Ref.: 25069-800-DU001-A

CamSight Fusion Block

# Interface Control Document

| | | | |
|---|---|---|---|
| Approved by | GERMAIN Nicolas | Product Manager | GERMAIN Nicolas 2026.01.30 '00'01+ 14:29:11 |
| Checked by | RANNOU Julien | Camsight Activity Manager | Julien RANNOU — Signature numérique de Julien RANNOU Date : 2026.01.30 14:32:49 +01'00' |
| Written by | LANGUILLE Antoine | Embedded Software Engineer | LANGUILLE Antoine — Signature numérique de LANGUILLE Antoine Date : 2026.01.30 14:20:21 +01'00' |
| | Name | Title | Visa |



bertin technologies
SAS (French Simplified Joint Stock Company) with registered capital of 18 000 000 Euros ◢ 422 511 204 RCS VERSAILLES ◢
French APE business sector code 7112B
Head Office: Parc d'Activités du Pas du Lac ◢ 10 bis, avenue Ampère Montigny-le-Bretonneux (78) ◢ France ◢ +33 (0)1 39 30 60 00
Postal Address: BP 284 ◢ 78053 Saint-Quentin-en-Yvelines Cedex ◢ FRANCE
https://www.bertin-technologies.com

# LIST OF REVISIONS

| Rev. | Date | Modifications<br>(main reasons, sections and pages affected) | Writer / Checker |
|---|---|---|---|
| A | 11/12/2025 | First issue | ALA/JRA |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# TABLE OF CONTENTS

# 1. SCOPE

This Interface Control Document (ICD) provides detailed information about the CamSight Fusion Block (FB) camera interfaces. This document's purpose is to provide guidance for integrating this camera into other video equipment and systems, by describing the camera's powering requirements, interfaces, and connectors pinouts.

This document is intended to be used by engineers and technicians in charge of integrating the camera into their systems. It assumes a basic understanding of video signal processing, electrical engineering, and communication protocols. This document should be used as a general guide, and any specific requirements or questions should be addressed to Bertin Technologies directly.

## 1.1. Documentation

| | | |
|---|---|---|
| AD001 | Compagnon software user manual | 25069-800-DU002 |
| AD002 | Updater software user manual | 25069-800-DU003 |
| AD003 | | |

## 1.2. Acronyms

| | |
|---|---|
| SWaP | Size, Weight and Power |
| FB | Fusion Block |
| LS | Low SWaP |
| FoV | Field of View |
| UART | Universal Asynchronous Receiver Transmitter |
| SDI | Serial Digital Interface |
| WIP | Work In Progress |

# 2. OVERVIEW

CamSight Fusion Block (FB) is an Original Equipment Manufacturer (OEM) digital core camera developed by Bertin Photonics. It has been designed to meet users' requirements of Low SWaP cameras offering easy integration within optronic systems while providing thermal infrared and visible spectrum imaging.

The CamSight FB camera is composed of a thermal infrared sensor (CamSight LS IR camera submodule) coupled with a visible spectrum image sensor allowing for multi-spectrum image capture. The sensor details are given in Table 1.

Based on patented Bertin Photonics' shutterless and image fusion technologies, it allows for low latency and freeze-free real time image delivery in thermal infrared, visible spectrum or a fusion of both, making it suitable for a wide range of applications.

| Sensor/Camera | Thermal IR (CamSight LS) | Visible | CamSight Fusion Block |
|---|---|---|---|
| Manufacturer | LYNRED | PYXALIS | Bertin Photonics |
| Reference | ATTO640D-02 | HDPyx230G | CSFB |
| Spectrum | LWIR | RGB and NIR | LWIR and RGB |
| Resolution | 640x480 | 1944x1204 | 1920x1080 |
| Pixel pitch | 12 microns | 3.2 microns | N/A |

**Table 1: CamSight FB sensors characteristics**

CamSight FB is available in 4 different assemblies, each corresponding to a specific Field of View and SWaP requirement, mechanical blueprints and characteristics are given for each module.

Several additional electronics boards are available to change the video output to better suit the users' needs. These boards are mounted on top of the "Core Power Board" as shown in Figure 1.

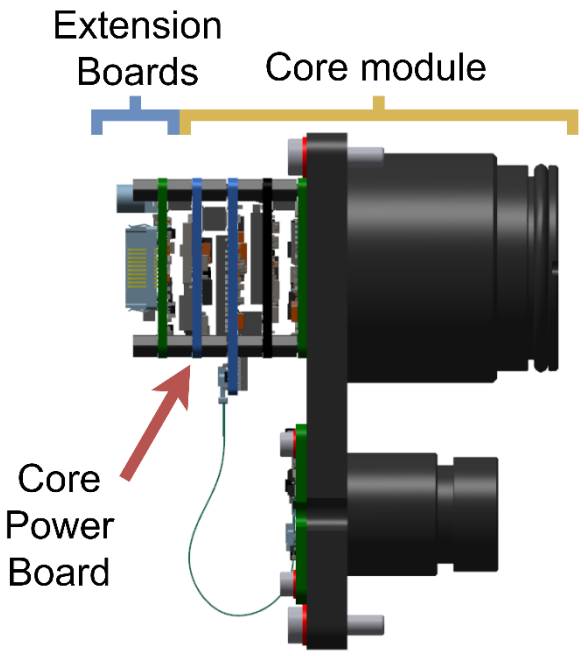



**Figure 1: CamSight FB Electronic stack**

# 3. MECHANICAL INTERFACE

## 3.1. FOV modules

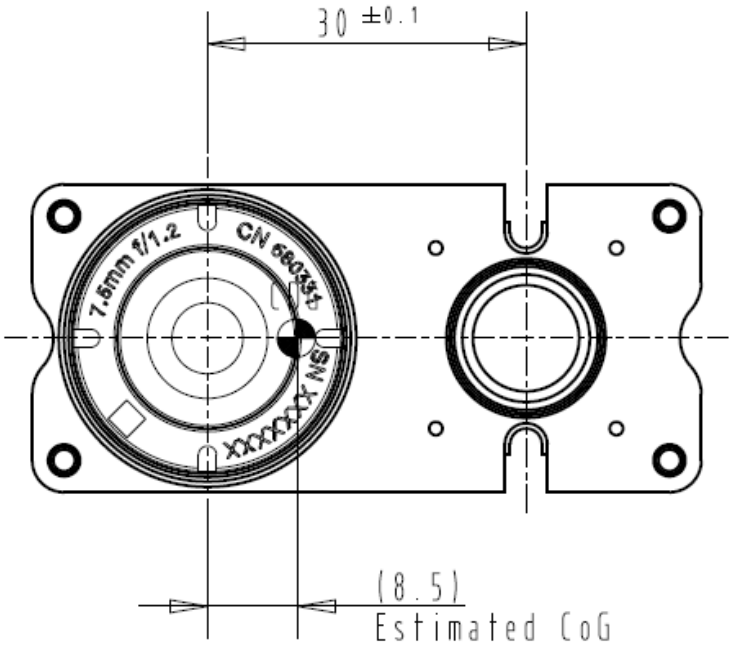
### 3.1.1. Wide Field (WF)

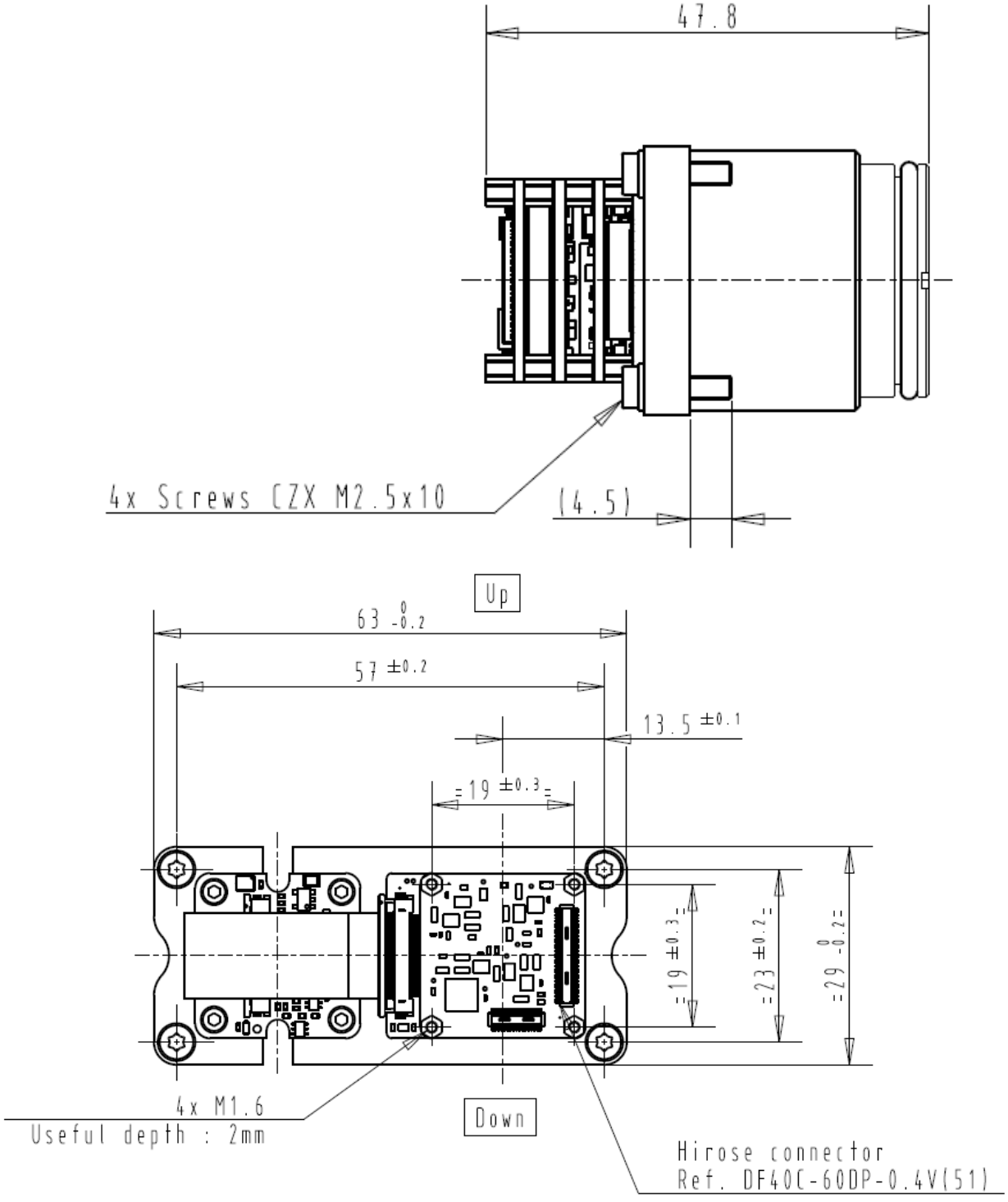
**Figure 2: WF Core Module 3D rendering**

| Dimensions | Value |
|---|---|
| Height | 29mm |
| Width | 63mm |
| Depth | 47.8mm |
| Weight | 98g |
| IR Field of View | 61.1° x 45.8° |
| Visible Field of View | 73° x 45° |

**Table 2: WF Core characteristics**

NOTE: The module depth was measured without any additional boards on top of the Power Core board. If additional electronic boards are mounted on the back of the camera, their depth should be added to the given camera depth to calculate the final camera size.





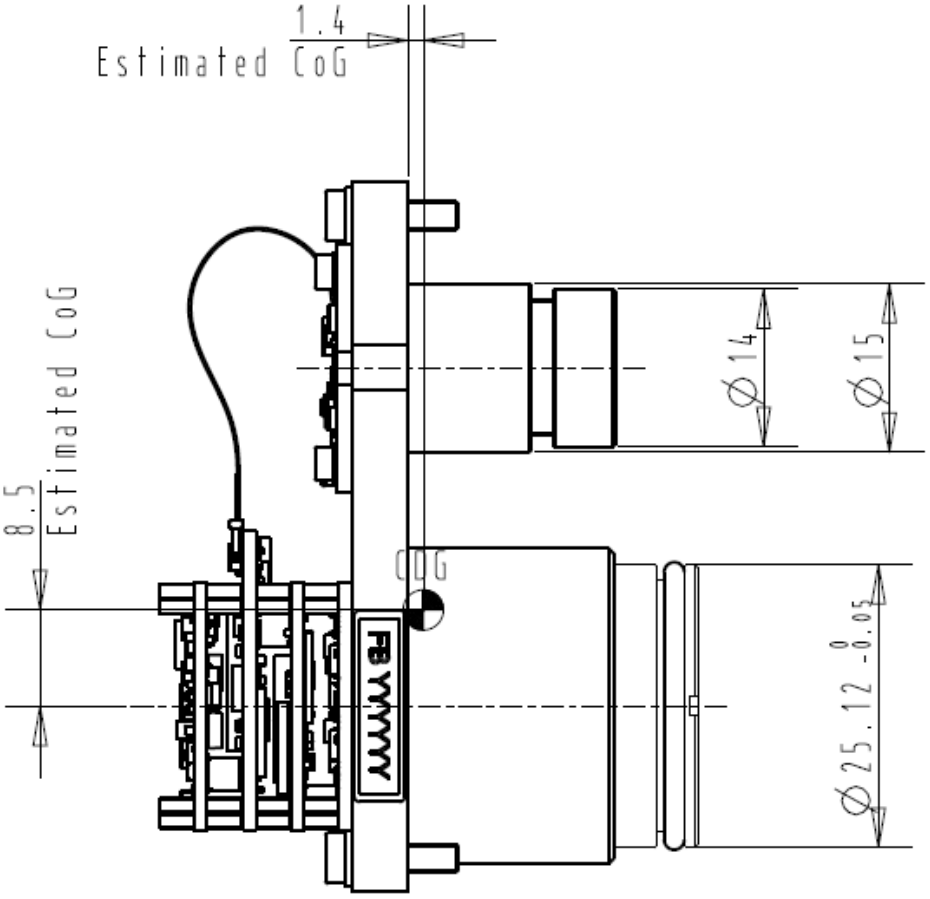



**Figure 3: WF Core mechanical blueprint**

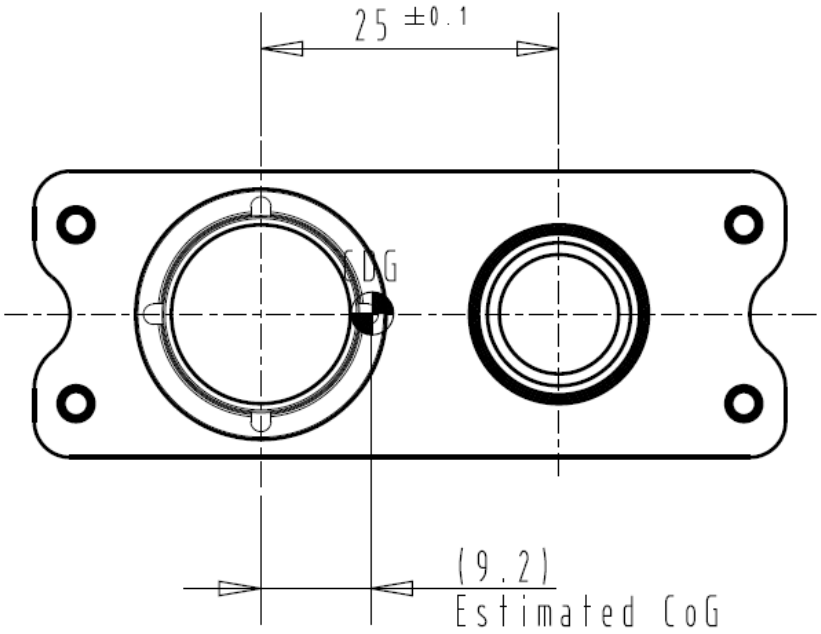
### 3.1.2. Medium Field (MF)

