EV76C570
2 Mpixels B&W and Color CMOS Image sensor

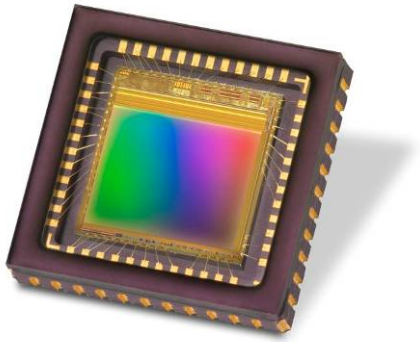
For the latest data sheet, please visit www.sunnywale.com

FEATURES

- 2 million (1600 x 1200) pixels, 4.5 µm square pixels with micro-lens
- Optical format 1/1.8”
- 50 fps@ full resolution

Embedded functions:

- Image Histograms and Context output
- Sub-sampling / binning
- Multi ROI (including 1 line mode)
- Defective pixel correction
- PLL with 5 to 50 MHz input frequency range
- High dynamic range capabilities
- Time to Read improvement (good first image, abort image)



Timing modes:

- Global shutter in serial and overlap modes
- Rolling shutter and Global Reset modes
- Output format 8 or 10 bits parallel plus synchronization
- SPI controls
- Control input pins: Trigger, Reset
- Light control output
- 3.3 V and 1.8 V power supplies

PERFORMANCE CHARACTERISTICS

- Low power consumption (200mW)
- High sensitivity at low light level
- Operating temperature [-30° to +65°C]
- Peak QE > 48%

AVAILABLE SENSOR TYPES

- B&W
- Color (Bayer arrangement)
- 60 frames per second option

APPLICATIONS

- Surveillance IP/CCTV cameras
- Industrial machine vision
- Biometrics/Medical Imaging
- 2D barcode reading

INTRODUCTION

The EV76C570 is a 2 million pixels CMOS image sensor designed with e2v's proprietary Eye-On-Si CMOS imaging technology. It is ideal for many different types of application where superior performance is required. The innovative pixel design offers excellent performance in low-light conditions with an electronic global (true snapshot) shutter, and offers a high-readout speed at 50 fps in full resolution. A 60 fps capable version is also available. Its very low power consumption makes it well suited for use in battery powered applications.

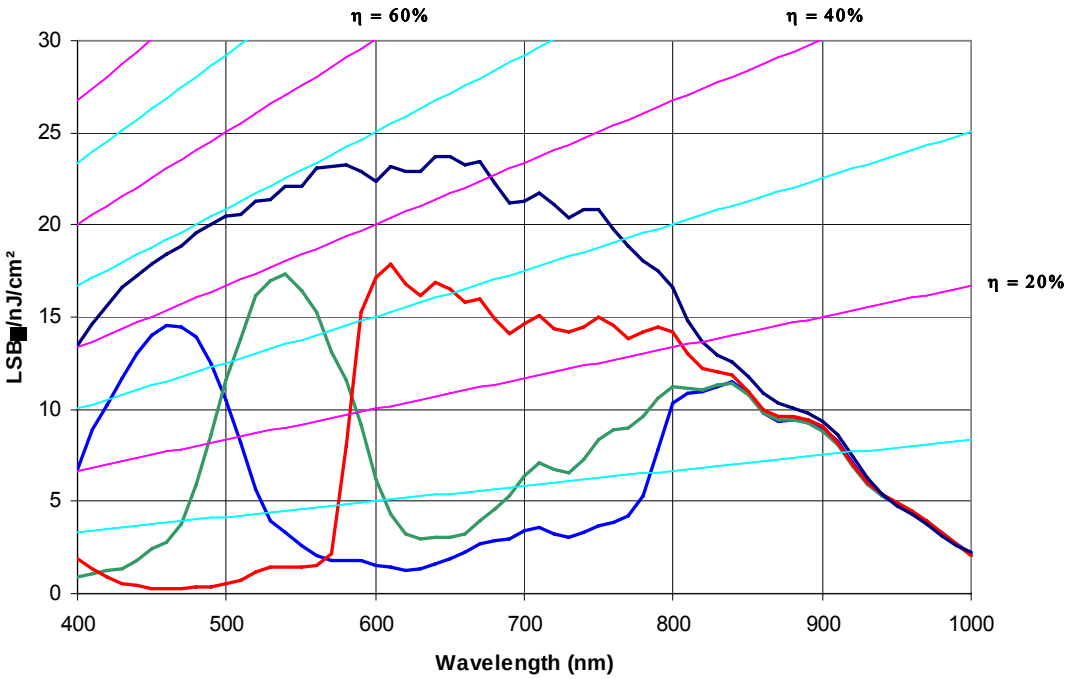
1. TYPICAL PERFORMANCE DATA

Table 1: Typical electro-optical performances @ 25°C and 65°C, nominal pixel clock

Parameter		Unit	Typical Value		
Sensor characteristics	**Resolution**	pixels	1600 (H) x 1200 (V)		
	Image size	mm inches	7.2 (H) x 5.4 (V) – 9 (diagonal)		
	Pixel size (square)	µm²	4.5 x 4.5		
	Aspect ratio		4/3		
	Max frame rate	fps	50 @ full format – 60 capability(6)		
	Pixel rate	Mpixels / s	114 -> 120		
	Bit depth	bits	10		
Pixel performance			**@ T_A 25°C**		**@ T_A 65°C**
	Dynamic range	dB	> 52 (1b)	> 66 (1a)	"
	Qsat	ke-	7.8 (1b)	12.6 (1a)	"
	SNR Max	dB	39 (1b)	41 (1a)	"
	MTF at Nyquist, λ=550 nm	%	63		
	Dark signal(2)	LSB_{10}/s	40		770
	DSNU(2)	LSB_{10}/s	17		240
	PRNU (3) (RMS)	%	< 1		
	Responsivity (4)	LSB_{10}/(lux/s)	7400		
Electrical interface	**Power supplies**	V	3.3 & 1.8		
	Power consumption : **Functional(5)** **Standby**	 mW µW	 < 210 33		

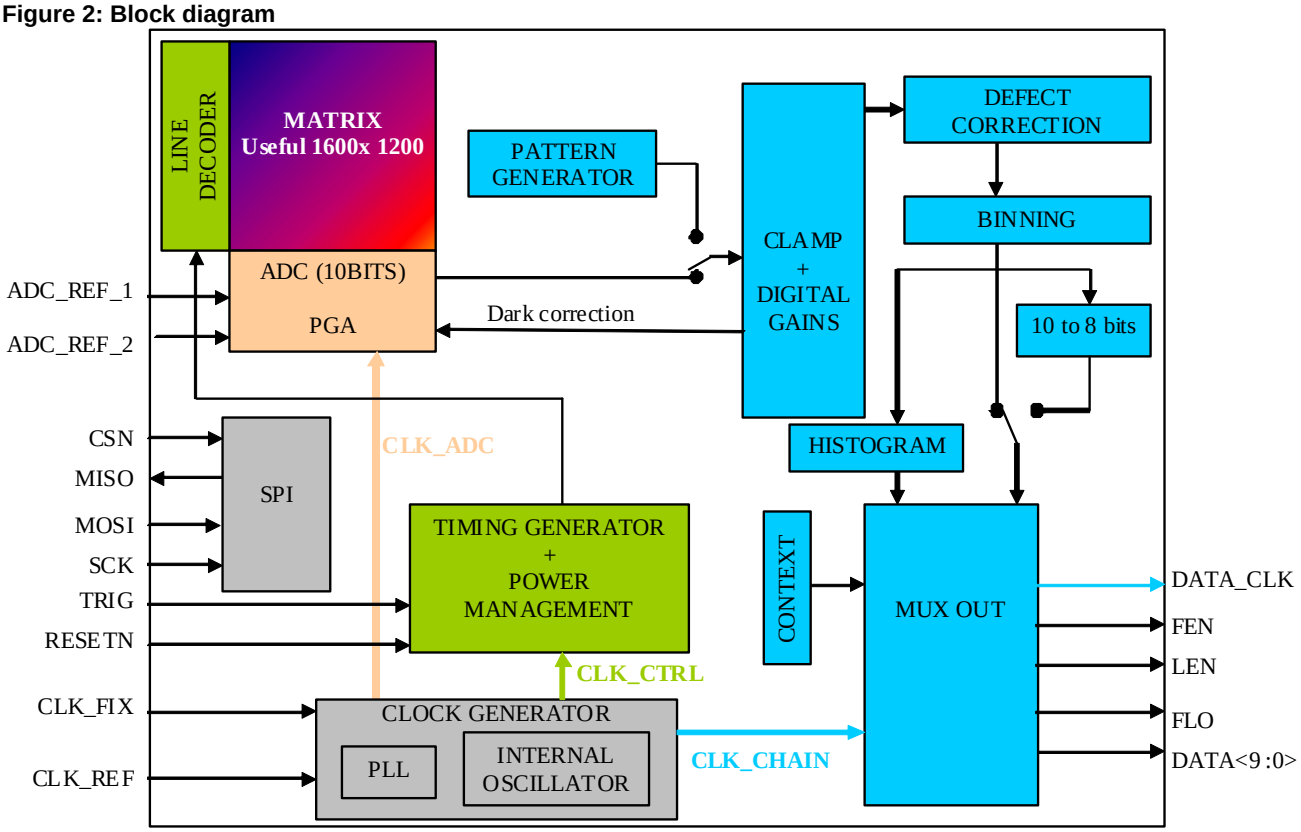
(1a): in electronic rolling shutter (ERS) mode
(1b): in global shutter (GS) mode
(2): min gain, 10 bits
(3): measured @ Vsat/2, min gain
(4): 3200K, window without AR coating, IR cutoff filter BG38 2 mm
(5): @ 50 fps & full format & with 10 pF on each output
(6): see ordering codes for 60fps version

Figure 1: Spectral response & quantum efficiency (η)



2. SENSOR OVERVIEW

Figure 2: Block diagram



Note: each color represents a clock domain (see § 18)

Detailed descriptions of the I/O signals and blocks are given in the datasheet sections listed in Table 2. See § 24 for the device pinout information.

Table 2 : Quick reference table for block diagram

Signal name	I/O	Description	Reference
ADC_REF1&2	I	ADC reference voltage	
CSN	I	SPI chip select	
MISO	O	SPI data output	
MOSI	I	SPI data input	
SCK	I	SPI clock	
TRIG	I	Trigger input	
CLK_REF	I	Reference clock input	
CLK_FIX	I	Fixed clock input	
ResetN	I	Sensor reset	
DATA<9:0>	O	10-bit data output bus	
FEN	O	Vertical sync output	
LEN	O	Horizontal sync output	
FLO	O	Illumination control output	
DATA_CLK	O	Output clock	

List of blocks	Reference
Matrix	
ADC + PGA	
Clamp + digital gain	
Defect correction	
Binning	
Histogram	
10->8 bits	
Context	
Mux out	
Timing and power management	
Clock generator	
Pattern generator	
SPI	

SUMMARY

3. DOCUMENT CONVENTIONS AND ACRONYMS

3.1 Acronyms

Table 3: Glossary of acronyms

B&W	Black and white
CDS	Correlated double sampling
DNC	Do not connect
DSNU	Dark signal non-uniformity
ERS	Electronic rolling shutter
FPN	Fixed pattern noise
fps	Frames per second
GR	Global reset
GS	Global shutter
IR	Infrared
LSB	Least significant bit
MIMR	Multiple integration multiple ROI
MSB	Most significant bit
MSL	Moisture sensitivity level
PGA	Programmable gain amplifier
PRNU	Photo response non-uniformity
ROI	Region of interest
Sat	Saturation value
SIMR	Single integration multiple ROI
SPI	Serial peripheral interface
TBC	To be confirmed
TBD	To be defined
Tint	Integration time

3.2 SPI register and bit names

SPI registers and bit name are shown in blue bold italics as follows:

example_reg_name	For the entire register
example_bit_name in **< example_reg_name>**	For part of the register

3.3 Numbering Conventions

Hexadecimal numbers are prefixed by “h”.
In register descriptions, decimal numbers are prefixed by “d”.

