A Real-time Control Computer for the E-ELT

Document GF-PDR-03

Green Flash Requirements Specification

Version 1.1

17th January 2016

l'Observatoire de Paris
PLDA
MICROGATE
Durham University

Change Record

Version	Date	Author(s)	Remarks
0.1	14 Feb 2015	Nigel Dipper	Initial skeleton version
0.2	8 May 2015	Eddy Younger	Reformatted; updates from AGB, EJY
0.3	29 May 2015	Nigel Dipper	Updates from NAD
0.4	30 Oct 2015	Eddy Younger	Added derived performance requirements
0.5	4 Nov 2015	Nigel Dipper	Minor corrections
0.6	12 Nov 2015	Eddy Younger	Minor changes; added METIS performance requirements; added table of requirements and requirement numbering.
0.7	17 Nov 2015	Eddy Younger	Additional requirements identified in existing text
0.8	27 Nov 2015	Eddy Younger	Revised METIS performance requirements
0.9	17 December 2015	Eddy Younger	Comments from Roberto, Damien
0.10	17 December 2015	Eddy Younger	Corrections from NAD
1.0	8 January 2016	Eddy Younger	Renumber and release for PDR
1.1	17 January 2016	Eddy Younger	Reformatted

Table of Contents

Acronyms and abbreviations

Table 1 Acronyms and Abbreviations

AIT	Assembly, integration and testing
AO	Adaptive Optics
CORBA	Common Object Request Broker Architecture
CoDR	Conceptual design review
DDS	Data Distribution Service
DM	Deformable Mirror
DMC	Deformable Mirror Controller
E-ELT	European Extremely Large telescope
ESO	European Southern Observatory
FFT	Fast Fourier transform
(s)FPDP	(serial) Front Panel Data Port
FSM	Fast Steering Mirror
ICD	Interface Control Document
LGS	Laser Guide Star
NGS	Natural Guide Star
PDR	Preliminary Design review
RTC	Real-Time Computer
RTCS	Real-Time Control System
TCS	Telescope Control System
WFS	Wavefront Sensor

1 Scope

This document specifies the requirements for a prototype real-time control system (RTCS) for Adaptive Optics, at the scale of first-light instruments for the E-ELT. The functional- and performance requirements derive primarily from known or anticipated requirements of three particular instruments:

1. the MAORY multi-conjugate adaptive optics (MCAO) system designed to feed the MICADO first-light imaging camera. These requirements flow down from the science requirements of MICADO to the requirements for the MAORY AO system and thus to the RTCS.
2. The HARMONI integral-field spectrograph
3. The METIS mid-infrared imager/spectrograph

The prototype RTCS, whilst targeted at these instruments, aims also to provide the core of a system capable of meeting the requirements of the other major instruments proposed for the E-ELT. Some of the prototype requirements hence reflect its application to these other instruments.

The prototype RTCS will be a complete system. It will include a low latency ‘hard real-time’ computer, a ‘soft real-time’ computer that supports the calibration, configuration and control of the system, the interfaces between these elements, the interfaces to the wave-front sensor (WFS) cameras and to the deformable mirrors (DM) and all of the required software.

In addition, the RTCS must interface to the E-ELT telescope Control System (TCS). This interface is not yet well defined. The requirements for the interface flow down from the control strategy of the E-ELT active optics and other control systems and from the AO control strategy within the RTCS.

This initial draft combines both system requirements and user requirements.

2 Summary of instrument AO requirements

All instruments on the E-ELT are likely to require an AO system to achieve their science goals. Here we consider the known instrument requirements, and seek to map these onto a specification for a generic RTCS. The telescope deformable mirror is known as M4 and has a total of 5316 actuators of which 4326 are in the pupil.

