIC-1 Telescope drive, star guider, TPoints - CC **89**
21.1 Commands to be received from CC 89
21.1.1 Commands concerning the drive 89
21.2 Data sent to CC 90
21.2.1 Data concerning the drive 91
21.2.2 Data concerning the communication 91
21.2.3 Drive format definition 91
21.2.4 Special Report 92
21.2.5 Data concerning the star guider 92
21.2.6 Data concerning the communication 93
21.2.7 Star guider format definition 93

22 MAGIC-2 Telescope drive, star guider, TPoints - CC **95**
22.1 Commands to be received from CC 95
22.1.1 Commands concerning the drive 95
22.2 Data sent to CC 96
22.2.1 Data concerning the drive 97
22.2.2 Data concerning the drive communication 97
22.2.3 Drive format definition 97
22.2.4 Data concerning the star guider 98
22.2.5 Data concerning the star guider communication 98
22.2.6 Star guider format definition 99
22.2.7 Special TPoint report 99

23 Pyrometer - CC **101**
23.1 Data sent to CC 101
23.1.1 Commands concerning the pyrometer 101
23.1.2 Commands concerning the communication 101
23.1.3 Pyrometer format definition 102

24 MAGIC-1 and MAGIC-2 Camera Control - CC **103**
24.1 Commands to be received from CC 103
24.2 Data sent to CC 104
24.2.1 Data concerning the communication 104
24.2.2 Format definition 105

25 MIR (Level 3 trigger and MAGIC-2 readout, trigger and calibration control)- CC 107
25.1 Commands to be received from CC . . . 107
25.2 Data sent to CC . . . 107
25.2.1 Data concerning the MAGIC-2 trigger communication . . . 107
25.2.2 MAGIC-2 trigger format definition . . . 108
25.2.3 Data concerning the MAGIC-2 pulsar configuration . . . 108
25.2.4 Data concerning the MAGIC-2 pulsar configuration communication . . . 109
25.2.5 MAGIC-2 pulsar configuration format definition . . . 109
25.2.6 Data concerning the MAGIC-2 calibration . . . 109
25.2.7 Data concerning the MAGIC-2 calibration communication 110
25.2.8 MAGIC-2 calibration format definition . . . 110
25.2.9 Data concerning the MAGIC-2 receivers . . . 110
25.2.10 Data concerning the MAGIC-2 receivers communication . 110
25.2.11 MAGIC-2 receivers format definition . . . 111
25.2.12 Data concerning the MAGIC-2 readout cooling . . . 111
25.2.13 Data concerning the MAGIC-2 readout cooling communication . . . 111
25.2.14 MAGIC-2 readout cooling format definition . . . 112
25.2.15 Data concerning the L3 trigger cooling . . . 112
25.2.16 Data concerning the Level 3 trigger communication . . . 112
25.2.17 Level 3 trigger format definition . . . 112

26 MAGIC-1 and MAGIC-2 active mirror controls - CC 115
26.1 Commands to be received from CC . . . 115
26.1.1 Commands concerning the Mirror control . . . 115
26.2 Data sent to CC . . . 117
26.2.1 Data concerning the communication . . . 117
26.2.2 Format definition . . . 117

27 Lidar - CC 119
27.1 Commands to be received from CC . . . 119
27.2 Data sent to CC . . . 119
27.2.1 Data concerning the communication . . . 120
27.2.2 Format definition . . . 120
27.2.3 Special Report . . . 120

28 MAGIC-1 trigger - CC 123
28.1 Commands to be received from CC . . . 123
28.1.1 Commands concerning the Trigger control . . . 123
28.1.2 Commands concerning the Trigger monitor . . . 124
28.2 Data sent to CC . . . 124

28.2.1 Data concerning the subsystem status 124
28.2.2 Data concerning the communication 124
28.2.3 Data concerning the trigger 125
28.2.4 Format definition 125

29 Burst Alarm - CC **127**
29.1 Data received from the CC 127
29.2 Data sent to the CC 127
29.2.1 Normal reports 127
29.2.2 Special alarm reports 128

30 Auxiliary systems - CC **131**
30.1 Commands to be received from CC 131
30.1.1 Commands concerning the Auxiliary systems 131
30.2 Data sent to CC 131
30.2.1 Data concerning the communication 131
30.2.2 Format definition 132
30.2.3 Special Report 132

31 Control output data format **133**
31.1 Main control, report or .rep file 133
31.2 CC status report 134
31.3 Electronic runbook or .rbk file 135
31.4 Run summary or .run file 135
31.5 Run summary HTML or .run.html file 137
31.6 DC current control file 137

32 Computer names and TCP/IP port assigments **139**

Chapter 1

Introduction

1.1 Design characteristics

The control system of the MAGIC Telescope is shown in a rough outline in figure 1.1. The main characteristics are:

Distributed design: The control system is split up into functional units which correspond to the independent subsystems of the telescope. In addition there is a central control (**CC**) computer which coordinates the subsystems and provides the user interface during normal observations. If all subsystems work well, the user only has to communicate with CC.

For maintenance, testing and development each subsystem can run in standalone mode without communicating with CC or other subsystems.

Communication via Ethernet using TCP/IP sockets and/or file sharing: All subsystems and CC communicate with each other via standard TCP/IP socket connections or NFS file sharing in a standard Ethernet network.

Central Ethernet Switch: The Ethernet architecture is simple hub-spokes design, i.e. there is a single Ethernet Switch to which all subsystems are connected.

Time synchronization: Synchronous clocks in the computers are instrumental to maintain the communication. All computers in the control system are synchronized to UTC within ∼100 ms using the NTP protocol implemented through the *xntpd* Linux daemon. They rely on the standard time server of the Spanish Rediris network (currently *hora.rediris.es*) and on a local time server in the auxiliary systems computer (*wwwint.magic.iac.es*). The computer clock in the auxiliary systems computer is synchronized to UTC directly through a GPS receiver and works as NTP time server. This local time server assures time synchronization when the connection to the IAC network is down.

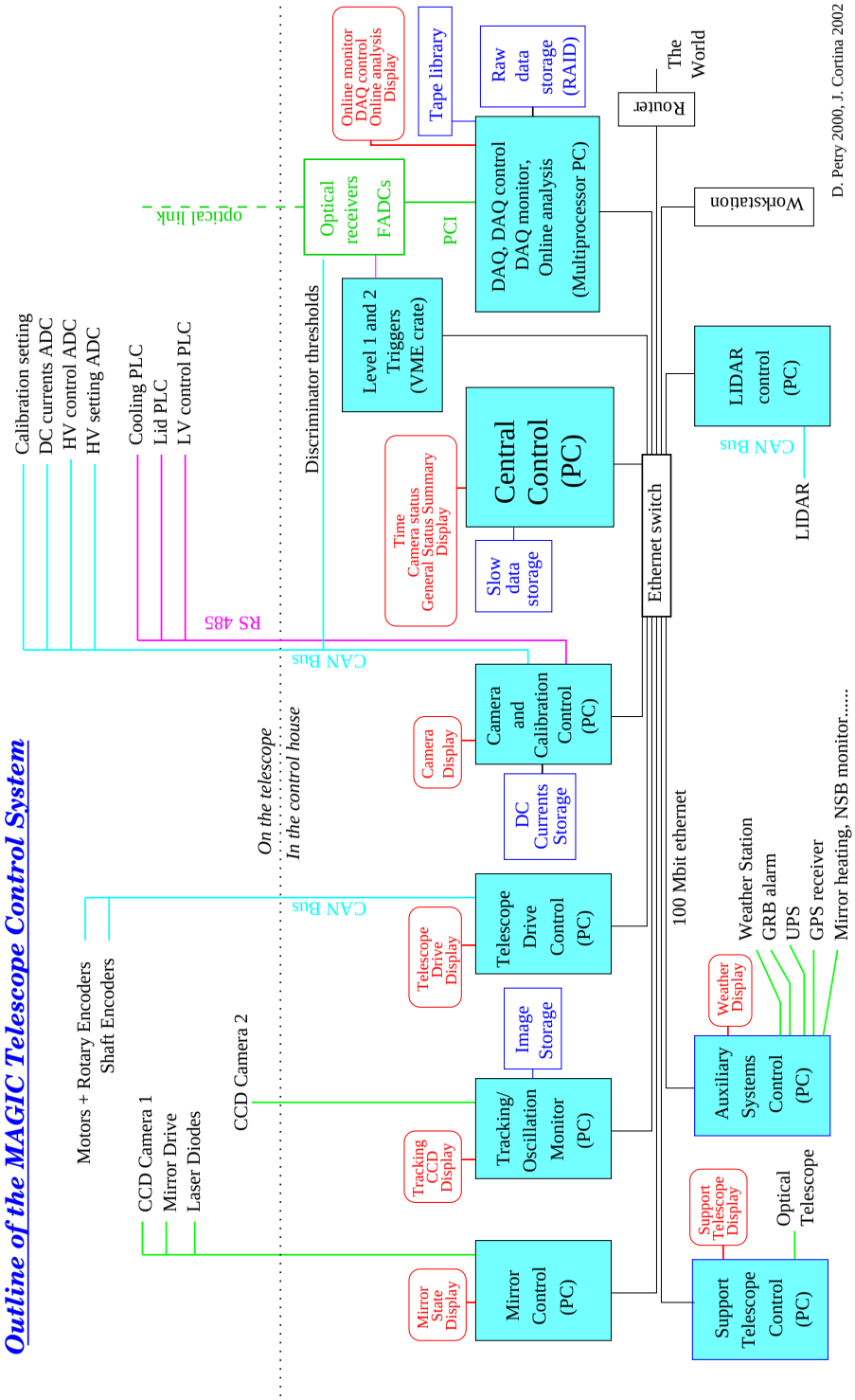

Figure 1.1: Outline of the MAGIC telescope's control and DAQ system.

Communication via CAN where possible: The "slow control" of the telescope's camera makes extensive of use the CANbus protocol. CAN is **strongly** encouraged as our standard for control subsystems. Even auxiliary instruments normally controlled under RS-232 or RS-485 should be linked to CAN over corresponding adapters.

Electromagnetical decoupling: All communication with systems outside the control room/house will use optical coupling in order to keep the control/daq computers safe from interference picked up by long (160 m) cable connections with the telescope. Where possible, optical fibre connections are used. The PCI CAN controller card used for camera control is optically decoupled.

1.2 Distributed development

One of the main ideas about the control system is the "distributed development". This means each group develops their subsystem such that

1. the system can run stand-alone without any CC
2. there is an interface for an outside system to connect to the subsystem and set the major parameters and receive status reports.

What is exactly meant by the second point, depends on the subsystem and will be described in later chapters of this text.

The effect of the distributed development is two-fold. On the one hand, the developers can easily share their work since they only have to agree on a relatively simple communication protocol with CC. This encapsulation enables them to fully test their system without having to buy additional hardware or having to transport their system to a different site. When the subsystem works reliably in stand-alone mode, it can within a relatively short period of time be integrated with the CC and the other systems.

On the other hand, the encapsulation of the subsystems leads to easier maintenance of the total system once it is implemented at the telescope site. Errors can be located more easily and upgrades of individual hardware or software elements influence only individual subsystems.

Part I

The subsystems

This part describes the tasks and implementation of the control of the individual subsystems of the telescope.

Chapter 2

MAGIC-1 and MAGIC-2 Data Acquisition

This system controls:

1. DAQ (the actual FADC readout)
2. communication with the trigger system
3. DAQ Monitor (DAQ status display etc.)
4. Online Analysis (preliminary determination of the gamma signal)
5. ROOTifier and Storage (saves data in final ROOT raw data format)

2.1 General tasks

The DAQ consists of several elements:

1. The DAQ control
2. The FADC readout
3. The event builder (ROOTifier) and data storage
4. The DAQ monitor
5. The online analysis

The DAQ control is responsible for coordinating and setting up all parts of the DAQ and for communicating with the user/CC.

2.1.1 Startup/Selftest

DAQ control should keep the setup of the DAQ system in a setup file (ASCII format) which is read at startup. The system is then brought into the state described by the setup file, i.e. ready to start taking data.

The system should test the availability of all DAQ elements.

A log file in ASCII format keeps information on all remarkable incidents.

2.1.2 Configuration

A comfortable user interface for configuring all elements of the DAQ is provided with the possibility to store the setups in files. The parts of the setup which are interesting for the MC simulation of the observation are stored with the events data.

2.1.3 Taking data

The user/CC starts data taking by telling DAQ control to do so. The DAQ system does not perform any checks on the telescope's position etc but only reads out triggered events, extracts the FADC information, builds events, feeds them to online analysis and stores them.

Data taking is split in *data runs*. A data run stops when a given number of events have been taken. This number of events will be defined so that the data run files are easy to handle. CC has no control over run start/stop and can only tell the DAQ to start/stop a *run sequence*. Hence run numbering is fully under the control of the DAQ. A summary of the run which contains run time, number of events etc (to be defined) is stored and sent to CC as a special report everytime a run ends.

A run number is defined[5] through the noon date and a 5 digit number starting in 0 at the beginning of each night, e.g., 2002102500001 stands for the first run in the night of 25 October 2002.

Appropriate statistical information about the trigger rates at various levels and the available disk space should be displayed for the user on the screen.

When CC tells the DAQ to stop a run sequence (using the *STOP DAQ* command) the DAQ goes to *daqtest mode*. This happens regularly when the operator wants to move to another source, he/she is changing the high voltage, is not ready for data taking, is testing the system... In *daqtest mode*, data taking is inmediately started without storage, i.e. the DAQ reads the events, statistical information is produced normally, the events are fed to online analysis and reports are sent to CC normally but final storing is skipped.

CC can issue a command to finish data taking after the current run (*LAST RUN* command). The DAQ waits for the specified number of events to finish, closes the data file and goes to *daqtest*.

Alternatively CC may start a run sequence defined by a maximum number of events. The DAQ takes as many runs as necessary to achieve this number of events and stops. This operation mode is useful for calibration and test runs.