4. PRECAUTIONS FOR USING THE DEVICE

4.1 Absolute maximum ratings

Table 4: Absolute maximum ratings

Parameter	Value
VDD18D Digital supply voltage	-0.25V; 2.2 V
VDD18A Analog supply voltage	-0.25V; 2.2 V
VDD33A Analog supply voltage	-0.25V; 4 V
DC voltage at any input pin	-0.25V; V_{DD18D} +0.25V
Storage temperature	-40°C to + 85°C
Operating temperature	-30°C to + 65°C

- Stresses above those listed under “Absolute Maximum Ratings” might cause permanent device failure. Functioning at or above these limits is not recommended. Exposure to absolute maximum ratings for extended periods might affect reliability.
- All power pins with the same name must be connected to the same power supply.
- All grounds must be connected.

4.2 ESD

The EV76C570 is resistant up to 2 kV (HBM). To avoid accumulation of charges and to prevent electrical field formation, the following precautions must be taken during manipulation:

- Wear anti-static gloves or finger cots, anti-static clothes and shoes.
- Protect workstation with a conductive ground sheet.
- Use conductive boxes.

4.3 Cleaning the window

The EV76C570 sensor is an optical device. All precautions must be taken to prevent dust or scratches on the input window.
If the window needs to be cleaned, use the following procedure.

4.3.1 Equipment

- Ethanol
- Cleaning medium (wipes, optical paper, cotton buds),
- Filtered blow-off gun (preferably with static charge neutralizing capability),
- Area protected from electrostatic discharges and equipped with ground straps,

4.3.2 Preparations

- Wear vinyl gloves or finger cots without talcum powder.
- Make use of anti-ESD equipment: ground straps, ionizers etc.

4.3.3 Recommendations

- Never clean with a dry cleaning medium,
- Soak the cleaning medium with alcohol and do not pour it directly on the window.
- Clean the window only if necessary.

4.3.4 Operating procedure

- Clean the glass window with an air-jet (using the blow-off gun).
- If stains or dust remain:
    - Soak the cleaning medium with alcohol and wipe the glass window in a single movement from one side to another.
    - Always use a clean part of the cleaning medium for each new attempt.
    - Adapt the speed of the wiping action to let alcohol evaporate without leaving traces.
    - Optionally, use the blow-off gun to clean the window once more.

5. STANDARDS COMPLIANCE

The EV76C570 sensor conforms to the following standards:

- RoHS compliant
- Product qualification according to JEDEC JESD47
- MSL 3 compliant

6. STANDARD CONFIGURATION

6.1 Sensor settings

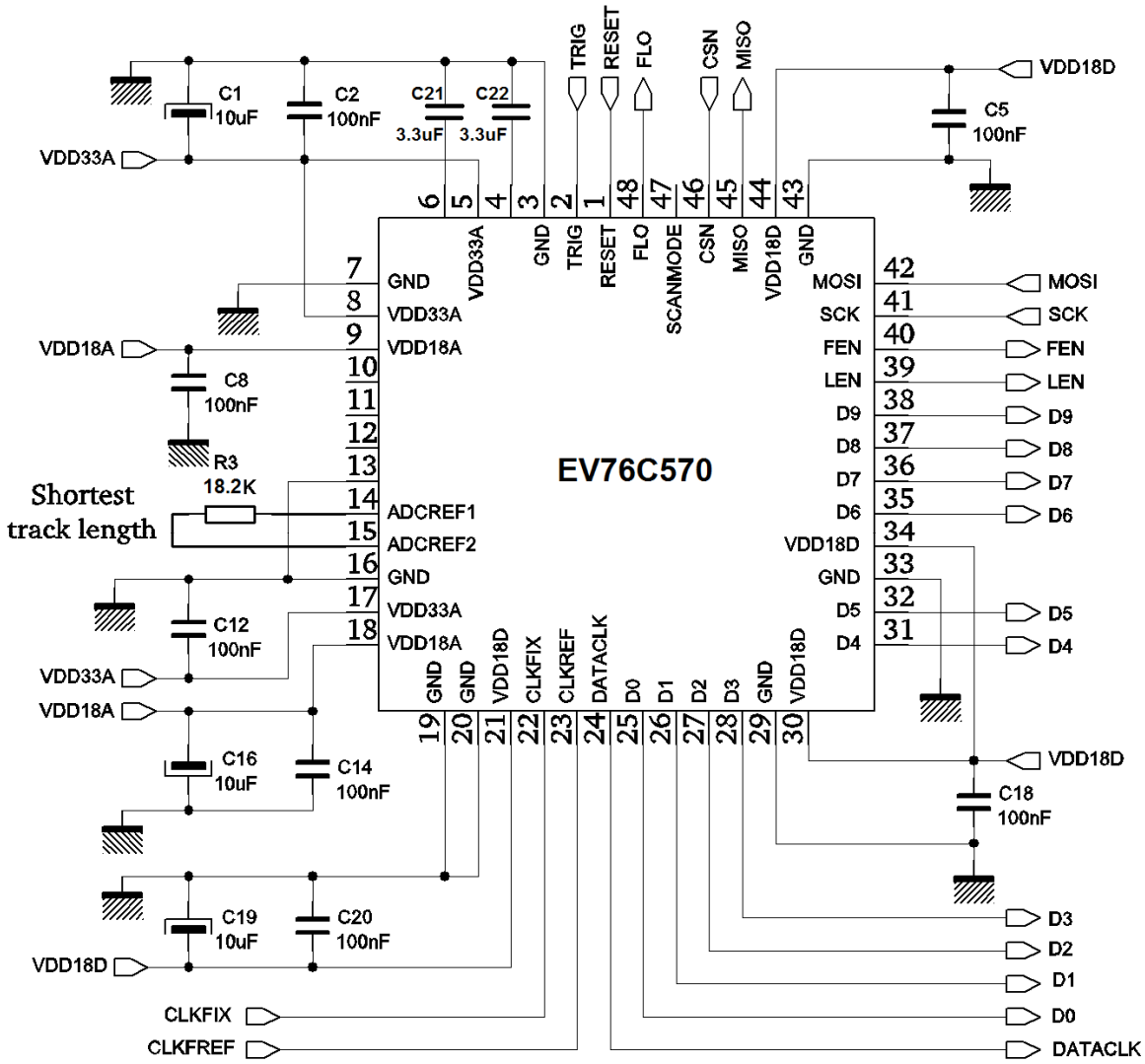
The static configuration required to allow image capture is as follows:

- All ground pins connected
- All power supply pins with the same name connected together.
- SPI pins connected to the host controller
- 1.8 V pins and 3.3 V pins powered-on
- Input clock driving the CLK_REF input pin
- RESETN pin held at high level after the power-on sequence. See § **21.1.1**
- STANDBY state is deactivated by writing 0 in the *stdby_rqst* bit in the <reg_ctrl_cfg> register. See § **20.3.8**
- Image capture is triggered by an active level on the TRIG pin or setting the *trig_rqst* bit in the <reg_ctrl_cfg> register. See § **20.3.8**

For improved performance, VDD33A and VDD18A must be noise-free. The best way to decouple VDD33A and to increase the power supply rejection ratio is to use a linear regulator dedicated to the image sensor. For VDD18A, an inductance can be used.

6.2 Application information

Figure 3: Required external components



It is recommended to use X7R for all the 100nF capacitors.
Reset pin has an internal pull-up.

6.3 Electrical levels

Table 5: DC Characteristics @ 25°C

Parameter	Symbol	Value			Unit
		Min	Typ	Max	
Analog power supply relative to GND	VDD33A	3.15	3.3	3.45	V
Digital power supply relative to GND	VDD18D	1.6	1.8	2	V
Analog power supply relative to GND	VDD18A	1.6	1.8	2	V
Power supply consumption(1)	P		210		mW
Supply current at 50 fps VDD33A pin	I_{VDD33A}		30		mA
Supply current at 50 fps VDD18A pin	I_{VDD18A}		30		mA
Supply current at 50 fps VDD18D pin	I_{VDD18D}		30		mA
Standby supply current VDD33A pin	I_{VDD33A}		0		mA
Standby supply current VDD18A pin	I_{VDD18A}		70		µA
Standby supply current VDD18D pin (2)	I_{VDD18D}		5	100	µA
Idle supply current VDD33A pin	I_{VDD33A}		9		mA
Idle supply current VDD18A pin	I_{VDD18A}		2		mA
Idle supply current VDD18D pin	I_{VDD18D}		5		mA
CMOS IN/OUT					
Input voltage low level	V_{IL}			0.3 V_{DD18}	V
Input voltage high level	V_{IH}	0.7 V_{DD18}			V
Input pin capacitance(3)	C_{IN}		4		pF
Output voltage low level	V_{OL1}			0.2	V
Output voltage high level	V_{OH1}	V_{DD18}–0.2			V
Output current @ V_{OH}(4)	I_{OH}	-10			mA
Output current @ V_{OL}(4)	I_{OL}			10	mA
Input leakage current(5)	I_L	-1		1	µA
ADC_REF current(6)	I_{ADC_REF}		100		µA