Instrument	AO mode	LGS	NGS	DMs	Frame rate
Harmoni	SCAO LTAO	 4, 74x74 subaps	 1:	 1: M4	800Hz
Micado/ Maory	SCAO MCAO	 6, 74x74 subaps	 3: (low order – 8x8 ?)	 2 or 3: M4 + 1 or 2	500Hz
METIS	SCAO LTAO	- 6	3 (8x8 ?) 3 (8x8)	M4 M4	1000Hz
MOSAIC	MOAO	6: 74x74 subaps	5: 74x74 subaps	11: M4 + 10 off 64x64	250Hz
HIRES	GLAO	6: 74x74 subaps	1–3: (low order – 8x8 ?)	1: M4	500Hz
EPICS	XAO	No LGS	1: 200x200 subaps	2: M4 + 201x201	2 - 3kHz

Table 2.1: E-ELT instrument AO requirements

Clearly, the specific requirements of each of these instruments differ widely, in terms of the numbers of WFSs and DMs, the frame rate and the AO mode. A system designed specifically for Micado/MAORY will not meet the requirements of the other listed instruments. The optimum approach to the design of AO RTC systems for the E-ELT therefore will be to create a generic design/architecture which can support as many of these instruments as possible via a modular approach to both the core processing pipeline and the external interfaces to WFS and DMs. This generic system will be provided by GreenFlash and can be customised to suit individual instruments.

3 Hard Real-time Data Pipeline

This is the low latency data pipeline system, for which low jitter is essential. The requirements for jitter differ depending on the instrument being served.

R3.10 The real-time data pipeline must be capable of accepting all wavefront sensor information at up to the maximum frame rates, and processing this with the algorithms specified below, before

R3.20 delivering DM demands with a latency of less than 2 frame times between first pixel received and last actuator delivered.

R3.30 The design of the real-time pipeline must be modular such that the required performance (throughput, latency, and jitter) may be achieved by appropriate choice of hardware:

1. sufficiently powerful sequential computing units, as these become available
2. sufficient numbers of concurrent processing units/coprocessors operating in parallel

The implications are that the design should be highly parallel, with sequential bottlenecks reduced to a minimum (ideally eliminated).

4 Soft Real-time co-processor

R4.10 The soft real-time coprocessor shall receive telemetry data in real-time from the hard real-time data pipeline, and use this to compute updated, optimised parameters for the pipeline. Key functions of the soft real-time co-processor are

- **R4.30** matrix inversion to generate new control matrices, at a rate TBD
- **R4.40** calculation of phase covariance.
- **R4.50** offload of systematic low-order modes to the telescope active optics system

These are computationally demanding operations which need to be offloaded from the pipeline processor so as not to degrade its performance. They will call for substantial computational power.

Specific requirements for the soft real-time co-processor flow down from the specific instrument specifications.

5 Telemetry

R5.10 The hard real-time pipeline shall output diagnostic and statistical telemetry for the use of optimization and logging systems (see 5.2); these may be a part of the soft real-time co-processor, or may be separate systems.

R5.12 The telemetry data will consist of at least the following:

- Raw and calibrated pixels (temporally subsampled at a configurable rate) from all WFS
- Sub-aperture and frame statistics (total flux, …)
- Slopes
- Algorithm parameters
- DM demands
- Status

5.1 Data Transport and Middleware

The hard real-time pipeline requires a very low latency, very low jitter data transport between its modules in order to achieve the top-level requirements. Minimum complexity and overhead is essential here. Since this is essentially point-to-point transport between functional modules of the pipeline, complex middleware is not required and so a simple high bandwidth low-latency transport is preferred, such as shared memory, OpenMPI or simple UDP sockets.

R5.14 Telemetry shall be published using DDS middleware. However, the middleware will be abstracted from the data sources and consumers and can be replaced with an alternative if such is identified or required.

Middleware for control and status is TBD. It could be DDS; more likely to be CORBA. Again, it is a requirement that the control and status middleware is well abstracted.

5.2 AO Data Capture and Recording

R5.20 A hardware/software system is required which is able to capture and record all AO telemetry data. This is quite separate to the recording of science data. The most demanding case in terms of performance requirements is to capture all possible telemetry sources undecimated at the maximum frame rate. The maximum bandwidths required are therefore instrument specific.

R5.22 The ability to record contiguous frames of undecimated data for a period of the order of 10s is a requirement, particularly for calibration purposes. This will dictate a buffer size within the data capture system.

6 Modularity

Green Flash is an experimental system designed to evaluate a number of technologies and algorithms, including emerging technologies not available at the time of writing. As such, its design is required to be modular, with standardized interfaces which facilitate the replacement of functional components.

Critical E-ELT interfaces such as the instrument-TCS interface are not yet defined or known. It is therefore essential that there is minimal coupling between these interfaces and the RTC core software in order for design and development of this software to proceed without the risk of extensive rework in the future.

