Programme: Technology Development

Project/WP: Common Instrument Systems Development

ELT Instrument Adaptive Optics Real-Time Computer - Real-Time MUDPI Stream Protocol

Document Number: ESO-310635

Document Version: 1

Document Type: Specification (SPE)

Released On: 2018-06-28

Document Classification: ESO Internal [Confidential for Non-ESO Staff]

Released by ESO PDM SYSTEM
Digitally signed by Released by ESO PDM SYSTEM
Date: 2018.06.28 16:30:45 +02'00'

Prepared by:	Suárez Valles, Marcos
Validated by:	Downing, Mark
Validated by:	Marchetti, Enrico
Approved by:	Casali, Mark
	Name

Authors

Name	Affiliation
Marcos Suárez Valles	ESO

Change Record from previous Version

Affected Section(s)	Changes / Reason / Remarks
All	First version of the document.

Contents

1. Introduction

1.1 Scope

This document describes the Real-Time MUDPI Stream (RTMS) protocol to be used for low-latency, deterministic communication by the Adaptive Optics (AO) Real-Time Computer (RTC) Hard Real-Time Core (HRTC) of the European Extremely Large Telescope (ELT) First Light Instrument's.

RTMS is applicable to the following AO RTC communication channels:

- Communication between the HRTC and the AO Wavefront Sensors (WFS) over the AO RTC Real-Time Network -i.e. reception of real-time pixel image data.
- Communication between the HRTC and the ELT Central Control System (CCS) over the ELT Deterministic Network for the purpose of commanding M4 and M5 - i.e. transmission of real-time command data and reception of echo signals.
- Communication between the HRTC and the internal Instrument's Corrective Optics actuators over the AO RTC Real-Time Network -i.e. transmission of real-time command data and possibly reception of echo signals.

In addition, RTMS may be used for internal communication amongst the various computing units in the HRTC Real-Time Pipeline -i.e. transmission and reception of intermediate computing pipeline results within a control loop cycle.

This document provides the protocol's generic packet layout common to all applications. The detailed packet descriptions for communication to the relevant ELT and Instrument sensors and actuators is provided by the corresponding ICD documents

1.2 Protocol Version

This document describes RTMS at its version 0.1. It is based on Multicast UDP Interface (MUDPI) at version 1.

1.3 Definitions, Acronyms and Abbreviations

This document employs several abbreviations and acronyms to refer concisely to an item, after it has been introduced. The following list is aimed to help the reader in recalling the extended meaning of each short expression:

AO	Adaptive Optics
B	Byte
CCS	Central Control System
DM	Deformable Mirror
ELT	European Extremely Large Telescope
HRTC	Hard real-Time Core
IP	Internet Protocol
M4	ELT Quaternary Mirror
M5	ELT Tip-Tilt Stabilisation Mirror
MTU	Maximum Transmission Unit
MUDPI	Multicast UDP Interface
OSI	Open Model Interconnection
PDU	Protocol Data Unit
RTC	Real-Time Computer
RTMS	Real-Time MUDPI Stream

TBC	To Be Confirmed
TBD	To Be Defined
UDP	User Datagram Protocol
WFS	Wavefront Sensor

Frame/packet: the terms *packet* and *frame* are used throughout this document to refer to Protocol Data Units (PDU) as follows:

- The term *packet* is used in connection to PDUs at the Network layer or above.
- The term *frame* is used in connection to PDUs at the Data Link Layer or below.

The Open System Interconnection (OSI) model is assumed.

Data sample: a data sample is a data item of either scalar or array type. It is comprised of one (i.e. scalar case) of more (i.e. array case) individual numerical values, hereafter referred to as **payload data units**.

2. Related Documents

2.1 Applicable Documents

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

AD references shall be specific about which part of the target document is the subject of the reference.

AD1 MUDPI Standard

ESO-302020 Version 3

3. Overview

RTMS is built on top of MUDPI (see AD1) to address specific needs of data propagation as part of real-time processing pipelines in a standardised way:

- Signalling the start/stop of data samples independently of in-band data transmission, thus enabling preparation/clean-up actions on receiver's side.
- Providing out-of-band data encoding sender's status information, thus allowing the detection of error conditions, the invalidation of data samples, etc.
- Supporting the transmission of application-specific, out-of-band information per data sample, relevant to the receiver's processing logic.
- Facilitating in-place computations involving in-band-data, by controlling memory alignment, padding, endianness, etc.
- Simplify receiver's processing logic (and user analysis of data from protocol dissection tools) by expressing sizes in payload data units.