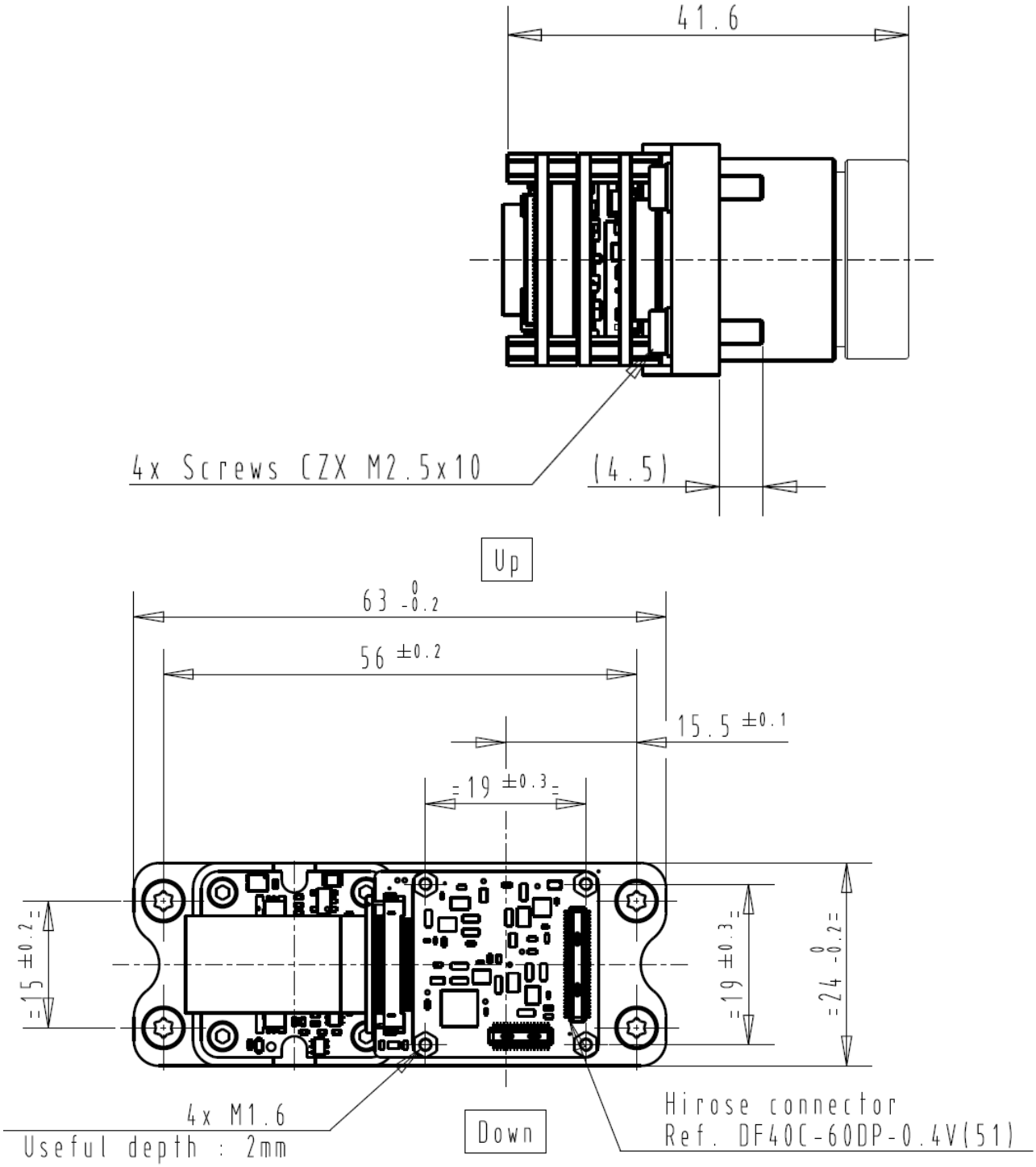
**Figure 4: MF Core Module 3D rendering**

| Dimensions | Value |
|---|---|
| Height | 24mm |
| Width | 63mm |
| Depth | 41.6mm |
| Weight | 78g |
| Field of view IR | 32.0° x 24.0° |
| Field of View VIS | 42.0° x 24.3° |

**Table 3: MF Core characteristics**

NOTE: The module depth was measured without any additional boards on top of the Power Core board. If additional electronic boards are mounted on the back of the camera, their depth should be added to the given camera depth to calculate the final camera size.





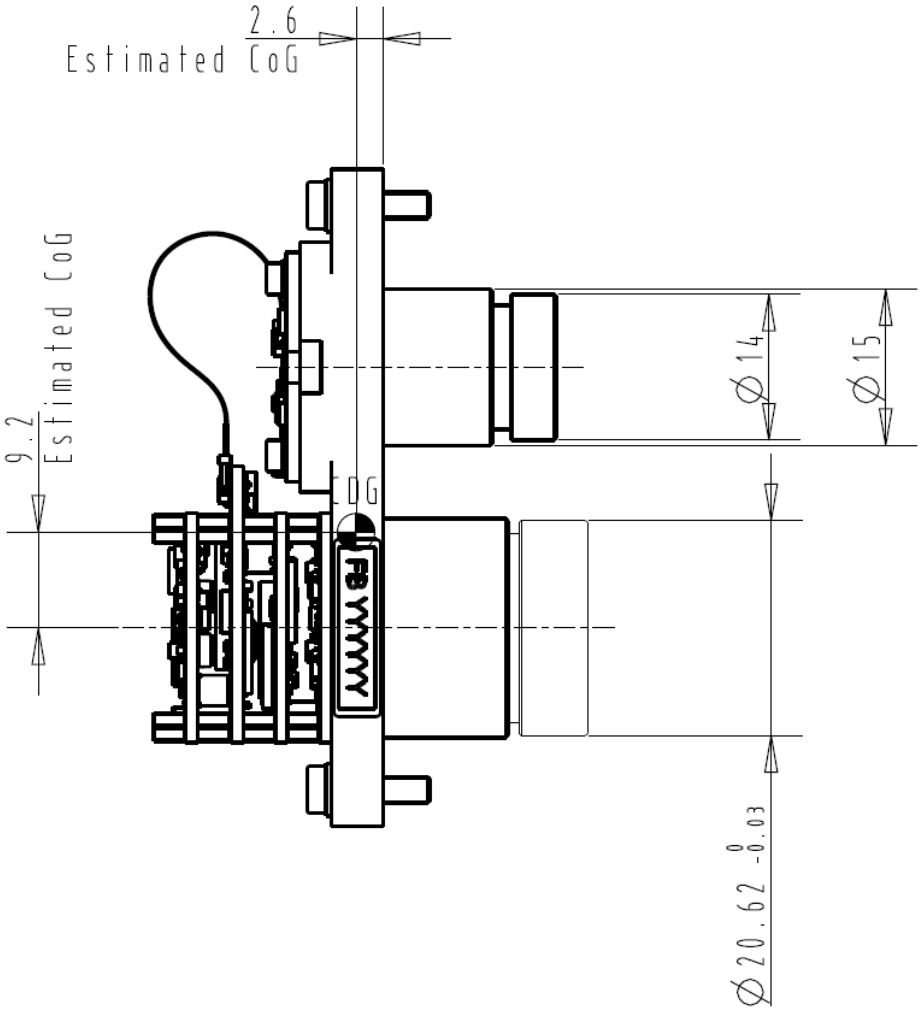



**Figure 5: MF Core mechanical blueprint**

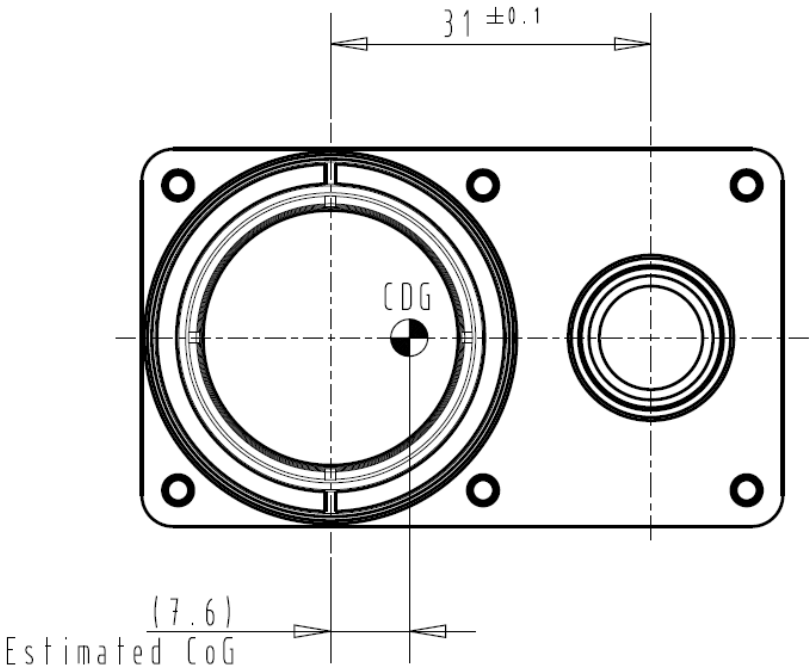
### 3.1.3. Narrow Field (NF)

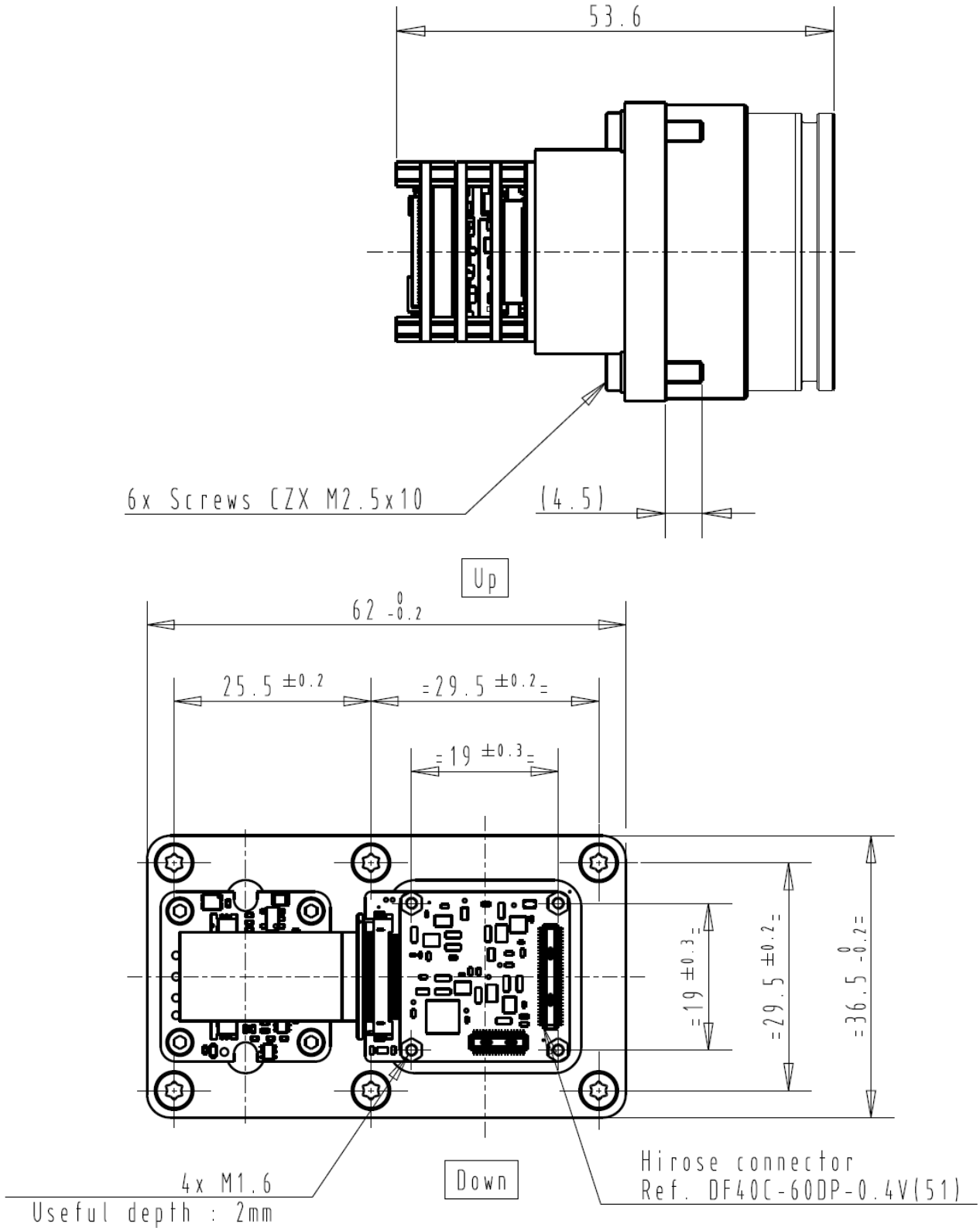
**Figure 6: NF Core Module 3D rendering**

| Dimensions | Value |
|---|---|
| Height | 36.5mm |
| Width | 62mm |
| Depth | 53.6mm |
| Weight | 116g |
| Field of View IR | 18.0° x 13.5° |
| Field of View VIS | 21.7° x 12.3° |
| Mounting screws | 4x M02 |

**Table 4: NF Core mechanical characteristics**

NOTE: The module depth was measured without any additional boards on top of the Power Core board. If additional electronic boards are mounted on the back of the camera, their depth should be added to the given camera depth to calculate the final camera size.





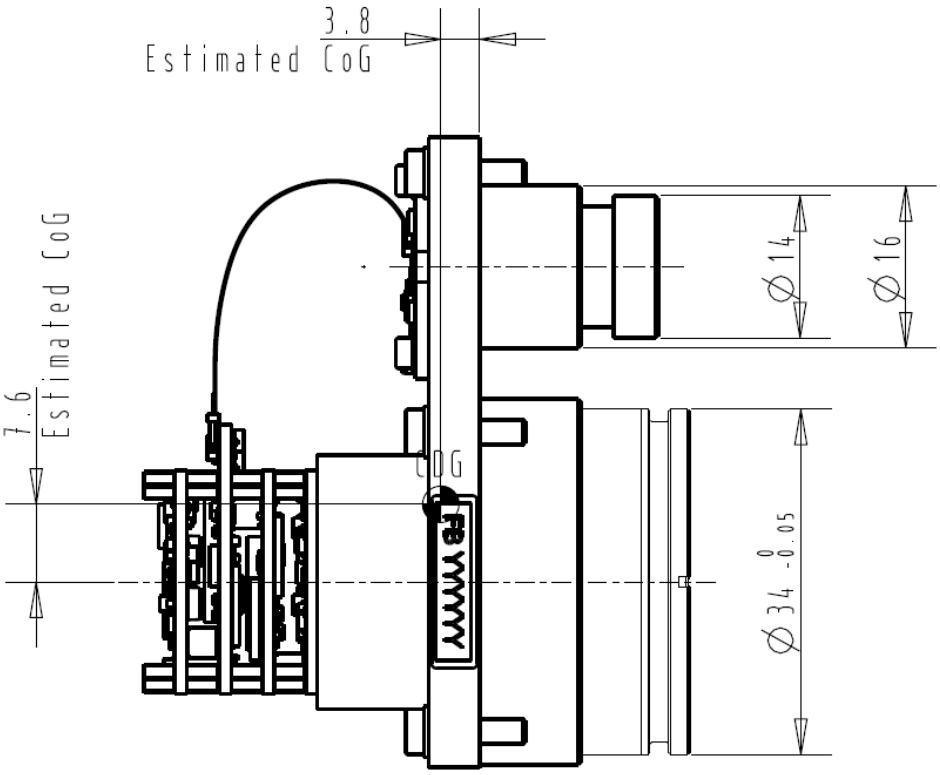



**Figure 7: NF Core mechanical blueprint**

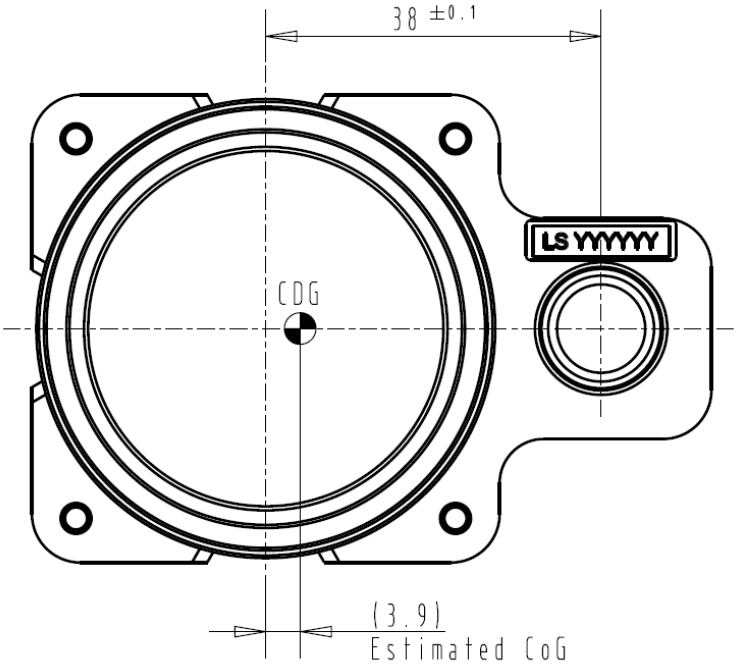
### 3.1.4. Very Narrow Field (VNF)