R6.10 The real-time pipeline core logic shall be decoupled from the middleware.

R6.20 Interfaces with (control- and telemetry) middleware shall be implemented as modules. The interfaces between these modules and the core software shall be a) minimal; and b) abstracted, in the sense that details of the hardware/middleware are hidden from the core software.

R6.30 It shall be possible to replace a software module with another which uses a different algorithm to perform the same basic function; for example, to replace the reconstructor specified in this document with an experimental algorithm, or to replace an input or output module in order to interface with alternative hardware. The inter-module interfaces shall allow replacement of modules possible without requiring a full rebuild of the system.

R6.40 The following components (at least) shall be implemented as modules:

1. Data pipeline input module
2. Pixel calibrator (sect. 8.2)
3. Windowing algorithm (sect. 8.2)
4. Centroider (sect. 8.2)
5. Reconstructor (sect. 8.3)
6. DM drive temporal filter (sect. 8.3)
7. DM drive output module

It shall not be necessary to recompile/rebuild the full software in order to replace components at this specified level of modularity.

7 Interfaces

7.1 General

R7.10 Hardware and software interfaces in Green Flash will conform as far as possible/practical to evolving ESO/Sparta standards.

R7.12 The inter-module data format for all data pipeline modules shall be the SPARTA frame format, unless otherwise stated. The frame number shall be retained throughout the pipeline as the means to relate the data at all stages to the initial pixel frame from which it is derived.

7.2 WFS to RTC Interface

This interface is responsible for the input of data to the hard real-time data pipeline. The WFS cameras are expected to have onboard pixel processing capabilities, and may be capable of producing raw pixels, calibrated pixels, pre-processed (e.g. thresholded) pixels or even centroids. This 'intelligent interface' may be located in the cameras or in hardware within the hard real-time computer.

R7.20 The hard real-time data pipeline shall be configurable to accept raw pixels, calibrated pixels, or centroids.

R7.22 This interface shall be modular to enable different types of WFS camera to be connected, including non-intelligent directly-connected cameras.

R7.24 As a minimum, the following camera types should be suported/supportable: Cameralink, sFPDP, AIA, GbE and 10GbE.

7.3 RTCS to DM Interface

R7.30 The RTCS shall generated demands to drive one or more DMs. One of these will be the E-ELT M4 mirror; instruments may contain their own DM(s) in addition to this. This interface shall be modular, to enable support for one or more of:

1. Interface to the telescope M4 (defined by ESO)
2. DAC card(s) contained within the RTC itself
3. External drive electronics, connected via (for example) ethernet or fpdp
4. Separate intelligent DM controller(s) (DMCs)

This module shall accept demands from the RTCS core in SPARTA frame format (see **R7.12)** and turn these into output suitable for the specific DM being driven – either commands to an external mirror controller or calls into the driver for a locally-hosted DAC card.

7.4 Telemetry Interface

The hard real-time data pipeline shall publish diagnostic, statistical and telemetry data to the soft real-time system as described in section 5 (See **R5.10** , **R5.12**). This data shall be published actively and asynchronously by the data pipeline.

Telemetry data shall be published via DDS (see **R5.14**).

R7.40 Each telemetry data sample shall contain a frame number. In the case of status data, the sample should contain the frame number of the last frame handled.

R7.42 Data shall be identified and timestamped in such a way that it is possible to relate calculated quantities (slopes, DM demands etc) to the input data and parameters from which they were generated, and also to contemporaneous science observations.

7.5 Control, Configuration and Status Interface

R7.50 The hard real-time system shall provide an interface and an API for configuration, calibration, control and status reporting. Configuration, calibration and control shall be possible via

1. An engineering GUI running on the hard real-time host itself (see **R10.10**).

2. A control interface which is able to receive commands from remote sources.

Status reporting shall be via the engineering GUI and by status messages transmitted via the telemetry interface as follows:

1. **R7.54** Regular status messages at a user configurable frequency. These shall be identified by the frame number of the last frame fully processed by the pipeline. They shall contain all relevant status for each component of the pipeline together with overall status information for the system as a whole (e.g. "control on", "loop closed", etc.). The frequency of these massages shall be configurable via the command interface.
2. **R7.56** Asynchronous Status Changed messages - these shall have the same format as periodic status messages, and shall be generated whenever the state of the hard real-time system changes due to an external event or a command.