Calibration or pedestal events will probably be concurrently with cosmic events with a frequency set by the calibration control, but special calibration/pedestal runs are also possible.

In calibration runs the calibration system steered by CaCo generates triggers that are sent directly to L2T. L2T is instructed to accept only calibration triggers. The DAQ is told to start a run defined by a given number of events. At the end of a calibration run, the DAQ appends the average response for each pixel to its special report to CC.

In pedestal runs the calibration system generates random triggers and feeds them to L2T, which is programmed to accept only this kind of triggers. At the end of the run, the DAQ appends the average response for each pixel to its special report to CC.

If the DAQ runs low on diskspace (only space for 20 minutes of data at the rate measured over the last minute is left) it should give a warning to the user/CC and start an acoustical signal which continues until the run is stopped or more diskspace becomes available. If it runs out of diskspace, it should switch to *standby*.

2.1.4 DAQ monitor

This subsystem has not been implemented. It is partly taken over by the general DAQ program.

2.1.5 Online analysis

The online analysis (OA) is a MARS executable which performs the alpha plot during data-taking. Its rate, about 300 Hz, is larger than the acquisition rate on average.
This executable has to be launched in "muxdaq", the pc where data are written by DAQ. Therefore, when the observation of a source begins, Arehucas sends to DAQ the command to start the online analysis. At that moment, DAQ launches the OA program through a script.
The online analysis stops when the data of the analyzed source have all been processed. Anyway shifters can decide to start and stop the OA manually.

The online analysis has been installed starting from Arehucas version 070926-0.

Chapter 3

MAGIC-2 Data Acquisition

3.1 General tasks

(to be written by the subsystem developers)

3.2 Controlled Parameters

(to be written by the subsystem developers)

3.3 States and State Transitions

(to be written by the subsystem developers)

3.4 Implementation

(to be written by the subsystem developers)

Chapter 4

Camera oscillation monitors

This subsystem has not been implemented.

Chapter 5

MAGIC-1 and MAGIC-2 Telescope drive and star guider

Contrary to the original design, the telescope drive and the star guider are implemented as one single program. They communicate however through different reports and they are still considered as different susbsystems.

5.1 General tasks

5.1.1 Startup/Self-Test

The shaft encoders and position sensors have to be powered up and tested. For the shaft encoders, the zero positions have to be set.

The availability and proper functioning of the drive motors has to be tested.

The interlock with the fence around the telescope has to be verified.

5.1.2 Coordinate transformations

The system works in three coordinate systems:

1. Shaft encoder (SE) positions
2. local coordinates (ALT/AZ) where we choose AZ = 0 to be in the North and AZ = 90° to be in the East.
3. the celestial coordinates (RA/DEC)

All positions (both targets and the actual position of the telescope) and the position errors (e.g. the tracking error) should be displayed permanently in all three coordinate systems.

5.1.3 Positioning

The *positioning error* is defined by the two components of the angular distance between the target position and the actual position of the telescope.

Positioning the telescope means to move it such that the positioning error is minimized as quickly as possible.

The target position can be given by the user in one of the three coordinate systems:

1. If the target position is given in the SE or the ALT/AZ system, the telescope's target position in local coordinates is not time dependent, i.e. once the target position is reached, the telescope will not have to move any more in the ideal case. Forces like wind or uneven balancing of the support, however, can still move it. The positioning error in this case can be used to test the presence of and the reaction of the telescope to such forces.

2. If the target position is given in RA/DEC coordinates (the standard case) the telescopes corresponding local coordinates are time dependent and the telescope has to be moved permanently in order to minimize the positioning error. This mode is called *tracking*. Note that the user provides RA/DEC for a given epoch that is NEVER epoch-of-date. The drive system is responsible for converting epoch into epoch-of-date. This is done so to centralize the place where epoch-of-date coordinates are calculated: we want to avoid that both CC and Drive have to do the conversion.

Before a target position is accepted it has to be checked as to whether it is a possible position of the telescope at the approximate time the telescope will reach it (since the telescope has very fast slewing, this means essentially the point of time when the positioning is enabled).

Possible positions are defined by the end positions of the drive and the parameter ranges of the coordinate systems.

The user should be offered the choice whether during the positioning the tracking error (see below, interaction with the tracking monitor) should be used instead of the positioning error.

In both cases, the positioning error should be continuously monitored.

5.1.4 Positioning error monitoring

A permanent monitoring and statistical evaluation of the deviation of the actual and the target position of the telescope should take place with a graphical display of the positioning error over the last five minutes (if possible the time window should be variable).

The user should be offered the choice to write a logfile which contains the positioning error in regular time intervals such that the behaviour can be analysed off-line.

5.1.5 Interaction with the tracking monitor

The tracking monitor will deliver an estimation of the *tracking error*, i.e. the two components of the angular distance between the true position of the target and the true position of the optical axis of the telescope (in ALT/AZ coordinates).

The user should be offered the choice whether during the positioning the tracking error or the positioning error (see above) should be minimized. In both cases, the positioning error should be continuously monitored.

If the tracking monitor cannot provide a tracking error measurement (i.e. it sends UNDETERMINED or STOPPED), the positioning error is used instead and a message to the user is issued.

For determining the guide star position at zero positioning error, the drive system has to synchronize itself with the tracking monitor and send it a trigger signal via a socket connection when it measures a zero positioning error.

It should be investigated whether the parameters of the bending correction (see next section) can be derived from the comparison of tracking and positioning error in an automatic way. If so, this process should be implemented as a special routine which can be started by the user on demand.

5.1.6 Bending correction

The positioning should be done taking into account the deformation of the telescopes support by using a mathematical model of the bending behaviour. This *bending model* should be parameterized and the parameters adjustable through a user interface.

5.1.7 Safety cross-checks

Before moving to a new position, the drive system should check whether the moon or the sun (when above the local horizon) will come near the field of view. In case this is so, the system should refuse to execute the positioning and give an error message to the user.

There should be a possibility of an override of this lock, but it should be made sure that the user is aware of the danger. Possibly, the user should be asked for a password.

5.1.8 Stopping

In order to react to emergencies, it should be possible to stop the telescope's movement as fast as technically possible.

5.1.9 Shutdown

When asked to do so, the system should go through an orderly shutdown sequence trying to make sure that the drive is left in a state such that the telescope will not be damaged, e.g. by sunlight, when left in that state for a longer time.

5.1.10 Parameter data base

All parameters of the drive system should be stored in one database (ASCII file) which can be accessed by the user and is read at startup

5.1.11 Interaction with CC

The system can be in two major modes: stand-alone and CC.

After startup the system should try to contact CC. If CC is available (CC reports are received regularly), and its global state is not "error", the system should go to CC mode. In this mode the system accepts commands from CC. The definition of the interface with CC can be found in section 19.

The system goes to stand-alone mode if CC is no longer available (no reports are received). CC is in error state, or the operator sets it accordingly on the system control panel. In this mode the system does not accept CC commands, but tries to send reports normally.

5.2 Controlled Parameters

(to be written by the subsystem developers)

5.3 States and State Transitions

(to be written by the subsystem developers)

5.4 Implementation

(to be written by the subsystem developers)

Chapter 6

Pyrometer

6.1 General tasks

The pyrometer is a system installed on the left side of the MAGIC mirror surface. It measures the sky temperature [K] defined as the integral thermal flux of the infrared radiation between 8-14 micro-wavelengths.
The pyrometer has Germanium lenses with a FOV of 2 degrees and an optical axis aligned to the MAGIC optical axis. These lenses focus the radiation on a thermoelectric sensor which measure the radiation temperature. The integral flux is then calculated according to the Planck's law under the hypothesis of a perfect black body (emission coefficient = 1).
The sky temperature is also used to evaluate the cloudiness which gives a quick estimation of the sky quality. It is defined as:

$$\frac{T_{meas} - T_{low}}{T_{up} - T_{low}} \tag{6.1}$$

where T_{meas} is the measured sky temperature and $T_{up} = 200K$, $T_{low} = 250K$ two values which indicate the best and the worst weather conditions respectively.
The pyrometer has another small sensor which manages to measure the air temperature and the humidity. Thus, the correlation between the sky temperature and these parameters can be studied offline.

The sky temperature, the air temperature, the cloudiness and the humidity are updated every 10 seconds, displayed on the pyrometer monitor and sent to the CC.

6.1.1 Startup/Shutdown

The pyrometer control software is always running on pc5 and every 10 seconds it receives from the CC a report including the information about the telescope

position. The pyrometer data are taken when the lid is opened: it happens only with the telescope NOT in the park position.

Chapter 7

MAGIC-1 Camera and Calibration Control

This system controls the majority of the elements in the telescope camera. Most of the control is implemented using CANbus. We shall refer to the camera and calibration control as *CaCo*.

7.0.2 Power supplies

The following power supplies have to be controlled in the camera:

1. HV power supplies
2. Power supply for pixel electronics (preamps, current monitors)
3. Power supply for optical transmitters
4. Power supply for calibration instrumentation in the camera lid
5. Power supply for temperature sensors
6. Power supply for lid movement
7. Power for cooling device
8. Oscillation monitor LEDs power on/off
9. Active Mirror Control LEDs power on/off
10. Power supply for central pixel.
11. Voltages and currents of the active load and 175V power supply to power the PMT amplification stages.

In addition CaCo is responsible for setting and monitoring other power supplies:

1. FADC power supplies.

All these items should be switchable separately, but there should also be the possibility to switch them on all at once with only one command (CaCo must internally perform a controlled switch-on in order not to generate current peaks!).

7.0.3 Setting/reading of parameters in the camera

The following parameters have to be set or read out:

1. HV: The high voltage values for 600 channels (in present design only 577 are needed, keep rest as spare) should be settable and readable.

2. Currents: The DC currents of 600 channels (as above) should be read out at a rate of 10 Hz with an accuracy of 2 Bytes. This corresponds to a data rate of 12 kB/s = 96 kbit/s.

3. Temperature sensors: Two (2) temperature sensors should be read out at a rate of 1 Hz.

4. Settings of the cooling device

5. Settings of the optical transmitters (?)

6. Lid position sensors

   Sensor for the state of the camera lid: open / closed. Depending on the construction, up to 4 sensors are needed to ensure that the lid state is correctly determined.

7.0.4 Motors

1. Lid drive: Depending on the nature of the lid movement mechanism, different signals are necessary.

7.1 General tasks

CaCo monitors a miscellanea of elements in the telescope camera. It also allows to set some of its parameters.

7.1.1 Startup/Selftest

CaCo keeps the setup of the system in a setup file (ASCII format) which is read at startup. The system is then brought into the state described by the setup file, i.e. ready to start taking data.

The system should test the availability of all control elements.

7.1.2 Configuration

A comfortable user interface for configuring all elements of the control should be provided with the possibility to store the setups in files. The parts of the setup which are necessary for further data analysis are stored in a data base.

7.1.3 Normal operation

(to be written by the subsystem developers)

7.1.4 Shutdown

At shutdown, it should be made sure that all volatile data is stored such that the system can be restored to the same status at the next startup if the user wishes to do so.

7.1.5 Interaction with CC

The system can be in two major modes: stand-alone and CC.

After startup the system should try to contact CC. If CC is available (CC reports are received regularly), and its global state is not "error", the system should go to CC mode. In this mode the system accepts commands from CC. The definition of the interface with CC can be found in section 19.

The system goes to stand-alone mode if CC is no longer available (no reports are received). CC is in error state, or the operator sets it accordingly on the system control panel. In this mode the system does not accept CC commands, but tries to send reports normally.

7.2 Controlled Parameters

(to be written by the subsystem developers)

7.3 States and State Transitions

(to be written by the subsystem developers)

7.4 Implementation

(to be written by the subsystem developers)

7.4.1 Communication with the calibration system

CaCo takes also the task of controlling the calibration light source. The calibration procedure is described in [6].

Chapter 8

MAGIC-2 camera control

8.1 General tasks

(to be written by the subsystem developers)

8.2 Controlled Parameters

(to be written by the subsystem developers)

8.3 States and State Transitions

(to be written by the subsystem developers)

8.4 Implementation

(to be written by the subsystem developers)

Chapter 9

MAGIC-2 calibration control

9.1 General tasks

(to be written by the subsystem developers)

9.2 Controlled Parameters

(to be written by the subsystem developers)

9.3 States and State Transitions

(to be written by the subsystem developers)

9.4 Implementation

(to be written by the subsystem developers)

Chapter 10

MAGIC-1 and MAGIC-2 active mirror controls

This system controls the mirror adjustment system: CCD camera and mirror panel step motors. It also controls the mirror heating system.

10.1 General tasks

10.1.1 Startup/Selftest

After startup, the system should test the availability of all mirror drives and go through a laser test of the whole reflector to determine the absolute positions of all the panels. Initialization is done once a day 20-30 minutes before data taking at the zenith position. After the initialization the system goes from "undefined" to "deadjusted" state and is ready to accept "laseradjust" or "adjust" commands.

The system knows the position of the telescope in alt-azimuth coordinates through the CC reports (10 second rate). This allows the system to select the appropriate table in case of adjustment without lasers.

Once the mirrors have been adjusted, the system goes to "adjusted" state. The system remains in this state until the telescope moves to an alt/az position which is too far (limits to be defined) from the position where the mirrors were adjusted. Then it goes to "deadjusted".

The system goes into "error" state whenever a large enough number of mirror panels cannot be adjusted, whenever the CCD camera fails or whenever a laser cannot be switched off. CC should close the camera if a laser cannot be switched off. The system knows that a laser has not been switched off only if the laser spot does not dissapear in the CCD image, so for security reasons CC must also close the camera if the CCD fails.