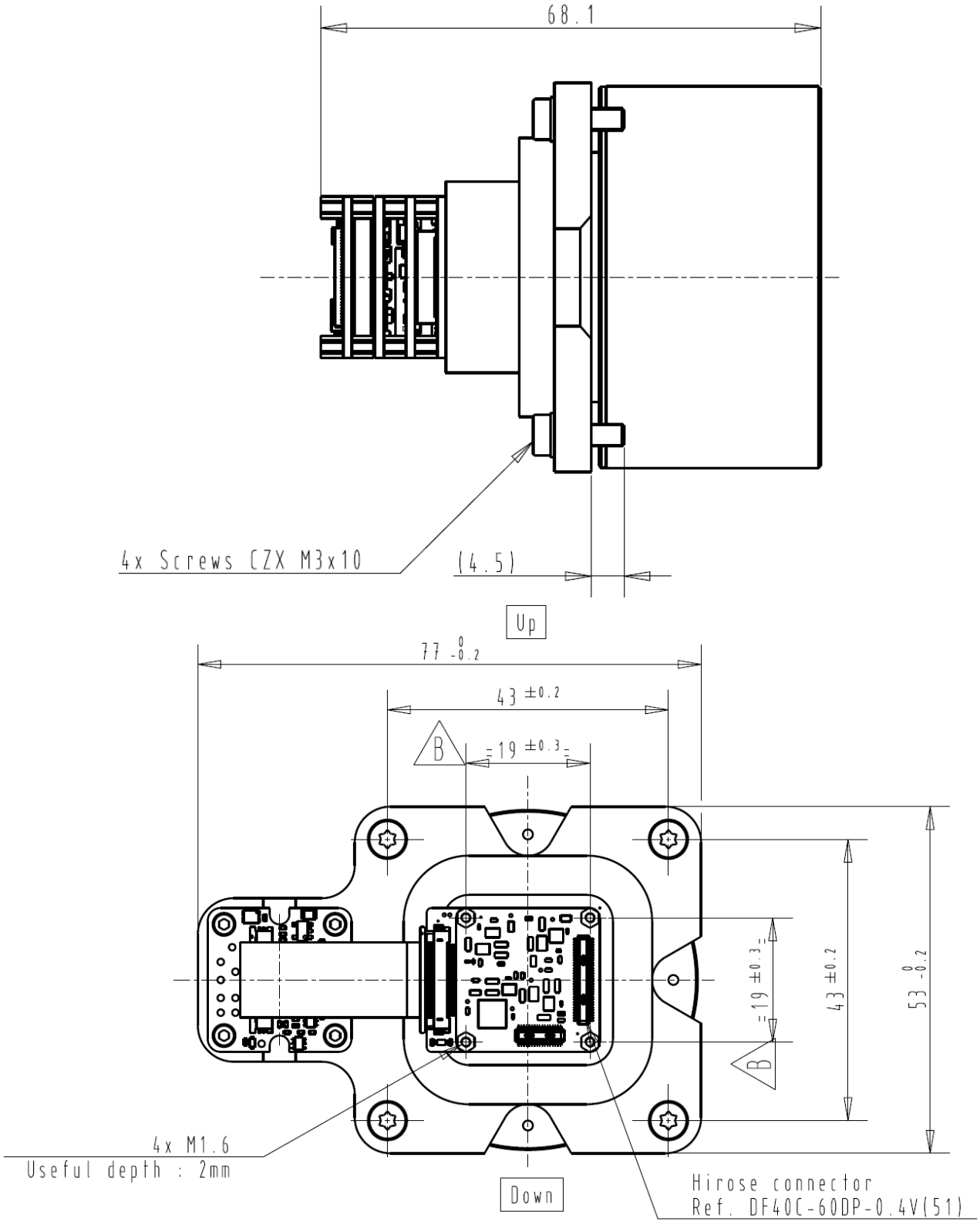

**Figure 8: VNF Core Module 3D rendering**

| Characteristic | Value |
|---|---|
| Height | 53mm |
| Width | 77mm |
| Depth | 68.1mm |
| Weight | 218g |
| Field of View IR | 10.9° x 8.2° |
| Field of View VIS | 14.1° x 8.0° |

**Table 5: VNF Core characteristics**

NOTE: The module depth was measured without any additional boards on top of the Power Core board. If additional electronic boards are mounted on the back of the camera, their depth should be added to the given camera depth to calculate the final camera size.





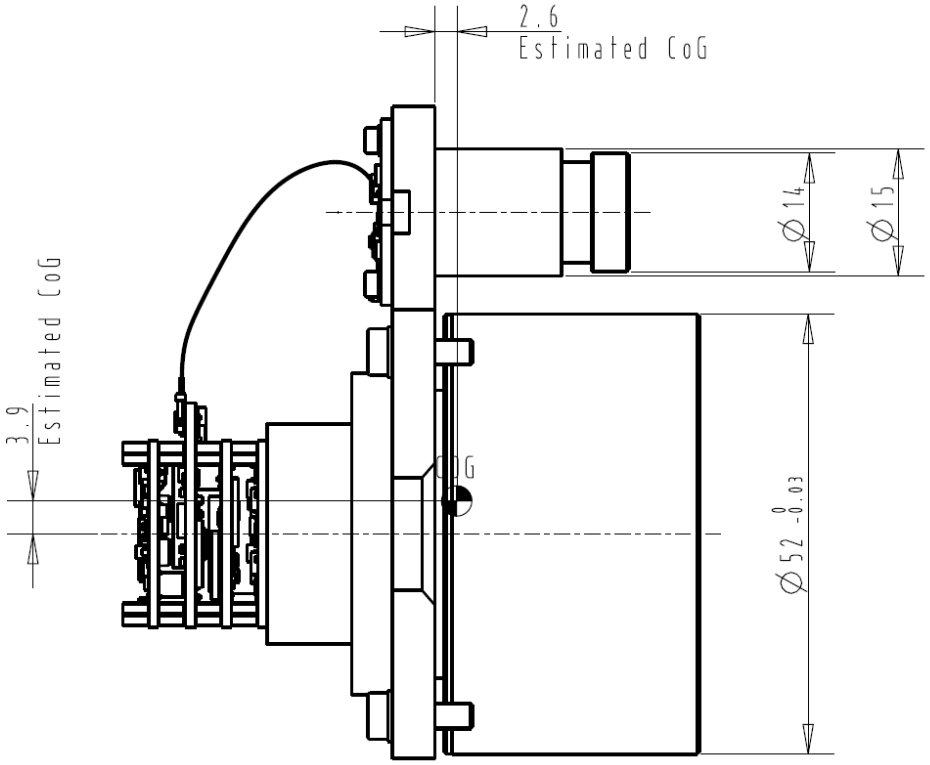



**Figure 9: VNF Narrow mechanical blueprint**

## 3.2. Electronic boards

### 3.2.1. Core Power Board

The Core Power Board is the base camera output board, providing essential functions to the camera electronics and is required to connect all subsequent electronic boards.

NOTE: The hatched areas around the screws should be left free of components.

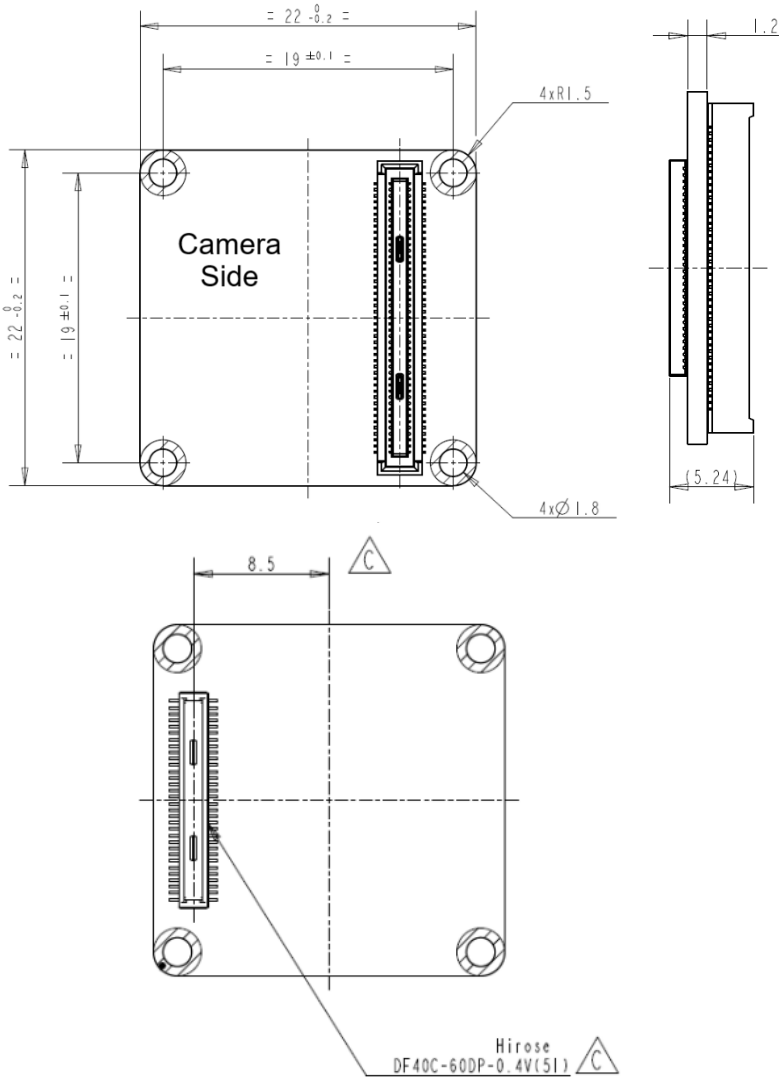



**Figure 10: Core Power Board mechanical blueprint**

### 3.2.2. Barrel connector board

NOTE: The hatched areas around the screws should be left free of components.

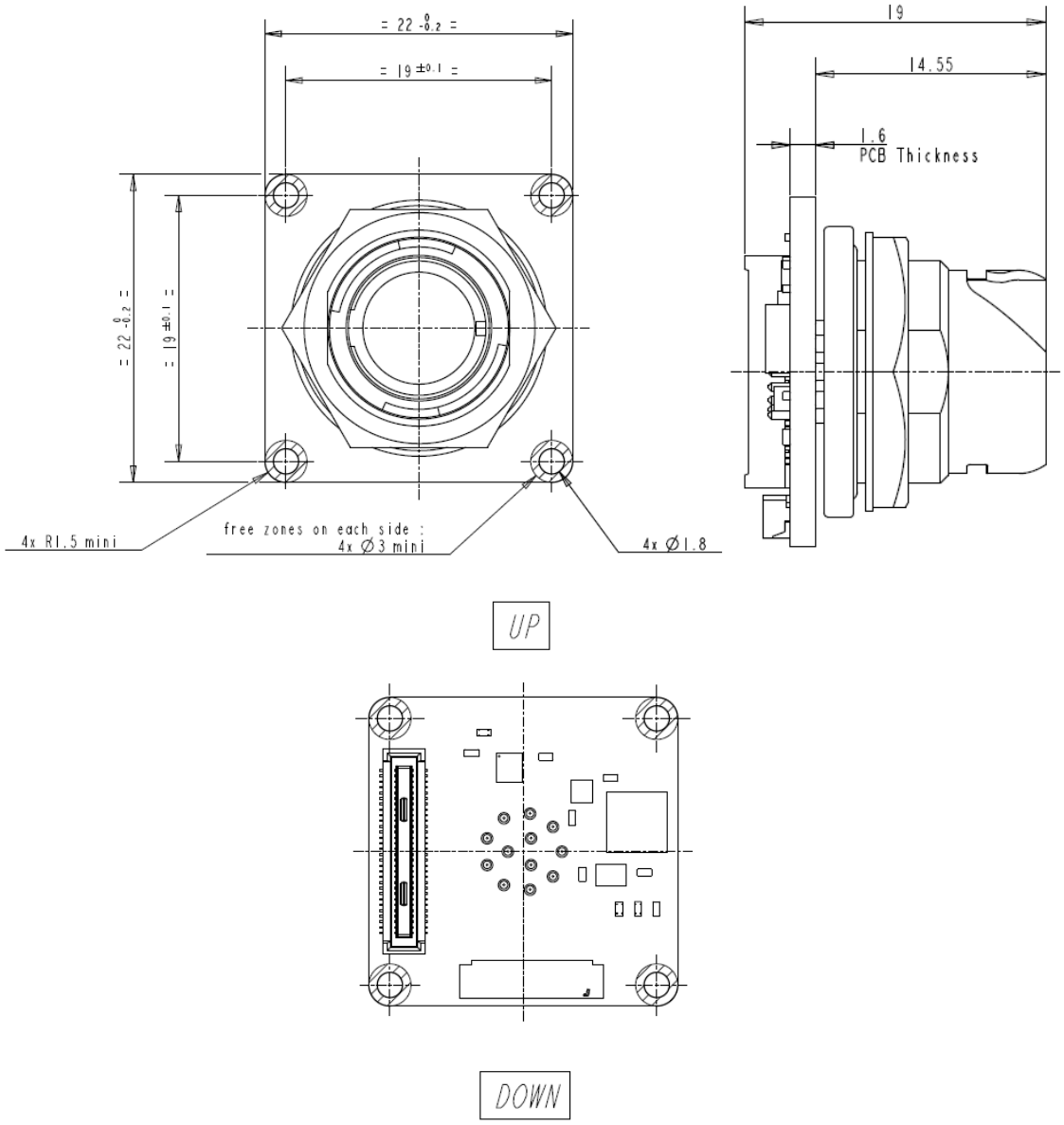



**Figure 11: Barrel Connector Board mechanical blueprint**

# 4. ELECTRICAL INTERFACE

## 4.1. Core Power Board

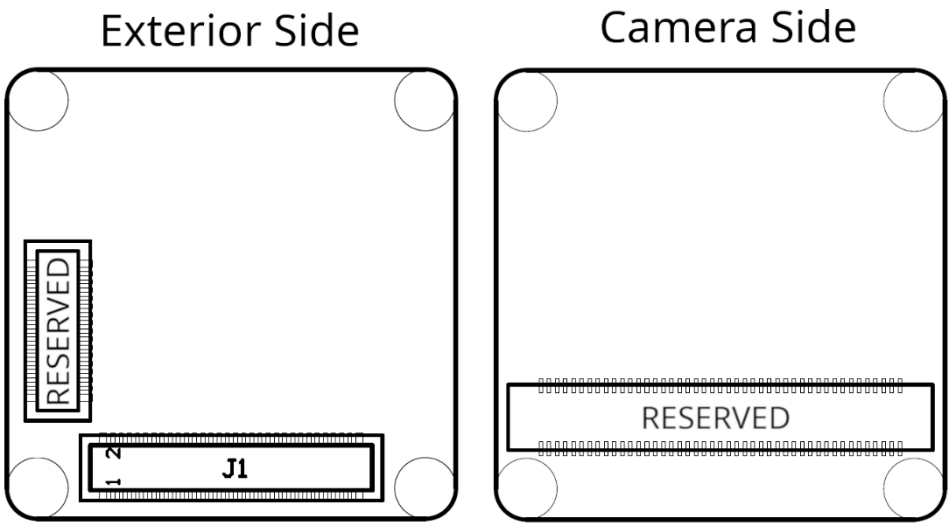



**Figure 12: Core Power Board connectors layout**

| Connector | Connector reference | Mating connector(s) |
|---|---|---|
| J1 | DF40C-60DP-0.4V(51) | DF40HC(3.0)-60DS-0.4V(51) |