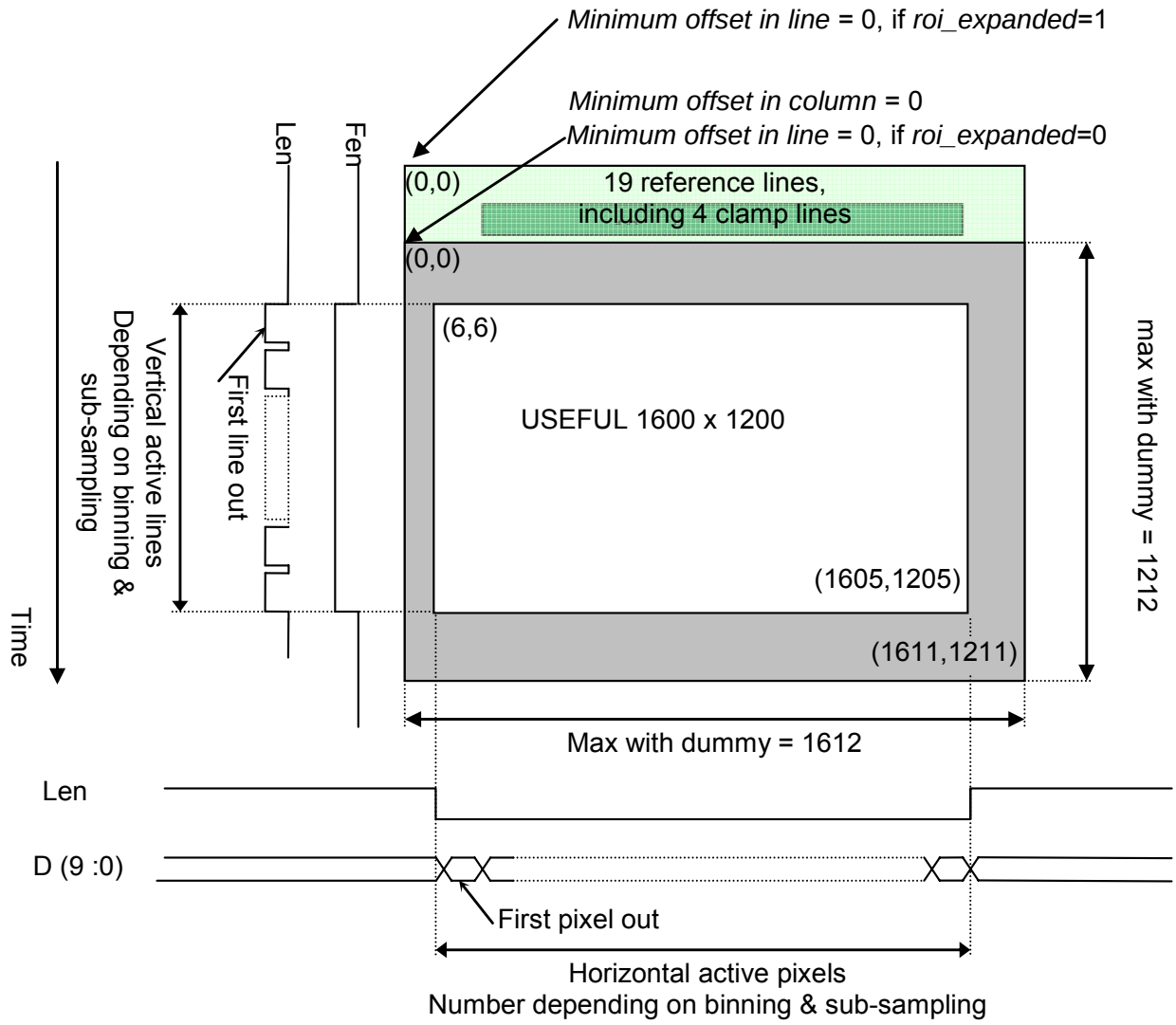
Note(1): Digital output loads =10 pF
Note(2): I_{VDD18D} with SPI on, without communication & without CLK_REF input.
Note(3): CLCC48 package
Note(4): On all output pins
Note(5): On all digital input pins
Note(6): On ADC_REF pins

7. MATRIX

7.1 Useful area definition

The useful area is 1600 x 1200 pixels as shown in Figure 4.
19 optically shielded reference lines included 4 lines to allow the black level adjustment.
6 dummy illuminated pixels surround the useful area.

Figure 4: Area description



7.2 CFA (Color Filter Array)

The following CFA types are implemented:

- Monochrome
- RGB Bayer filter
- Other types are available on request.

Table 6: Color of first pixel using the flip functions

		RoiX_0l / RoiX_0c			
Flip H	**Flip V**	**Odd / Odd**	**Odd / Even**	**Even / Odd**	**Even / Even**
No	No	Red	Green Red	Green Blue	Blue
No	Yes	Green Blue	Blue	Red	Green Red
Yes	No	Green Red	Red	Blue	Green Blue
Yes	Yes	Blue	Green Blue	Green Red	Red

RoiX_0l stands for **Roi1_0l_1, Roi2_0l_1, Roi3_0l_1 & Roi4_0l_1**. See Section 17.3.11, 17.3.12, 17.3.13, and 17.3.14 respectively.
RoiX_0c stands for **Roi1_0c_1, Roi2_0c_1, Roi3_0c_1 & Roi4_0c_1**. Section 17.3.11, 17.3.12, 17.3.13, and 17.3.14 respectively.

It is recommended to keep:

• Roi_w_1 + Roi_0c_2 even
• Roi_h_1 + Roi_0l_2 even

Flip H & Flip V are under roi_flip_h & roi_flip_v control. See <*reg_miscel2*> in § **20.3.4**

7.3 Pixels

The matrix is composed of five transistor (5T) pixels. This structure supports either global shutter (GS) mode or electronic rolling shutter (ERS) mode. (See § **21.2**)

7.4 Lens Chief Ray Angle (CRA) compensation

In order to better focus the light rays on the photodiode, the EV76C570 micro lenses are radially shifted to match the exit angles due to the external application lens. This results in improved efficiency and reduced corner shading.
This shift is linearly applied from center (0 shift) to corner (α angle). α is the corner CRA (Chief Ray Angle) defined as a mean value of the telecentricity of optics lenses that would be used with the sensor.

The sensor, optimized for a corner CRA of 12°, can be used with a range of telecentricity from 5° to 20° (estimated for Fnumber f#/1.2).

Figure 5 : Microlenses shifting

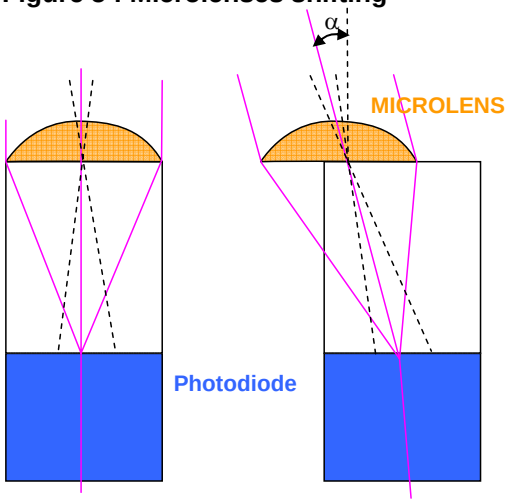
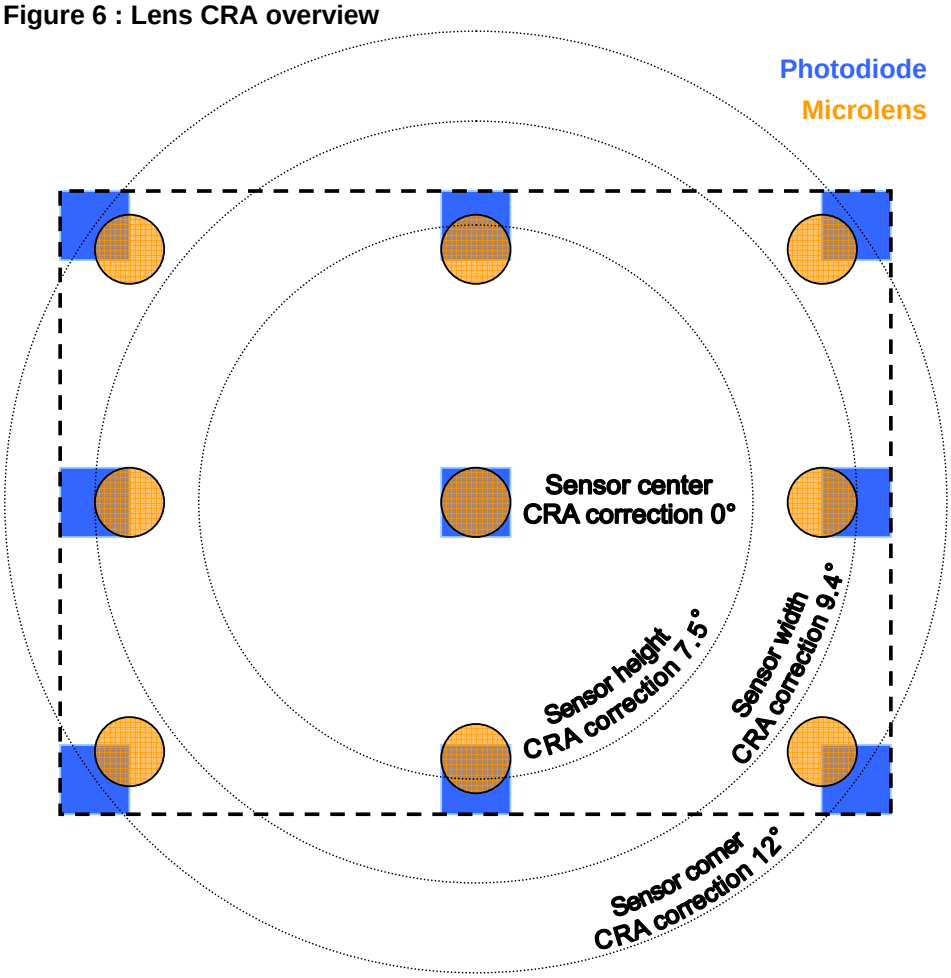


Figure 6 : Lens CRA overview



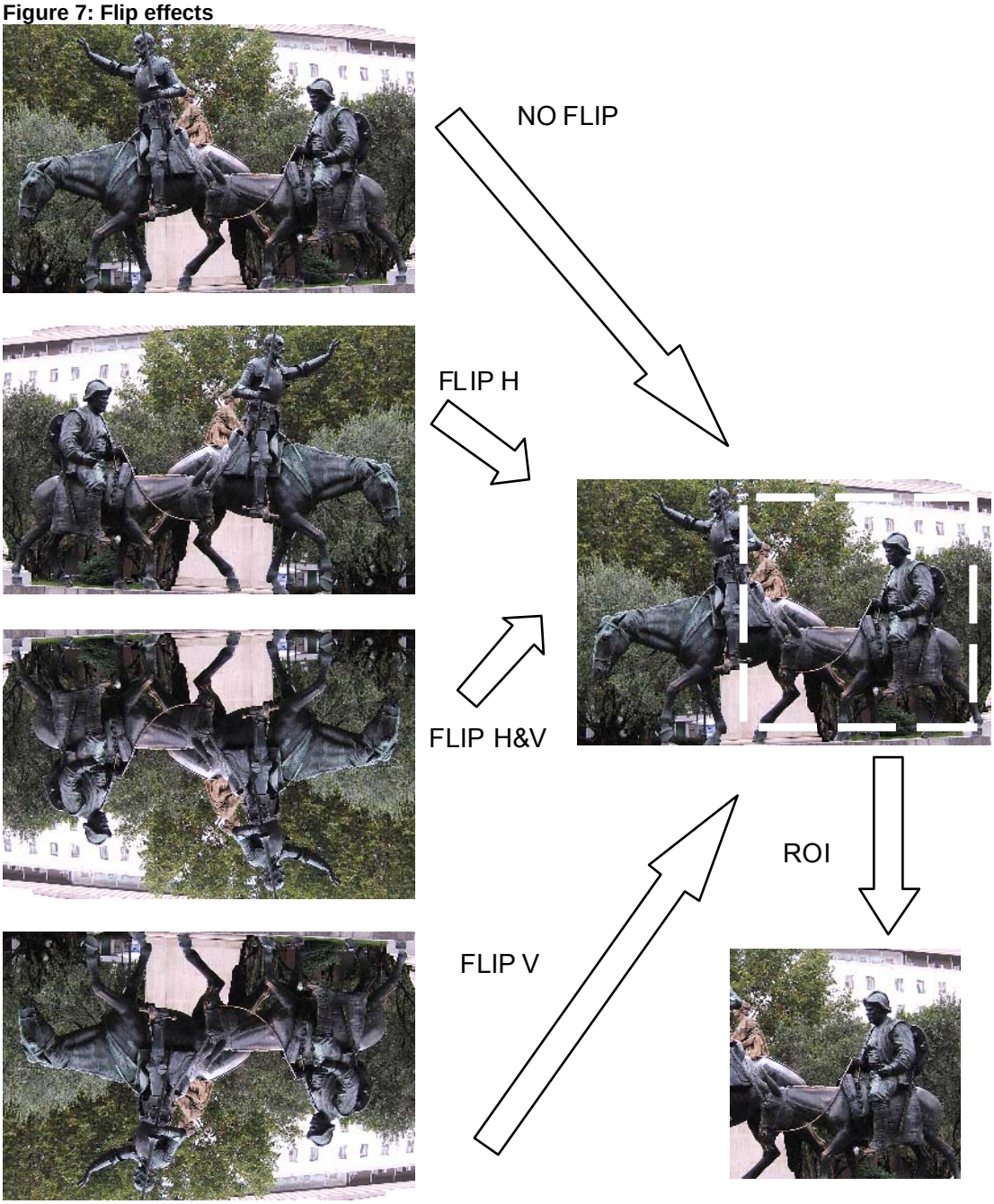
7.5 Region of interest (ROI)