10.1.2 Mirror adjustment

The user has the following options:

1. Individual mirror adjustment with Laser/CCD measurement (for experimenting with individual mirrors); afterwards the user can save the value to a file

2. Individual adjustment without Laser/CCD measurement; the position of the mirror is restored to a value which is read from a file

3. Individual adjustment without Laser/CCD measurement; the position of the mirror is set to a value which is given by the user; afterwards the user can save the value to a file

4. Global mirror adjustment with Laser/CCD measurement (like the individual adjustment, but performed automatically for all mirrors); afterwards the user can save the values to a file [PRIORITARY].

5. Global mirror adjustment without Laser/CCD measurement; the positions of all mirrors are read from a file and restored accordingly [PRIORITARY].

The filenames are given by two coordinates (ALT and AZ) and a key letter (a, b, c, ..., z). This enables the system later to restore the adjustment which is closest to the present position of the telescope. In order to avoid confusion, no other filenames are allowed.

10.1.3 Mirror heating

The heating is not controlled through the AMC as originally intended. It rather works in manual or autonomous mode. What follows is just for completeness.

For the heating first a short description on what Munich told us how it works. There is a general heating which can be turned on and off for all mirrors. For deicing a high current can be sent to the mirror panels, as this current is very high only on sector (25%) of the telescope dish can be deiced at a time. The software must circle through all sectors. How long to deice each sector may depend on temperature and dewpoint. As this information is available to CC we recommend that CC decides for how long to deice and send AMC a value of seconds for how long this burst-heating should be applied to each sector.

10.1.4 Shutdown

The shutdown sequence should offer the user to restore the default adjustment (ALT = 90 dgerees, AZ = 0 degrees) and then switch all devices off.

10.1.5 Interaction with CC

The system can be in two major modes: stand-alone and CC.

After startup the system should try to contact CC. If CC is available (CC reports are received regularly), and its global state is not "error", the system should go to CC mode. In this mode the system accepts commands from CC. The definition of the interface with CC can be found in section 19.

The system goes to stand-alone mode if CC is no longer available (no reports are received). CC is in error state, or the operator sets it accordingly on the system control panel. In this mode the system does not accept CC commands, but tries to send reports normally.

10.2 Controlled Parameters

(to be written by the subsystem developers)

10.3 States and State Transitions

(to be written by the subsystem developers)

10.4 Implementation

(to be written by the subsystem developers)

Chapter 11

Lidar

Currently being implemented.

Chapter 12

M1 Level 2 trigger, i.e. prescaler

The Level 2 trigger control (L2TC) is the program that interacts with the trigger system (Level 1 and Level 2) and the Central Control (CC). The Level 2 trigger is located in two VME crates and a CPU hosts the control program.

The L2TC interfaces to:

1. the Level 1 trigger system, by an I/O card that selects the fold multiplicity;
2. the Level 2 trigger system, by means of memory boards (the so-called SMART boards);
3. the trigger rate monitor, by reading scalers and reporting the results to the CC.

12.1 General tasks

12.1.1 Startup/Selftest

At startup, the program executes an initialization script and checks for the presence of the needed boards. It makes sure that the parameters are correctly written into memory by reading them and making a comparison with default values. If these operations are not successful the program will go into the appropriate error state.

Also at startup, a TCP server and client threads are launched. The former waits for a connection from the CC, the latter to send reports to the CC. The client connection is attempted every 10 seconds until it is succesfully opened and reports to CC are started.

12.1.2 Trigger Tables

Upon instruction by the CC, the L2TC loads the trigger table on the L2 memories and configures the L1 trigger if necessary. The trigger table is a collection of

ASCII files resident on the local disk, containing commands that are executed by the L2TC. The information contained in every trigger table is:

- the L1 trigger multiplicity (NN = 2, 3, 4, 5);
- the L2 trigger setup, as a collection of data to be downloaded into the boards' memories (if enabled, a check on the write operation is performed);
- the factors to load into the prescaler registers;
- the maximum output rates expected for every trigger.

So far, the information to be stored in these files has not beed defined in its final form. However, a future TDAS note is expected to document the trigger table format. In principle, every new trigger table should be documented in a specific TDS note, showing the details of implementation and expected performances.

12.1.3 Trigger Rates Monitor

The L2TC checks the scaler counts at fixed time intervals (1 sec typically), it computes the rates and sends a report to Central Control. The trigger rates are compared to some preset values and an error message is issued when the rates are above these limits. Since the meaning of the trigger bits can vary changing trigger table, these rate limits are written in the trigger table itself and used by the L2TC to monitor the performance of the trigger system.

12.1.4 Stopping

The L2TC stops triggers when told to do that by the CC (**STOP!** command). The prescaler outputs are disabled so that no triggers reach the FADC system.

12.1.5 Shutdown

The L2TC is supposed to be run 24 h and only rarely to be shutdown. The shutdown should require a password and an email should be sent to the person in charge of the system.

12.1.6 Interaction with CC

The system can be in two major modes: stand-alone and CC.

After startup the system should try to contact CC. If CC is available (CC reports are received regularly), and its global state is not "error", the system should go to CC mode. In this mode the system accepts commands from CC. The definition of the interface with CC can be found in section 19.

The system goes to stand-alone mode if CC is no longer available (no reports are received). CC is in error state, or the operator sets it accordingly on the system control panel. In this mode the system does not accept CC commands, but tries to send reports normally.

12.2 Controlled Parameters

(to be written by the subsystem developers)

12.3 States and State Transitions

12.3.1 State "Error"

An error in the trigger programming may cause the subsystem to enter this state. When the subsystem is in this state, no further commands are accepted. The error must be recovered in order to get the system working. This operation cannot be accomplished via network commands, the system can be recovered only from the trigger console.

12.3.2 State "Idle"

The subsystem has performed succesfully the initial startup and it is ready to accept a trigger table to load or, if already loaded, to enable triggers.

12.3.3 State "Loading"

The system is busy while loading the trigger table into the L2 trigger system.

12.3.4 State "Ready"

The system has loaded succesfully a trigger table in the L2 trigger system.

12.3.5 State "Active"

The subsytem has received the command **START** from CC. The triggers are enabled by un-masking the prescaler(s) outputs.

12.3.6 State "Stopped"

The subsytem has received the command **STOP!** from CC. The triggers are disabled by masking off the prescaler(s) outputs.

12.4 Implementation

(to be written by the subsystem developers)

Chapter 13

M2 trigger: prescaler

13.1 General tasks

(to be written by the subsystem developers)

13.2 Controlled Parameters

(to be written by the subsystem developers)

13.3 States and State Transitions

(to be written by the subsystem developers)

13.4 Implementation

(to be written by the subsystem developers)

Chapter 14

Level 3 or stereo trigger

14.1 General tasks

(to be written by the subsystem developers)

14.2 Controlled Parameters

(to be written by the subsystem developers)

14.3 States and State Transitions

(to be written by the subsystem developers)

14.4 Implementation

(to be written by the subsystem developers)

Chapter 15

Burst Alarm

The Burst Alarm is the subsystem which monitors the GCN (Gamma-ray bursts Coordinates Network) for any alert on GRBs, and so to communicate such alerts to the Central Control providing the "trigger" for the repositioning of the telescope. The system is a daemon, a program running in background in a stand alone mode, and is meant to work 24 hours a day without interruptions.

The Burst Alarm System communicates with

- the GCN, for a continuos 24-hours monitoring for GRBs alerts;
- the CC, for a prompt noticing of the alert.

15.1 General Tasks

15.1.1 Startup/Selftest

The Burst Alarm system is managed by a program called *gspot* (Gamma Sources POinting Trigger), a system which monitors the presence of GRB alerts by a 24-hours no-stop communication with GCN, in order to communicate to the CC such alerts as soon as they happen and then to provide a "trigger" on the repositioning of the telescope.

The system is a daemon, a program running in background in a stand alone mode, and is meant to work 24 hours a day without interruptions.

At startup *gspot* tests the availability of the Internet connection and try to establish a socket connection to GCN. If there is no Internet connection or no GCN availability it continues to attempt to establish a connection to GCN. At startup *gspot* tries also to contact CC and start the communication with it. If CC is not available this does not affect the GRB monitoring, it continues to monitor GCN while trying to establish the communication with CC.

As told above, the *gspot* is meant to operate in a no-stop 24-hours a day, with very rarely interruptions (e.g. systems maintenance). In such cases the *gspot* can

be easily operated by a script which manages basics operations. In order to do this the operator must log into www machine as GrB user and then run the script

~/gspot {start|stop|restart|reload|status}

where:

- **start**: to start *gspotd* (gspot daemon). It first checks whether there is any other *gspotd* process running, and if so it does not start anything;
- **stop**: to stop *gspotd*;
- **restart|reload**: stop and restart *gspotd*;
- **status**: tells whether *gspotd* is running or not.

The Burst Monitor system *gspot* is also present in the boot sequence of www machine, so that in case of reboot of this machine the operator does not need to start up manually the system.

15.1.2 Normal operations

Normal operations are defined as the those managed when the system is not in alert status. The *gspot* operates onto two sides:

1. GCN: *gspot* continues to get standard *I'm alive* reports from GCN. If a problem concerning the communication (GCN problem, Internet problem...) occours then it starts a procedure to manage this situation and then re-establish the communication;
2. CC: *gspot* opens a *client* and a *server* for the interaction with CC. The *client* continues to send to the CC standard reports bringing the status of the system and of its communication with other systems. Obviously if there is a problem on the communication between the *gspot client* and the CC it cannot be reported, so that the system is considered as *unavailable* by the CC. The *server*, instead, gets reports from CC reporting its status. The definition of the interface with CC can be found in chapter 29.

Sometimes could happen that on www the *xntpd* process, the daemon which get the UTC from a time server, dies so that the *gspot* could loose the right time and become *unavailable* at the CC monitor. In such cases you must report the problem to any administrator and to the Burst Alarm responsible (common operators have no access to such tasks).

15.1.3 Alert

When an alert bringing GRB's coordinates comes from GCN, *gspot* calculates the observability of the coordinates according to some parameters, currently fixed and not settable by the user. Then it immediately sends a GRB-ALARM report to the CC, **EVEN IF THE COORDINATES ARE CONSIDERED NOT OBSERVABLE**. If the CC is not available it continues to attempt to send it the GRB-ALARM report for 2 hours. In any case ALL the informations recived from the alert are stored into an ASCII file, with also local informations (e.g. Alt/Az if coordinates are present), increasing the *GRB data base*.

Further documentation can be found in TDAS-??.

15.2 Controlled Parameters

As already mentioned the *gspot* is a daemon meant for no-stop 24-hours working in a *stand alone* mode, without been controlled by the CC neither by the operators. Unless something nasty happens it also does not have to be started up or shut down, but it must be only monitored.

15.3 States and State Transitions

The status of the Burst Alarm system can be fast cecked looking by the web page (not yet available)

http://magic.iac.es/BurstAlarm.html

The status is defined by four substati:

1. GCN status: te status of the communication with GCN;
2. CC *client* status: the status of the subsystem's *client* communicating (sending GRB-REPORTS) with CC;
3. CC *server* status: the status of the subsystem's *server* communicating (getting CC-REPORTS) with CC;
4. ALARM status: the status of the subsystem's mainframe concerning the presence of an alert.

15.3.1 GCN status

The GCN status, i.e. the status of the system's communication with GCN, is so defined:

- state **0**: ERROR - something nasty happened, maybe this thread died, you must check the global status of the system;
- state **3**: waiting for GCN connection;
- state **4**: communication with GCN is OK.

15.3.2 CC *client* status

The CC *client* status is the status of the *gspot's client* connected to CC's *server*, sending to CC all the routine GRB-REPORTS and the ALERT-REPORTS, and is so defined:

- **0**: ERROR! Something nasty happened, probably the program or this thread died, you must check the global status of the system;
- **4**: communication to CC *server* OK;
- **5**: connecting to CC.

15.3.3 CC *server* status

The CC *server* status is the status of the *gspot's server* connected to CC's *client*, getting from CC all the CC-REPORTS, and is so defined:

- **0**: ERROR! Something nasty happened, probably the program or this thread died, you must check the global status of the system;
- **3**: waiting for a CC connection.
- **4**: communication to CC *server* OK;

15.3.4 Alarm status

The Alarm status is the status of the subsystem's mainframe and is so defined:

- **0**: NO ALARM, the system is operating in normal mode;
- **8**: YELLOW ALARM, *gspot* got an alert from GCN but the GRB's coordinates are considered, by the system, not observable. The system starts anyway to send to CC the GRB-ALARM report since it recives from it a comfirm (i.e. the GRB-ALARM report echoed back) that CC got it. If after 2 hours the CC doesn't have sent a confirmation the *gspot* stops to send the GRB-ALARM report. The alert is anyway stored and sent via e-mail to MAGIC's grb mailing list;

- **9**: RED ALARM, *gspot* got an alert from GCN and the GRB's coordinates are considered, by the system, observable. The system starts to send to CC the GRB-ALARM report since it recives from it a comfirm (i.e. the GRB-ALARM report echoed back) that CC got it. If after 2 hours the CC doesn't have sent a confirmation the *gspot* stops to send the GRB-ALARM report. The alert is stored and sent via e-mail to MAGIC's grb mailing list.

Some remarks concerning the Alarm State: 1)when the Burst Alarm system is in Alarm State **8** or **9** it stores the whole package recived from GCN into an ASCII file, increasing the so called GRB's data-base; 2)The iteration of sending the GRB-ALARM report untill the Central Control confirms it is due in order not to loose alerts because of communication problems or shift time features (like when an alert occours few time before arehucas starts).

15.4 Implementation

Further implementations are going on with the following priority:

1. an interface for checking the system's status, essentially a web page where the operators can check the *gspot* status in details;

2. a database with all the GRB's alerts, bringing also further informations concerning what happened at the notice of these alerts;

3. an interface to allow operators to set some parameters of *gspot*, like the parameters to filter the GCN alerts and to calculate the GRB's observability.

The last point is currently not considered essential and won't be implemented ASAP. The reason is that the small number of alerts that MAGIC gets morally forces us to consider all alerts bringing GRB's coordinates. Thus such implementation is taken into account for the possibility that in the future, with the next satellites dedicated at the study of GRBs, the number of alerts could increase, so that the CC could need to filter furthermore the alerts.