**Table 6: Core Power Board connectors reference**

| Max voltage | Signal name | I/O | Pin number | Pin number | I/O | Signal name | Max voltage |
|---|---|---|---|---|---|---|---|
| 1.8V | Parallel video b0 | OUT | 1 | 2 | OUT | HD-SDI | ±2.5V |
| 1.8V | Parallel video b1 | OUT | 3 | 4 | | **GND** | |
| 1.8V | Parallel video b2 | OUT | 5 | 6 | | Reserved | |
| 1.8V | Parallel video b3 | OUT | 7 | 8 | | Reserved | |
| 1.8V | Parallel video b4 | OUT | 9 | 10 | | **GND** | |
| 1.8V | Parallel video b5 | OUT | 11 | 12 | | Reserved | |
| 1.8V | Parallel video b6 | OUT | 13 | 14 | | Reserved | |
| 1.8V | Parallel video b7 | OUT | 15 | 16 | | **GND** | |
| | **GND** | | 17 | 18 | IN | UART Rx | 1.8V |
| 1.8V | Parallel video b8 | OUT | 19 | 20 | OUT | UART Tx | 1.8V |
| 1.8V | Parallel video b9 | OUT | 21 | 22 | BOTH | I2C SDA | 1.8V |
| 1.8V | Parallel video b10 | OUT | 23 | 24 | BOTH | I2C SCL | 1.8V |
| 1.8V | Parallel video b11 | OUT | 25 | 26 | | **GND** | |
| 1.8V | Parallel video b12 | OUT | 27 | 28 | IN | Camera Reference Clock P | 1.8V |
| 1.8V | Parallel video b13 | OUT | 29 | 30 | IN | Camera Reference Clock P | 1.8V |
| 1.8V | Parallel video b14 | OUT | 31 | 32 | | **GND** | |
| 1.8V | Parallel video b15 | OUT | 33 | 34 | | Reserved | |
| | **GND** | | 35 | 36 | | Reserved | |
| 1.8V | Parallel video VSYNC | OUT | 37 | 38 | | Reserved | |
| 1.8V | Parallel video HSYNC | OUT | 39 | 40 | | Reserved | |
| 1.8V | Parallel Video Display Enable | OUT | 41 | 42 | | Reserved | |
| | **GND** | | 43 | 44 | | **GND** | |
| 1.8V | Parallel video pixel clock | OUT | 45 | 46 | BOTH | GPIO2 | 1.8V |
| | **GND** | | 47 | 48 | BOTH | GPIO0 | 1.8V |
| | Reserved | | 49 | 50 | BOTH | GPIO1 | 1.8V |
| 1.8v | Video Trigger | IN | 51 | 52 | | **GND** | |
| | **GND** | | 53 | 54 | OUT | 1.8V | 1.8V |
| 5V | 5V | IN | 55 | 56 | | **GND** | |
| 5V | 5V | IN | 57 | 58 | IN | 4.5V | 5.5V |
| | **GND** | | 59 | 60 | IN | 4.5V | 5.5V |

**Table 7: Core Power Board J1 connector pinout**

## 4.2. Barrel connector Board

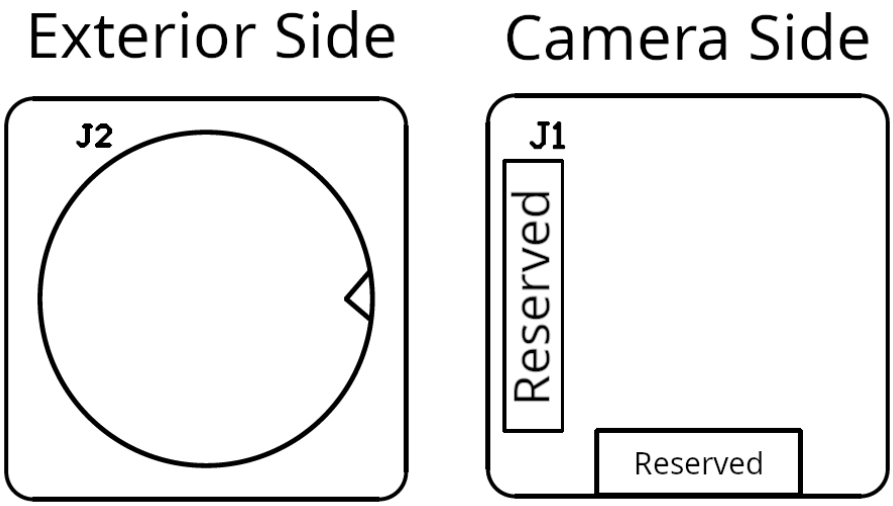



**Figure 13: Barrel Connector Board connectors layout**

| Connector | Connector reference | Mating connector(s) |
|---|---|---|
| J1 | Reserved | Base OEM Board J4 connector |
| J2 | Hirose - LF10WBRB-12PD | Hirose - LF10WBP-12S(31) |

**Table 8: Barrel Connector Board connectors references**

For a J2 to USB & HD-SDI cable example, ask Bertin Technology for details about schematic **024489-465-PE001-A**.

| Pin number | I/O | Signal name | Max voltage |
|---|---|---|---|
| 1 | | **GND** | |
| 2 | IN | 5V | 5.5V |
| 3 | IN | UART Rx | 3.3V |
| 4 | OUT | UART Tx | 3.3V |
| 5 | | Reserved | |
| 6 | | Reserved | |
| 7 | | Reserved | |
| 8 | OUT | HD-SDI | ±2.5V |
| 9 | | **GND** | |
| 10 | | Reserved | |
| 11 | | Reserved | |
| 12 | IN | 1.8V | 1.8V |

**Table 9: Barrel Connector Board J2 connector pinout**

## 4.3. Power Supplies

### 4.3.1. Core Power Board

| Input pin | Requirement | Min | Max |
|---|---|---|---|
| VCC Digital | Voltage | 2.5V | 5.1V |
| | Current | | 280mA |
| | Ripple | | 60mV* |
| VCC Analog | Voltage | 4.5V | 5.1V |
| | Current | | 100mA |
| | Ripple | | 30mV* |

**Table 10: Board 2 Power Supplies requirements**

*: Avoid SMPS with operating frequency lower than 1MHz for better image quality. Prefer linear regulators with high PSRR.

### 4.3.2. Barrel Connector Board

| Input pin | Requirement | Min | Max |
|---|---|---|---|
| Power +5V | Voltage | 4.5V | 5.5V |
| | Current | | 300mA |
| | Ripple | | 60mV* |

**Table 11: Board 2 Power Supplies requirements**

*: Avoid SMPS with operating frequency lower than 1MHz for better image quality. Prefer linear regulators with high PSRR.

## 4.4. Serial electrical interface

To configure and communicate with CamSight FB cameras, a serial UART interface is available on all electronic boards and extension boards.

**WARNING:** Depending on the electronic board you are connecting to, the logic levels can be either 1.8V or 3.3V, refer to your board electrical interface for the specific line voltage to use.

| UART Parameter | Value |
|---|---|
| Default baud rate | 115200 baud/s |
| Start bit | 1 |
| Stop bit | 1 |
| Number of data bits | 8 |
| Parity bit | None |

**Table 12: UART Serial interface parameters**

# 5. IMAGE PROCESSING

## 5.1. Visible Thermal Fusion

The camera includes a smart pixel-level fusion algorithm for visible and thermal infrared images:

This image processing algorithm aims to generate a more informative composite image by intelligently fusing visible and infrared (IR) inputs at the pixel level. The goal is to enhance detection and reveal hidden elements by increasing contrast and dynamic range, while preserving as much relevant information from each source image as possible to faithfully reconstruct the observed scene.

Two prerequisites are essential for the algorithm to function properly:

1. The images must be temporally synchronized beforehand.
2. The images to be fused must have the same resolution.

To achieve this, a synchronization module in the design performs a pixel index alignment upstream of the fusion module.

In addition, a zoom function is also applied to the IR channel prior to fusion, resizing IR images to match the resolution of the visible sensor: 1920×1080 (16:9 format). Since IR images have a native resolution of 640×480 (4:3 format), the process involves vertical interpolation by a factor of 2.25.

Consequently, this explains **the presence of two vertical black bars** on either side of the upscaled IR image.

Therefore, during the fusion operation, **only the central ROI (1440x480) is used as the effective fusion zone.** On the left and right edges, the visible image is fused with IR black bars, resulting in darker regions after fusion processing.

**Figure 14: IR/Visible/Fusion video snapshots**

## 5.2. Infrared Band

### 5.2.1. Shutterless operation

CamSight FB cameras feature Bertin shutterless technology allowing for low-latency and freeze-free image delivery without a mechanical shutter. This technology compensates for thermal drift and other noise sources affecting image quality in traditional shuttered cameras.

To achieve the stated thermal range, the camera is calibrated at regular temperature intervals throughout the whole thermal range. When transition occurs between two calibration temperatures points some artifacts can be observed on the image, typically lasting less than half a second.

The shutterless operation of the CamSight FB camera offers many advantages, such as low latency and freeze-free image delivery in real-time. Several post processing algorithms are integrated into the camera to deliver high quality output images.

### 5.2.2. Column noise correction

Each pixel columns is individually corrected inside the sensor, therefore variations between each column reference bolometer cause a type of non-uniformity, resulting in column noise.

The available “Column noise correction” algorithm works by detecting variations between columns of pixels and correcting them, effectively smoothing out column’s variations.

NOTE: This algorithm may sometimes create artefacts on vertical structures when they are too big in the image.

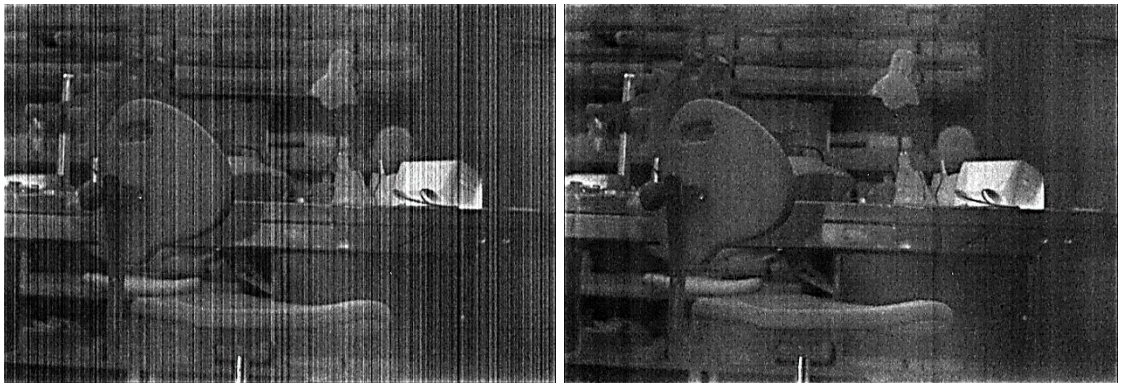


**Figure 15: Left: column noise correction off / Right: column noise correction ON (Sharpness is increased by 75% on both images to better observe the effect on thumbnails.)**

### 5.2.3. Vignetting correction (flattening filter)

Temperature gradients on the sensor cause a whitening of the image center and darkening of the corners, also called vignetting.

This non-uniformity is a common issue with shutterless cameras and can significantly degrade the image quality. The algorithm works by analyzing the image to detect areas with different temperature gradients and applying a correction factor to each pixel based on its location.

NOTE: This algorithm can sometimes create artefacts on temperature uniform surfaces, such as the sky. This is because the algorithm may interpret the uniform temperature as a temperature gradient and apply a correction factor that is not needed.

**Figure 16: Vignetting correction turned off (left) / on (right)**
**(brightness is increased by 30% on both images to better observe the effect on thumbnails.)**

### 5.2.4. Edge enhancement: sharpening or contouring

To further improve image quality, the camera is equipped with an edge enhancement algorithm. This algorithm has two modes: Sharpening, and Contouring.

In sharpening mode, the object's edges are enhanced, resulting in a sharper image.

In contouring mode, an edge detection algorithm is applied to the image, highlighting only the scenes edges, which can be particularly useful when aiming with a holographic thermal aiming sight. It is recommended that image polarity is inverted, to have contours in white over a black background instead of black edges over a white background.

**Figure 17: Sharpening filter turned off (left) / on (right)**

**Figure 18: Contouring filter turned off (left) / on (right**

### 5.2.5. Dynamic Contrast Enhancement

The camera includes two dynamic contrast enhancement algorithms that maximize the scene's contrast based on histogram equalization techniques: CLHE (Contrast-Limited Histogram Equalization) and CLAHE (Contrast-Limited Adaptive Histogram Equalization).

The CLHE algorithm gathers the image histogram over a user defined ROI (Region of Interest). This algorithm has a uniform impact on the image, but the overall result is dependent on the hottest point inside the ROI.

The CLAHE algorithm divides the image into 16 ROIs and processes the histogram of each part of the image separately. The final contrast correction is then interpolated to reduce boundary effects between regions. The result is a much more uniform image but pixels with the same brightness no longer necessarily have the same temperature.

The level of contrast correction is adjustable by the end user to set the scene output contrast.

Depending on the scenes encountered in the final application, CLHE or CLAHE should be selected for best results.

**Figure 19: CLHE algorithm (left) / CLAHE algorithm (right)**

### 5.2.6. Automatic Histogram Control

The camera is equipped with a function to restrict AHC (Automatic Histogram Control) gain. This functionality aims at reducing noise amplification in low contrast settings. This feature works against the CLHE/CLAHE algorithm, effectively limiting the image output contract to reduce visual noise.

Users can tune two parameters, namely AHC black limit and AHC white limit, to limit the output image brightness range in the lower values (black) and/or higher values (white).

The AHC gain limits are set as a percentage of the output dynamic. A black limit of 20% will leave at least the 20% of the lower output brightness range unused (brightnesses values from 0 to 51 for MONO8 output), therefore the image will appear grey in the dark spots instead of black and likewise for the white limit.

The CLHE/CLAHE algorithms equalize the image histogram by clipping it at a user defined threshold and redistributing the clipped energy over the image histogram.

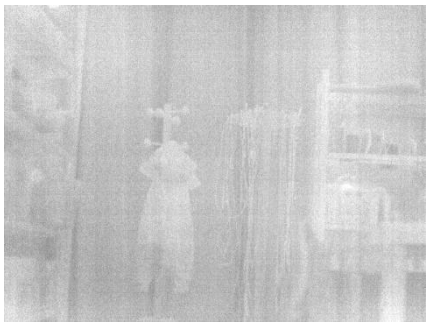
In the situation of low contrast (narrow histogram peak), this redistribution has very little impact on the image overall histogram, the input/output lookup table graph is therefore very steep, resulting in small input brightness variations such as noise becoming large variations in output brightness, amplifying the image noise.

To counteract this phenomenon, the energy redistribution is artificially stretched over a range of brightnesses larger than the image histogram, effectively spreading this amplification, lowering the lookup table graph slope, resulting in a lower noise amplification at the cost of reducing the output brightness range.