7.5.1 Flip functions

Flip functions are available to allow the application to use any type of lens (with or without mirror). The flip functions are controlled by programming the *roi_flip_h* and *roi_flip_v* bitfields in the <reg_miscel2> register. See § **20.3.4**

The shielded lines for dark reference are always read first (except in expanded ROI mode selected by the *roi_expanded* bit in the <reg_miscel2> register (See § **20.3.4)** when whole lines may be read.

Figure 7: Flip effects



7.5.2 ROI definition

The ROI selection is done on the flipped/un-flipped image. All ROIs are defined in relation to the matrix and useful pixel area (as shown in **Figure 4**). The ROIs are defined before sub-sampling, defect correction and binning.

7.5.2.1 Sub-sampling and Windowing

The sub-sampling function causes the sensor to read only 8 pixels over the selected factor. For example, a sub-sampling factor of 8 over 16 means that the sub-sampling ratio is 1:2. For color sensors, the algorithm is more complicated due to the Bayer organization.

Sub-sampling is programmable with a ratio 1 to 32 in steps of 0.125. Different sub-sampling factors can be defined for horizontal and vertical directions. They are programmable using SPI commands: ***roiX_subs_v*** & ***roiX_subs_h*** in **<reg_roiX*>** (where with X is the number of the ROI 1, 2, 3 or 4) see §**20.3.11**, **20.3.12**, **20.3.13**, & **20.3.14** respectively.

Windowing defines the size and position of the ROI.Windowing is defined in two dimensions: horizontal and vertical. The minimum width of the window is 16 columns and the minimum height is 1 line. The user has to define the height, width and offsets of the ROI through the SPI control bus for each ROI used. (See **7.5.3**).

Windowing, Sub- sampling and then Binning are possible on the same image.

Figure 8: Combination of windowing, sub-sampling and binning

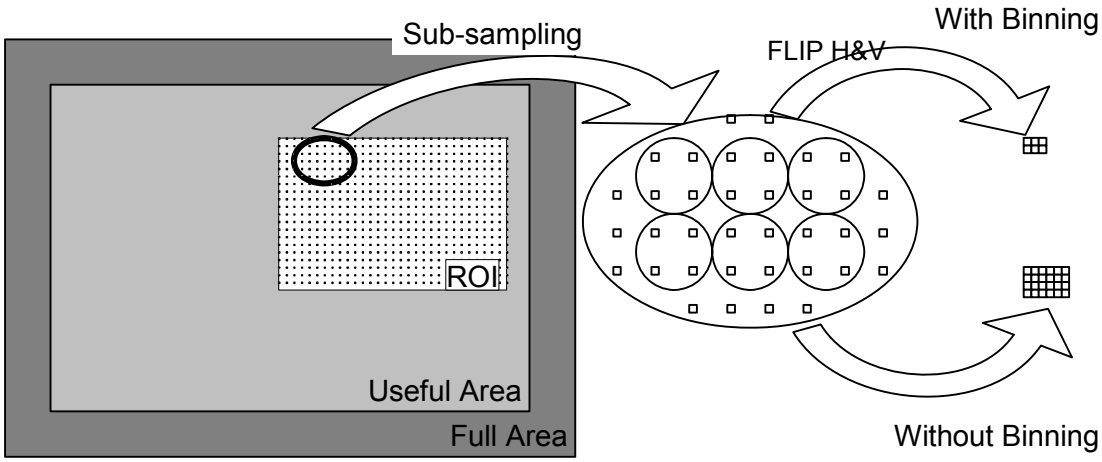
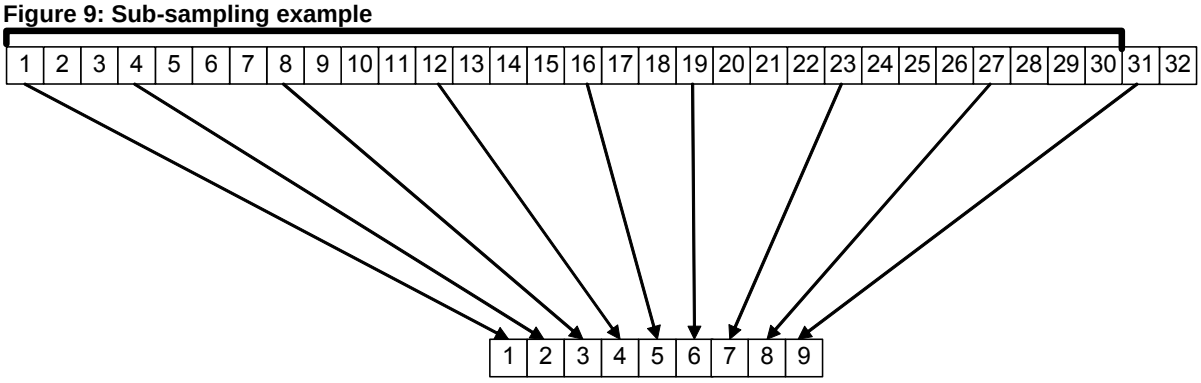


Figure 9: Sub-sampling example



With an 8/30 sub-sampling factor only these pixels (or lines) will be read:

- On the first group of 30 pixels: 1, 4, 8, 12, 16, 19, 23, 27
- On the second group of 30 pixels: 31, 34, 38…
- On the third group of 30 pixels: 61…
- …

Roughly, the sub-sampled image format will be multiplied by 8/30=1/3.75. For more precise calculation of the output image size the following formulas must be used.

7.5.2.2 Calculating the image output size

Image output sizes (width_out and height_out) are determined by the following equations depending on:

- B&W or color version (***color_en*** in **<reg_miscel2>** See § **20.3.4**)
- Sub-sampling factor (***roiN_subs_v*** & ***roiN_subs_h*** in **<reg_roiN*> see** § **20.3.11**, **20.3.12**, **20.3.13**, & **20.3.14** respectively),
- Defect correction activation (***roi_ddc_en*** in **<reg_chain_cfg>** see § **20.3.7**),
- Binning activation (***roiN_binning_en*** in **<reg_chain_cfg>** see § **20.3.7**.

If roiN_binning_en = 0 AND color_en = 0

For ROI 1:

$$ROI\_width = INT\left(\frac{8\times roi1\_w\_1}{roi1\_subs\_factor+8}\right) + ent\left(\frac{8\times roi1\_w\_2}{roi1\_subs\_factor+8}\right)$$

For ROI 2, 3 and 4:

$$ROI\_width = INT\left(\frac{8\times roiN\_w}{roiN\_subs\_factor+8}\right)$$

If (roiN_binning_en = 1 AND color_en = 0) OR (roiN_binning_en = 0 AND color_en = 1)

For ROI 1:

$$ROI\_width = 2\times\left[INT\left(\frac{4\times roi1\_w\_1}{roi1\_subs\_factor+8}\right) + INT\left(\frac{4\times roi1\_w\_2}{roi1\_subs\_factor+8}\right)\right]$$

For ROI 2, 3 and 4:

$$ROI\_width = 2\times INT\left(\frac{4\times roiN\_w}{roiN\_subs\_factor+8}\right)$$

If roiN_binning_en = 1 AND color_en = 1

For ROI 1:

$$ROI\_width = 4\times\left[INT\left(\frac{2\times roi1\_w\_1}{roi1\_subs\_factor+8}\right) + INT\left(\frac{2\times roi1\_w\_2}{roi1\_subs\_factor+8}\right)\right]$$

For ROI 2, 3 and 4:

$$ROI\_width = 4\times INT\left(\frac{2\times roiN\_w}{roiN\_subs\_factor+8}\right)$$

Then, *width_out* is:

$$\Rightarrow width\_out = \frac{ROI\_width - 4\times ddc\_en}{2^{roiN\_binning\_en}}$$

If roiN_binning_en = 0 AND color_en = 0

For ROI 1:

$$ROI\_height = INT\left(\frac{8 \times roi1\_h\_1}{roi1\_subs\_factor + 8}\right) + INT\left(\frac{8 \times roi1\_h\_2}{roi1\_subs\_factor + 8}\right)$$

For ROI 2, 3 and 4:

$$ROI\_height = INT\left(\frac{8 \times roiN\_h}{roiN\_subs\_factor + 8}\right)$$

If (roiN_binning_en = 1 AND color_en = 0) OR (roiN_binning_en = 0 AND color_en = 1)

For ROI 1:

$$ROI\_height = 2 \times \left[INT\left(\frac{4 \times roi1\_h\_1}{roi1\_subs\_factor + 8}\right) + INT\left(\frac{4 \times roi1\_h\_2}{roi1\_subs\_factor + 8}\right)\right]$$

For ROI 2, 3 and 4:

$$ROI\_height = 2 \times INT\left(\frac{4 \times roiN\_h}{roiN\_subs\_factor + 8}\right)$$

If roiN_binning_en = 1 AND color_en = 1

For ROI 1:

$$ROI\_height = 4 \times \left[INT\left(\frac{2 \times roi1\_h\_1}{roi1\_subs\_factor + 8}\right) + INT\left(\frac{2 \times roi1\_h\_2}{roi1\_subs\_factor + 8}\right)\right]$$

For ROI 2, 3 and 4:

$$ROI\_height = 4 \times INT\left(\frac{2 \times roiN\_h}{roiN\_subs\_factor + 8}\right)$$

Then, *height_out* is:

$$\Rightarrow height\_out = \frac{ROI\_height - 4 \times ddc\_en}{2^{roiN\_binning\_en}}$$

Notes:

- INT( ) takes the integer part of the division result.
- N stands for ROI number (1, 2, 3 or 4).
- If defect correction is active, the minimum ROI size is 5; defect correction must be disabled for smaller ROI size.