The above becomes important when some notion of *stream* is to be supported -i.e. data samples are periodically transmitted between fixed endpoints, with transmission time comparable to the period. E.g. this the case of WFS images transmitted in parallel to readout: the start of image may be signalled before the first packet of pixels is available, error conditions are likely to be detected somewhere in the middle of a data sample, WFS-specific time-stamping data may be provided for end-to-end latency measurements, data copy operations by the receiver are to be avoided, etc.

As in the MUDPI case, in-band RTMS data is not self-descriptive: the packets for a given data sample contain no indication about the type, size, encoding, etc. of its payload data units. This information is assumed to be available system-wide using the *Topic Id* field in the MUDPI packets as an index into a system data dictionary.

RTMS includes no provisions for reliable communication. Basic tracking of the data sample transmission process, verification of data reception completion and recovery from out-of-order delivery may be done by the receiver using the *Frame Id* and *Num Frames* fields in the MUDPI header.

This document focuses on the specification of the MUDPI payload to be used by RTMS, as well as on the semantics of specific MUDPI header fields.

3.1 Real-Time MUDPI Stream Framing

Each RTMS packet is encapsulated in a MUDPI packet, which is in turn encapsulated in a User Datagram Protocol (UDP) packet[1], in turn encapsulated in an Internet Protocol (IP) packet[2], in turn encapsulated in an Ethernet Layer 2 frame[3]:

[1] See: https://en.wikipedia.org/wiki/User_Datagram_Protocol#Packet_structure
[2] See: https://en.wikipedia.org/wiki/IPv4#Packet_structure
[3] See: https://en.wikipedia.org/wiki/Ethernet_frame#Structure

<table>
<tr><td>Ethernet Header
(14/18 B)[4]</td><td colspan="5">Ethernet Payload
(min: 46/42 B max: MTU) [5]</td><td>Ethernet CRC
(4 B)</td></tr>
<tr><td></td><td>IPv4 Header
(≥ 20 B)[6]</td><td colspan="4">IPv4 Data
(≤ MTU – 20 B)</td><td></td></tr>
<tr><td></td><td></td><td>UDP Header
(8 B)</td><td colspan="3">UDP Data
(≤ MTU – 28 B)</td><td></td></tr>
<tr><td></td><td></td><td></td><td>MUDPI Header
(32 B)</td><td>MUDPI Payload
(≤ MTU – 62 B)</td><td>MUDPI Checksum
(4 B)</td><td></td></tr>
<tr><td></td><td></td><td></td><td colspan="3">RTMS Packet</td><td></td></tr>
</table>

RTMS makes use of MUDPI multi-part messages: the data sample being transmitted is split into several packets, all of them tagged with the same sample identifier.

Three types of RTMS packets are distinguished:

- Leader packets:
  They are comprised of meta-data signalling the start of the transmission of a data sample. They indicate the data sample size and may additionally include application-specific information.

  The leader packet is the first packet in the MUDPI multi-part transmission. A single leader packet shall be output by the transmitter at the beginning of the transmission, before the first payload packet is transmitted.

- Payload packets:
  They are comprised of payload data units from the data sample, together with some general-purpose meta-data (e.g. used for facilitating data sample re-assembling at destination) and possibly some data padding.

  One or more payload packets shall be transmitted between the leader and trailer packets. The concrete number depends on the size of the data sample, the Maximum Transmission Unit (MTU) and application-specific constraints.

- Trailer packets:
  They are comprised of meta-data signalling the end of the transmission of a data sample. They may additionally include application-specific information.

  The trailer packet is the last packet in the MUDPI multi-part transmission. A single trailer packet shall be output by the transmitter at the end of the transmission, after the last payload packet has been transmitted.

3.2 Ethernet and IP Framing Considerations

The corresponding Ethernet Layer 1 frame includes as well a 56-bit preamble followed by an 8-bit Start Frame Delimiter (SFD):

[4] The size of the Ethernet frame header is 14 bytes are used when no IEEE 802.1Q tag is present; 18 bytes otherwise.
[5] The minimum size of the Ethernet frame payload is 46 bytes when no IEEE 802.1Q tag is present; 42 bytes otherwise. The maximum size is the MTU.
[6] The size of the IP header is 20 bytes when no *Options* field is present; it will be bigger otherwise.

Preamble	SFD	Ethernet Layer 2 Frame
(7 B)	(1 B)	(min: 64 B max: MTU + 18/22 B) [7]

The Ethernet Layer 1 frame is followed by an idle time between packets (i.e. Inter-packet Gap) of minimum 96-bit:

Ethernet Layer 1 Frame	Inter-packet Gap
(min: 72 B max: MTU + 26/30 B) [8]	(≥12 B)

The MTU is equal to the maximum size of an IP packet that can be transmitted without IP fragmentation[9]. If IP fragmentation is to be avoided[10], the biggest MUDPI payload that can be propagated has a size of MTU – 62 bytes. This size includes both the actual RTMS payload data as well as any required, per-packet, meta-data.