**Figure 20 : Original image with low contrast and high noise (no AHC)**

Black Limit = 0%
White Limit = 10%

Black Limit = 10%
White Limit = 10%



Black Limit = 10%
White Limit = 0%

In scenes with higher contrast, the effects of the output brightness range reduction will become noticeable with high limit values.

| Black Limit = 0%<br>White Limit = 0% | Black Limit = 20%<br>White Limit = 20% | Black Limit= 40%<br>White Limit = 40% |
|---|---|---|

### 5.2.7. AHC speed temporal filter

When the AHC changes, a user parameter is available to adjust the speed at which the new gain parameters are applied.

A recursive filter is employed to limit the AHC speed, where the filter coefficient is represented as $\alpha = \frac{1}{2^k}$, meaning that with each iteration the gain output will change by $\alpha$ percents until the target value is reached.

This parameter can be adjusted within a range from 0 (immediate update) to 8 (very slow AHC response). Iterations time is typically 50ms.

### 5.2.8. Gamma correction

A gamma correction can be applied to the image to modify it's luminosity and contrast. Gamma values higher than 1 result in an overall darkening of the image while a gamma value less than 1 results in an overall lightening of the image.

The gamma correction value can be set by the user depending on the imaged scene.

NOTE: The gamma computation is approximated, resulting in a lack of brightness boosting for low values of gamma (3rd order approximation of $(1+x)^\gamma$).

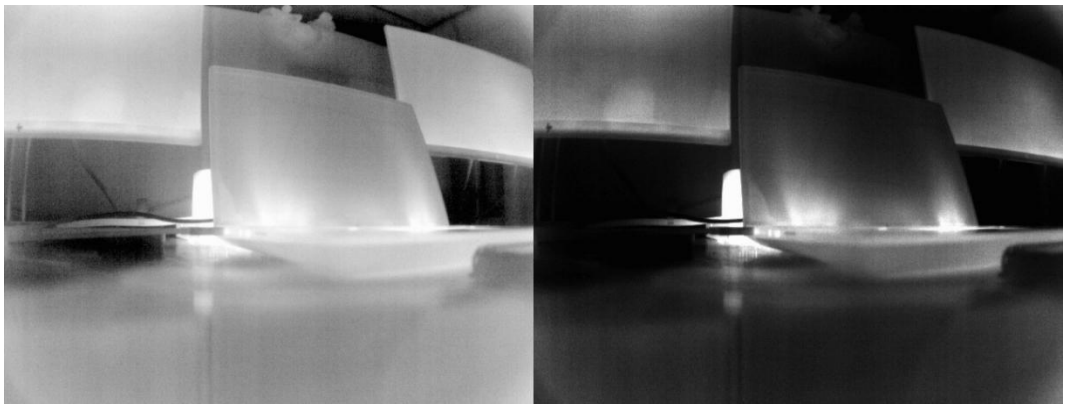


**Figure 21: Gamma value set to 0,5 / Gamma set to 2**

### 5.2.9. Image polarity

The color representing hot zones in the image, also called its polarity, can be chosen by the user.

If lighter colors are used to indicate hot zones, the camera is in "white hot" polarity.

If the camera is using darker colors to represent hot zones, it is set in "black hot" polarity.

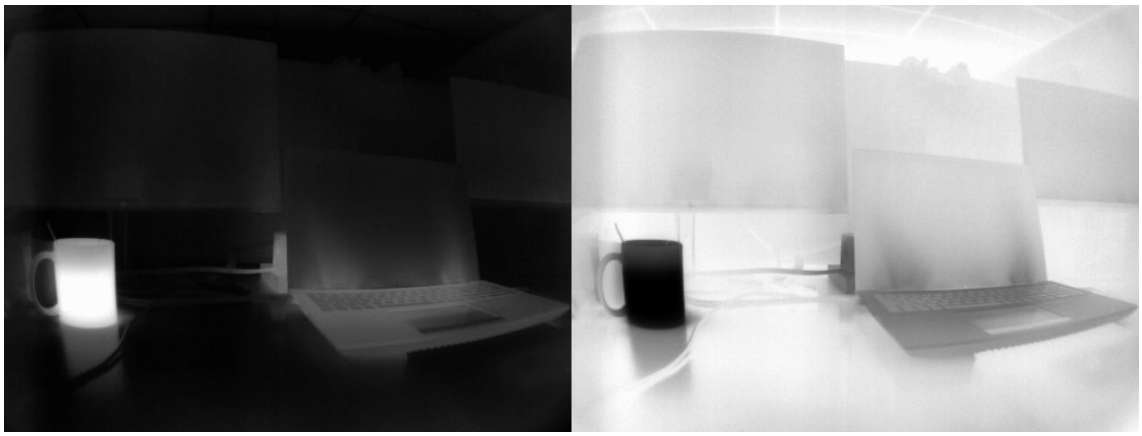


**Figure 22: White hot image / black hot image**

## 5.3. Visible band

### 5.3.1. Automatic gain control

The camera is equipped with a function to control the visible sensor gain automatically (Automatic Gain Control) gain. This allows the user to set a target brightness for the visible image and the camera will modify the sensor gain accordingly.

Users can tune three parameters, namely AGC target, AGC thresholds and AGC Filter coefficient.

The AGC target is the target brightness the user wants to maintain, it is expressed as a percentage between an all-black image at 0% and an all-white image at 100%.

To prevent flashing the user can also choose the minimum brightness deviation from the target value that will trigger a gain update, the AGC thresholds expressed as percentage points. A target brightness of 50% with a threshold of 5pp will trigger a new AGC adjustment if the brightness goes above 55% or below 45%.

### 5.3.2. AGC Temporal filter

When the AGC changes, a user parameter is available to adjust the speed at which the new gain parameters are applied.

A recursive filter is employed to limit the AGC speed, where the filter coefficient is represented as $\alpha = \frac{1}{2^k}$, meaning that with each iteration the gain output will change by $\alpha$ percents until the target value is reached.

This parameter can be adjusted within a range from 0 (immediate update) to 8 (very slow AGC response). Iterations time is typically 50ms.

### 5.3.3. Dynamic contrast enhancement (CLHE)

The camera includes a dynamic contrast enhancement algorithm in the visible channel, similar to the thermal infrared channel, that maximizes the scene's contrast based on histogram equalization techniques: CLHE (Contrast-Limited Histogram Equalization).

The CLHE algorithm gathers the image histogram over a user defined ROI (Region of Interest). This algorithm has a uniform impact on the image, but the overall result is dependent on the hottest point inside the ROI.

The level of contrast correction is adjustable by the end user to set the scene output contrast.

# 6. VIDEO OUTPUTS

| Framerate | Parallel | HD-SDI/3G-SDI |
|---|---|---|
| 25Hz | Supported | Supported |
| 30Hz | | |
| 50 Hz | WIP | WIP |
| 60 Hz | | |

**Table 13: CamSight FB interfaces framerate**

| Electronics Boards | Parallel | HD-SDI |
|---|---|---|
| Base OEM | Supported | Supported |

**Table 14: CamSight FB interfaces hardware availability**

## 6.1. Parallel video output

CamSight FB outputs the Visible/IR/Fusion video via a parallel16 bits interface. The parallel interface signals are detailed in Table 15 and a chronogram is given in Figure 23.

NOTE: The pixel output framing is not fixed. Pixels are delivered when available. With high zoom values, bursts of successive lines can be observed.

| Pin Name | Signal name | Description |
|---|---|---|
| VIDEO_OUT_PCLK | Pixel Clock | Synchronization and data signals are set on falling edge of PCLK. |
| VIDEO_OUT_HSYNC | Horizontal Synchronization | Signals the start and end of a line of pixels excluding blanking.<br>Logic ‘1’ if outputting a horizontal line of pixels.<br>Logic ‘0’ otherwise |
| VIDEO_OUT_VSYNC | Vertical Synchronization | Signals the start and end of the image excluding blanking.<br>Logic ‘1’ if outputting an image.<br>Logic ‘0’ otherwise |
| VIDEO_OUT_DE | Data Enable | Confirms valid data is transmitted.<br>Prior to firmware version v2.1.1, the DE signal can fall to '0' at any time during a line.<br>Starting from firmware version v2.1.1, the DE signal remains at '1' throughout the entire line. |
| VIDEO_OUT_D0 to VIDEO_OUT_D15 | Video bus | Video data transmitted over 16 bits with a YCb-Cr:4:2:2 format.<br>• D0 -7: Chrominance CbCr (Cb first)<br>• D8-15: Luminance Y |

**Table 15: Parallel output signals**

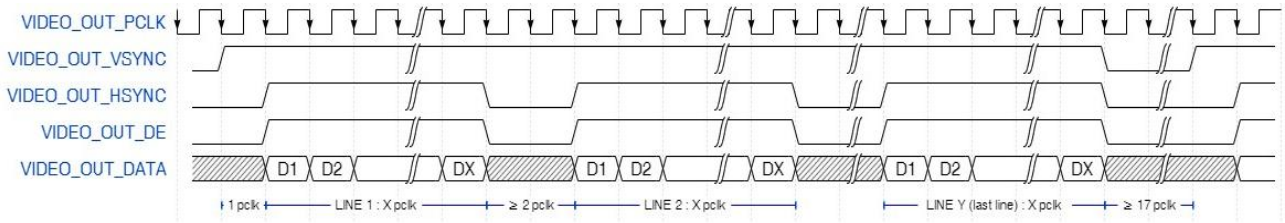



**Figure 23 : Parallel video framing with resolution XxY (Firmware v2.1.1 and above)**

## 6.2. HD-SDI output

CamSight FB offers an HD-SDI video output with progressive video data, meaning the odd field and the even field are time related. This video presentation method is less prone-to-artifacts than regular HD-SDI video.

The progressive HD-SDI interface transmits video signals with a resolution of 1920x1080 pixels. The interface is 75ohm and transmits a digital signal at a rate of 2Gb/s. The video signal is encoded using 8-bit digital coding and is transmitted in a serial format.

# 7. SOFTWARE INTERFACE

## 7.1. Protocols

The CamSight® camera is always acts as a slave device. Therefore, a master device must be connected to the camera to initiate communication in any protocol.

### 7.1.1. MAVLink V2

The CamSight MAVLink Protocol is based on MAVLink v2 protocol. All transactions are composed of 1 command packet sent by the master device and 1 response packet sent by the camera.

#### 7.1.1.1. MAVLink versioning

The CamSight MAVLink Protocol is versioned. All devices implementing this protocol must ensure compatibility by adhering to the protocol implementation guidelines.

The `GET_CAMERA_PROTOCOL_VERSION` command is used by the master device to identify the available camera command set.

Historical versions of the CamSight MAVLink Protocol is not versioned (doesn't implement the `GET_CAMERA_PROTOCOL_VERSION` command) and is identified in this document as version 0.

#### 7.1.1.2. MAVLink generator

Pymavgen is used to generate automatically the MAVLink code libraries to interface with the Camera: https://github.com/mavlink/mavlink

This tool can generate the MAVLink protocol implementation in a variety of languages, from embedded C to Python or C#, Rust etc.

The library generator program 'mavgenerate.py' takes as input an XML file describing the command set to be implemented in the library. An XML file is available for each version of CamSight MAVLink Protocol on request.

The output of the MAVLink library generator is a file or directory of files depending on the target language.

**WARNING:** The generated output should be used directly as a library and not modified, if regenerated, the modified files will be overwritten. Prefer creating wrappers instead of modifying code directly.

The MAVLink library sits as a data encoder/decoder, the programmer must use the higher level API to send and receive messages as well as indicate to the generated library how to interface with the hardware serial communication (read/write functions).

Here is an example of this implementation for a C language target:

```
#ifndef MAVLINK_H
#define MAVLINK_H
//------------------------------------------------------------------------------
// HRo: configuration part for MavLink messaging
#include "mavlink_types.h"
#include "serial_interface.h"

#define MAVLINK_USE_CONVENIENCE_FUNCTIONS          1

#define MAVLINK_SEND_UART_BYTES                    serial_interface_send_bytes
extern mavlink_system_t mavlink_system;
//------------------------------------------------------------------------------
```

#### 7.1.1.3. Channel addressing

In the CamSight FB cameras, a CamSight LS serves as the IR camera providing the IR video stream. CamSight LS already integrates a MAVLink serial link to configure and operate the IR camera, but additional functionality is needed to operate the Visible and Fusion video streams.

The MAVLink control architecture of the CamSight FB is given in Figure 24.

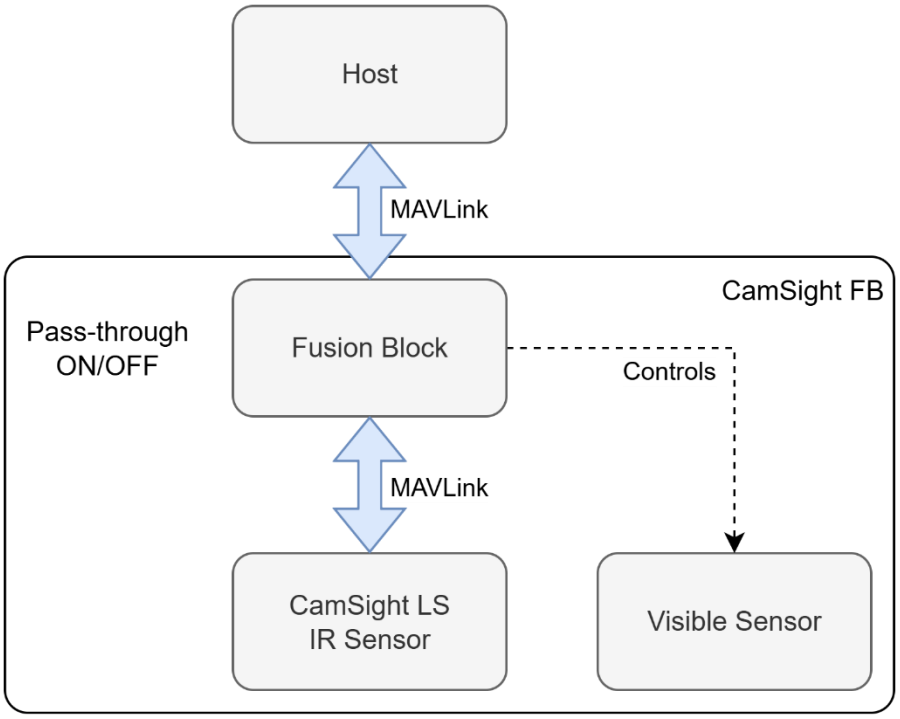



**Figure 24: MAVLink control architecture**

Therefore, a MAVLink pass-through has been integrated in the Fusion Block allowing for MAVLink commands addressed to the IR Sensor to be forwarded to the CamSight LS subsystem.

This passthrough is controlled through by sending the `SET_TARGET_MSG` command to toggle this pass-through on/off.

#### 7.1.1.4. MAVLink transactions

The communication between the camera and the host is structured as a “Request/Answer” or “Command/Response” exchange. The camera is in a slave role and only responds to commands sent by the host as shown in Figure 25.

The answers for SET commands are `MESSAGE_ACK/MESSAGE_NACK` packets for SET messages. For GET messages, the same message is echoed with the PAYLOAD section filled by the camera.