•

7.5.3 Multi ROI

The multi-ROI offers two different and separate modes:

- The MIMR (Multiple Integration Multiple ROI) mode allows the user to define an acquisition cycle comprising up to 4 ROI cycle(s) (See ***roi_max_id*** in **<reg_chain_cfg>** in § **20.3.7)**
- The SIMR (Single Integration Multiple ROI), for the first ROI of the multi ROI cycle only, allows 1, 2 or 4 areas of interest to be acquired within the same integrated image. In SIMR mode, the sensor outputs only the configured zones and concatenates them to form a single image (see Figure 11 and Figure 12).

Each ROI has its own specific parameters (see Table 7) and parameters that are common to all ROIs (see Table 8).

Table 7: ROI specific parameters

Parameter	Description	Register bitfield name			
		ROI1	ROI2	ROI3	ROI4
ROI configuration	Defines the ROI dimensions and position in the total field of view	***Roi1_0l_1*** ***Roi1_h_1*** ***Roi1_0c_1*** ***Roi1_w_1*** ***Roi1_0l_2*** ***Roi1_h_2*** ***Roi1_0c_2*** ***Roi1_w_2*** In <**reg_roi1**>	***Roi2_0l_1*** ***Roi2_h_1*** ***Roi2_0c_1*** ***Roi2_w_1*** In <**reg_roi2**>	***Roi3_0l_1*** ***Roi3_h_1*** ***Roi3_0c_1*** ***Roi3_w_1*** In <**reg_roi3**>	***Roi4_0l_1*** ***Roi4_h_1*** ***Roi4_0c_1*** ***Roi4_w_1*** In <**reg_roi4**>
Integration times	For each ROI, two integration times have to be defined : one in number of lines and one in sub-line times	***Roi1_t_int_ll*** ***Roi1_t_int_clk*** In <**reg_roi1**>	***Roi2_t_int_ll*** ***Roi2_t_int_clk*** In <**reg_roi2**>	***Roi3_t_int_ll*** ***Roi3_t_int_clk*** In <**reg_roi3**>	***Roi4_t_int_ll*** ***Roi4_t_int_clk*** In <**reg_roi4**>
Analog and digital gains		***Roi1_ana_gain*** ***Roi1_dig_gain*** In <**reg_roi1**>	***Roi2_ana_gain*** ***Roi2_dig_gain*** In <**reg_roi2**>	***Roi3_ana_gain*** ***Roi3_dig_gain*** In <**reg_roi3**>	***Roi4_ana_gain*** ***Roi4_dig_gain*** In <**reg_roi4**>
Vertical and horizontal sub-sampling factors		***Roi1_subs_v*** ***Roi1_subs_h*** In <**reg_roi1**>	***Roi2_subs_v*** ***Roi2_subs_h*** In <**reg_roi2**>	***Roi3_subs_v*** ***Roi3_subs_h*** In <**reg_roi3**>	***Roi4_subs_v*** ***Roi4_subs_h*** In <**reg_roi4**>
Binning factor	Binning is performed after the sub-sampling if this is used. Each ROI can have its own binning factor.	***Roi1_binning_en*** In <**reg_chain_cfg**>	***Roi2_binning_en*** In <**reg_chain_cfg**>	***Roi3_binning_en*** In <**reg_chain_cfg**>	***Roi4_binning_en*** In <**reg_chain_cfg**>
Repetition count	Each ROI will be repeated several times before reading the next ROI	***Roi1_rep_nb*** In <**reg_roi1**>	***Roi2_rep_nb*** In <**reg_roi2**>	***Roi3_rep_nb*** In <**reg_roi3**>	***Roi4_rep_nb*** In <**reg_roi4**>
Wait time	Wait time after the end of the ROI repetition	***Roi1_t_wait_ext*** In <**reg_roi1**>	***Roi2_t_wait_ext*** In <**reg_roi2**>	***Roi3_t_wait_ext*** In <**reg_roi3**>	***Roi4_t_wait_ext*** In <**reg_roi4**>

Table 8: ROI common parameters

Parameter	Description	Register bitfield names
Binning factor divider	The binning result may be divided by 1, 2 or 4 to either keep the maximum amount of information or reduce the noise.	***Binning_div_factor*** in <**reg_chain_cfg**>
Flip configuration	(see Flip effect section)	***Roi_flip_h*** and ***roi_flip_v*** In <**reg_miscel2**>
Readout mode		***Roi_readout_mode***
Digital color gains (for color sensor)		***gb_dig_gain;gr_dig_gain*** In <**reg_dig_gain_gb_gr**> ***b_dig_gain;r_dig_gain*** In <**reg_dig_gain_b_r**>
Wait time at the end of each frame		***Roi_t_wait***
Line length		***Line_length***
Clamp configuration and offsets	Depends on MIMR, SIMR or High Dynamic configurations.	

Figure 10: Multi ROI cycle

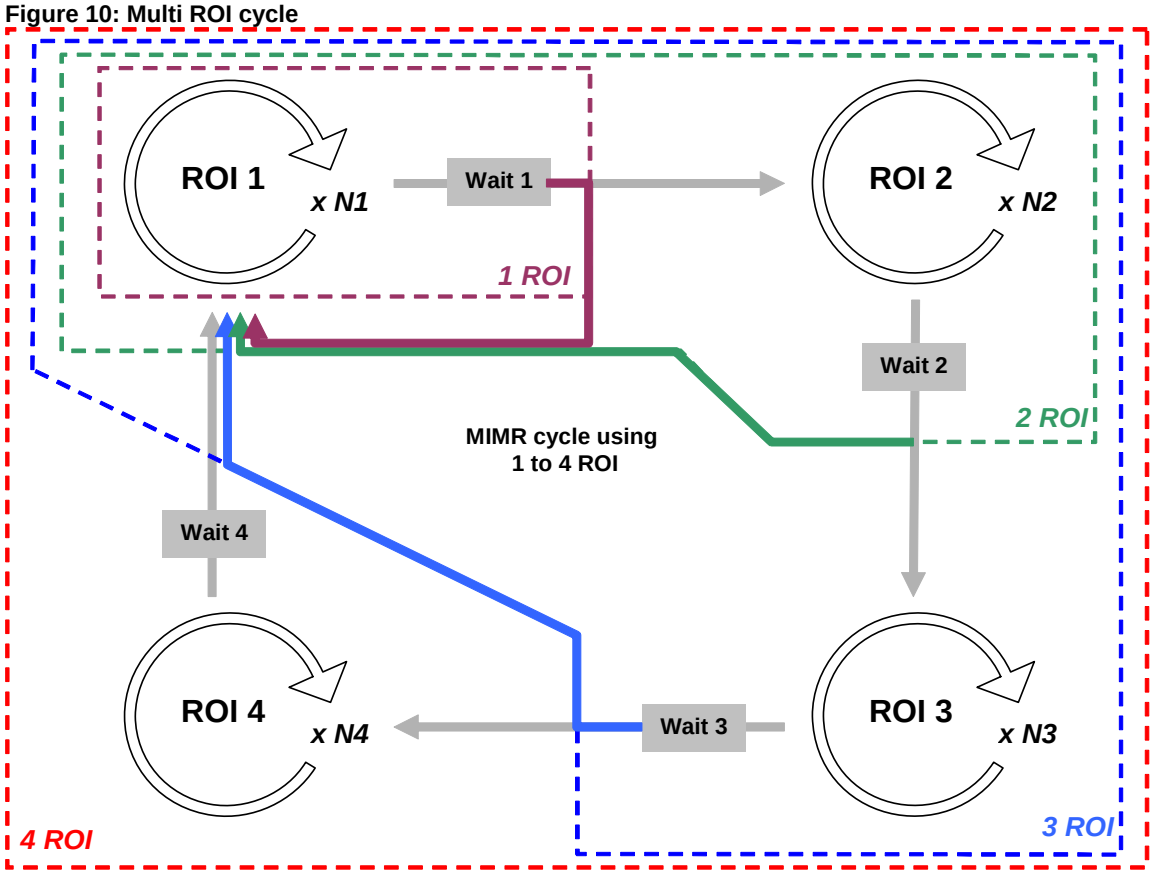
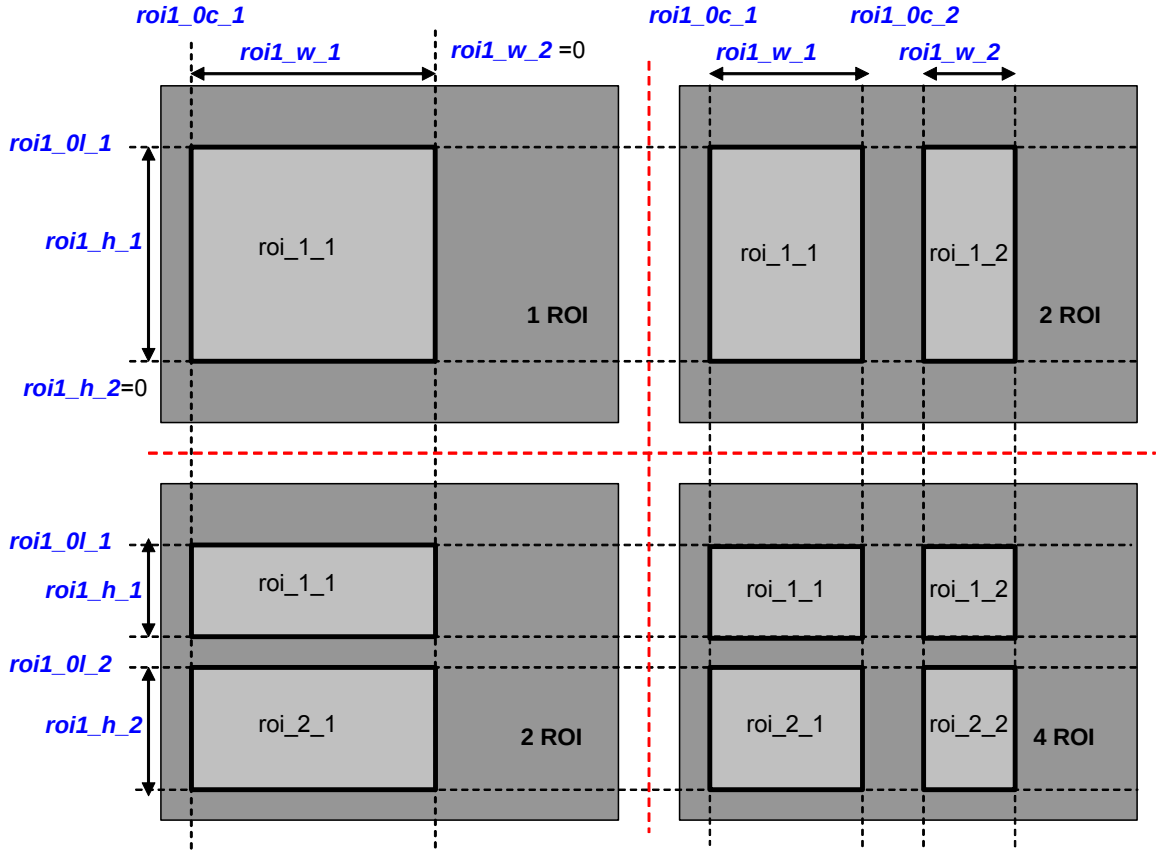


Figure 11: SIMR parameters



All the ROI 1 registers are described in § **20.3.11**.