Command set

The hard real-time system shall provide an interface and API for the execution of commands from a local GUI (for testing) or from a remote client (see **R7.50).**

R7.60 Sending commands to the hard real-time system shall not increase the latency of the data pipeline. It should be possible to send all commands whilst the pipeline is running.

R7.62 Any command that affects the data pipeline shall be applied to a full frame and never to a partial frame. Thus all data pipeline parameters (such as a reconstructor matrix) must be double-buffered so that the data pipeline only uses the new data when its upload is complete.

In this context, it should also be possible to:

- Load to the buffer NOT being used by the pipeline and switch buffers between frames
- Load new data to a specific buffer (A or B) that is NOT being used by the pipeline
- Switch buffers without upload of new parameters (useful to switch rapidly between 2 values, matrices, etc.)

R7.64 There shall be software locks to ensure that a second command will not be processed whilst a command is still executing.

R7.66 Since commands are thus 'blocking', there shall be an abort command and a timeout for every command.

R7.68 On completion (or timeout or abort), commands shall return an acknowledgement with the status (success, timeout, error etc) and the last frame number. Every command resulting in a change of status shall result in the transmission of a status frame with the new status, including the last frame number, via the telemetry stream as stated by **R7.56**.

8 Functional Requirements and Specifications

8.1 Top-Level Requirements

The RTC shall accept pixel input from the wavefront sensors, and produce output to drive DMs (see **R7.20**, **R7.30).**

8.2 Wavefront Processing

Wavefront processing comprises all functions concerned with processing raw pixels from the wavefront sensors into slopes.

R8.10 Wavefront processing shall be modularised such that the following implementations are possible:

1. Wavefront processing implemented entirely in CPU. The hard real-time system will receive raw pixel data from the Cameras, and shall implement all required wavefront processing functionality.
2. Wavefront processing in CPU with accelerator support. As above but some or all processing offloaded to accelerators to reduce latency.
3. Centroid and slope calculation in CPU (with or without coprocessors). The pixel calibration will be performed in the intelligent camera interfaces, and the calibrated pixels fed to the hard real-time system.
4. Wavefront processing entirely in intelligent camera interfaces. The slopes will be calculated entirely by the cameras or by the intelligent interface module in the RTC box.

Pixel calibration

Pixel calibration shall be performed by the data pipeline prior to calculation of centroids, and will consist of the following steps:

- Map pixels to subapertures
- Bad-pixel handling
- Background image subtraction
- Flat-field (gain) compensation
- Thresholding (noise reduction)
- Pixel weighting
- Apply global power factor - (pixel intensity)^n

R8.11 Calibration procedures shall be implemented which are able to generate bad-pixel maps, flat-field and background images, and pixel thresholds for each detector.

R8.12 It shall be possible to update pixel calibration parameters in real time without interrupting the data pipeline, introducing additional latency, or causing data corruption.

Thresholding

The classical way to compute the centre of gravity is first to apply a threshold, in order to reject some of the noise.

Two thresholding algorithms exist, which are not equivalent:

Algorithm 1:

```
if ( value<threshold ) value=0
```

Algorithm 2:

```
value = value – threshold
if ( value<0 ) value=0
```

Both algorithms (1 and 2) should be implemented.

R8.13a It shall be possible to select which of the 2 algorithms is used when the data pipeline starts.

R8.13b It shall be possible to switch between the two algorithms whilst the data pipeline is running, without interrupting the pipeline.

Centroiding and slope calculation

R8.14a The baseline centroiding algorithm shall be weighted centre-of-gravity.

R8.14b Correlation centroiding, either directly or by FFT, shall be implemented as an alternative algorithm.

R8.15 It shall be possible to replace this algorithm with other algorithms, as described in sect. 6.

WFS Windowing

Open-loop operation of a WFS requires a special treatment of the computation of the center of gravity of the Hartmann spots. In closed loop operation, the spot is supposed to remain close to the centre of the window in which the computation is done. In open-loop however, the WFS spot images are expected to move significantly over time relative to the detector.

Notice that the notion of window was inherited from the early ages of adaptive optics, when real-time computers were not sufficiently powerful to achieve complex calculations on the whole image. This notion of window allows the computation to be done in a parallel way. Now, it becomes more and more necessary to update this method, and a variety of other algorithms can be imagined:

Basic Windowing

In this method, the centroid computation is done within a fixed window for each spot. This is the method used in SPARTA.