Chapter 16

Auxiliary systems

The auxiliary systems server controls

- the Weather Station
- the NSB sensor (a single photomultiplier which monitors the night sky brightness)???
- the smoke sensors???
- the GRB monitor
- the fence lock???
- the mirror heating???

and provides the data to CC such that certain safety checks can be performed.

16.1 General tasks

16.1.1 Startup/Selftest

At startup, the system should power up all auxiliary systems and verify that they are available.

16.1.2 Weather

The weather parameters provided by the weather station are read out every 40 seconds and a graphical display of their values over a given time window (10 minutes up to 12 hours) is given such that trends can be seen. This information is also uploaded to a web page that can be accessed from outside and inside the computer system.

The latest values should be provided to CC.

A log file (ASCII format) with 10 minute averages should be written.

The width of the time window of the display should be kept variable.

For the value of the *humidity* and the *dew point* it should be possible to fix alarm limits which, when violated, set off a warning message and an acoustical signal which can only be stopped by CC or a local user with password.

16.1.3 NSB

Not implemented

The NSB monitor should be readout every second and the time history of the value over a given time (10 minutes up to 12 hours) graphically displayed.

The latest values should be provided to CC.

For comparison the measured values of a previous night can be superimposed on the plot.

The 1 minute averages are stored in log files (ASCII format).

16.1.4 Smoke sensors

Not implemented

The smoke sensors should be read out every second and the result displayed on the screen. If smoke is detected, a graphical display with the position of the sensor should be shown and an acoustical signal started which can only be stopped by CC or a local user with password.

16.2 Controlled Parameters

(to be written by the subsystem developers)

16.3 States and State Transitions

(to be written by the subsystem developers)

16.4 Implementation

(to be written by the subsystem developers)

Chapter 17

Central Control

17.1 General tasks

The task of CC is to provide the user with a simple interface to control the telescope in an orderly manner and with reproducible results.

17.1.1 Startup/Selftest

CC is the first of all subsystems which should be started up. At startup, it starts polling the other subsystems for availability. As soon as they are up, the communication is established and this fact is displayed to the user.

17.1.2 Communication with the other subsystems

CC is communicating directly with all subsystems except the online analysis via socket connections or file sharing. The parameters delivered by the subsystems are compared to safety limits/standard values and warnings or safety reactions result when the parameters seem to show a problematic state. In the worst cases, CC can tell a subsystem to stop and shut down.

Commands from CC to the other subsystems are executed by them as if the user had given them directly at the subsystems' consoles.

In order to avoid accidental interference by unexperienced users, the subsystems, when controlled by CC, are locked for direct interaction at their console except for functions which only concern the display characteristics.

A special protocol is defined for each subsystem (see chapter II).

17.2 Auxiliary elements

A number of elements in the telescope are controlled directly by CC:

- **Settings of the optical receivers of M1**, namely the **discriminator threshold** level and the **time delays** of the individual pixel trigger signals. This is done through a serial connection directly from the CC computer to the front of the optical receiver rack. Both parameters are set in arbitrary units from 0 to 255 using the corresponding utilities in the CC GUI. The discriminator thresholds are saved as a special receiver report file starting on Summer 2008. Its format in defined in section 31.2.

- **Weather parameters**. The weather station is read out over a serialo connection by a program running on *wwwint*. The weather parameters are written onto a file that is accessed over NFS by CC every second. A mean of the parameters every 3 seconds is displayed on screen and saved to the control file. If the parameters are outside the allowed range CC shuts down the system.

- **Difference between Rubidium time and GPS time**. The time stamp for the DAQ is provided by a rubidium clock coupled to a GPS receiver. The difference between both devices is monitored through a NIM module and reported over a serial RS232 connection to the second port of pc1 (CC). This difference, measured in μs, is readout by the CC every 10 seconds, sent as a part of the CC report to the subsystems and saved to the .rep file.

17.3 Controlled Parameters

CC keeps track of the state of the whole telescope by communicating with the sub-systems and permanently updating the global system state vector which is defined in table 17.3. State 9 is always "not-available by CC" and is defined only at CC.

System ID	State 0	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8
1.0 - CC	error	standby	data run	initialization	DAQ test	cal run	ped run	domcal run	lin run
1.1 - Weather	error				ok		warning	alarm	
1.2 - UPS	error	battery low			net high	battery high	net low		
1.3 - Thresholds	error	mismatch		configuring		nominal	control IPR		
1.4 - Time delays	error	mismatch		configuring		nominal			
1.5 - GPS	error					nominal			
1.6 - Schedule	error					nominal			
2 - DAQ	error	standby	daqtest	configuring	data run	cal run	ped run	point run	
3 - SumTrigger	error	idle	loading	ready	active	inactive			
4 - Drive	error	stopped		slewing	tracking				
4 - StarGuider	error	stopped	standby	await ZPET	monitoring	choose star			
5 - Pyrometer	error				ok				
6.0 - CaCo	error	alarm	blocked	warm-up	hot	hv ramping	ok	init	shutd
6.1 - Cal.Control	error	blocked	waiting	firing				init	shutd
6.2 - Cam.Sentinel	error	all ok	sun	bad atm	disabled	DC alarm	warm-up	cc timeout	
6.3 - Cam.HV.PS	error	mismatch	tripped	ramping	off	nominal	limit curr		
6.4 - Cam.Lid	error	safety lim passd			closed	open	moving	stopped	
6.5 - Cam.LV	error	LV alarm	LV tripped		LV off	LV on			
6.6 - Cam.Cool	error	alarm			off	ok	temp warn	cond warn	
6.7 - Cam.HV	error	mismatch			hv off	hv nominal			
6.8 - Cam.DC	error	DC alarm		hot		ok	warm		
6.9 - Caos.LEDs	error				off	on			
6.10 - FADC.fans	error				on	off			
6.11 - Cal.CanBus	error			ok					
6.12 - Cal.IO	error		waiting	firing					
6.13 - Cal.LV	error		off	on					
6.14 - AMC.LEDs	error				off	on			
7.0 - AMC	error	parked	initialized	readjust	adjusted	laser	moving		
7.1 - Mirror Heating	error	off		on	deicing				
8 - Lidar	error				ok				
11 - L2T	error	idle	loading	ready	active	inactive			
12 - AUX	error	stopped			OK				
13 - GRB	error			waiting	OK	connecting		yellow al	red al

System ID	State 0	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8	State 9	State 10	State 11
12 - Pulsars	error	idle	loading	ok								
13 - Trigger2	error	idle	loading	ready	active	inactive						
14.0 - Drive2	error	stopped		slewing	tracking							
14.1 - StG2	error	stopped	standby	wait ZPET	monitoring	choose star						
14.2 - TPoint	failed	succeeded										
16.0 - CaCo2	error	data mode	AMC mode	tpoint mode	standby	blocked			shutdown			
16.1 - Lid	error	passed		tpoint	closed	open	moving	stopped				
16.2 - Sentinel	disabled	ok	bad cond		DC alarm							
16.3 - LV	off	on	error									
16.4 - HV	off	on	error	mismatch								
16.5 - PD	error	nominal	off									
16.6 - Bias	off	nominal	error									
16.7 - Attenuat.	off	nominal	error									
16.8 - DC	off	nominal	error									
16.9 - AMCLeds	closed	open	error									
16.10 - StgLeds	closed	open	error									
16.11 - PulseInj	error	on	off									
16.12 - Pixel Temp	error	fine	hot									
16.13 - Clus Temp	error	fine	hot									
16.14 - Cam Temp	error	fine	hot									
16.15 - Humidity	error	fine	humid									
17.0 - AMC2	error	parked	initialized	readjust	adjusted	laser adj	moving	undef				
18.0 - Rec2	error	idle	loading	ok				burning				
18.1 - Delays	error		loading	nominal								
18.2 - DT	error		loading	nominal	IPR control							
18.3 - PulseWidth	error		loading	nominal								
19.0 - DAQ2	error	standby	daqtest	configuring	data run	cal run	ped run	domcal run	unavail	lin run	pinj run	pedsub run
19.1- DominoCal	disactive	test	active									
20.0 - Cooling2	error			ok	warning			burning				
20.1 - IntCool	error			ok	warning	problem	configuring					
20.2 - ExtCool	error		off	ok								
20.3 - PSCool	error		off	ok								
21 - Calib2	error	ready	waiting for ready	firing	configuring	undefined						
22 - L3	error	idle	loading	ready	active	inactive						

17.4 Implementation

The latest version of the CC, SuperArehucas, is a Labview version 8 program. Labview is commercial software developed by National Instruments (see [2] for more details).

17.4.1 The main loop event

The main loop event of CC goes through the following steps:

1. Determine subsystem states (except camera currents) by reading their reports
2. Adjust subsystem displays
3. Determine global state
4. Adjust global state display, available global commands, and vetos for subsystem commands
5. Update global time display
6. Process commands.
7. Register next main loop event at 0.2 seconds from now

Part II

Communication protocols

Chapter 18

Generic subsystem-CC interaction

Please note that there have been major changes in this section. No standalone/CC modes exist any more in the old sense.

Formats are defined in C-style according to [1], section B 1.2.

18.1 Guidelines regarding interaction with CC

1. The subsystem can be in two major modes: stand-alone and CC.

2. CC sends reports to all the subsystems every 10 seconds. A subsystem knows that CC is no longer available if it does not get CC reports for more than 30 seconds. It should then change its "communication state" in the report, so that CC is aware of the problem (1=ok, 2=problem). A subsystem is in CC mode whenever it regularly receives CC reports. A subsystem in CC mode must be ready to receive and execute CC commands.

3. What has to be done in standalone mode is reached depends on the subsystem. In general it should go on with its normal operation until some security timeout is reached Only then it must "protect itself" from whatever danger it foresees. For instance CaCo may close the camera and switch off the pixel HV, Drive may go to park, etc.

4. The subsystems send reports to CC every second. This is done whenever a TCP connection to CC can be established, no matter if CC is available. If CC does not get any report from a given subsystem in 5 seconds, the subsystem will be considered as unavailable (n/a in table 17.3, page 67). This is an extra state not provided by the subsystem itself but to be included in the state machine in CC.

5. CC decision loop requires knowing the state of each subsystem at a given time. There is no time in the decision loop to request for a report and wait for it. If the connection to some subsystem is broken, it must be known at that time. So this report/state update should then be asynchronous with the decision loop.

18.1.1 Connection/Control commands

The LOCK, ULOCK and ADIOS commands are obsolete.

18.1.2 Data from subsystems to be kept: send to CC

If there's any chance some data from a subsystem operation can be needed in a offline analysis, it should be sent to CC so that it can store in a central place or forward it to the system that will do this. So, what is to be kept has to be specified in each subsystem specific chapter, and sent according the agreed format.

18.2 Implementation

The subsystem reports and CC commands were chosen to be delivered through network by TCP/IP streams. This requires the end points of the socket connection interpreting the strings recieved according a defined protocol.

18.2.1 Communication flows

The communication between CC and each subsystem can be classified by its direction:

- Subsystem Report: CC $\longleftarrow$ Subsys
- CC Commands: CC $\longrightarrow$ Subsys.

It is then natural to use a different TCP/IP socket for each flow, especially given that being connected (and sending reports) is not tied to be in CC mode (receiving commands).

This way, in a subsystem one has a *commandListener* socket that has to listen and interpret the strings from it as commands (See section 21.1 and correspondent ones for other subsystems). On the other hand, the subsystem will have another socket *reportSender* to which it will write periodically its report (see sections 21.2.1 and correspondent ones for other subsystems), but won't have to listen to it. CC would have the complementary sockets *commandSender* and *reportListener* with complementary tasks.

"Always awaiting from CC" feature [1] is implemented by subsystem always running the *commandListener* TCP server which listens to CC *commandSender*, as a client.

[1] awaiting, be it reporting or not; being standalone or not.

Chapter 19

MAGIC-1 and MAGIC-2 DAQ - CC

The two DAQ programs are pretty similar so it makes sense to merge the chapters in one. The few differences are highlighted in the text.

The communication uses socket connections (TCP/IP).

19.1 Commands to be received from CC

19.1.1 Commands concerning the DAQ

The "projectname" string is a mean to classify the run, e.g. all data belonging to the same observing proposal can be labeled in this way.

The source coordinates are the source celestial coordinates as found in the CC catalogues and sent to the drive subsystem. Only the drive subsystem knows about EpochOfDate or local coordinates.