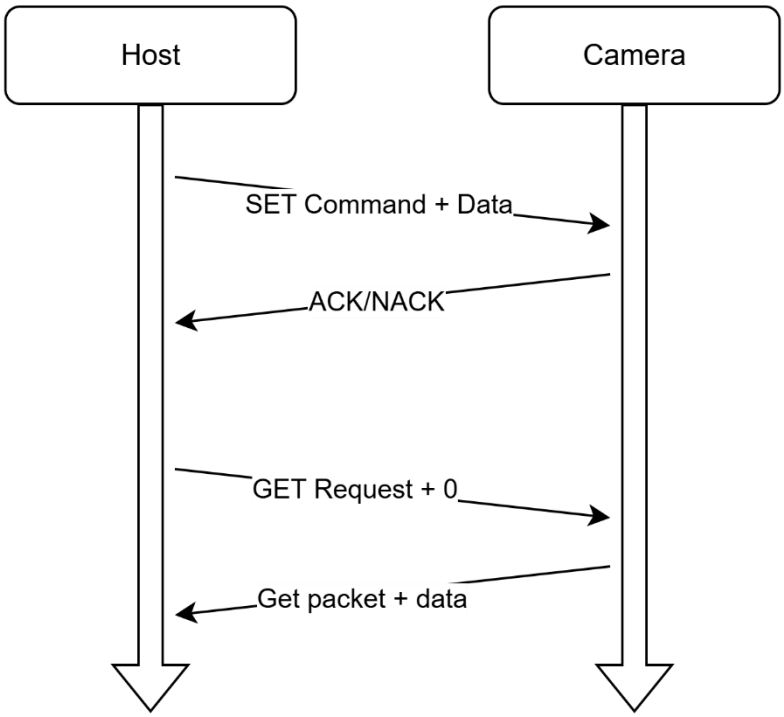



**Figure 25: MAVLink communication flow**

#### 7.1.1.4.1. Master to Slave command/request

Requests/Commands are sent from the host to the camera. Two types or Requests/Commands exist:

- GET commands, asking the camera to send back information
- SET commands, setting parameters values and configuration

All requests fall into these two categories, even if they don't contain the words "Get" or "Set".

Every Request/Command is structured as indicated by Table 1.

| B0 | B1 | B2-B3 | B4 | B5-B6 | B7-B9 | B10-B(N+9) | B(N+10)-B(N+11) |
|---|---|---|---|---|---|---|---|
| 0xFD | LEN | RESERVED | SEQ | RESERVED | MID | PAYLOAD | CRC16 |

**Table 1: MAVLink command format**

Bytes:

- 0xFD: Magic number/start byte for MAVLink V2 communication.
- LEN: Length of the PAYLOAD, expressed in bytes; can be from 0 to 255 bytes.
- RESERVED: Must be kept 0.
- SEQ: Sequence number, incremented at each message sent/received.
- RESERVED: Must be kept 0.
- MID: Message ID (24 bits) of the sent command.
- PAYLOAD of LEN bytes:
  - Command data depends on the command type.
  - 0 in case of GET commands
- CRC16: 16-bit CRC (MCRF4XX) computed over B1 to B(N+9) bytes.

NOTE: Each MAVLink packet is coded in little endian format.

#### 7.1.1.4.2. Slave to Master acknowledgement/response

Every request command is followed after some time by an acknowledgement or an answer (in the case of a GET command) whose structure is indicated by Table 2.

| B0 | B1 | B2-B3 | B4 | B5-B6 | B7-B9 | B10-(N+9) | B(N+10)-B(N+11) |
|---|---|---|---|---|---|---|---|
| 0xFD | LEN | RESERVED | SEQ | RESERVED | MID | PAYLOAD | CRC16 |

**Table 2: MAVLink answer format**

Bytes:

- 0xFD: Magic number/start byte for MAVLink V2 communication.
- LEN: Length of the PAYLOAD, expressed in bytes; can be from 0 to 255 bytes.
- RESERVED: Reserved value, must be kept 0.
- SEQ: Sequence number, incremented at each message sent/received.
- RESERVED: Reserved value, must be kept 0.
- MID: Message ID (24 bits) of the answered Request/Command.
- PAYLOAD of LEN bytes:
  - Requested data for GET commands
  - MESSAGE_ACK packet data for acknowledgement
- CRC16: 16-bit CRC (MCRF4XX) computed over all message bytes.

NOTE: Each MAVLink packet is coded in little endian format.

#### 7.1.1.4.3. Timing and errors recovery

The `MESSAGE_ACK` message is used as the default answer for SET commands. Unknown/Unimplemented messages are answered with a `MESSAGE_NACK` packet.

If a command fails to execute/raise an error, a `MESSAGE_NACK` packet is also emitted.

| Value | Min | Mean | Max |
|---|---|---|---|
| Time between message and answer | | 100ms | 1.5s |
| Number of recommended retries | 1 | 3 | |

**Figure 26: Error recovery recommendations**

#### 7.1.1.4.4. Streaming protocol

A streaming protocol is used to allow transferring huge amounts of data through 2 specific streaming commands. These streams can be either read or write streams.

The write and read streaming transfers are built around MAVLink transactions extended protocol messages: `WRITE_STREAM` and `READ_STREAM`.

##### 7.1.1.4.4.1. Write Stream Transfer format

The MAVLink master always initiates the transaction with a `WRITE_STREAM_START` indicating the type of data transferred, its length and destination address, the slave acknowledges the stream start by sending back a `MESSAGE_ACK`. If the slave sends a `MESSAGE_NACK` packet instead, the stream is aborted.

The stream packets are then sent by the master and individually acknowledged or not the slave device responding with `MESSAGE_ACK` or `MESSAGE_NACK` packet.

If a `MESSAGE_ACK` packet is received by the master, the next write packet is sent with its SEQ control byte incremented.

However, if a MESSAGE_NACK packet is received, the master resends the same stream packet with same SEQ control byte. If the slave device keeps sending a MESSAGE_NACK after this 2nd try at least 2 more times before cancelling the transfer.

At the end of a successful transfer, the master sends a STREAM_END packet containing the 32 bits CRC computed over all sent data bytes (PAYLOAD bytes only).

Note that in case of a failed transaction (MESSAGE_NACK response), the packet data should still be counted in the final CRC, as the local CRC is updated with each incoming packet, regardless of the transaction success.

The slave device computes and sends back the CRC32 of all received data (PAYLOAD bytes only) and sends it back via a STREAM_END packet.

The sequence of packets in a Read Stream Transfer is detailed in Table 16.

| Master request | Slave response |
|---|---|
| WRITE_STREAM_START | MESSAGE_ACK |
| WRITE_STREAM | MESSAGE_ACK |
| ... | ... |
| WRITE_STREAM | MESSAGE_ACK |
| STREAM_END | STREAM_END |

**Table 16: MAVLink Stream write transactions**

#### 7.1.1.4.4.2. Read Stream Transfer format

The MAVLink master always initiates the transaction with a READ_STREAM_START indicating the type of data transferred, its length and starting address, the slave acknowledges the stream start by sending back a MESSAGE_ACK. If the slave sends a MESSAGE_NACK packet instead, the stream is aborted.

The stream packets are then sent by the master and individually acknowledged or not the slave device responding with a READ_STREAM containing the read data byte or a MESSAGE_NACK packet indicating an error has occurred.

If a READ_STREAM packet is received by the master, the next read packet is sent with its SEQ control byte incremented.

However, if a MESSAGE_NACK packet is received, the master resends the same stream packet with same SEQ control byte. If the slave device keeps sending a MESSAGE_NACK after this 2nd try at least 2 more times before cancelling the transfer.

At the end of a successful transfer, the master sends a STREAM_END packet containing the 32 bits CRC computed over all sent data bytes (PAYLOAD bytes only).

Note that in case of a failed transaction (MESSAGE_NACK response), the packet data should still be counted in the final CRC, as the local CRC is updated with each incoming packet, regardless of the transaction success.

The slave device computes and sends back the CRC32 of all received data (PAYLOAD bytes only) and sends it back via a STREAM_END packet.

The sequence of packets in a Read Stream Transfer is detailed in Table 17.

| Master request | Slave response |
|---|---|
| READ_STREAM_START | MESSAGE_ACK |
| READ_STREAM | READ_STREAM |
| ... | ... |
| READ_STREAM | READ_STREAM |
| STREAM_END | STREAM_END |

**Table 17: MAVLink Stream read transactions**

### 7.1.2. MAVLink reference

#### 7.1.2.1. CAMERA COMMON USER

##### 7.1.2.1.1. MESSAGE_ACK

[RESERVED] Generic message to acknowledge messages and receive errors

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | COMMAND | | | | VALUE | | |

| B16 | B17 | B18 | B19 |
|---|---|---|---|
| VALUE | RESULT | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 9 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 8192 (0x2000)
- **B9-B12:** COMMAND Of type `uint32_t`. Acknowledged message ID
- **B13-B16:** VALUE Of type `uint32_t`, defaults to 0. Optional additional information.
- **B17:** RESULT Of type `uint8_t`. Message result

  Allowed values are defined by the MESSAGE_ACK_RESULT enum:
  - 0: MESSAGE_ACK_OK Message processed successfully
  - 1: MESSAGE_ACK_NOK Message processing failed

- **B18-B19:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.2. GET_SERIALNUMBER

[GET] Get the camera serial number (SN)

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | SERIAL_NUMBER | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 8194 (0x2002)
- **B9-B12:** SERIAL_NUMBER Of type `uint32_t`. Camera serial number
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.3. SHUTTER_CONTROL

[SET] Direct camera shutter control

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | COMMAND | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 8206 (0x200E)
- **B9:** COMMAND Of type `uint8_t`. Shutter command
  Allowed values are defined by the SHUTTER_CONTROL_CMD enum:
  - 0: SHUTTER_CONTROL_CMD_OPEN Open the camera shutter
  - 1: SHUTTER_CONTROL_CMD_CLOSE Close the camera shutter
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.4. GET_SHUTTER_POSITION

[GET] Get position of the shutter

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | SHUTTER_ CTRL | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12363 (0x304B)
- **B9:** SHUTTER_CTRL Of type `uint8_t`. Shutter command
  Allowed values are defined by the SHUTTER_CONTROL_CMD enum:
  - 0: SHUTTER_CONTROL_CMD_OPEN Open the camera shutter
  - 1: SHUTTER_CONTROL_CMD_CLOSE Close the camera shutter
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.5. SHUTTER_CHECK_PRESENCE

[GET] Check if the camera is equipped with a mechanical shutter

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | IS_PRESENT | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 9000 (0x2328)
- **B9:** IS_PRESENT Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.6. GET_CAMERA_PROT_VERS

[GET] Get the MAVLink protocol version

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | CAMERA_ PROTOCOL_ VERSION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12287 (0x2FFF)
- **B9:** CAMERA_PROTOCOL_VERSION — Of type `uint8_t`. MAVLINK camera communication protocol version
  Allowed values are defined by the CAMERA_PROTOCOL_VERSION enum:
  - 6: PROTOCOL_V6 — Camera communication protocol version 6
  - 5: PROTOCOL_V5 — Camera communication protocol version 5
  - 4: PROTOCOL_V4 — Camera communication protocol version 4
  - 3: PROTOCOL_V3 — Camera communication protocol version 3
  - 2: PROTOCOL_V2 — Camera communication protocol version 2
  - 1: PROTOCOL_V1 — Camera communication protocol version 1
  - 0: PROTOCOL_V0 — Camera communication protocol version 0
- **B10-B11:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.7. SET_GAMMA

[SET] Set the gamma contrast curve on IR sensor

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | VALUE | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12290 (0x3002)
- **B9-B12:** VALUE Of type `uint32_t`, between 327678 (min) and 163840 (max).
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.8. SET_CONTRAST

[SET] Set the histogram clip value for the CLHE/CLAHE algorithms

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | VALUE | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12292 (0x3004)
- **B9-B12:** VALUE Of type `uint32_t`, between 0 (min) and 30000 (max). CLHE/CLAHE clip threshold
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.9. INVERT_POLARITY

[SET] Set the camera contrast polarity

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12294 (0x3006)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.10. ROI_CONTROL

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12297 (0x3009)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.11. CONTRAST_CONTROL

[SET] Change the image processing contrast algorithm in use

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | TYPE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12300 (0x300C)
- **B9:** TYPE Of type `uint8_t`. Contrast algorithm used
  Allowed values are defined by the CONTRAST_TYPE enum:
  - 0: CONTRAST_CLHE Contrast algorithm used is CLHE (Contrast Limited Histogram Equalization)
  - 1: CONTRAST_CLAHE Contrast algorithm used is CLAHE (Contrast Limited Adaptative Histogram Equalization)
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.12. CAMERA_STATUS

[GET] Get complete status of the camera

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | CONTRAST | | | | LUMINOSITY | | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| LUMINOSITY | FOCUS_ ERROR | SHUTTER_ ERROR | FOCUS_ MODE | FOCUS_ ACTION | FOCUS_POSITION | | |

| B24 | B25 | B26 | B27 | B28 | B29 |
|---|---|---|---|---|---|
| FOCUS_ POSITION | NUC_MODE | NUC_STATUS | IR_POLARITY | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 19 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12303 (0x300F)
- **B9-B12:** CONTRAST — Of type `uint32_t`. Current CLHE/CLAHE clip threshold, see [SET_CONTRAST]
- **B13-B16:** LUMINOSITY — Of type `uint32_t`. Current gamma value, see [SET_GAMMA]
- **B17:** FOCUS_ERROR — Of type `uint8_t`.
- **B18:** SHUTTER_ERROR — Of type `uint8_t`. Shutter status
- **B19:** FOCUS_MODE — Of type `uint8_t`.

- **B20:** FOCUS_ACTION Of type `uint8_t`.
- **B21-B24:** FOCUS_POSITION Of type `uint32_t`, between -100 (min) and 100 (max). [Deprecated] Previously used for cameras with focusing abilities
- **B25:** NUC_MODE Of type `uint8_t`. Current NUC mode, see [NUC_CONTROL]
- **B26:** NUC_STATUS Of type `uint8_t`.
- **B27:** IR_POLARITY Of type `uint8_t`. Current image polarity, see [INVERT_POLARITY]
- **B28-B29:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.13. SET_CUSTOM_SPEED

[SET] Enable/Disable custom MAVLink UART speed

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12308 (0x3014)
- **B9:** ENABLE Of type `int8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.14. SET_ZOOM_PARAMS

[SET] Set the camera zoom parameters

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_FACTOR | | | | Y_FACTOR | | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| Y_FACTOR | X_CENTER | | | | Y_CENTER | | |

| B24 | B25 | B26 |
|---|---|---|
| Y_CENTER | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 16 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12310 (0x3016)
- **B9-B12:** X_FACTOR Of type `uint32_t`, between 65536 (min) and 524288 (max).
- **B13-B16:** Y_FACTOR Of type `uint32_t`, between 65536 (min) and 524288 (max).
- **B17-B20:** X_CENTER Of type `uint32_t`. X zoom center, horizontal distance in pixels from image top-left corner
- **B21-B24:** Y_CENTER Of type `uint32_t`. Y zoom center, vertical distance in pixels from image top-left corner
- **B25-B26:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.15. SET_ZOOM_METHOD

[SET] Set the zoom algorithm used

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | METHOD | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12311 (0x3017)
- **B9:** METHOD Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.16. GET_ROI

[GET] Get the Region Of Interest's coordinates used for the CLHE histogram calculation

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X1 | | X2 | | Y1 | | Y2 |

| B16 | B17 | B18 |
|---|---|---|
| Y2 | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12315 (0x301B)
- **B9-B10:** X1 Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X2 Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y1 Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y2 Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.17. GET_ZOOM_CONFIG

[GET] Get the complete zoom configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_FACTOR | | | | Y_FACTOR | | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| Y_FACTOR | X_CENTER | | | | Y_CENTER | | |

| B24 | B25 | B26 | B27 |
|---|---|---|---|
| Y_CENTER | METHOD | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 17 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12316 (0x301C)
- **B9-B12:** X_FACTOR Of type `uint32_t`, between 65536 (min) and 524288 (max). X resize factor, see [SET_ZOOM_PARAMS]
- **B13-B16:** Y_FACTOR Of type `uint32_t`, between 65536 (min) and 524288 (max). Y resize factor, see SET_ZOOM_PARAMS
- **B17-B20:** X_CENTER Of type `uint32_t`. X zoom center, see [SET_ZOOM_PARAMS]
- **B21-B24:** Y_CENTER Of type `uint32_t`. Y zoom center, see [SET_ZOOM_PARAMS]
- **B25:** METHOD Of type `uint8_t`. Zoom method, see [SET_ZOOM_METHOD]
- **B26-B27:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.18. GET_CONTRAST_TYPE