It should however be noted that the size of a window must be independent from the pitch of the windows. Eventually, the size of a window may be larger than the window pitch, so that a given pixel may belong to several windows. This situation cannot be handled by SPARTA.

R8.16 Fixed windowing as described shall be implemented.

R8.17 The size of the window shall be independent of the window pitch.

Adaptive Windowing

The difference from the basic windowing is that windows may be moved all together with the average wavefront slope. Again as for basic windowing, there is a requirement for the size of a window to be independent from the pitch of the windows.

There are some dangers with this method:

WARNING 1: this procedure may be unstable in some cases.

WARNING 2: As the boxes are moving, it is important that the centres of gravity of the spots are expressed in a coordinate system that *does not move* with the windows, but which is fixed with respect to the pixels of the CCD.

R8.18 Adaptive windowing as described shall be implemented.

R8.19 It shall be possible to define windows such that

- They are non-square (i.e. rectangular)
- They are defined by their top-left pixel coordinate and their X and Y extent
- Their geometries are different for each subaperture

R8.20 Temporal filtering of window position updates shall be implemented to avoid instabilities.

R8.22 It is expected that the windowing algorithms used will be AO mode and instrument specific. The algorithms must be readily replaceable as described in sect. 6.

Reference Slopes

R8.24 The soft real-time system shall compute the reference slopes to be used for residual slope calculation. These will be averages of a large number (100 to 1000) of consecutive slope measurements. These reference slopes will be calculated in the soft real-time system based on the slope telemetry data transmitted from the hard real-time system. The reference slope values will be updated by external command. The update rate for reference slopes will depend on the frame rate but is likely to be in the range 0.5 to 10 Hz.

R8.26 The slope data in the telemetry stream must be sub-sampled so that data from contiguous frames can be used to calculate the reference slopes.

R8.28 The hard real-time system shall provide an interface to update reference slopes in real time without interrupting the pipeline, causing data corruption, or introducing additional latency.

8.3 Reconstruction

Reconstructor Algorithms

The data pipeline shall use the calculated slopes to reconstruct the incoming wavefront.

R8.30 The design of the reconstructor stage shall allow the actual reconstructor algorithm to be changed without change to the interfaces with the slope calculation stage and the DM control stage; this will allow alternative algorithms to be used without a redesign of the data pipeline.

Calibration

R8.32 It must also be possible to perform manual calibration procedures in order to calculate interaction matrices for the mirror(s). As a minimum it shall be possible to upload a DM drive vector from a file, and to drive a single actuator on demand. Calculation of the interaction matrices will be performed by the soft real-time system using telemetry data from the hard real-time system.

Filtering, clipping and anti-windup

R8.34 Temporal filtering of the DM demands generated by the reconstructor shall be performed.

R8.36 The exact nature of this filtering is not fixed at this time; it is likely that several algorithms will be required depending on AO mode and instrument. The hard real-time software design must facilitate the use of alternative filtering algorithms, as described in sect. 6.

R8.37 Anti-saturation shall be performed on the demands sent to the DM control stage, but not on the calculated values used in recurrence relations used to calculate subsequent demand vectors. This will ensure that the DM demands remain within a safe range. It shall be possible to enable/disable anti-saturation controls. The nature of the anti-saturation control is dependent on the actual mirror technology, and may differ for force-limited and voltage-limited mirrors. Saturation management for the E-ELT M4 is handled by the M4 mirror drive system, which is a telescope system external to the instrument RTCS

R8.38 The number of actuators that are in saturation must be recorded. If the number is greater than a configurable maximum, an error shall be generated.

R8.39 Anti-windup shall be implemented for the DMs. The most basic algorithm attempts to maintain the average actuator value at the middle of the allowed range. More advanced DM control algorithms (such as Kalman or H_∞) may well remove the need for anti-windup algorithm. It shall be possible to enable/disable this processing via the command interface.

8.4 Optimisation

R8.40 The data pipeline shall output (optionally temporally sub-sampled) pixel, centroid, slope and DM drive data in real time to the soft real-time system as described in sect 5.

R8.42 The soft real-time system will use this data to optimise the parameters used by the data pipeline, including but not limited to reference slopes, reconstructor matrices, temporal filter and predictive control parameters.