1. start data run

   CC sends:

```
"DARUN %d %d"
" %s %s"
" %03d %02d %05.2f"
" %03d %02d %05.2f"
" %1.1s %04d"
" %03d %02d %05.2f"
" %03d %02d %05.2f\n"
 numtriggers(0=infinite), run_number,
 projectname, sourcename,
 SourceRaHour, SourceRaMin, SourceRaSec,
```

```
 SourceDecDeg, SourceDecMin, SourceDecSec
 SourceEpochChar, SourceEpochDate, TrackingMode
 TelescopeRaHour, TelescopeRaMin, TelescopeRaSec,
 TelescopeDecDeg, TelescopeDecMin, TelescopeDecSec
```

2. start pedestal run

   CC sends:

```
"PERUN %d %d\n",
 numtriggers, run_number
```

3. start calibration run

   CC sends:

```
"CARUN %d %d"
" %s %s"
" %03d %02d %05.2f"
" %03d %02d %05.2f"
" %1.1s %04d"
" %03d %02d %05.2f"
" %03d %02d %05.2f\n"
 numtriggers, run_number,
 projectname, sourcename,
 SourceRaHour, SourceRaMin, SourceRaSec,
 SourceDecDeg, SourceDecMin, SourceDecSec
 SourceEpochChar, SourceEpochDate, TrackingMode
 TelescopeRaHour, TelescopeRaMin, TelescopeRaSec,
 TelescopeDecDeg, TelescopeDecMin, TelescopeDecSec
```

4. start linearity run

   CC sends:

```
"INRUN %d %d"
" %s %s"
" %03d %02d %05.2f"
" %03d %02d %05.2f"
" %1.1s %04d"
" %03d %02d %05.2f"
" %03d %02d %05.2f\n"
```

```
numtriggers(=0 for M2 as of SA 090616), run_number,
projectname, sourcename,
SourceRaHour, SourceRaMin, SourceRaSec,
SourceDecDeg, SourceDecMin, SourceDecSec
SourceEpochChar, SourceEpochDate, TrackingMode
TelescopeRaHour, TelescopeRaMin, TelescopeRaSec,
TelescopeDecDeg, TelescopeDecMin, TelescopeDecSec
```

5. start pedestal subtraction run

   CC sends:

   ```
   "PSRUN %d %d\n",
    numtriggers, run_number
   ```

6. start domino calibration run (only in M2 DAQ as of SA 090616-0)

   CC sends:

   ```
   "DCRUN 0 %d %s\n"
    run_number, projectname
   ```

7. abort domino calibration (only in M2 DAQ as of SA 090616-0)

   CC sends 2 commands, one immediately after the other:

   ```
   ABORTDOMCALIB
   ```

   and

   ```
   STOP
   ```

8. halt present run

   CC sends:

   ```
   "%STOP!\n"
   ```

   (For interrupting the run. Go to stand-by)

9. go to DAQ test

   CC sends:

   ```
   "DTEST\n"
   ```

   (For going from stand-by to daqtest run. This command is normally issued inmediately after a HALT! command.)

10. start online analysis
    (starting on Arehucas version 070926-0)

    CC sends:

    ```
    "OASTART %d %d %s %s\n"
     numtriggers(0=infinite), tel_num(0=M1 1=M2),
     sourcename (without evt. wobble extensions), random forest for OA (RF4OA)
    ```

19.2 Data sent to CC

Every second, the DAQ should send a report to CC (even if it is in stand alone mode, as long it is connected to CC).

19.2.1 Data concerning the DAQ

1. status (error, standby, daqtest, configuring, data run, cal run, ped run, test run) according to table 17.3 (page 67)
2. run number
3. last event number
4. run time elapsed in seconds
5. present first level trigger rate (events/s)
6. present storage rate (events/s)
7. present storage rate (kB/s)
8. present gamma candidate rate (events/s, from online analysis)
9. remaining disk space (GB)
10. remaining time until storage 90 % full with given storage rate (seconds)

11. The pixel charges of a random event in FADC counts. This event must have been recorded in the last second. The charge is calculated using a robust signal extractor, for the time being sliding window (4 slices). The pixel pedestal is substracted.

12. The pixel arrival times of the above event in FADC slices. They are calculated using the same signal extractor.

13. Information about the interleaved calibration events that are taken among the normal data events: number of events with the calibration tag during the last second and number of events that satify a given criterion that identifies them as calibration events (typically the charge in the pixels and the number of pixels over a given threshold)

19.2.2 Data concerning the communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

19.2.3 Format definition

Subsystem sends:

```
"DAQ-REPORT %s %02d %04d %02d %02d"
" %02d %02d %02d %03d"
" %02d %04d %02d %02d"
" %02d %02d %02d %03d"
" %08d %08d %06d"
" %6.1f %6.1f %06d %7.4f "
" %10.8u %8.6u"
" %1.d"
telescope, status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec,
comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec,
run_number, last_event_number, run_time_elapsed_sec,
trigrate_hz, storerate_hz, storerate_kbps, gammarate_hz,
remtime_sec, diskspace_MB,
domino calibration state

Starting on Arehucas version 050317-0 it also contains:

" <npix>*%04x  ATIME  <npix>*%02x"
```

```
10*Charge[iPixel], 10*Atime[iPixel]

replaced in Arehucas version 070416-0 by:

" <npix>*%05x  ATIME  <npix>*%03x"
Charge[iPixel], 10*Atime[iPixel])

in M2 this is (as of SA 090616-0)

" <npix>*%05x  ATIME  <npix>*%03x",
16*Charge[iPixel], 10*Atime[iPixel]

in M1 as of SA 1110240 is:

" <npix>*%05x  ATIME  <npix>*%03x",
Charge[iPixel], 10*Atime[iPixel]

The end of the report is:

" CALIB_EVENTS %03x %03x",
rate_cal_tag_events, rate_probable_cal_events
" OVER\n"
```

19.2.4 Special Report

Starting on Arehucas version 050317-0 the end reports for the different kind of runs contain much more information. Before that version the reports contained only the charge means and RMS for all the pixels.

The report at the start of any run has the following format for both telescopes:

```
"STARU %s %d %d %d"
" %d %d %d %d"
" %d %d %s %s"
 telescope number, run_start_year, run_start_month, run_start_day,
 run_start_hour, run_start_min, run_start_sec, run_start_msec
 run_number, subrun_number, project_name, run_type
```

The report at the end of a data run has the following format for both telescopes:

```
"ENDDR %2.2s %04d %02d %02d"
" %02d %02d %02d %03d"
```

```
" %010d %05d %22.22s %010d"
" COSMIC_MEAN <npix>*%05x"
" COSMIC_ATIME <npix>*%03x"
" COSMIC_HIT <npix>*%02x"
" CALIB_MEAN <npix>*%05x"
" CALIB_RMS <npix>*%04x"
" CALIB_ATIME <npix>*%03x"
" PED_MEAN <npix>*%04x"
" PED_RMS <npix>*%03x"
" CALIB_EVENTS %03x %03x",
" OVER\n"
 telescope (M1 or M2), end_run_year, end_run_month, end_run_day,
 end_run_hour, end_run_min, end_run_sec, end_run_msec
 run_number, subrun_number, project_name, number_events_written
 mean_cosmic_charge[iPixel],
 10*mean_cosmic_atime[iPixel],
 10*mean_cosmic_number_hits[iPixel]
 mean_calibration_charge[iPixel],
 rms_calibration_charge[iPixel],
 10*mean_calibration_atime[iPixel],
 mean_pedestal_charge[iPixel],
 rms_pedestal_charge[iPixel],
 number_cal_tag_events, number_probable_cal_events
```

The only variable that is not self-explanatory is mean_cosmic_number_hits. This is the percentage of cosmic events that have a charge greater than a given threshold.

The report at the end of a calibration , pulse injection or linearity run has the following format:

```
"END%.1sR %s %02d %04d %02d %02d"
" %02d %02d %02d %03d"
" %d %d %s %d"
" <npix>*%05x"
" <npix>*%04x"
" <npix>*%03x"
" CALIB_EVENTS %03x %03x",
" OVER\n"
 run_type, telescope (M1 or M2), status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
 mean_calibration_charge[iPixel],
```

```
 rms_calibration_charge[iPixel],
 10*mean_calibration_atime[iPixel],
 number_cal_tag_events, number_probable_cal_events
```

Here `run_type` is `C` for calibration, `J` for pulse injection and `N` for intensity (linearity) runs.

The report at the end of a pedestal run has the following format:

```
"ENDPR %s %02d %04d %02d %02d"
"  %02d %02d %02d %03d"
" %d %d %s %d"
" PED_MEAN <npix>*%04x"
" PED_RMS <npix>*%03x"
" OVER\n"
 telescope (M1 or M2), status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
 mean_pedestal_charge[iPixel],
 rms_pedestal_charge[iPixel],
```

This for special (`S`,`?`) and domino calibration (`L`) runs the format is simply:

```
"END%.1sR %s %02d %04d %02d %02d"
"  %02d %02d %02d %03d"
" %d %d %s %d"
" OVER\n"
 run_type, telescope (M1 or M2), status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
```

19.2.5 Run Reports stored in .rep file

At the beginning of each type of run:

```
"RUN-REPORT %s 00 %d %d %d"
" %d %d %d %d"
"START %d %d %s %s OVER\n"
 telescope, run_start_year, run_start_month, run_start_day
 run_start_hour, run_start_min, run_start_sec, run_start_msec
 run_number, subrun_number, project_name, run_type
```

At the end of a data run (D):

```
"RUN-REPORT %s 04 %04d %02d %02d %02d %02d %02d"
" STOP %d %d %s %d"
" COSMIC_MEAN <npix>*%05x"
" COSMIC_ATIME <npix>*%03x"
" COSMIC_HIT <npix>*%02x"
" CALIB_MEAN <npix>*%05x"
" CALIB_RMS <npix>*%04x"
" CALIB_ATIME <npix>*%03x"
" PED_MEAN <npix>*%04x"
" PED_RMS <npix>*%03x"
" CALIB_EVENTS %03x %03x",
" OVER\n"
 telescope, year, month, day , hour, minutes, sec, msec
 run number, sub run number, project name, number of events
 mean_cosmic_charge[iPixel],
 10*mean_cosmic_atime[iPixel],
 10*mean_cosmic_number_hits[iPixel]
 mean_calibration_charge[iPixel],
 rms_calibration_charge[iPixel],
 10*mean_calibration_atime[iPixel],
 mean_pedestal_charge[iPixel],
 rms_pedestal_charge[iPixel],
 number_cal_tag_events, number_probable_cal_events
```

At the end of a calibration (C), pulse injection (J) and intensity (linearity) (N) runs:

```
"RUN-REPORT %s %02d %02d %04d %02d %02d"
" %02d %02d %02d %03d"
" %d %d %s %d"
" <npix>*%05x"
" <npix>*%04x"
" <npix>*%03x"
" CALIB_EVENTS %03x %03x",
" OVER\n"
 telescope (M1 or M2), run_identifier, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
 mean_calibration_charge[iPixel],
```

```
 rms_calibration_charge[iPixel],
 10*mean_calibration_atime[iPixel],
 number_cal_tag_events, number_probable_cal_events
```

where `run_identifier` is 05 for calibration, 10 for pulse injection and 08 for intensity (linearity)

The run-report at the end of a pedestal run has the following format for M1:

```
"RUN-REPORT %s 06 %02d %04d %02d %02d"
"  %02d %02d %02d %03d"
" %d %d %s %d"
" PED_MEAN <npix>*%04x"
" PED_RMS <npix>*%03x"
" OVER\n"
 telescope (M1 or M2), status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
 mean_pedestal_charge[iPixel],
 rms_pedestal_charge[iPixel],
```

The report at the end of a domino calibration (`L`) run is simply:

```
"RUN-REPORT %s 07 %02d %04d %02d %02d"
"  %02d %02d %02d %03d"
" %d %d %s %d"
" OVER\n"
 run_type, telescope (M1 or M2), status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec
 run_number, subrun_number, project_name, number_events
```

Chapter 20

Camera oscillation monitor - CC

This subsystem has not been implemented.

Chapter 21

MAGIC-1 Telescope drive, star guider, TPoints - CC

The communication uses socket connections (TCP/IP).

21.1 Commands to be received from CC

21.1.1 Commands concerning the drive

1. set target position in celestial coordinates (RA/DEC) for a given epoch (never epoch-of-date!), move the telescope **in slow mode** and start tracking

   CC sends:

   ```
   "RADEC %c %02d %02d %05.2f %c %02d %02d %05.2f\n"
    rasign, rahours, ramin, rasec, decsign, decdeg, decarcmin, decarcsec
   ```

   `rasign` and `decsign` can be `+` or `-`.

2. set target position in celestial coordinates (RA/DEC) for a given epoch (never epoch-of-date!), move the telescope **in fast mode** and start tracking

   CC sends:

   ```
   "%.5s %03d %02d %05.2f %03d %02d %05.2f %04d\n",
   "GRB", rahours, ramin, rasec, decdeg, decarcmin, decarcsec, epoch
   ```

3. set target position in local coordinates (HA/DEC), move the telescope and go to "stopped" state

   CC sends:

```
"%.5s %03d %02d %05.2f %03d %02d %05.2f\n",
"HADEC", hahours, hamin, hasec, decdeg, decarcmin, decarcsec
```

4. set target position in local coordinates (ALT/AZ) move the telescope and go to "stopped" state

   CC sends:

   ```
   "%.5s %03d %02d %05.2f %03d %02d %05.2f\n",
   "ALTAZ", altdeg, altmin, altsec, azdeg, azmin, azsec
   ```

5. move to park position and go to "stopped" state

   CC sends:

   ```
   "%.5s %s\n", "PREPS PARK"
   ```

   or instead of "PARK"

   ```
   MOUNT, STORM (any other??)
   ```

   The exact coordinates of all these special positions are defined only in the drive subsystem.

6. stop drive (i.e. stop the telescope movement. Drive will go to standby state)

   CC sends:

   ```
   "%.5s\n", "STOP!"
   ```

21.2 Data sent to CC

Every second, the drive control and the star guider send a report to CC (if it is not in stand alone mode). The reports contain:

21.2.1 Data concerning the drive

1. status (error, parked, stopped, slewing, tracking) according to table 17.3 (page 67)
2. requested source position in RA, DEC, HA, Zd and Az.
3. MJD calculated by the drive.
4. present position in Zd and Az.
5. error in Zd and Az, in seconds

21.2.2 Data concerning the communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

21.2.3 Drive format definition

All this information is sent in one report:
Subsystem sends:

```
("DRIVE-REPORT %02d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d",
 comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
 lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" %1s %03d %02d %02d
   %1s %03d %02d %02d
   %1s %03d %02d %02d",
 requested_ra_sign ("+" or "-"), requested_ra_hours,
 requested_ra_min, requested_ra_sec,
 requested_dec_sign ("+" or "-"), requested_dec_hours,
 requested_dec_min, requested_dec_sec,
 requested_ha_sign ("+" or "-"), requested_ha_hours,
 requested_ha_min, requested_ha_sec)

(" %f",
 MJD)
```

```
(" %1s %03d %02d %02d
   %1s %03d %02d %02d
   %1s %03d %02d %02d",
 requested_zd_sign ("+" or "-"), requested_zd_hours,
 requested_zd_min, requested_zd_sec,
 requested_az_sign ("+" or "-"), requested_az_hours,
 requested_az_min, requested_az_sec)

(" %1s %03d %02d %02d
   %1s %03d %02d %02d
   %1s %03d %02d %02d",
 present_zd_sign ("+" or "-"), present_zd_hours,
 present_zd_min, present_zd_sec,
 present_az_sign ("+" or "-"), present_az_hours,
 present_az_min, present_az_sec)

(" %f %f",
 error_Zd_sec, error_Az_sec)
```

21.2.4 Special Report

No special report is defined.