All RTMS packets shall have the *Don't Fragment* (DF) bit in the *Flags* field of the IP packet header set to 1, so as to enable network equipment misconfiguration detection[11].

[7] The maximum size of the Ethernet Layer 2 frame is MTU+18 bytes when no IEEE 802.1Q tag is present; MTU+22 bytes otherwise.

[8] The maximum size of the Ethernet Layer 1 frame is MTU+26 bytes when no IEEE 802.1Q tag is present; MTU+30 bytes otherwise.

[9] When the size of an IP packet exceeds the MTU, a first IP fragment is sent, with the *More Fragments* (MF) bit set in the *Flags* field and containing the UDP header plus part of the payload of size up to the MTU. This is followed by a second IP fragment with the remaining IP payload, but no UDP header. It's up to the destination host to re-assemble the IP datagram and extract the single UDP datagram out of it. IP fragmentation is here usually done at the network stack of the operating system, or offloaded to the driver of the network card which can be further distinguished in a software or hardware implementation.

[10] IP fragmentation shall be avoided as it tends to come with side effects, the most relevant one being to force a network device to behave in a stateful way (i.e. have memory of some packets which were already forwarded in the past) which often involves moving a fragment out of the hardware forwarding path to central CPU based process/interrupt forwarding, which is a no-go for many cases, especially when latency/jitter/in-order delivery is important.

[11] If the DF flag is set and fragmentation is required to route the packet, then the packet is dropped. This enables the detection of network equipment which does not support the specified MTU: it will cause an interface failure (i.e. no data received) rather than more subtle performance degradation.

4. Packet Layout

RTMS packets start at offset 0 inside the MUDPI packet *Payload* field.

Big endianness applies to all packet fields unless specified otherwise.

4.1 Common Protocol Header

	Byte 0			Byte 1			Byte 2			Byte 3		
Byte	Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 23	Bit 24	. . .	Bit 31
0	Frame Info (8-bit)			Extended Info Size (8-bit)			Protocol Version (16-bit)					
4	Status (32-bit)											

4.1.1 Frame Info

8-bit field with the following structure:

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Packet Type			IsSimSource	Endianness	Reserved		

where:

- Bit field [0:2]: Packet Type:
  - 000b: Payload Packet
  - 001b: Trailer Packet
  - 100b: Leader Packet
  - Rest of values reserved for future protocol extension
- Bit 3: IsSimSource:
  - 0: Real source
  - 1: Simulated source
- Bit 4: Endianness
  - 0: Big-endian payload
  - 1: Little-endian payload
- Bit field [5:7]: Reserved for future use

The *Endianness* field indicates exclusively the endianness applicable to the *Payload* field in the protocol payload packets. It shall be set to 0 for leader and trailer packets.

The same endianness shall apply to all the payload packets involved in the transmission of a given data sample.

4.1.2 Extended Info Size

8-bit field in unsigned integer format, indicating the size (in bytes) of the optional *Extended Information* field. If set to 0, the *Extended Information* field shall not be present.

4.1.3 Protocol Version

16-bit field with the following structure:

Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15
Major Version Number			Minor Version Number		

where:

- Bit field [0:7]: Major Version Number
- Bit field [8:15]: Minor Version Number

Both the *Major Version Number* and *Minor version Number* fields contain 8-bit values in unsigned integer format.

4.1.4 Status

32-bit field with the following structure:

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit5	Bit 6	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 31
Error Severity				Stream Error Status				Error Code			Reserved		

where:

- Bit field [0:3]: Error Severity
  - 0000b: No Error
  - 1111b: Fatal Error
  - Rest of values reserved for future use
- Bit field [4:7]: Stream Error Status
  - 0000b: Valid Data Sample
  - 0001b: Incomplete Data Sample
  - 1111b: Discard Data Sample
  - Rest of values reserved for future use
- Bit field [8:15]: Error Code
  8-bit field in unsigned integer format indicating the precise error condition. The interpretation of this field may be application-specific in part/full (TBD).
- Bit field [16:31]: Reserved for future use

4.2 Leader Packet

	Byte 0			Byte 1			Byte 2			Byte 3		
Byte	Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 23	Bit 24	. . .	Bit 31
0	Common Protocol Header											
4												
8	Frame Size (32-bit)											
12	Extended Info (optional)											
...												

Leader packets shall have the *Frame Id* field in the MUDPI header set to 1.