R8.44 The hard real-time system shall provide an interface to update the calibrator, centroider and reconstructor parameters in real time, without stopping the data pipeline or introducing additional latency.

9 Performance Requirements

The latency requirement **R3.20** is 2 frame periods between arrival of first pixel to delivery of last actuator value (with a goal for Green Flash of 1 frame period). All processing and data transfers for each frame must be completed within this time. The pipeline processing consists of the following stages; these need not and should not be sequential, in the sense that each stage may begin processing a frame before the previous stage has finished processing it.

1. **Pixel calibration.** The computation for pixel calibration scales as $O(n_{pixels})$, and each pixel *may* be calibrated in parallel. This stage can begin *in principle* when the first pixel is received from the camera.
2. **Slope calculation.** For weighted-CoG method, computation is $O(n_{pixels})$ (or $O_{subapertures}$ when a "N brightest pixels" calculation is used), and each subaperture (for a Shack-Hartmann WFS) *may* be processed in parallel. *In principle* this stage may begin when the first calibrated pixel is received from the calibration stage (Shack-Hartmann, though typically it begins when the first full calibrated subaperture is received), or when a half-frame of calibrated pixels is received (Pyramid WFS).
3. **Reconstruction.** For a MVM reconstructor, computation is $O(n_{subapertures} \times n_{actuators})$; there are a variety of approaches to parallelising and otherwise optimising MVM. In principle this stage may begin when the first slope is received from the previous stage.
4. **DM commands delivery.** Generally this cannot happen until the reconstruction stage is complete. In particular for the E-ELT, commands to drive M4 are delivered as Zernicke coefficients which *cannot* be calculated fully until the complete reconstructed wavefront is available.

Of these processes, reconstruction is generally the most computationally demanding for systems with large DM command vectors. In the calculations below, we allow half the latency budget for the reconstruction (1 frame time *required*, ½ frame time *goal*). It is presumed that the data transfer between computation stages is small relative to the computation – zero-copy methods used wherever possible.

9.1 Pipeline frequency

The pipeline frequency is the same as the WFS frame rate and DM update rate.

9.2 Pixel Rates

Pixel rates can be derived from frame rate and detector sizes.

9.3 Latency

The required maximum latency from the arrival of the last pixel of an iteration to the beginning of DM demand output, is 2 frames (with a goal of 1 frame).

9.4 Jitter

The maximum acceptable jitter in the latency value is not yet known and will be instrument specific. ESO suggest an upper bound of 10% of frame period. The goal for Green Flash is 100uS peak-to-peak jitter in the latency over a 1 second period.

9.5 Power consumption

Minimising power consumption is a design goal for H2020 projects. Minimising the power requirements of Green Flash will reduce the power-budget for any instruments which make use of the Green Flash design and hence also reduce the cooling requirements. Since these instruments are installed at altitude in remote locations, both power requirements and cooling are important considerations.

9.6 Derived performance requirements

Performance requirements for the various E-ELT instruments may be derived from the information in Table 2.1. For each instrument, the most demanding case is taken in the case where a range of possibilities is quoted.

- As stated above, the latency requirement is taken as 2 full frames.
- The data rate is taken as $framerate * pixelbytes * \sum WFS\ pixels$, where each LGS WFS is assumed to be 1600x1600 pixels at 2 bytes/pixel, and each NGS WFS is assumed to be 800*800 pixels at 2 bytes/pixel
- MAC performance assumes MVM reconstructor, and is calculated assuming 74x74 subapertures per LGS WFS, and 8x8 subapertures per NGS WFS unless shown otherwise in Table 2.1, and 75x75 actuators for M4:
  MAC/s = $framerate * (2 * \sum subapertures) * \sum DMactuators$

Table 9.1 shows the derived requirements for each of the instruments listed in Table 2.1:

Instrument	*Frame Rate (Hz)*	*Latency (mS)*	*Jitter (uS)*	*Pixel Rate (Gb/s)*	*TMAC/s for MVM*
HARMONI	800	2.5	125	139.3	0.2
MAORY/ MICADO	500	4	200	138.2	0.46
METIS SCAO	1000	2	100	10.2	2.2 GMAC/s
METIS LTAO	1000	2	100	194.6	0.37
MOSIAC	250	8	400	74.2	1.4
HIRES	500	4	200	138.2	0.28
EPICS	3000	0.667	33	122.9	11.05

Table 9.1: derived performance requirements for E-ELT instruments

Performance requirements for the XAO EPICS instrument are included, however the Green Flash prototype will not attempt to meet these. The design should however be scaleable such that these requirements *could* be met by a larger-scale implementation of the prototype design.