21.2.5 Data concerning the star guider

1. status (error, parked, stopped, slewing, tracking) according to table 17.3 (page 67)
2. Mispointing in zenith and azimuth angle, in arcminutes (up to Arehucas version 050317-0 the units were deg^2/arcmin)

   Starting on Arehucas version 050317-0 it also contains:
3. Nominal zenith and azimuth angle of the telescope.
4. Coordinates of the center of the camera, in pixels.
5. NUmber of "identified stars", i.e., number of bright spots identified by the filter in the starguider window (filter = black box in the starfield; identified spots are marked by a white cross)
6. Number of "correlated stars", i.e., number of spots, that could be correlated with catalogue positions of stars (catalogue positions of stars are marked with blue circles in the starguider display)

7. Brightness of the sky.

8. MJD.

21.2.6 Data concerning the communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

21.2.7 Star guider format definition

All this information is sent in one report:
Subsystem sends:

```
("STARG-REPORT %02d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d",
 comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
 lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" %f %f",
 mispointing_za_arcmin, mispointing_az_arcmin)
```

Starting on Arehucas version 050317-0 it also contains:

```
(" %c %d %d %d",
 nominal_za_sign ("+" or "-"), nominal_za_deg,
 nominal_za_min, nominal_za_sec)

(" %c %d %d %d",
 nominal_az_sign ("+" or "-"), nominal_az_deg,
 nominal_az_min, nominal_az_sec)

(" %f %f",
 camera_center_x_pix, camera_center_y_pix)

(" %d", number_identified_stars)

(" %f", brightness)

(" %f", mjd)
```

```
WARNING! Starting on 2006/06, the last three variables changed to:

(" %d", number_correlated_stars)

      (" %f", brightness)

      (" %f", mjd)

      (" %f %f", dummy1, dummy2)

      (" %d", number_identified_stars)
```

Chapter 22

MAGIC-2 Telescope drive, star guider, TPoints - CC

The communication uses socket connections (TCP/IP).

22.1 Commands to be received from CC

22.1.1 Commands concerning the drive

1. set target position in celestial coordinates (RA/DEC) for a given epoch (never epoch-of-date!), move the telescope **in slow mode** and start tracking

   CC sends:

   ```
   "%.5s %03d %02d %05.2f %03d %02d %05.2f %04d\n",
   "RADEC", rahours, ramin, rasec, decdeg, decarcmin, decarcsec, epoch
   ```

2. set target position in celestial coordinates (RA/DEC) for a given epoch (never epoch-of-date!), move the telescope **in fast mode** and start tracking

   CC sends:

   ```
   "%.5s %03d %02d %05.2f %03d %02d %05.2f %04d\n",
   "GRB", rahours, ramin, rasec, decdeg, decarcmin, decarcsec, epoch
   ```

3. set target position in local coordinates (HA/DEC), move the telescope and go to "stopped" state

   CC sends:

```
"%.5s %03d %02d %05.2f %03d %02d %05.2f\n",
"HADEC", hahours, hamin, hasec, decdeg, decarcmin, decarcsec
```

4. set target position in local coordinates (ALT/AZ) move the telescope and go to "stopped" state

   CC sends:

   ```
   "%.5s %03d %02d %05.2f %03d %02d %05.2f\n",
   "ALTAZ", altdeg, altmin, altsec, azdeg, azmin, azsec
   ```

5. move to park position and go to "stopped" state

   CC sends:

   ```
   "%.5s %s\n", "PREPS PARK"
   ```

   or instead of "PARK"

   ```
   MOUNT, STORM (any other??)
   ```

   The exact coordinates of all these special positions are defined only in the drive subsystem.

6. stop drive (i.e. stop the telescope movement. Drive will go to standby state)

   CC sends:

   ```
   "%.5s\n", "STOP!"
   ```

22.2 Data sent to CC

Every second, the drive control and the star guider send a report to CC (if it is not in stand alone mode). The reports contain:

22.2.1 Data concerning the drive

1. the string "M2" which specifies the telescope
2. status (error, parked, stopped, slewing, tracking) according to table 17.3 (page 67)
3. requested source position in RA, DEC, HA, Zd and Az.
4. MJD calculated by the drive.
5. present position in Zd and Az.
6. error in Zd and Az, in seconds

22.2.2 Data concerning the drive communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

22.2.3 Drive format definition

All this information is sent in one report:
Subsystem sends:
DRIVE-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
DRIVE2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
DRIVE2 comm state, year, month, day, hour, minuts, seconds, ms
(requested coordinates)
%s %d %d %d
sign, Ra [hour], Ra [min], Ra [sec]
%s %d %d %d
dec [deg], dec [min], dec [sec]
%s %d %d %d
sign, HA [hour], HA [min], HA [sec]
%f
mjd
%s %d %d %d
sign, Zd [deg], Zd[min], Zd [sec]
%s %d %d %d
sign, Az[deg], Az [min], Az[sec]
(current coordinates)
%s %d %d %d
sign, Zd [hour], Zd [min], Zd [sec]

%s %d %d %d
Az [deg], Az [min], Az [sec]
%f %f
error Zd [sec], erros Az [sec]
%d %d
cosy lock, starguider mode

22.2.4 Data concerning the star guider

1. the string "M2" which specifies the telescope

2. status (error, parked, stopped, slewing, tracking) according to table 17.3 (page 67)

3. Mispointing in zenith and azimuth angle, in arcminutes (up to Arehucas version 050317-0 the units were deg^2/arcmin)

   Starting on Arehucas version 050317-0 it also contains:

4. Nominal zenith and azimuth angle of the telescope.

5. Coordinates of the center of the camera, in pixels.

6. NUmber of "identified stars", i.e., number of bright spots identified by the filter in the starguider window (filter = black box in the starfield; identified spots are marked by a white cross)

7. Number of "correlated stars", i.e., number of spots, that could be correlated with catalogue positions of stars (catalogue positions of stars are marked with blue circles in the starguider display)

8. Brightness of the sky.

9. MJD.

22.2.5 Data concerning the star guider communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

22.2.6 Star guider format definition

All this information is sent in one report:

Subsystem sends: STARG-REPORT M2 04 2011 10 25 01 54 11 384 01 2011 10 25 01 54 02 124 2.734 0.802 + 021 31 026 + 060 06 025 031.8 264.2 0028 92.8 55859.079298 5 10 0065

STARG-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
STG2-state, year, month, day, hour, minutes, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
STG2-comm-state, year, month, day, hour, minutes, seconds, ms
%f %f
mis-Zd[min], mis-Az [min]
%s %d %d %d
nominal Zd (sign, deg, min, sec)
%s %d %d %d
nominal-Az(sign, deg, min, sec)
%f %f %d
center-X, center-Y, n
%f %f
sky-brightness, MJD
OVER

22.2.7 Special TPoint report

Subsystem sends:

TPOINT-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
TPOINT2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
TPOINT2 comm state, year, month, day, hour, minuts, seconds, ms
%s
Tpoint type
%8.4f %8.4f
requested Az, requested elev
%8.4f %8.4f
measured Az, measured elev
%8.4f %8.4f
measured misAz, measured misElev
%8.4f %8.4f
measured misAz, measured misElev
%12.6f
MJD
%f %f %f %f %f %f

cam ctr mag, artificial mag, camera ctr x, camera ctr y, star x, star y
%d %d %d %d
no. LEDs, no. circles, no. detected star, no. correlated stars
%f %f %s
sky brightness, magnitude, star name
OVER

Chapter 23

Pyrometer - CC

This subsystem was implemented in AREHUCAS version 20070515. The communication uses socket connection (TCP/IP).

23.1 Data sent to CC

Every second, the pyrometer should send a report to CC, but its information change every 10 seconds.

23.1.1 Commands concerning the pyrometer

1. status ("error", "ok") according to table 17.3 (page 67)
2. Sky Teperature [K]
3. Cloudiness [%]
4. Lid status ("open", "closed")
5. Air Temperature [K]

23.1.2 Commands concerning the communication

1. time when the information was written (UTC)
2. status of the communication (1=ok, 2=problem)
3. time when the last command was received (UTC)

23.1.3 Pyrometer format definition

CC sends:

```
("%d %d %d %d %d %d %d %d"
 status, writeyear, writemonth, writeday,
 writehour, writemin, writesec, writemsec)

(" %d %d %d %d %d %d %d %d"
 comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
 lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" %f %f %d %f"
 skyT, cloudiness, lidstatus, airT)

("OVER\n")
```

Chapter 24

MAGIC-1 and MAGIC-2 Camera Control - CC

The communication will use socket connections (TCP/IP).

24.1 Commands to be received from CC

`PINON` Turns on the pulser power for pulse injection. No arguments;

`PINOF` Turns off the pulser power for pulse injection. No arguments;

`AT001` Sets the attenuator value to one pixel.
Format: `AT001 %04d%02d , pix_num, atten_val`;

`ATFIL` Sets the attenuator values reading them from a file.
Format: `ATFIL %s , filename`;

`ATPIX` Sets the attenuator value to more than one custom pixel.
Format: `ATPIX %04d%02d%04d%02d... , pix_num, atten_val, pix_num, atten_val ...`;

`HVPIX` Sets the HV value to one pixel.
Format: `HVPIX %04d%04d , pix_num, hv_val`;

`HVALL` Sets the HV value to all pixels.
Format (without HPDs): `HVPIX 1039*%04d , 1039*hv_val`;

`HVFIL` Sets the HV values reading them from a file.
Format: `HVFIL %s , filename`;

`BSPIX` Sets the Bias value to one pixel.
Format: `BSPIX %04d%04d , pix_num, bs_val`;

`BSALL` Sets the Bias value to all pixels.
Format (without HPDs): `BSPIX 1039*%04d , 1039*bs_val`;

`BSFIL` Sets the Bias values reading them from a file.
Format: `BSFIL %s , filename`;

`LVONN` Turns on the low voltage. No arguments;

`LVOFF` Turns off the low voltage. No arguments;

`AMCON` Turns on the AMC leds. The intensity is given in steps of 3276.8 (?).
Format: `AMCON %d , led_intensity`;

`AMCOF` Turns off the AMC leds. No arguments;

`LEDON` Turns on the Starguider leds. The intensity is given in steps of 3276.8 (?).
Format: `LEDON %d , led_intensity`;

`LEDOF` Turns off the Starguider leds.

`LDOPN` Opens the lids. No arguments;

`LDCLS` Closes the lids. No arguments;

`LDTPN` Puts the lids in target/tpoint mode. No arguments;

`LDSTP` Stops the lids. No arguments;

`POSCH` Sent to signal that the telescope is moving. Used for HPD control. No arguments.

24.2 Data sent to CC

Every second, the camera control should should send a report to CC (even if it is in stand alone mode, as long it is connected to CC). The camera report contains:

24.2.0.1 Data concerning the Camera2 Subsystem

1. a string ("M1" or "M2") which specifies the telescope.
2. status (...) according to table 17.3 (page 67)
3. .....
4. State of the camera lids...

24.2.1 Data concerning the communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

24.2.2 Format definition

All this information is sent in one report:

Subsystem sends:

CAMERA-REPORT %s %02d %04d %02d %02d %02d %02d %02d %03d
telescope-number, camera-state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
camera-comm-state, year, month, day, hour, minuts, seconds, ms
%1d %1d %1d %1d %1d %1d %1d %1d %1d %1d %1d %1d %1d %1d
camera-lid, camera-stg-led, cam amc led, cam HV, cam DC, cam bias, cam pixels temp, cam attenuators, cam cluster temp, cam sentinel, cam temp, cam humidity, cam LV, cam pulseinj
HV (1039x%04d)
HV readout [V]
DC (1039x%04.2f)
DC currents [uA]
BS (1039x%04d)
Bias [V]
PD (1039x%03.1f)
PD currents [mA]
TP (1039x%05.1f)
pixel temperature [deg]
AT (1039x%02d)
Attenuators readout [V]
TC (338x%05.1f)
cluster temp [deg]
TU (8x%05.1f)
camera temp [deg]
HU %04.1f %04.1f
Humidity
OVER

Chapter 25

MIR (Level 3 trigger and MAGIC-2 readout, trigger and calibration control)- CC

The communication uses socket connections (TCP/IP).

25.1 Commands to be received from CC

(to be written)

25.2 Data sent to CC

Every second, MIR should should send a report to CC (even if it is in stand alone mode, as long it is connected to CC) for each of the sub-subsystems which are controlled by the same program.

25.2.0.1 Data concerning the MAGIC-2 trigger

1. the string "M2" which specifies the telescope
2. status (...) according to table 17.3 (page 67)
3. .....
4. State of the camera lids...

25.2.1 Data concerning the MAGIC-2 trigger communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

25.2.2 MAGIC-2 trigger format definition

All this information is sent in one report:
Subsystem sends: TRIGGER-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %02d %03d
TRiGGER2-state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
TRIGGER2 comm state, year, month, day, hour, minuts, seconds, ms
%s %s
trigger2 table name, L1T2 table name
%d
multiplicity
%s
L2T2 table name
(8x(%s %03d))
presc fact description, presc factor (unknown, unknown,cal2, ped2, pulseinj, L1T2, unknown, unknown)
%d %d %d %d
life time LSB, life time MSB, dead time LSB, dead time MSB
(19x%f)
macrocells rate [Hz]
%f %f
dummy, dummy
(8x%f)
rates after prescaler [Hz]
(8x%f)
rates before prescaler [Hz]
%f %f %f %f
dummy, dummy, L1T rate, L2T rate
OVER

25.2.3 Data concerning the MAGIC-2 pulsar configuration

1. the string "M2" which specifies the telescope
2. status (...) according to table 17.3 (page 67)
3. .....
4. State of the....