4.2.1 Frame Size

32-bit field in unsigned integer format, containing the number of payload data units in the data sample being transmitted. It can account for a maximum of 2^{32} payload data units.

e.g. The biggest square WFS pixel image that could be transmitted via RTMS has $(2^{16}-1)x(2^{16}-1)$pixels.

4.2.2 Extended Info

Optional field, only valid if the *Extended Info Size* field in the frame header is set to a value different from 0. If this field is present, it contains a number of bytes given by the *Extended Info Size* field. The structure and format of these data is application-specific.

e.g. For WFS supporting them, the following sub-fields may be embedded in the *Extended Info* field of the leader packet:

	Byte 0			Byte 1			Byte 2			Byte 3		
Byte	Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 23	Bit 24	. . .	Bit 31
12	**Window Size X** (32-bit)											
16	**Window Size Y** (32-bit)											
20	**Window Start X** (32-bit)											
24	**Window Start Y** (32-bit)											

where *Window Size X*, *Window Size Y*, *Window Start X* and *Window Start Y* are 32-bit fields in unsigned integer format.

Note that the above extended information may be used as a provision for a hypothetical WFS operational mode where the window location on the detector is changed from one frame the next one during acquisition and the receiver identifies this condition via the WFS data stream itself.

4.3 Payload Packet

LP	Byte 0			Byte 1			Byte 2			Byte 3		
Byte	Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 23	Bit 24	. . .	Bit 31
0	**Common Protocol Header**											
4												
8	**Payload Size** (16-bit)						**Leading Padding Size** (8-bit)			**Trailing Padding Size** (8-bit)		
12	**Payload Tag** (32-bit)											
16	**Leading Padding**											
. . .												

	Leading Padding (cont'd)	**Payload**
	Payload (cont'd)	
. . .	**Payload (cont'd)**	
	Payload (cont'd)	
	Payload (cont'd)	**Trailing Padding**
	Trailing Padding (cont'd)	
. . .	**Trailing Padding (cont'd)**	
	Trailing Padding (cont'd)	
	Trailing Padding (cont'd)	

The need for an *Extended Info* field by the payload packets is TBC.

4.3.1 Payload Size

16-bit field in unsigned integer format, containing the number of payload data units in the packet's *Payload* field. It can account for a maximum payload size of 65,535 data units.

The value of the *Payload Size* field shall not include the leading and trailing padding.

4.3.2 Leading Padding Size

8-bit field in unsigned integer format, containing the number of padding bytes in the packet's *Leading Padding* field. It can account for a maximum padding of 255 bytes.

4.3.3 Trailing Padding Size

8-bit field in unsigned integer format, containing the number of padding bytes in the packet's *Trailing Padding* field. It can account for a maximum padding of 255 bytes.

4.3.4 Payload Tag

32-bit field containing the ordinal position that the first PDU in the packet's *Payload* field occupies within the overall data sample -counting starts at 1. That is, the *Payload* field of any given payload packet contains the closed interval [k, $k+N-1$] of contiguous PDUs from the overall data sample, where:

- k = content of field *Payload Tag*
- N = content of field *Payload Size*

4.3.5 Leading Padding

Field located immediately before the packets' *Payload* field that contains a number of padding bytes given by the *Leading Padding Size* field. It may be used for controlling the alignment of the payload memory image on the receiver's side[12].

[12] A particular memory alignment is required by CPU-based computing nodes if the CPU vector processors (e.g. Intel AVX requires 32-byte alignment) are to be used for maximum performance. Otherwise, costly misaligned accesses are required at the beginning and end of a data block, degrading performance and making the SW more complex.

4.3.6 Payload

Field that contains a number of payload data units given by the *Payload Size* field.

The payload data units inside the *Payload* field are arranged with the endianness indicated in the *Endianness* field of the packet's *Common Protocol Header*.

4.3.7 Trailing Padding

Field located immediately after the packet's *Payload* field that contains a number of padding bytes given by the *Trailing Padding Size* field. It may be used for controlling the size of the payload memory image on the receiver's side[13].

4.4 Trailer Packet

	Byte 0			Byte 1			Byte 2			Byte 3		
Byte	Bit 0	. . .	Bit 7	Bit 8	. . .	Bit 15	Bit 16	. . .	Bit 23	Bit 24	. . .	Bit 31
0	Common Protocol Header											
4												
8	Extended Info (optional)											
…												

Trailer packets shall have the *Frame Id* field in the MUDPI header set to the same value as the *Num Frames* field.

In those applications requiring it, the *Extended Info Section* of the WFS trailer packet may be populated with additional status information.

--- End of document ---

[13] Data blocks of size multiple of a minimum block are required by CPU-based computing nodes if the CPU vector processors (e.g. Intel AVX requires 32-byte or 64-byte blocks depending on the version) are to be used for maximum performance.