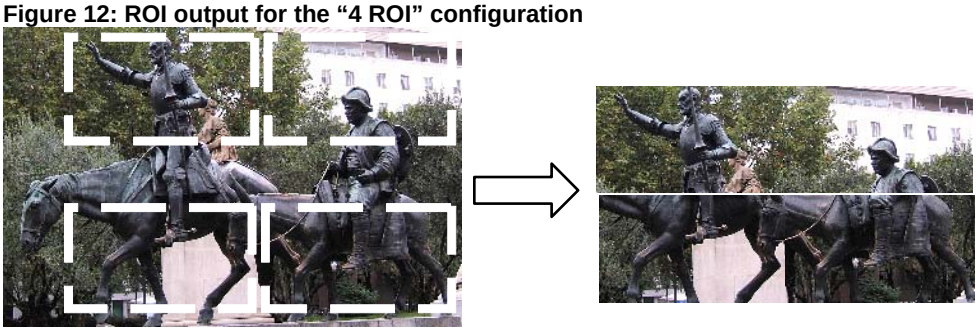
- If the ROI_1_2 width and ROI_2_1 height are null, only ROI_1_1 is read. The user has to choose:
  - ROI_1_1 horizontal (***roi1_0c_1***) and vertical (***roi1_0l_1***) offsets.
  - ROI_1_1 horizontal (***roi1_w_1***) and vertical (***roi1_h_1***) dimensions.

- If the ROI_1_2 width is greater than 0 and ROI_2_1 height is null only the ROI_1_1 and the ROI_1_2 are read. The user has to choose:
  - ROI_1_1 horizontal (***roi1_0c_1***) and vertical (***roi1_0l_1***) offsets.
  - ROI_1_1 horizontal (***roi1_w_1***) and vertical (***roi1_h_1***) dimensions.
  - ROI_1_2 horizontal (***roi1_0c_2***) offset. (ROI_1_2 vertical offset is the same as for ROI_1_1)
  - horizontal (***roi1_w_2***) width (ROI_1_2 height is the same as for ROI_1_1)

- If the ROI_1_2 width is null and ROI_2_1 height is greater than 0 only the ROI_1_1 and the ROI_2_1 are read. The use has to choose:
  - ROI_1_1 horizontal (***roi1_0c_1***) and vertical (***roi1_0l_1***) offsets.
  - ROI_1_1 horizontal (***roi1_w_1***) and vertical (***roi1_h_1***) dimensions.
  - ROI_2_1 vertical (***roi1_0l_2***) offset. (ROI_2_1 horizontal offset is the same as for ROI_1_1)
  - ROI_2_1 height (***roi1_h_2***) (ROI_2_1 width is the same as ROI_1_1)

- If the ROI_1_2 width and ROI_2_1 height are greater than 0, then 4 ROI_1_1, ROI_2_1, ROI_1_2 and ROI_2_2 are read. The user has to choose:
  - ROI_1_1 horizontal (***roi1_0c_1***) and vertical (***roi1_0l_1***) offsets.
  - ROI_1_1 horizontal (***roi1_w_1***) and vertical (***roi1_h_1***) dimensions.
  - ROI_2_1 ROI_2_2 vertical (***roi1_0l_2***) offset and (***roi1_h_1***) height
  - ROI_1_2 horizontal (***roi1_0c_2***) offset and (***roi1_w_2***) width.

Figure 12: ROI output for the "4 ROI" configuration



When using the defect correction (***roi_ddc_en*** = 1) there is:

- A 4-column (or 2 if binning function is enabled) black border between ROI_1_1& ROI_1_3 and ROI_1_2 & ROI_1_4
- A 4-line (or 2 if binning function is enabled) black border between ROI_1_1& ROI_1_2 and ROI_1_3 & ROI_1_4.

7.5.3.1 High dynamic range configuration

A special MIMR configuration using two integration times can be used to provide high dynamic images.

The first integration time image and the following second integration image are combined without any image loss. For example:

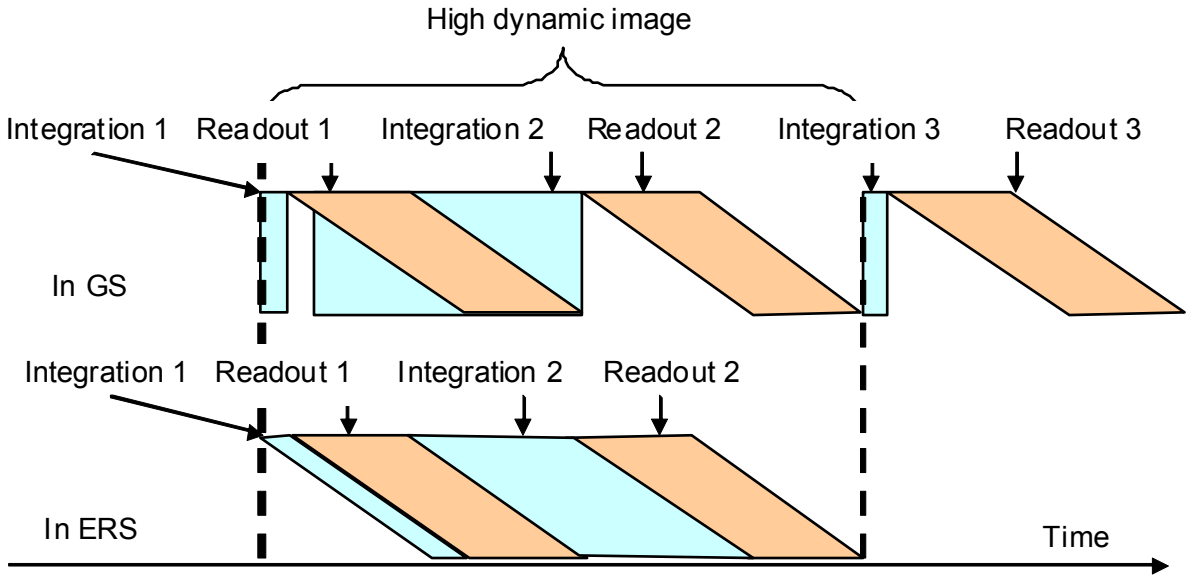
- Image 1 with a short integration time
- Image 2 with N time longer integration time
- A computed image may be calculated by summing image 2 + [image 1 with each of its pixel values multiplied by N]

In this mode, only two ROIs are used. They must have the same:

- Position and dimensions.
- Binning
- Sub-sampling factor
- Repetition factor (1)
- ROI mode (SIMR must not be used)

To prevent motion distortion it is recommended to perform the short integration time first.

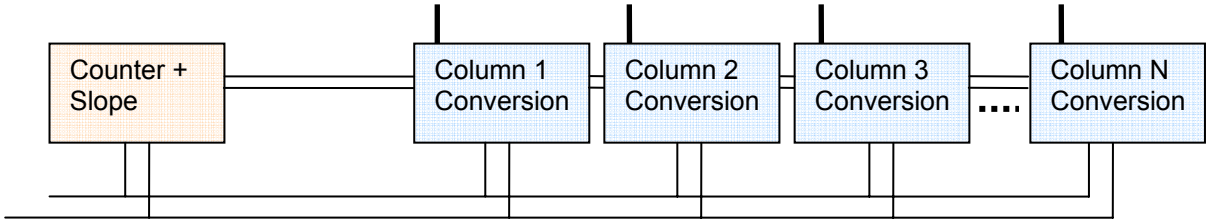
Figure 13: Dual integration time mode for high dynamic



8. 10 BIT ADC

Digital conversion is done by a high speed 10-bit column ADC. All the pixel values of the same line are converted in parallel.

Figure 14: Principle of the column Analog to Digital Converter



8.1 Analog gain

The analog gain is done by a slope adjustment. There are 7 available values, all programmable via SPI. Each ROI has its own analog gain:

- ***roi1_ana_gain***; for ROI 1 (see **reg_roi1*** § **20.3.11**)
- ***roi2_ana_gain***; for ROI 2 (see **reg_roi2*** § **20.3.12**)
- ***roi3_ana_gain***; for ROI 3 (see **reg_roi3*** § **20.3.13**)
- ***roi4_ana_gain***; for ROI 4 (see **reg_roi4*** § **20.3.14**)

8.2 External resistor choice

The ADC gain value is set through an external resistor connected between ADC_REF_1 and ADC_REF_2 pins. An internal protection against a short circuit between these two pins is included in the design.

$$R_{EXT} = \frac{K}{CLK\_ADC} - 80$$

Where K = 2.08 × 10^{12}, CLK_ADC is in Hertz and R_{EXT} is in Ohms.

With a 114 MHz ADC clock, the resistor value is 18.2 kΩ.

8.3 Analog Gain Tolerances

Table 9: ADC gain tolerances

Gain	1	1.5	2	3	4	6	8
Reference	89.09	59.09	44.18	29.64	22.30	14.90	11.27
Precision (%)	-	0.5	1	1	1	1	2

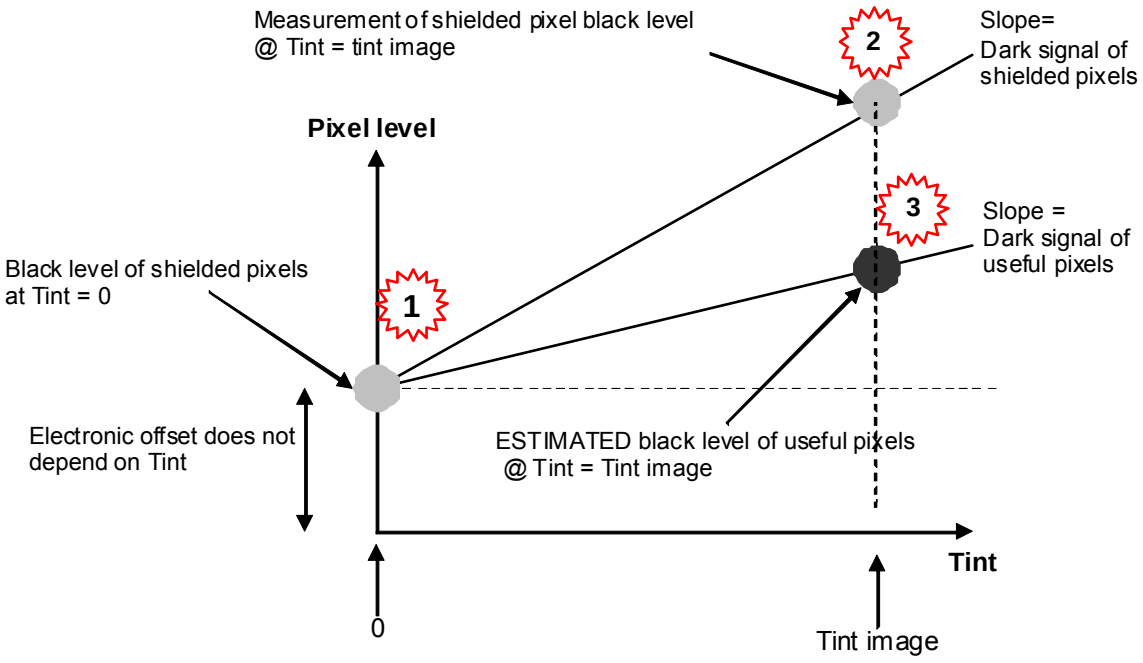
9. CLAMP & OFFSET ADJUSTMENT

The purpose of the automatic black level adjustment function (or clamp) is to cancel:

- The offset due to pixel dark current (offset variable with temperature and integration time).
- The analog chain offset (mainly due to comparator offset).