25.2.4 Data concerning the MAGIC-2 pulsar configuration communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

25.2.5 MAGIC-2 pulsar configuration format definition

All this information is sent in one report:
Subsystem sends: PULSAR-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
PULSAR2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
PULSAR2 comm state, year, month, day, hour, minuts, seconds, ms
%*s*
pulsar table name
(56x%05.2f)
domino temp
(56x%05.2f)
domino voltage1
(56x%05.2f)
domino voltage2
(56x%05.2f)
domino power supply voltage
(56x%05.2f)
positive current mezzanine
(56x%05.2f)
negative current mezzanine
(56x%x)
domino status
OVER

25.2.6 Data concerning the MAGIC-2 calibration

1. the string "M2" which specifies the telescope

2. status (...) according to table 17.3 (page 67)

3. .....

4. State of the....

25.2.7 Data concerning the MAGIC-2 calibration communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

25.2.8 MAGIC-2 calibration format definition

All this information is sent in one report:
Subsystem sends: CAL-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
CALIB2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
CALIB2 comm state, year, month, day, hour, minuts, seconds, ms
(4x%02d)
temp TCU board [deg]
%04x %04x %04x
mask firing channels, mask triggering channels, mask direct triggering channels switch
%01d %d %01d %d %01d %d
cal switch, cal freq [Hz], ped switch, ped freq [hz], pulse inj switch, pulse inj freq [Hz]
%01d %03d %02d %02d %02x
calib box status, temp cal box [deg], filter wheel 1, filter wheel 2, mask relay
(starting from SA version 20090731-0)
%01d %d
cal laser status, minutes since cal box is on
OVER

25.2.9 Data concerning the MAGIC-2 receivers

1. the string "M2" which specifies the telescope
2. status (...) according to table 17.3 (page 67)
3. .....
4. State of the....

25.2.10 Data concerning the MAGIC-2 receivers communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

25.2.11 MAGIC-2 receivers format definition

All this information is sent in one report:
Subsystem sends:
RECEIVER-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
RECEIVERS2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
RECEIVERS2 comm state, year, month, day, hour, minuts, seconds, ms
%s %02d %s %02d %s %02d
dt table name, dt state, delay table name, delay state, width table name, width state
RECTEMP (84x%02d)
receiver2 temperatures [deg]
IPR (1039x%e)
IPR values
DT (1039x%05d)
discr threshold
(not always) TD (1039x%04d)
time delays
TW (1039x%04d)
pulse widths
OVER

25.2.12 Data concerning the MAGIC-2 readout cooling

1. the string "M2" which specifies the telescope

2. status (...) according to table 17.3 (page 67)

3. .....

4. State of the....

25.2.13 Data concerning the MAGIC-2 readout cooling communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

25.2.14 MAGIC-2 readout cooling format definition

All this information is sent in one report:
Subsystem sends: COOLING-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %02d %03d
COOLING2 state, year, month, day, hour, minuts, seconds, ms
%02d %04d %02d %02d %02d %02d %02d %03d
COOLING2 comm state, year, month, day, hour, minuts, seconds, ms
INTCOOLING %02d %03d %03d %03d EXTCOOLING %02d
int cooling state, 3 cooling crate temp [deg], ext cooling state
PSCRATES %02d %02d %02d %02d %02d %02d
cooling ps states
VOLT %05.2f %05.2f %05.2f %05.2f %05.2f %05.2f
cooling PS Volts
CURR %05.2f %05.2f %05.2f %05.2f %05.2f %05.2f
cooling PS currents
TEMP %02d %02d %02d %02d %02d %02d OVER
PS temp [deg]
OVER

25.2.15 Data concerning the L3 trigger cooling

1. the string "M2" which specifies the telescope
2. status (...) according to table 17.3 (page 67)
3. .....
4. State of the....

25.2.16 Data concerning the Level 3 trigger communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

25.2.17 Level 3 trigger format definition

All this information is sent in one report:
Subsystem sends: L3T-REPORT M2 %02d %04d %02d %02d %02d %02d %02d %03d
L3T state, year, month, day, hour, minuts, seconds, ms

%02d %04d %02d %02d %02d %02d %02d %03d
L3T comm state, year, month, day, hour, minuts, seconds, ms
%s
L3T table name
ANGLE %5.1f %5.1f %5.1f %5.1f
M1 altitude, M1 azimuth, M2 altitude, M2 azimuth
DELAY (4x%d)
(4x delay_in)
(4x%d)
(4x width_in)
(4x%d)
(4x delay_out)
MASK %x %x %x
pmask, cmask, tmask
COUNTS %d
trigger number
(4x%d)
(4x counter)
OVER

Chapter 26

MAGIC-1 and MAGIC-2 active mirror controls - CC

The communication uses socket connections (TCP/IP).

26.1 Commands to be received from CC

26.1.1 Commands concerning the Mirror control

1. Initialize AMC and set to Altitude 90.0, Azimuth 0.0, On completion AMC will go to state Initialized or Error.

   CC sends:

   ```
   "%.5s\n","INAMC"
   ```

2. Adjust mirrors using table. On completion AMC will go to state Adjusted or Error

   CC sends:

   ```
   "%.5s\n","ADJST"
   ```

3. Adjust mirrors using lasers. On completion AMC will go to state Laser Adjusted or Error

   CC sends:

   ```
   "%.5s\n","LSADJ"
   ```

4. Break actual command execution. If a command is being executed it will aborted and the AMC will go to stated Undefined or Reajust.

   CC sends:

   ```
   "%.5s\n","BREAK"
   ```

5. Not really defined till now what to do. Will set the Mirrors to Altitude 90.0 and Az 0.0, On completion AMC will go to state Parked.

   CC sends:

   ```
   "%.5s\n","SHUTD"
   ```

6. Turn mirror heating on.

   CC sends:

   ```
   "%.5s\n","HEAT+"
   ```

7. Turn mirror heating off.

   CC sends:

   ```
   "%.5s\n","HEAT-"
   ```

8. De-ice, i.e. heat up in "burst mode". If the heating is not turned onwhen deicing is requested it will be turned on and willl be kept on after deicing

   CC sends:

   ```
   "%.5s %04d\n","DEICE num_seconds_to_deice"
   ```

9. Break de-icing. Will only stop the deicing procedure but will keep the heating on.

   CC sends:

   ```
   "%.5s\n","BRDIC"
   ```

26.2 Data sent to CC

Every second, the mirror control should should send a report to CC (even if it is in stand alone mode, as long it is connected to CC). The report should contain:

26.2.0.1 Data concerning the Mirror control

1. the string "M1" or "M2" which specifies the telescope
2. status (error, parked, init...) according to table 17.3
3. number of panels installed, number of panels with errors
4. (Starting Arehucas version 050401:) "AMC focus" (0=AMC focussed to 10km, 1=AMC focussed to infinity, 2=AMC focussed to Roque lamp without corrections)
5. (Starting Arehucas version 060808:) The "AMC focus", which was actually never used, is now distance_to_10km_focus_position, i.e, deviation (in [mm]) from the standard focal length of 17000 mm (= 10km focussing), or what is to say, a value of '0' means default (correct) focus

26.2.1 Data concerning the communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

26.2.2 Format definition

All this information is sent in one report:
Subsystem sends:

```
("AMC-REPORT %s %02d %04d %02d %02d %02d %02d %02d %03d",
  telescope("M1" or "M2"), status, writeyear, writemonth, writeday,
  writehour, writemin, writesec, writemsec)

  (" %02d %04d %02d %02d %02d %02d %02d %03d",
  comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
  lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

  (" PANELS %03d %03d",
  num_panels_installed, num_panels_with_errors)

  Starting Arehucas version 050401, modified version 060808:
```

```
("  INF %d",
distance_to_10km_focus_position)

("  OVER\n")
```

Chapter 27

Lidar - CC

The communication uses socket connections (TCP/IP).

27.1 Commands to be received from CC

1. Autostart the lidar.

   CC sends:

   ```
   "%.9s\n","AUTOSTART"
   ```

2. Start a laser run (take lidar data). At the end of the run, the lidar sends a laser report to Arehucas through a special report (see below).

   CC sends:

   ```
   "%.8s\n","TAKEDATA"
   ```

3. Shut down the lidar.

   CC sends:

   ```
   "%.8s\n","SHUTDOWN"
   ```

27.2 Data sent to CC

Every second, the lidar sends a report to CC. The report contains:

27.2.0.1 Data concerning the lidar status

1. the string "M0" which indicates that this subsystem is not associated to a specific telescope (M1 or M2)
2. status (error=0, ok=4) according to table 17.3
3. status of the laser (error=0, ok=4)

27.2.1 Data concerning the communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

27.2.2 Format definition

All this information is sent in one report:
Subsystem sends:

```
("LIDAR-REPORT M0 %01d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d",
comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" LASER_STATUS %01d",
laser_status)

(" OVER\n")
```

27.2.3 Special Report

After a laser run, the lidar sends a special report with the digitized data. The format is as follows (the length of the array and the size of the elements is not defined yet):

```
("LASER-REPORT M0 %04d %02d %02d %02d %02d %02d %03d",
  year, month, day,
  hour, min, sec, msec)
```

```
("  2000 * %4d,
  data_samples)

  (" OVER\n")
```

Chapter 28

MAGIC-1 trigger - CC

The communication will use socket connections (TCP/IP).

28.1 Commands to be received from CC

28.1.1 Commands concerning the Trigger control

1. enable the trigger system
   CC sends:

   ```
   "%.5s\n", "START"
   ```

   The subsystem enables the output stage of the prescaler(s) and goes into the "Active" state, meaning that the trigger system is active and that triggers are sent to the FADC system.

2. disable the trigger system
   CC sends:

   ```
   "%.5s\n", "STOP!"
   ```

   The subsystem disables the output stage of the prescaler(s) and goes into the "Stopped" state, meaning that the trigger system is not active and that triggers are not sent to the FADC system.

3. load a trigger table
   CC sends:

   ```
   "%.5s\n", "LOADT TABLE_NAME"
   ```

The subsystem loads the trigger table into the trigger system and goes into the "Loading" state, meaning that the trigger system is busy, therefore not in an active state. When this operation is finished the system goes into the "Ready" state meaning that it is ready to accept the **START** command and enable triggers.

28.1.2 Commands concerning the Trigger monitor

1. enable trigger rates report
   CC sends:

   ```
   "%.5s\n", "RATES"
   ```

   The subsystem enables the periodic reading of the scalers counts, computes the rates and sends a report to the CC.

2. disable trigger rates report
   CC sends:

   ```
   "%.5s\n", "RATE!"
   ```

   The subsystem disables the periodic reading of the scalers counts and the reporting to CC.

28.2 Data sent to CC

Every second (typically), the trigger control should send a report to CC (even if it is in stand alone mode, as long it is connected to CC). The report should contain:

28.2.1 Data concerning the subsystem status

1. the string "M1" which specifies the telescope
2. status (error, idle, loading, ready, active, stopped) according to table 17.3 (page 67)

28.2.2 Data concerning the communication

1. status of the communication (1=ok, 2=problem)
2. time when the last command was received (UTC)

28.2.3 Data concerning the trigger

1. Trigger table name, L1 and L2 table names, multiplicity.
2. Unprescaled rates (Hz) on the 19 L2T macrocells.
3. Description of the prescaling fields and prescaling factors.
4. Livetime and deadtime (5 times; only the last one is valid for analysis).
5. L2T and L1T rates.
6. Prescaled rates (Hz) on the 19 L2T macrocells.
7. Individual Pixel Rates in Hz. Only the 340 pixels that are included in the trigger are reported. The pixels are not in regular spiral numbering but in receiver board numbering.
8. The same IPR but in spiral numbering and separated by blanks for all the 397 inner pixels.

28.2.4 Format definition

All this information is sent in one report:
Subsystem sends:

```
("TRIGGER-REPORT M1 %02d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d",
 comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
 lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" %s",
 trigger_table_name)

(" 32*%06.1f",
 dummy rates)

(" %s %d %s",
 L1_trigger_table_name, Trigger_Multiplicity, L2_trigger_table_name)

(" %s %d %s %d %s %d %s %d %s %d %s %d %s %d %s %d ",
description_prescaling_field_1, prescaling_factor_1....
description_prescaling_field_8, prescaling_factor_8)
```

```
     (" %d %d %d %d",
      livetime1_least_significant_byte, livetime1_most_significant_byte,
deadtime1_least_significant_byte, deadtime1_most_significant_byte)

......

     (" %d %d %d %d",
      livetime5_least_significant_byte, livetime5_most_significant_byte,
deadtime5_least_significant_byte, deadtime5_most_significant_byte)

     (" 20*%d",
blank, blank,
unpresc_bit0, ... unpresc_bit7
presc_bit0, ... presc_bit7
blank, blank)

     (" %d %d",
      L1T_rate_Hz, L2T_rate_Hz)

     (" 18*%d",
      dummy_rate0 .... dummy_rate17 )

     (" %05440x", individual_pixel_rates)

     (" 397*%09d",
      individual_pixel_rates)

     (" OVER\n")
```

Chapter 29

Burst Alarm - CC

If the communication between the CC and the Burst Alarm system is on, the *gspot* program continues to exchange reports with the main Central Control. The communication is managed by a TCP-IP socket connection.

29.1 Data received from the CC

Every 10 seconds the *gspot* gets a status report from CC as defined in 31.2. It gets them by a TCP-IP socket *server*. No commands have to be recived from the CC.

29.2 Data sent to the CC

The *gspot* sends to the CC data by a TCP-IP socket connection in two kind of reports: normal GRB-REPORTS and special GRB-ALARM reports.