The goal for the Green Flash prototype RTC is to meet and ideally exceed the most demanding of these instrument requirements (with the exception of EPICS as noted). Table 9.2 lists the performance design goals for the prototype.

	Required	*Goal*
R9.10 Frame Rate (Hz)	800	1000
R9.20 Pixel Rate (Gb/s)	200	250
R9.30 TMAC/s	1.5	3.0
R9.40 Max. Latency (mS)	2.0	1.0
R9.50 Max. Jitter (uS)	100	100

Table 9.2: Green Flash prototype performance requirements

10 User Interface

R10.10 An engineering graphical user interface (GUI) shall be provided with (at least) the following capabilities:

- **R10.20 I**t shall be possible to start, stop and pause (and then restart) the hard real-time data pipeline.
- **R10.30** It shall be possible to initialize and update all operational parameters.
- **R10.40** It shall be possible to visualize all status information.
- **R10.50** It shall be possible to visualise any real-time/diagnostic telemetry stream (suitably sub-sampled) produced by the hard real-time system.
- **R10.60** It shall be possible to export the GUI display to a remote workstation (possibly with some loss of real-time visualization capability).

R10.70 In addition to this engineering GUI, a remote user GUI will be produced. This will make use of the command interface to the hard real-time system described in sect. 7.5.

11 Stability and Robustness

R11.10 The RTC shall detect and reject invalid/erroneous data from all sources. Warnings shall be issued when this occurs.

R11.20 The RTC shall reject commands from the user which are not valid in its (the RTC's) current state; for example, if it has not been fully initialised.

R11.30 No sequence of user commands shall cause the RTC to crash or to enter an invalid or inconsistent state. It shall be robust to receipt of an arbitrary sequence of characters from a controlling terminal or equivalent source of input.

R11.40 The RTC shall be robust to disconnection or failure of any input during operation. Such events shall be detected by the RTC which shall continue operation if possible (in a degraded state if necessary), or terminate gracefully if continuation is not possible. Warnings shall be issued when such events occur.

12 Table of Requirements

This section summarises the Green Flash requirements in table form. The requirements are fully described in the sections of this document referenced by the ***Source*** column.

Requirement No	***Source***	***Description***
R3.10	Section 3	Real-time pipeline shall accept and process WFS data at the full frame-rate
R3.20	Section 3	Latency shall be <= 2 frame periods
R3.30	Section 3	RTC design shall be modular such that performance goals may be achieved by appropriate hardware choices
R4.10	Section 4	Soft-real-time processor shall consume real-time telemetry data from the hard-real-time pipeline
R4.30	Section 4	Soft real-time processor shall perform matrix inversion to compute control matrices
R4.40	Section 4	Soft real-time processor shall compute phase covariance
R4.50	Section 4	Offload low-order modes to telescope
R5.10	Section 5	Hard-real-time telemetry
R5.12	Section 5	Hard-real-time telemetry data streams
R5.14	Section 5.1	Telemetry shall be DDS-based
R5.20	Section 5.2	Telemetry data recording
R5.22	Section 5.2	Capability to record undecimated data from all sources for an (as-yet-unspecified) time period
R6.10	Section 6	RTC pipeline core shall be decoupled from the middleware

Requirement No	*Source*	*Description*
R6.20	Section 6	Interfacing with middleware shall be implemented as modules with middleware-independent interfaces to the core.
R6.30	Section 6	Core pipeline functions shall be modularised
R6.40	Section 6	Minimum acceptable level of modularity
R7.10	Section 7.1	Interfaces shall conform to emerging ESO E-ELT interfaces as far as possible
R7.12	Section 7.1	Real-time pipeline inter-module data format shall be SPARTA frame format
R7.20	Section 7.2	RTC shall be configurable to accept raw pixels, calibrated pixels or centroids as input
R7.22	Section 7.2	RTC input interface shall be modularised to enable different camera types to be used
R7.24	Section 7.2	Cameralink, sFPDP, AIA, GbE and 10GbE cameras shall be supported
R7.30	Section 7.3	RTC output to DM(s) shall be modular, to support the listed devices (at least)
R7.40	Section 7.4	All telemetry frames shall contain a frame number
R7.42	Section 7.4	All telemetry frames shall be timestamped and all related frames shall be identifiable
R7.50	Section 7.5	RTC shall provide interface/API for command/control/configuration/status reporting
R7.54	Section 7.5	RTC shall report status regularly via GUI and telemetry
R7.56	Section 7.5	RTC shall report status changes asynchronously