The black level adjustment is active up to 65 °C with 200 ms integration time
Black level adjustment can be automatic or manual. This is selected by the ***clamp_auto_en*** bit in the **<reg_clamp_cfg>** register. See § **20.3.17**
In order to compensate possible differences in dark current generation between masked pixels and useful pixels, the automatic black level correction works as follows:

Figure 15: Clamp principle



For each frame acquisition:

- A first measurement is done on a shielded pixel with a very short integration time (fixed to the minimum possible time) to determine the hardware offset of the acquisition chain (chain_offset).
- A second measurement is done to determine the dark signal mean value of a shielded pixel for the configured integration time (shld_pix_level).
- The dark signal of a useful pixel is deduced from these 2 measurements and from the ratio between useful and shielded pixels (V0_ratio). This ratio is configurable via the ***v0_gain*** bit field in the **<reg_clamp_cfg>** register. See § **20.3.17**

useful dark signal = (shielded pixel level – chain offset) x V0_ratio + chain offset

A lock mechanism guarantees a constant correction offset as long as the difference between the new correction offset and the current correction is less than a threshold configurable by ***clamp_lock_th*** in **<reg_clamp_cfg>** see § **20.3.17**. This mechanism is necessary to ensure offset stability during a video stream. It can be bypassed using ***clamp_lock_en*** in **<reg_clamp_cfg>** see § **20.3.18**

Offset can be adjusted using either ***clamp_add_offset*** (if ***clamp_auto_en*** = '1' in ***<reg_clamp_cfg>***) or ***clamp_manual_offset*** (if ***clamp_auto_en*** = '0' in ***<reg_clamp_cfg>***) in **<reg_clamp_offset>** see § **20.3.17.**

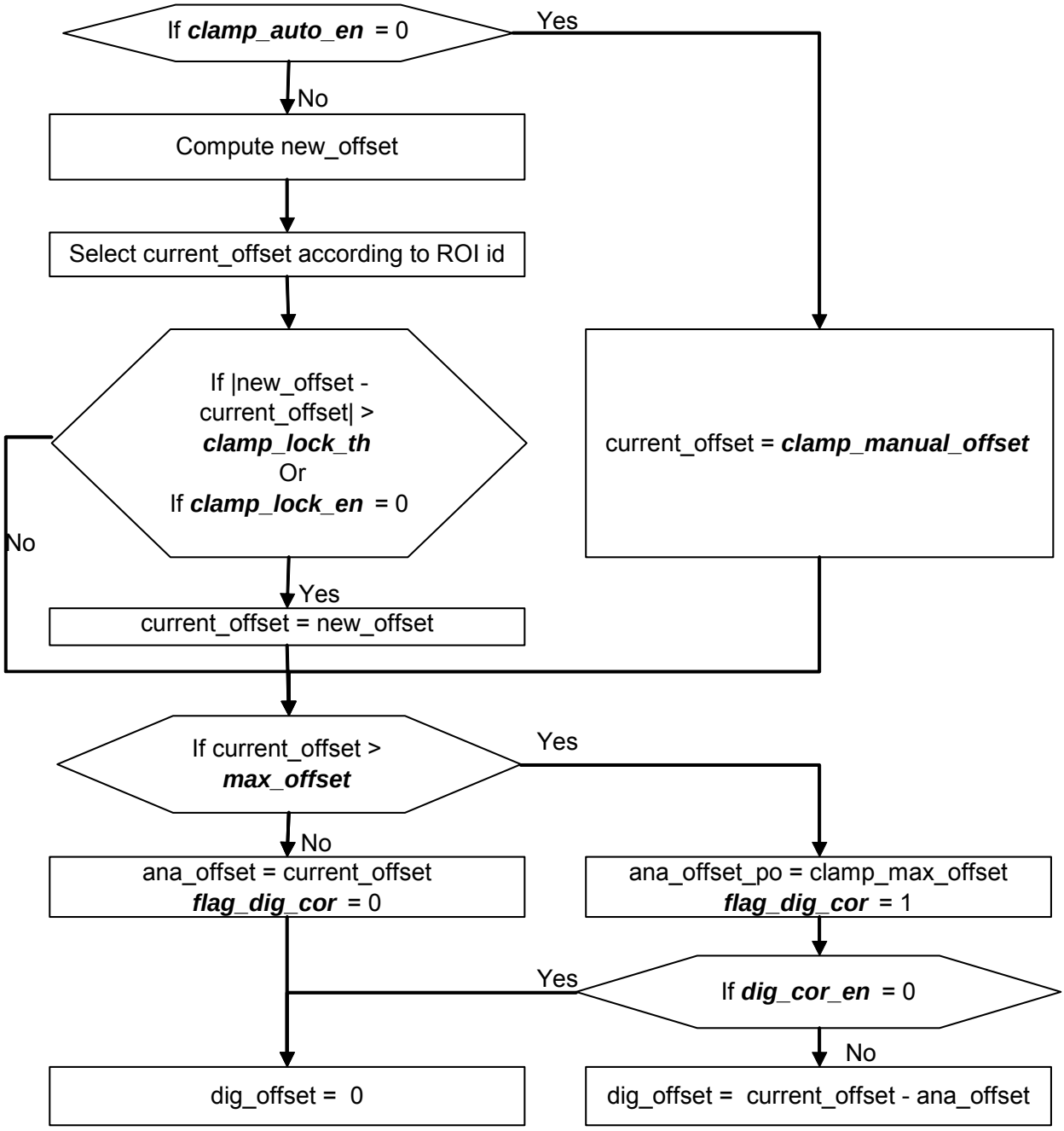
The ***flag_dig_cor*** flag in the **<fb_status>** register indicates if a digital correction is needed or not (see § **20.3.23**).

If the analog correction allowed by <**max_offset**> is saturated, a digital correction can be activated by setting <**dig_cor_en**>.

If <**dig_cor_en**> = 1 and analog offset is saturated, then the maximum data output level will be limited.

Digital and analog offsets are output in two feedback registers ***fb_ana_offset*** & ***fb_dig_offset*** in **< fb_clamp>** see § **20.3.22**

Figure 16: Clamp algorithm



10. DIGITAL GAIN

This block applies one global gain followed by four digital gains (for the Bayer or WRGB CFA structures) configurable by 8-bit SPI registers.

In **B&W products**, only the global gain is used.

To allow good precision with low gains the 8-bits for programming the digital gain are used as follow:
- The 2 MSB are used for precision P
- The 6 LSB are used to control the gain G (from 0 to 63)

The ROIX digital gains (***roiX_dig_gain)*** follow this rule:

$$Gain = 2^P \times \left(1 + \frac{G}{64}\right)$$

- For P=0 Gain varies from 1 to 1.98 in steps of 0.015
- For P=1 Gain varies from 2 to 3.97 in steps of 0.031
- For P=2 Gain varies from 4 to 7.94 in steps of 0.062
- For P=3 Gain varies from 8 to 15.88 in steps of 0.125

In **color products**, the four digital gains can be used to balance the four color channels (blue, green blue, green red and red):
- The 2 MSB are used for precision P
- The 6 LSB are used to control the gain G (from 0 to 63)

The four digital color gains (***gb_dig_gain***; ***gr_dig_gain***; ***b_dig_gain***; ***r_dig_gain***) follow this rule:

$$Gain = 2^{P-2} \times \left(1 + \frac{G}{64}\right)$$

- For P=0 Gain varies from 0.25 to 0.5 in steps of 0.004
- For P=1 Gain varies from 0.5 to 0.99 in steps of 0.008
- For P=2 Gain varies from 1 to 1.98 in steps of 0.016
- For P=3 Gain varies from 2 to 3.97 in steps of 0.0.31

11. DEFECTIVE PIXEL CORRECTION

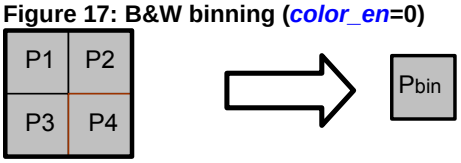
A multidirectional 3x3 median filter (with maximal weighting) is implemented and can be enabled by programming ***roi_ddc_en*** in **< reg_chain_cfg >**. See § **20.3.7**

This filter is compatible with B&W and color products (Bayer or WRGB). All pixels of the ROI are corrected: this correction deletes 2 pixels all around the input picture so the ROI output is reduced by 2 pixels in each line and column (See § **7.5.2.1**).

12. BINNING

Two Binning 2x2 modes are implemented:
- A binning for monochrome sensors (Figure 17),
- A binning for color sensors (Figure 18).

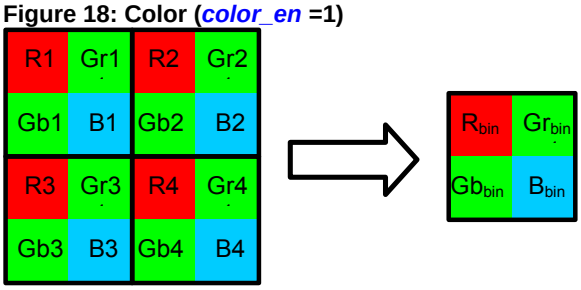
Figure 17: B&W binning (*color_en*=0)



$$P_{bin} = \frac{1}{k}\sum_{i=1}^{4} P_i$$

The k parameter (see ***binning_div_factor***) divides the sum by 1, 2 or 4.

Figure 18: Color (*color_en* =1)



$$X_{bin} = \frac{1}{k}\sum_{i=1}^{4} X_i$$

With X = B, Gb, Gr or R.

The k parameter (see ***binning_div_factor***) allows dividing the sum by 1, 2 or 4.
The binning respects the Bayer pattern to add only same color pixels.

When k= 4 → Average by 4 → Saturation remains the same and noise on the image is reduced by a factor 2.
When k=2 or 1, the sum is clipped at the value 1023
The dimensions of the binning output image are half the input image dimensions.