29.2.1 Normal reports

The *gspot* sends every second a status report to the CC defined as follows:

```
("GRB-REPORT %02d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d\n",
comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" OVER\n")
```

The first part of the report tells the status of the system, concerning the monitoring of the GCN, with the current date. The second part of the report refers to the communication *gspot*-CC, i.e. to the CC *server* status as defined in 15.3. If there is a problem with the communication TO Central Control (i.e. a problem concerning the CC *client* status) it obviously cannot be reported, so the system is considered *unavailable*. The data sent with this report are:

- GCN status (see 15.3)
- current date (year, month, day, hours, minutes, seconds, milliseconds)
- CC *server* status (see 15.3)
- last CC-REPORT date (year, month, day, hour, minutes, seconds, milliseconds)

29.2.2 Special alarm reports

In case of an alert a special report is sent to the CC by the *gspot client*, and it is so defined:

```
("GRB-ALARM %02d %04d %02d %02d %02d %02d %02d %03d",
alertstatus, curryear, currmonth, currday,
currhour, currmin, currsec, currmsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d %06
   %02d %02d %05.2lf %02d %02d %05.2lf %06d\n",
comstatus, grbyear, grbmonth, grbday,
grbhour, grbminute, grbdsec, grbmsec,
difftime, grbRA_H, grbRA_M, grbRA_S,
grbDEC_D, grbDEC_M, grbDEC_S, coordERR)

(" OVER\n")
```

The data sent with this kind of report are the following:

- ALARM status (see 15.3)
- current date (year, month, day, hour, minutes, seconds, milliseconds)
- CC *server* status (see 15.3)

- GRB's onset date (year, month, day, hour, minutes, seconds, milliseconds)
- the time difference between current date and GRB's onset date in seconds
- GRB's Right Ascension J2000 (hours (int), minutes (int), seconds (double))
- GRB's Declination J2000 (deg (int), minutes (int), seconds (double))
- error on GRB's coordinates in arcsec.

Chapter 30

Auxiliary systems - CC

The communication will use socket connections (TCP/IP).

30.1 Commands to be received from CC

30.1.1 Commands concerning the Auxiliary systems

There are no commands to set the alarm limits. These are set in the subsystem itself and CC is informed about in special report.

1.

30.2 Data sent to CC

Every second, the auxsys control should should send a report to CC (even if it is in stand alone mode, as long it is connected to CC). The report should contain:

30.2.0.1 Data concerning the auxiliary systems

1. Status (error, parked, standby, slewing, tracking) according to table 17.3 (page 67)

30.2.1 Data concerning the communication

1. status of the communication (1=ok, 2=problem)

2. time when the last command was received (UTC)

30.2.2 Format definition

All this information is sent in one report:
Subsystem sends:

```
("AUX-REPORT %02d %04d %02d %02d %02d %02d %02d %03d",
status, writeyear, writemonth, writeday,
writehour, writemin, writesec, writemsec)

(" %02d %04d %02d %02d %02d %02d %02d %03d\n",
comstatus, lastcmdyear, lastcmdmonth, lastcmdday,
lastcmdhour, lastcmdminute, lastcmdsec, lastcmdmsec)

(" OVER\n")
```

30.2.3 Special Report

- Diferent alarm/warning levels

Chapter 31

Control output data format

The general rule is to transfer over TCP all the control data that is relevant for the physical analysis to **CC**. CC will save it to disk in the format defined below (section 31).

There is only **one additional control data file** which is saved locally by CaCo and this is only because the data throughput is large enough to make the TCP transfer cumbersome. This control file contains the pixel currents.

All the control subsystems generate **log files** for internal control and debugging. The format of these log files is not fixed but ASCII format in a fashion similar to the standard subsystem reports to the CC is advisable. CC also generates an ASCII log file with **.dbg** extension.

31.1 Main control, report or .rep file

CC generates a main file with control data. This file is stored in ASCII format to the *logbooks* directory. with a **.rep** (for "report") extension. A new file is created every time CC is started.

The name of the file has the following format:

```
("CC_%04d_%02d_%02d_%02d_%02d_%02d.rep", YYYY, MM, DD, hh, mm, ss)
```

where the recorded information is the UTC date and time: YYYY stands for year, MM for month, DD for day in the month, hh for hour (24 h), mm for minute, ss for second

CC saves control data to this file at startup and shutdown, and every 10 seconds at UTC second 05, 15... The control data consists of:

1. The last standard report of all subsystems in the format of the report.
2. The last special report of all subsystems in the format of the report.
3. A CC report (see 31.2).

The data rate to the CC report file is determined by the DC currents (4 bytes per channel), HV settings (4 bytes per channel), discriminator thresholds (1 byte per channel) and time delays (1 byte per channel). At 0.1 Hz this results in ∼550 bytes/s, that is, ∼2 MB/hour and ∼24 MB for a 12 h night.

31.2 CC status report

CC produces a report similar to the reports sent by the subsystems over TCP.

1. Status according to table 17.3 (page 67)
2. Status of all the M1 subsystems, $s1-62$, according to table table 17.3 (page 67).
3. Telescope current position in local and celestial coordinates ("telzenith", "telazimuth"...) as reported by the drive subsystem.
4. Mean temperature in °C, mean solar irradiation in Watt/m^2, mean wind speed in km/h and mean relative humidity in % from the weather station. This information is retrieved from *wwwint* through a file mounted over NFS.
5. Difference between the Rubidium clock time and the time provided by the GPS receiver, in μs. This information is readout through the second serial port of the CC computer.
6. Status of the computer system UPS (charge in arbitrary units - still not properly defined!). This information is retrieved from *wwwint* through a file mounted over NFS.
7. Information about the source which is scheduled at this particular time. Bear in mind that the observers may actually decided to observe a different object.
8. Status of all the M2 subsystems, according to table table 17.3 (page 67).

CC sends:
CC-REPORT M0 %02d %04d %02d %02d %02d %02d %02d %03d
CCstate, year, month, day, hour, minute, sec, ms
%01d
weather state (wind+humidity, 0=error, 4=ok, 7=warning, 8=alarm)
%01d %01d %01d %01d
DAQ1 state, DominoCalibration1 state, drive1 state, stg1 state
%01d %01d %01d %01d %01d
CaCo1 state, CaCo1 LID state, CaCo1 sentinel state, CaCo1 LV state, CaCo1 HV state

%01d %01d %01d %01d %01d %01d
AMC1 state, L2T1 state, pulsar1 state, receiver1 state, DT1 state, calib1 state
%01d %01d %01d
Lidar state, aux state, GRB state
%05.2f %05.2f %05.2f %05.2f
current M1 Zd [deg], current M1 Az [deg], req DEC [deg], req RA [deg]
%05.2f %05.2f %05.2f %05.2f %05.2f %05.2f
mean T, pressure, wind speed, mean hum, ups charge, Rub-GPS
SCHEDULE %s %d
sourcename, category
%01d %01d %01d %01d
DAQ2 state, DominoCalibration2 state, drive2 state, stg2 state
%01d %01d %01d %01d %01d
CaCo2 state, CaCo2 LID state, CaCo2 sentinel state, CaCo2 LV state, CaCo2 HV state
%01d %01d %01d %01d %01d %01d
AMC2 state, L2T2 state, pulsar2 state, receiver2 state, DT2 state, calib2 state,
%01d
Readout cooling state
%05.2f %05.2f
current M2 Zd [deg], current M2 Az [deg]
%s
LightConditions (“Moon”, “No_Moon”, “Twilight”, “Day”)
OVER

31.3 Electronic runbook or .rbk file

CC generates an electronic runbook with comments written by the operators and the most relevant data taking events, e.g. run start/stop. The electronic runbook file is stored in ASCII format to the *logbooks* directory with an **.rbk** extension. The operators enter information to the runbook file using a utility in the CC graphical interface. Comments are tagged with UTC date and time in the format [YYYY-MM-DD HH:MM:SS] and separated with empty lines.

31.4 Run summary or .run file

CC generates an ascii file with .run extension where every line corresponds to a run and contains global information about this run or the values of some parameters (rate, mean DC, etc) at the time the run ends. Every entry is separated by a space.

There is one separate file for each telescope. A short description of the fields can be found in the third line of the file. This line starts with the string "#FORMAT:" and is followed by a list of variable names. Starting on Arehucas version 111205-1 (after upgrading to new calibration boxes and new DRS4 readouts), the format for both telescopes is the same and as follows:

```
1 Tel number
2 Run number
3 Subrun number
4 Run type
5 Start date
6 Start time
7 Stop date
8 Stop time
9 Source name
10 Zd[deg]
11 Az[deg]
12 Number of events
13 Project name
14 L1 Trigger table
15 L2 Trigger table
16 Mean trigger rate [Hz]
17 L1 Trigger rate [Hz]
18 L2 Trigger rate [Hz]
19 DAQ Trigger rate [Hz]
20 DAQ Store Rate [Hz]
21 HV settings
22-25 Inner mean DC, RMS, max DC, pix with max DC
26-29 Outer mean DC, RMS, max DC, outer pix with max DC
30-33 Inner mean HV, RMS, max HV, pix with max HV
34-37 Outer mean HV, RMS, max HV, outer pix with max pix
38 MJD
39 Test run
40 Moon conditions
41 Discriminator thresholds table
42 Trigger delays table
43 Telescope_RA
44 Telescope_DEC
45 Observation mode
46 Source_RA
47 Source_DEC
48 DT mean
49 DT max
```

```
50 Pix with DT max
51 IPR mean
52 IPR max
53 Pix with IPR max
54 Sumt
55 wheel pos1
56 wheel pos2
57 L3 table name
58 L3T rate
59 cycle
60 PI
61 working group
62 proposal name
```

31.5 Run summary HTML or .run.html file

This file contains an HTML table with the same information as the .run file up to the HV settings ((Starting Arehucas version 041113) plus the calibration script name). The header of the file contains the description of the files.

31.6 DC current control file

CaCo generates an ASCII control file with the currents measured with 10 Hz frequency. The HV settings and discriminator threshold are not saved to this file.

The ASCII format of each entry (line) of the DC current files is:

```
("DC +%02d +%02d %02d %02d %02d %03d 577*%05d\n",
hvstatus1, hvstatus2, hour, minute, second, ms, dc-currents_nA)
```

The DC currents are saved with 2 byte precision at 10 Hz for the whole camera. This makes up ∼120 kB/s data rate.

Chapter 32

Computer names and TCP/IP port assigments

The computer names and IPs are defined on table 32.1. All the computers are members of the magic.iac.es (161.72.130.xx) subdomain in the IAC domain. The system is accessed only through www.magic.iac.es, IP number 161.72.130.130 (See note on IT standardization[4].)

The TCP ports for subsystem communication with CC are also tabulated. E.g. the aux system sends reports to the "report port" opened in the CC computer and receives commands through the "command port" opened locally. CC is added for completeness: it actually opens TCP ports complementary to the subsystem ports, e.g. CC sends commands to CaCo over the CaCo Cmd port 7406, which is opened remotely in CaCo; and receives reports from CaCo in the local port 7306.

Subsystem name	IP name	IP node	Report port (remote in CC)	Cmd port (local in subsystem)
CC (SuperArehucas)	pc1, superarehucas	161.72.130.1	-	-
SuperArehucas2	pc2, superarehucas2	161.72.130.2	-	-
AMC 1	pc7, amc	161.72.130.7	7307	7407
AMC 2	pc17, amc2	161.72.130.17	7317	7417
Calib 1	pc10, mir1	161.72.130.10	7341	7448
Calib 2	pc20, mir2	161.72.130.20	7321	7418
Camera Control 1	pc6, caco, caco1	161.72.130.6	7306	7406
Camera Control 2	pc16, caco2	161.72.130.16	7316	7416
DAQ M1	pc9, daq1	161.72.130.9	7309	7409
DAQ M2	pc19, daq2	161.72.130.19	7319	7419
Drive/Starguider 1	pc4, drive, drive1	161.72.130.4	7304	7404
Drive/Starguider 2	pc14, drive2, drivemagic2	161.72.130.14	7314	7414
Pulsar 1	pc10, mir1	161.72.130.10	7342	7448
Pulsar 2	pc20, mir2	161.72.130.20	7312	7418
Receivers 1	pc10, mir1	161.72.130.10	7348	7448
Receivers 2	pc20, mir2	161.72.130.20	7318	7418
SumTrigger 1	pc3, sum1	161.72.130.3	7303	7403
SumTrigger 2	pc13, sum2	161.72.130.13	7313	7413
Trigger 1	pc10, mir1	161.72.130.10	7344	7448
Trigger 2	pc20, mir2	161.72.130.20	7314	7418
Cooling	pc20, mir2	161.72.130.20	7320	7420
GRB	grb, www	161.72.130.130	7330	7430
Level 3 Trigger	pc20, mir2	161.72.130.20	7322	7418
Lidar	pc8, lidar	161.72.130.8	7308	7408
Pyrometer	pc5, pyrometer	161.72.130.5	7305	7405

Table 32.1: Computer names and IPs, and TCP report and command ports for communication with CC. The GRB monitor runs in www, the external router.

Bibliography

[1] Kernigan, B.W., & Ritchie, D.M., 1988, "The C programming language, 2nd ed.", Prentice-Hall.

[2] NI web page http://www.ni.com/labview and Labview User Manual available as part as the Labview software distribution.

[3] Simple Wrapper Interface Generator `http://www.swig.org`

[4] MAGIC-TDAS note xxxx, J. A. Coarasa, at the MAGIC general web page (http://hegra1.mppmu.mpg.de) and the control web page (http://magic.ifae.es/c̃ortina/control/welcome.html)

[5] MAGIC-TDAS 00-09/2, J. Cortina and J. A. Coarasa, "Description of the DAQ raw data format", at the MAGIC general web page (http://hegra1.mppmu.mpg.de).

[6] MAGIC MAGIC-TDAS 02-04, M. Gaug and T. Schweizer, "Calibration analysis", at the MAGIC general web page (http://hegra1.mppmu.mpg.de).