[GET] Get IR sensor contrast algorithm used

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | TYPE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12320 (0x3020)
- **B9:** TYPE Of type `uint8_t`. Contrast algorithm used
  Allowed values are defined by the CONTRAST_TYPE enum:
  - 0: CONTRAST_CLHE Contrast algorithm used is CLHE (Contrast Limited Histogram Equalization)
  - 1: CONTRAST_CLAHE Contrast algorithm used is CLAHE (Contrast Limited Adaptative Histogram Equalization)
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.19. GET_FIRMWARE_ID

[GET] Get both FPGA and RISCV firmware ID

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | FPGA_VERSION | | RISCV_VERSION | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12321 (0x3021)
- **B9-B10:** FPGA_VERSION — Of type `uint16_t`. FPGA build version
- **B11-B12:** RISCV_VERSION — Of type `uint16_t`. RISCV build version
- **B13-B14:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.20. SET_FLIP_H

[SET] Enable/Disable horizontal flip

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12323 (0x3023)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.21. GET_FLIP_H

[GET] Get the horizontal flip status

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12322 (0x3022)
- **B9:** ENABLE Of type `uint8_t`. Horizontal flip status, see [SET_FLIP_H]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.22. SET_FLIP_V

[SET] Enable/Disable vertical flip

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12325 (0x3025)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.23. GET_FLIP_V

[GET] Get the vertical flip status

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12324 (0x3024)
- **B9:** ENABLE Of type `uint8_t`. Vertical flip status, see [SET_FLIP_V]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.24. SET_COLUMN_CORRECTION

[SET] Enable/Disable the IR sensor column correction

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12326 (0x3026)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.25. GET_COLUMN_CORRECTION

[GET] Get IR sensor column correction value

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12327 (0x3027)
- **B9:** VALUE Of type `uint8_t`. Column correction status, see [SET_COLUMN_CORRECTION]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.26. SET_VIGNETTING_CORRECTION

[SET] Enable/Disable IR sensor vignetting correction

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12328 (0x3028)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.27. GET_VIGNETTING_CORRECTION

[GET] Get IR sensor vignetting correction value

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12329 (0x3029)
- **B9:** VALUE Of type `uint8_t`. Vignetting correction status, see [SET_VIGNETTING_CORRECTION]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.28. SET_VIDEO_SELECT

[SET] Set the camera's video output format

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12332 (0x302C)
- **B9:** VALUE Of type `uint8_t`. Video output selection
  Allowed values are defined by the VIDEO_SELECT_PARAM enum:
  - 0: RAW_OUTPUT Raw mode 640x480, 14 bits, Parallel format
  - 1: NUC_OUTPUT NUC mode 640x480, 14 bits, Parallel format
  - 2: COLUMN_OUTPUT Column correction 640x480, 14 bits, Parallel format
  - 3: VIGNETTING_OUTPUT Vignetting correction 640x480, 14 bits, Parallel format
  - 4: EDGE_OUTPUT Edge correction 640x480, 14 bits, Parallel format
  - 5: LUT_OUTPUT LUT correction 640x480, 8 bits, Parallel format
  - 6: RFU_OUTPUT Reserved for Future Use
  - 7: ZOOM_OUTPUT Zoom 720x576, 8 bits, Parallel format
  - 8: BT656_OUTPUT BT656 720x576, 14 bits, BT656 format
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.29. GET_VIDEO_SELECT

[GET] Get the camera's video output format

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12333 (0x302D)
- **B9:** VALUE Of type `uint8_t`. Video output selection
  Allowed values are defined by the VIDEO_SELECT_PARAM enum:
  - 0: RAW_OUTPUT Raw mode 640x480, 14 bits, Parallel format
  - 1: NUC_OUTPUT NUC mode 640x480, 14 bits, Parallel format
  - 2: COLUMN_OUTPUT Column correction 640x480, 14 bits, Parallel format
  - 3: VIGNETTING_OUTPUT Vignetting correction 640x480, 14 bits, Parallel format
  - 4: EDGE_OUTPUT Edge correction 640x480, 14 bits, Parallel format
  - 5: LUT_OUTPUT LUT correction 640x480, 8 bits, Parallel format
  - 6: RFU_OUTPUT Reserved for Future Use
  - 7: ZOOM_OUTPUT Zoom 720x576, 8 bits, Parallel format
  - 8: BT656_OUTPUT BT656 720x576, 14 bits, BT656 format
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.30. SET_SDI_OUTPUT

[SET] Enable/Disable the SDI video output

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12342 (0x3036)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.31. GET_SDI_OUTPUT

[GET] Get output video SDI status

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12343 (0x3037)
- **B9:** VALUE Of type `uint8_t`. SDI video status, see [SET_SDI_OUTPUT]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.32. SET_PARALLEL_OUTPUT

[SET] Enable/Disable the Parallel video output

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12344 (0x3038)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.33. GET_PARALLEL_OUTPUT

[GET] Get output video Parallel status

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12345 (0x3039)
- **B9:** VALUE Of type `uint8_t`. Parallel video status, see [SET_PARALLEL_OUTPUT]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.34. SET_MIPI_OUTPUT

[SET] Enable/Disable the MIPI video output

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12346 (0x303A)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.35. GET_MIPI_OUTPUT

[GET] Get output video MIPI status

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12347 (0x303B)
- **B9:** VALUE Of type `uint8_t`. MIPI video status, see [SET_MIPI_OUTPUT]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.36. GET_BIT

[GET] Get the Built In Test register. See documentation for details.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | BIT | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12358 (0x3046)
- **B9-B12:** BIT Of type `uint32_t`. Built in test
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.37. GET_CAMERA_TEMPERATURE

[GET] Get the system temperatures. See documentation for details.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | FPGA_TEMPERATURE | | | | SENSOR_TEMPERATURE | | |

| B16 | B17 | B18 |
|---|---|---|
| SENSOR_TEMPERATURE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12359 (0x3047)
- **B9-B12:** FPGA_TEMPERATURE — Of type `uint32_t`, in mK. FPGA temperature expressed in mK
- **B13-B16:** SENSOR_TEMPERATURE — Of type `uint32_t`, in mK. External sensor temperature expressed in mK
- **B17-B18:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.38. GET_TYPE

[GET] Get the camera and sensor type

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | TYPE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12288 (0x3000)
- **B9:** TYPE Of type `uint8_t`. Camera model and sensor type

  Allowed values are defined by the CAMERA_TYPE enum:
  - 0: TYPE_VISIBLE Visible sensor
  - 1: TYPE_INFRARED Infrared sensor
  - 2: CAMSIGHT_LS CamSight LS camera model
  - 3: CAMSIGHT_HD CamSight HD camera model
  - 4: CAMSIGHT_HDLP CamSight HD LP camera model
  - 5: CAMSIGHT_LP CamSight LP camera model
  - 6: FOR_IRGC FOR IR GC camera model
  - 7: FOR_IRPC FOR IR PC camera model
  - 8: FOR_VIS FOR Visible camera model
  - 9: SMARTSIGHT_IR SmartSight IR camera model
  - 10: SMARTSIGHT_VIS SmartSight Visible camera model

- 11: CAMSIGHT_METEO CamSight Meteo camera model
- 12: CAMSIGHT_IA CamSight IA camera model
- 13: CAMAXE CamAxe camera model
- 21: CAMSIGHT_FUSION_BLOCK CamSight Fusion Block model

- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.39. GET_RESOLUTION

[GET] Get the camera image size

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | WIDTH | | | | HEIGHT | | |

| B16 | B17 | B18 |
|---|---|---|
| HEIGHT | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12289 (0x3001)
- **B9-B12:** WIDTH Of type `uint32_t`. Sensor image width in pixels
- **B13-B16:** HEIGHT Of type `uint32_t`. Sensor image height in pixels
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.40. FOCUS_CONTROL

[SET] Set the focus mode

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | TYPE | | | | VALUE | | |

| B16 | B17 | B18 |
|---|---|---|
| VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12293 (0x3005)
- **B9-B12:** TYPE Of type `uint32_t`. Focus mode
  Allowed values are defined by the FOCUS_EVENT enum:
  - 0: FOCUS_GOTO Set motor target position
  - 1: FOCUS_SPEED Set motor speed
  - 2: FOCUS_AUTO Switch to auto mode
  - 3: FOCUS_MANUAL Switch to manual mode
  - 4: FOCUS_FAST Start fast research mode
  - 5: FOCUS_FULL Start Full research mode
- **B13-B16:** VALUE Of type `uint32_t`. Focus position or speed
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.41. NUC_CONTROL

[SET] Set Non-Uniformity Correction mode

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | MODE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12295 (0x3007)
- **B9:** MODE Of type `uint8_t`. Enable or disable the NUC correction
  Allowed values are defined by the NUC_MODE enum:
  - 0: NUC_DISABLE Disable NUC correction on the video pipe.
  - 1: NUC_AUTO_TEMPERATURE NUC is automatically managed by camera, auto-triggers a NUC request when the temperature drifts past a certain threshold
  - 2: NUC_ENABLE Enable NUC correction on the video pipe.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.42. NUC_REQUEST

[SET] Trigger the acquisition of a new Non-Uniformity Correction

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | OPTION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12296 (0x3008)
- **B9:** OPTION Of type `uint8_t`. NUC request type
  Allowed values are defined by the NUC_REQUEST_OPTION enum:
  - 0: NUC_REQUEST_OPTION_NONE No option. NUC is directly processed
  - 1: NUC_REQUEST_OPTION_WITH_SHUTTER Process the NUC with shutter closed, automatically reopens the camera shutter after NUC
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.43. ENABLE_GAIN

[SET] Enable/Disable gain correction

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12312 (0x3018)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.44. ENABLE_OFFSET

[SET] Enable/Disable offset correction

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12313 (0x3019)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.45. ENABLE_BPR

[SET] Enable/Disable bad pixel replacement

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | ENABLE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12314 (0x301A)
- **B9:** ENABLE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.46. GET_SENSOR_CONFIG

[GET] Return complete IR sensor configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | GSK | | | | GFID | | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| GFID | GMS | | | | TINT | | |

| B24 | B25 | B26 | B27 | B28 | B29 |
|---|---|---|---|---|---|
| TINT | GAIN_ ENABLED | OFFSET_ ENABLED | BPR_ ENABLED | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 19 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12317 (0x301D)
- **B9-B12:** GSK — Of type `uint32_t`, in mV. GSK value
- **B13-B16:** GFID — Of type `uint32_t`, in mV. GFID value
- **B17-B20:** GMS — Of type `uint32_t`. GMS value
- **B21-B24:** TINT — Of type `uint32_t`. TINT value
- **B25:** GAIN_ENABLED — Of type `uint8_t`. Gain correction status

- **B26:** OFFSET_ENABLED Of type `uint8_t`. Offset correction status
- **B27:** BPR_ENABLED Of type `uint8_t`. BPR correction status
- **B28-B29:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.47. SAVE_PARAMS

[SET] Make a request to store the parameters

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | PARAMS_ TYPE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12362 (0x304A)
- **B9:** PARAMS_TYPE Of type `uint8_t`. The type of parameters to store
  Allowed values are defined by the CAMERA_PARAMS_TYPE enum:
  - 0: CAMERA_PARAMS_USER User facing parameters, documented and editable by users
  - 1: CAMERA_PARAMS_FACTORY Factory settings, not user editable
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.48. SET_TRIG_MODE

[SET] Set the trigger mode, internal or external, in the trigger IP

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | MODE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12364 (0x304C)
- **B9:** MODE Of type `uint8_t`. Trigger source
  Allowed values are defined by the MAV_TRIG_MODE_ENUM enum:
  - 0: MAV_TRIG_MODE_INTERNAL New frame is automatically triggered by the camera
  - 1: MAV_TRIG_MODE_EXTERNAL New frame is triggered externally using the VIDEO_TRIG signal
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.1.49. GET_TRIG_MODE

[GET] Get the trigger mode adn the status of the trigger IP

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | MODE | STATUS | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 5 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12365 (0x304D)
- **B9:** MODE Of type `uint8_t`. Trigger source
  Allowed values are defined by the MAV_TRIG_MODE_ENUM enum:
  - 0: MAV_TRIG_MODE_INTERNAL New frame is automatically triggered by the camera
  - 1: MAV_TRIG_MODE_EXTERNAL New frame is triggered externally using the VIDEO_TRIG signal
- **B10-B13:** STATUS Of type `uint32_t`.
- **B14-B15:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.50. SET_OUTPUT_ENDIANESS

[SET] Set the parallel output endianness

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12366 (0x304E)
- **B9:** VALUE Of type `uint8_t`.
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.1.51. GET_OUTPUT_ENDIANESS

[GET] Get the parallel output endianness

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12367 (0x304F)
- **B9:** VALUE Of type `uint8_t`. Output endianness status, see [SET_OUTPUT_ENDIANESS]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2. FUSION BLOCK USER

##### 7.1.2.2.1. GET_FUSION_MODE

[GET] Get the fusion block output configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16640 (0x4100)
- **B9:** VALUE Of type `uint8_t`. Video output type
  Allowed values are defined by the FUSION_VIDEO_MODE enum:
  - 0: VIDEO_MODE_IR IR video output only
  - 1: VIDEO_MODE_VIS Visible video output only
  - 2: VIDEO_MODE_FUSION Fusion of IR and Visible video output
  - 3: VIDEO_MODE_USE_CURRENT Current output (could be IR, Visible or Fusion)
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.2. SET_FUSION_MODE

[SET] Set the fusion block output configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16641 (0x4101)
- **B9:** VALUE Of type `uint8_t`. Video output type
  Allowed values are defined by the FUSION_VIDEO_MODE enum:
  - 0: VIDEO_MODE_IR IR video output only
  - 1: VIDEO_MODE_VIS Visible video output only
  - 2: VIDEO_MODE_FUSION Fusion of IR and Visible video output
  - 3: VIDEO_MODE_USE_CURRENT Current output (could be IR, Visible or Fusion)
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.3. SET_TARGET_MSG

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | TARGET | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16642 (0x4102)
- **B9:** TARGET Of type `uint8_t`. MAVLink target to address future messages
  Allowed values are defined by the TARGET_MSG enum:
  - 0: FUSION_BLOCK MAVLink pass-through disabled
  - 1: IR_CAMERA MAVLink pass-through enabled
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.4. SET_VIS_AGC_PARAMS

[SET] Set the Automatic Gain Control algorithm parameters on the Visible channel.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 |
|---|---|---|---|---|---|
| MID | TARGET_ PERCENT | VALID_RANGE | ALPHA | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 3 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16643 (0x4103)
- **B9:** TARGET_PERCENT Of type `uint8_t`, in %. Target brightness, expressed as a percentage of the Visible video output range
- **B10:** VALID_RANGE Of type `uint8_t`. Brightness range around the target brightness outside of which a gain correction will be triggered, expressed as a percentage of the Visible video output range
- **B11:** ALPHA Of type `uint8_t`. Smoothing coefficient limiting gain variations between video frames, gain delta is limited to (100 * 1/2^alpha) % of the difference
- **B12-B13:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.5. SET_VIS_CLHE_PARAMS

[SET] Set the parameters of the Contrast Limited Histogram Equalization algorithm processed on the visible channel.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | THRESHOLD | | | | ALPHA | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 5 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16644 (0x4104)
- **B9-B12:** THRESHOLD Of type `uint32_t`. CLHE Histogram clip threshold. The pixels above this threshold will be redistributed over the whole histogram.
- **B13:** ALPHA Of type `uint8_t`. Smoothing coefficient limiting CLHE threshold variations between video frames, threshold delta is limited to (100 * 1/2^alpha) % of the difference
- **B14-B15:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.6. GET_VIS_AGC_PARAMS