Requirement No	*Source*	*Description*
R7.60	Section 7.5	Sending commands to hard-RTC shall not increase latency
R7.62	Section 7.5	Any command affecting data pipeline shall be applied only to full frames
R7.64	Section 7.5	There shall be software locks to serialise commands
R7.66	Section 7.5	Each command shall have a timeout; there shall be an "Abort" command
R7.68	Section 7.5	Commands shall return acknowledgement on completion
R8.10	Section 8.2	Wavefront processing shall be modularised
R8.11	Section 8.2	Calibration procedures to generate pixel calibration data shall be implemented
R8.12	Section 8.2	It shall be possible to update pixel calibration parameters whilst pipeline is running
R8.13a	Section 8.2	It shall be possible to select thresholding algorithm at startup
R8.13b	Section 8.2	It shall be possible to switch thresholding algorithms whilst pipeline is running
R8.14a	Section 8.2	Baseline centroiding algorithm shall be weighted centre-of-gravity
R8.14b	Section 8.2	Correlation centroiding shall be implemented
R8.15	Section 8.2	Different centroiding algorithms shall be supportable
R8.16	Section 8.2	Fixed windowing shall be implemented
R8.17	Section 8.2	Window size shall be independent of pitch

Requirement No	***Source***	***Description***
R8.18	Section 8.2	Adaptive windowing shall be implemented
R8.19	Section 8.2	Non-square windows shall be supported
R8.20	Section 8.2	Temporal filtering of window positions
R8.22	Section 8.2	Different windowing algorithms shall be supportable
R8.24	Section 8.2	Soft real-time system shall compute reference slopes
R8.26	Section 8.2	Slope telemetry data must provide contiguous frames for reference slope computation
R8.28	Section 8.2	Reference slopes shall be updateable whilst the pipeline is running
R8.30	Section 8.3	Different reconstructor algorithms shall be supportable
R8.32	Section 8.3	Manual DM calibration procedures shall be supported
R8.34	Section 8.3	Temporal filtering of DM demands
R8.36	Section 8.3	Different temporal filtering algorithms shall be supported
R8.37	Section 8.3	Anti-saturation control for DM actuator drive
R8.38	Section 8.3	Number of DM actuators in saturation shall be recorded
R8.39	Section 8.3	Anti-windup algorithm for DM actuator drive
R8.40	Section 8.4	Hard real-time pipeline shall publish telemetry in real time.
R8.42	Section 8.4	Soft real-time processor shall compute pipeline optimisations in real-time

Requirement No	*Source*	*Description*
R8.44	Section 8.4	It shall be possible to update the hard real-time pipeline parameters in real time without introducing latency
R9.10	Table 9.2	Frame rate >= 800Hz
R9.20	Table 9.2	Input pixel data rate >=200Gb/s
R9.30	Table 9.2	MVM performance >= 2TMAC/s
R9.40	Table 9.2	Max latency <= 2mS
R9.50	Table 9.2	Max. jitter <= 100uS
R10.10	Section 10	An engineering user interface (UI) shall be provided
R10.20	Section 10	It shall be possible to start, stop, pause and restart the real-time pipeline via the engineering UI
R10.30	Section 10	It shall be possible to initialise all parameters via the engineering UI
R10.40	Section 10	It shall be possible to visualise all status information via the engineering UI
R10.50	Section 10	It shall be possible to visualise any real-time or diagnostic telemetry stream via the engineering UI
R10.60	Section 10	It shall be possible to export the engineering UI to a remote workstation
R10.70	Section 10	A remote user GUI shall be produced
R11.10	Section 11	RTC shalll reject invalid/erroneous input
R11.20	Section 11	RTC shall reject invalid user input and commands
R11.30	Section 11	No sequence of user commands shall crash the RTC

Requirement No	*Source*	*Description*
R11.40	Section 11	RTC shall be robust to, and report, loss of inputs