[GET] Get the Automatic Gain Control algorithm parameters on the Visible channel

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 |
|---|---|---|---|---|---|
| MID | TARGET_ PERCENT | VALID_RANGE | ALPHA | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 3 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 16645 (0x4105)
- **B9:** TARGET_PERCENT — Of type `uint8_t`. Target brightness, see [SET_VIS_AGC_PARAMS]
- **B10:** VALID_RANGE — Of type `uint8_t`. Brightness threshold, see [SET_VIS_AGC_PARAMS]
- **B11:** ALPHA — Of type `uint8_t`. Smoothing coefficient, see [SET_VIS_AGC_PARAMS]
- **B12-B13:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.7. GET_VIS_CLHE_PARAMS

[GET] Get the parameters of the CLHE algorithm processed on the visible channel.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | THRESHOLD | | | | ALPHA | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 5 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16646 (0x4106)
- **B9-B12:** THRESHOLD Of type `uint32_t`. CLHE Histogram clip threshold, see [SET_VIS_CLHE_PARAMS]
- **B13:** ALPHA Of type `uint8_t`. Smoothing coefficient, see [SET_VIS_CLHE_PARAMS]
- **B14-B15:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.8. GET_FUSION_START_MODE

[GET] Get the fusion video start-up configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16647 (0x4107)
- **B9:** VALUE Of type `uint8_t`. Video output type at start-up
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.2.9. SET_FUSION_START_MODE

[SET] Set the fusion video start-up configuration

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | VALUE | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 16648 (0x4108)
- **B9:** VALUE Of type `uint8_t`. Video output type at start-up
  Allowed values are defined by the FUSION_VIDEO_MODE enum:
  - 0: VIDEO_MODE_IR IR video output only
  - 1: VIDEO_MODE_VIS Visible video output only
  - 2: VIDEO_MODE_FUSION Fusion of IR and Visible video output
  - 3: VIDEO_MODE_USE_CURRENT Current output (could be IR, Visible or Fusion)
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3. CAMSIGHT LS USER

#### 7.1.2.3.1. SET_FRAMERATE

[SET] Set video output framerate if the camera is not 9Hz locked

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | VALUE | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12330 (0x302A)
- **B9-B12:** VALUE Of type `uint32_t`, in Hz.
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.3.2. GET_FRAMERATE

[GET] Get video output framerate

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | VALUE | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12331 (0x302B)
- **B9-B12:** VALUE — Of type `uint32_t`, in Hz. Video framerate, see [SET_FRAMERATE]
- **B13-B14:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.3. SET_EDGE_PARAMS

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | MODE | GAIN | | | | THRESHOLD_1 | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| THRESHOLD_1 | | THRESHOLD_2 | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 13 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12334 (0x302E)
- **B9:** MODE Of type `uint8_t`. Type of edge filter
  Allowed values are defined by the EDGE_MODE_PARAM enum:
  - 0: EDGE_MODE_OFF No edge filter
  - 1: SHARPENING_MODE Sharpening filter
  - 2: CONTOURING_MODE Contouring filter
- **B10-B13:** GAIN Of type `uint32_t`. Global gain factor G, sharpening filter only, 8bits unsigned integer
- **B14-B17:** THRESHOLD_1 Of type `uint32_t`. Brightness threshold Th1, 8bits unsigned integer
- **B18-B21:** THRESHOLD_2 Of type `uint32_t`. Brightness threshold Th2, sharpening filter only, 8bits unsigned integer, must be stricly higher than Th1
- **B22-B23:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.3.4. GET_EDGE_PARAMS

[GET] Get video edge treatment parameters, see [SET_EDGE_PARAMS]

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | MODE | GAIN | | | | THRESHOLD_1 | |

| B16 | B17 | B18 | B19 | B20 | B21 | B22 | B23 |
|---|---|---|---|---|---|---|---|
| THRESHOLD_1 | | THRESHOLD_2 | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 13 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12335 (0x302F)
- **B9:** MODE Of type `uint8_t`. Type of edge filter
  Allowed values are defined by the EDGE_MODE_PARAM enum:
  - 0: EDGE_MODE_OFF No edge filter
  - 1: SHARPENING_MODE Sharpening filter
  - 2: CONTOURING_MODE Contouring filter
- **B10-B13:** GAIN Of type `uint32_t`. Global gain factor, sharpening filter only, 8bits unsigned integer
- **B14-B17:** THRESHOLD_1 Of type `uint32_t`. Brightness threshold Th1, 8bits unsigned integer
- **B18-B21:** THRESHOLD_2 Of type `uint32_t`. Brightness threshold Th2, sharpening filter only, 8bits unsigned integer, must be stricly higher than Th1
- **B22-B23:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.5. SET_ZOOM_OUTPUT_RESOLUTION

[SET] Set ouput zoom parameters

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_RESOLUTION | | | | Y_RESOLUTION | | |

| B16 | B17 | B18 |
|---|---|---|
| Y_ RESOLUTION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12338 (0x3032)
- **B9-B12:** X_RESOLUTION Of type `uint32_t`. Output image horizontal size, in pixels
- **B13-B16:** Y_RESOLUTION Of type `uint32_t`. Output image vertical size, in pixels
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.6. GET_ZOOM_OUTPUT_RESOLUTION

[GET] Get ouput zoom parameters

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_RESOLUTION | | | | Y_RESOLUTION | | |

| B16 | B17 | B18 |
|---|---|---|
| Y_ RESOLUTION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12339 (0x3033)
- **B9-B12:** X_RESOLUTION Of type `uint32_t`. Output image horizontal size, in pixels
- **B13-B16:** Y_RESOLUTION Of type `uint32_t`. Output image vertical size, in pixels
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.7. SET_COLORLUT

[SET] Set output video color mapping

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | COLORLUT_SEL | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12340 (0x3034)
- **B9-B12:** COLORLUT_SEL Of type `uint32_t`. LUT selection
  Allowed values are defined by the COLORLUT_PARAM enum:
  - 0: GREYSCALE Black and white greyscale, Mono 8bits format
  - 1: IRON Color LUT IRON, YCbCr:4:2:2 16bits format
  - 2: HOT Color LUT HOT, YCbCr:4:2:2 16bits format
  - 3: RAINBOW Color LUT RAINBOW, YCbCr:4:2:2 16bits format
  - 4: NCAR Color LUT NCAR, YCbCr:4:2:2 16bits format
  - 5: COOLWARM Color LUT COOLWARM, YCbCr:4:2:2 16bits format
  - 6: CUBEHELIX Color LUT CUBEHELIX, YCbCr:4:2:2 16bits format
  - 7: GREYSCALE_YCBCR Black and white greyscale, YCbCr:4:2:2 16bits format
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.8. GET_COLORLUT

[GET] Get output video color mapping

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 |
|---|---|---|---|---|---|---|
| MID | COLORLUT_SEL | | | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 4 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12341 (0x3035)
- **B9-B12:** COLORLUT_SEL Of type `uint32_t`. Selected LUT
  Allowed values are defined by the COLORLUT_PARAM enum:
  - 0: GREYSCALE Black and white greyscale, Mono 8bits format
  - 1: IRON Color LUT IRON, YCbCr:4:2:2 16bits format
  - 2: HOT Color LUT HOT, YCbCr:4:2:2 16bits format
  - 3: RAINBOW Color LUT RAINBOW, YCbCr:4:2:2 16bits format
  - 4: NCAR Color LUT NCAR, YCbCr:4:2:2 16bits format
  - 5: COOLWARM Color LUT COOLWARM, YCbCr:4:2:2 16bits format
  - 6: CUBEHELIX Color LUT CUBEHELIX, YCbCr:4:2:2 16bits format
  - 7: GREYSCALE_YCBCR Black and white greyscale, YCbCr:4:2:2 16bits format
- **B13-B14:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.9. SET_ROI1_inclusive

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12348 (0x303C)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.10. GET_ROI1_inclusive

[GET] Get the inclusive Region of Interest 1 coordinates, see [SET_ROI1_inclusive]

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12349 (0x303D)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.11. SET_ROI0_exclusive

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12350 (0x303E)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

#### 7.1.2.3.12. GET_ROI0_exclusive

[GET] Get the exclusive Region of Interest 0 coordinates, see [SET_ROI0_exclusive]

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12351 (0x303F)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.13. SET_ROI1_exclusive

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12352 (0x3040)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.14. GET_ROI1_exclusive

[GET] Get the exclusive Region of Interest 1 coordinates, see [SET_ROI1_exclusive]

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | X_START | | X_END | | Y_START | | Y_END |

| B16 | B17 | B18 |
|---|---|---|
| Y_END | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12353 (0x3041)
- **B9-B10:** X_START Of type `uint16_t`. ROI top-left corner horizontal distance from image top-left corner
- **B11-B12:** X_END Of type `uint16_t`. ROI bottom-right corner horizontal distance from image bottom-right corner
- **B13-B14:** Y_START Of type `uint16_t`. ROI top-left corner vertical distance from image top-left corner
- **B15-B16:** Y_END Of type `uint16_t`. ROI bottom-right corner vertical distance from image bottom-right corner
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.15. SET_AHC_GAIN_LIMITATION

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 |
|---|---|---|---|---|
| MID | BLACK_ LIMITATION | WHITE_ LIMITATION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 2 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12354 (0x3042)
- **B9:** BLACK_LIMITATION — Of type `uint8_t`, in %. Minimum LUT output value, as a percentage of total histogram size
- **B10:** WHITE_LIMITATION — Of type `uint8_t`, in %. Maximum LUT output value, as a percentage of total histogram size
- **B11-B12:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.16. GET_AHC_GAIN_LIMITATION

[GET] Set the Automatic Histogram Control gain limit for CLHE/CLAHE algorithms, see [SET_AHC_GAIN_LIMITATION].

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 |
|---|---|---|---|---|
| MID | BLACK_ LIMITATION | WHITE_ LIMITATION | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC — Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN — Data payload length in bytes, here equal to 2 byte(s)
- **B2-B3:** RESERVED — Reserved value, must be kept 0
- **B4:** SEQ — Sequence number, incremented at each command
- **B5-B6:** RESERVED — Reserved value, must be kept 0
- **B7-B8:** MID — Message ID, here equal to 12355 (0x3043)
- **B9:** BLACK_LIMITATION — Of type `uint8_t`, in %. Minimum LUT output value, as a percentage of total histogram size
- **B10:** WHITE_LIMITATION — Of type `uint8_t`, in %. Maximum LUT output value, as a percentage of total histogram size
- **B11-B12:** CRC16 — 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.17. SET_LUT_TEMPORAL_FILTER

[SET] Set the temporal smoothing coefficient for CLHE/CLAHE algorithms

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | N_FILTER | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12356 (0x3044)
- **B9:** N_FILTER Of type `uint8_t`, between 0 (min) and 8 (max). Smoothing coefficient limiting histogram variations between video frames, gain delta is limited to (100 * 1/2^coeff) % of the difference
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.18. GET_LUT_TEMPORAL_FILTER

[GET] Get the temporal smoothing coefficient for CLHE/CLAHE algorithms

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 |
|---|---|---|---|
| MID | N_FILTER | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 1 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12357 (0x3045)
- **B9:** N_FILTER Of type `uint8_t`, between 0 (min) and 8 (max). Smoothing coefficient limiting histogram, see [SET_LUT_TEMPORAL_FILTER]
- **B10-B11:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.19. UPDATE_CTRL

[SET] Control the update system.

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | CMD | PARAM1 | | | | PARAM2 | |

| B16 | B17 | B18 | B19 |
|---|---|---|---|
| PARAM2 | | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 9 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12360 (0x3048)
- **B9:** CMD Of type `uint8_t`. Command
- **B10-B13:** PARAM1 Of type `uint32_t`. Parameter 1
- **B14-B17:** PARAM2 Of type `uint32_t`. Parameter 2
- **B18-B19:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

### 7.1.2.3.20. GET_CAMERA_INFO

[GET] Get camera operating information

| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 |
|---|---|---|---|---|---|---|---|
| MAGIC | LEN | RESERVED | | SEQ | RESERVED | | MID |

| B8 | B9 | B10 | B11 | B12 | B13 | B14 | B15 |
|---|---|---|---|---|---|---|---|
| MID | STARTUP_NUMBER | | | | OPERATING_TIME | | |

| B16 | B17 | B18 |
|---|---|---|
| OPERATING_ TIME | CRC16 | |

NOTE: Each MAVLink packet is coded in little endian format.

Bytes:

- **B0:** MAGIC Magic number for MAVLink V2, equal to 253 (0xFD)
- **B1:** LEN Data payload length in bytes, here equal to 8 byte(s)
- **B2-B3:** RESERVED Reserved value, must be kept 0
- **B4:** SEQ Sequence number, incremented at each command
- **B5-B6:** RESERVED Reserved value, must be kept 0
- **B7-B8:** MID Message ID, here equal to 12361 (0x3049)
- **B9-B12:** STARTUP_NUMBER Of type `uint32_t`. Number of camera startups
- **B13-B16:** OPERATING_TIME Of type `uint32_t`. Total operating time in minutes
- **B17-B18:** CRC16 16-bit CRC (MCRF4XX) computed over header and data bytes

# 8. ADDITIONAL SOFTWARE

## 8.1. Compagnon Software

CamSight FB cameras are supplied with an updater tool to download new or update the camera firmware.

See AD001 for full documentation.

## 8.2. Updater Software

CamSight FB cameras are supplied with a companion software tool to set the camera parameters in factory.

See AD002 for full documentation.

# 9. ENVIRONMENTAL RATINGS AND COMPLIANCE

## 9.1. Environmental conditions

*It is the integrator's responsibility to verify the final compliance of the overall system with the camera integrated inside.*

- *Pre-compliance to STANAG 4370 AECTP 300*

| Parameter | Min | Max |
|---|---|---|
| Storage temperature | -40°C | 85°C |
| Operating temperature | -40°C | 50°C |
| Humidity | 5% | 95% (not condensing) |

**Table 18: Environnemental conditions compliance**

## 9.2. Shock and vibrations

*It is the integrator's responsibility to verify the final compliance of the overall system with the camera integrated inside.*

| Test | Test condition | Result |
|---|---|---|
| Shock | STANAG 4370 AECTP400 Ed3<br>X, Y, Z axes, 3 shocks per axis | Pass with casing |
| Vibration | STANAG 4370 AECTP400 Ed3<br>X, Y, Z axes, 8 hours per axis | Pass with casing |

**Figure 27: Shock and vibration compliance**

## 9.3. EMC

*It is the integrator's responsibility to verify the final compliance of the overall system with the camera integrated inside.*

- *Pre-compliance to STANAG 4370 AECTP 500 ED4 Ground army*

| Test | Test condition | Result |
|---|---|---|
| Conducted Susceptibility | NCS02 control and signals<br>20Hz-50 kHz | Pass with casing |
| | NCS07 BCI<br>10kHz to 200MHz | Pass with casing |
| | NCS08 BCI Impulse excitation Level 5A | Pass with casing |
| | NCS09 Damped sinusoidal transients<br>10kHz to 100MHz | Pass with casing |
| Radiated Susceptibility | NRS01 magnetic field<br>30 Hz - 100 kHz | Pass with casing |
| | NRS02 electrical field of<br>2MHz to 6GHz | Pass with casing |
| Radiated Emissions | NRE01 Magnetic field<br>30Hz to 100 kHz | Pass with casing |
| | NRE02 Electric field<br>10kHz to 6GHz | Pass with casing |

| Test | Test condition | Result |
|---|---|---|
| Conducted Emissions | NCE02<br>10kHz-10 MHz | Pass with casing |
| | NCE04 transient measurement | Pass with casing |
| | NCE05 Current measurement<br>30Hz-150 MHz | Pass with casing |

**Figure 28: EMC compliance**

## 9.4. Other standards

*It is the integrator's responsibility to verify the final compliance of the overall system with the camera integrated inside.*

| Compliance standard | Status |
|---|---|
| CE (Conformité Européenne) | Compliant |
| RoHS (Reduction of Hazardous Substances) | Compliant |
| REACH<br>(Registration, Evaluation, Authorization and Restriction of Chemicals) | Compliant |

**Figure 29: Other standards compliance**