13. HISTOGRAM

Four histograms can be computed (for color sensors):
- The first one with green blue pixels
- The second one with red pixels
- The third one with blue pixels
- The fourth one with green red pixels

To enable histogram calculation program ***roi_histo_en*** in **< reg_chain_cfg>.** See § **20.3.7**

The number of categories (bins) is selectable: 8, 16, 32 or 64 using ***hist_bin_nb*** in **<reg_chain_cfg>**. See § **20.3.7**

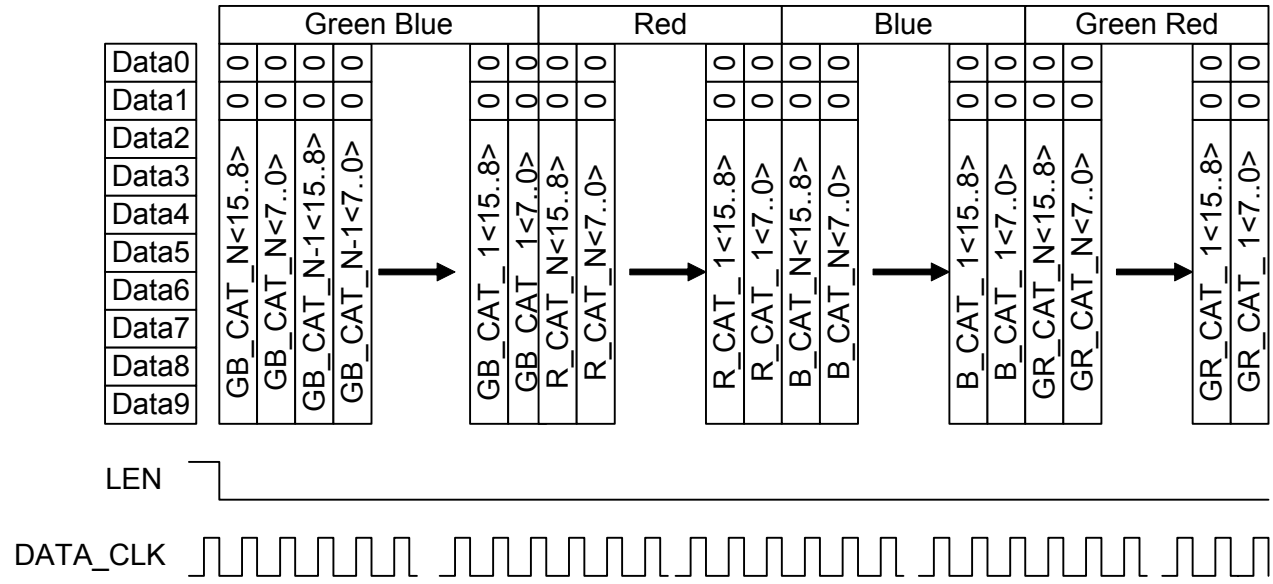
The histograms are output (see **Figure 21: Header and histogram** page **29**) with the number of bright pixels first.

Each category is coded on 16 bits and output on the 8 MSB of two successive pixels.

The 4 histograms are output serially without any delimiter.

The number of saturated pixels at zero and at 1023 are calculated and provided to the application in the footer. (See §**15**)

Figure 19: Histogram outputs



14. 10 TO 8 BIT COMPRESSION

To allow the use of 8-bit output, the amplitude range is redefined with 256 levels. 8databits are output on the 8 MSB. The transfer function is defined as in the following:

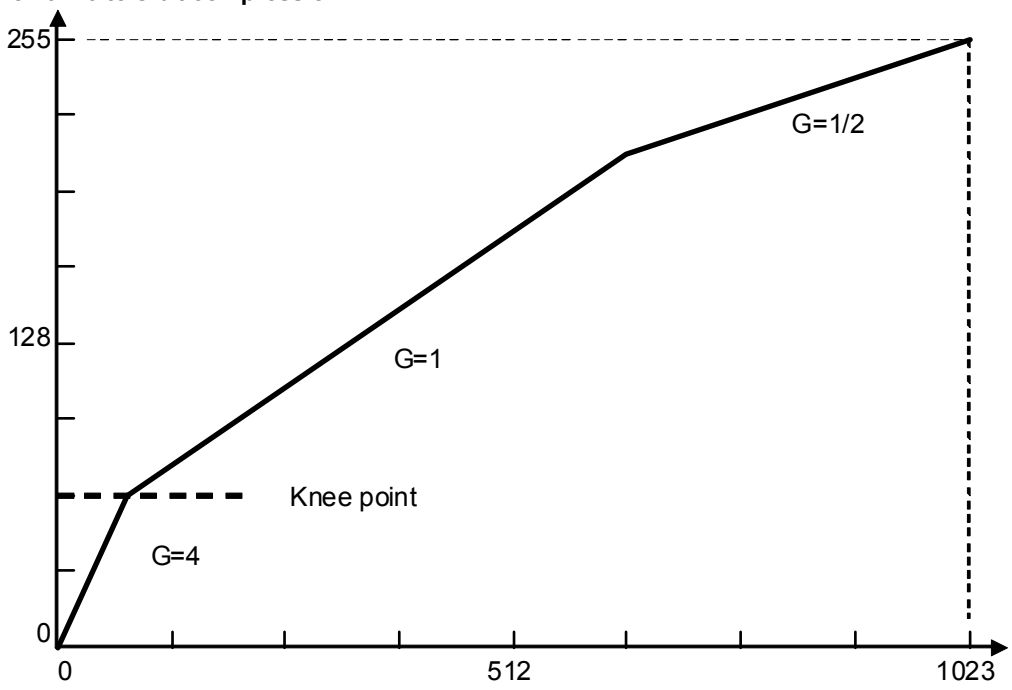
- The user has to choose the knee point K_N by programming ***range_coeff*** in **<reg_miscel1>**.
- The output value on 8 bits will follow these rules:

For $0 \leq$ **IN** $< K_N$ $\longrightarrow$ $OUT = IN$

For $K_N \leq$ **IN** $< 8 \bullet \left(128 - \frac{3}{4}K_N\right)$ $\longrightarrow$ $OUT = \frac{IN}{4} + \frac{3}{4}K_N$

For $8 \bullet \left(128 - \frac{3}{4}K_N\right) \leq$ **IN** < 1024 $\longrightarrow$ $OUT = \frac{IN}{8} + 128$

Figure 20: 10 to 8 bit compression



To enable this function use ***range_en*** in **<reg_chain_cfg>** see § **20.3.7.**
Using a knee point at 0 will only output the 8 MSB of 10-bit values to the 8 MSB of the output without any compression.

15. CONTEXT

This block inserts in the data stream, the configuration of the sensor used for the current image.

Insertion of the image context is under SPI control. See ***roi_context_out_en*** in **<reg_chain_cfg>** see § **20.3.7**

Each image has its own header and footer.

The output of context may be done with or without histogram output.

The context data are output on the first line and on the last line inside the FEN signal. The context is output as extra lines. If the stream is too long for the LEN (due to a small ROI) the output of the stream is not cut by the change of LEN state. This means that even for the context output the LEN duration is the same for the whole image comprising context and histograms.

Depending on **mask_idle_data** in ***<reg_miscel2>***, if the useful line length is too short, data may be truncated.

The data are output on the 8 MSB of the video output (the 2 LSB are left at 00).
The Figure 21 shows the location of the header and histogram data in the final frame structure:

Figure 21: Header and histogram

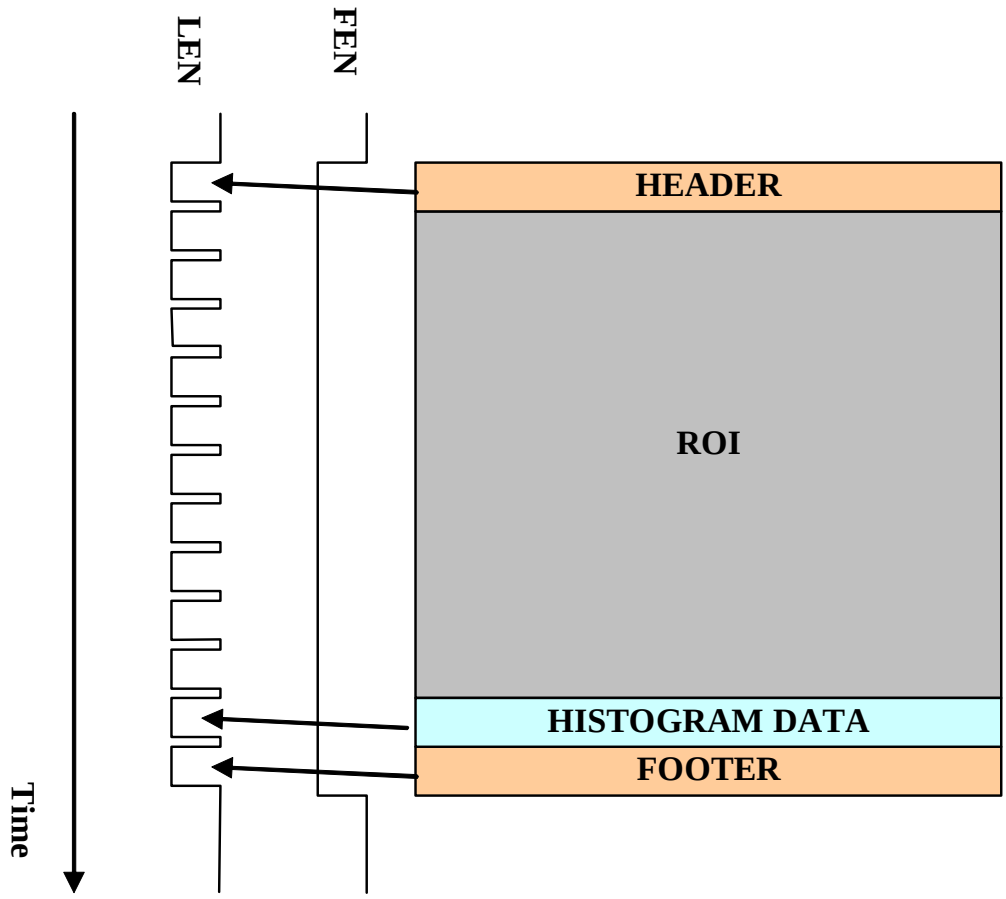


Table 10: Header content

Word count	Name	Description
0	“000000” & roi_id	ROI id
1	roi_nb	ROI number
2	“00000” & read_roi_0c_1[10:8]	Address of first column (MSB)
3	read_roi_0c_1[7:0]	Address of first column (LSB)
4	“00000” & read_roi_0l_1[10:8]	Address of first line (MSB)
5	read_roi_0l_1[7:0]	Address of first line (LSB)
6	“00000” & roi_width[10:8]	ROI width (MSB)
7	roi_width[7:0]	ROI width (LSB)
8	“00000” & roi_height[10:8]	ROI Height (MSB)
9	roi_height[7:0]	ROI Height (LSB)
10	t_int_ll[15:8]	Main ROI integration time in line (MSB)
11	t_int_ll[7:0]	Main ROI integration time in line (LSB)
12	“00” & t_int_clk[13:8]	MSB of extra ROI integration time in CLK_CTRL x *t_int_clk_mult_factor*
13	t_int_clk[7:0]	LSB of extra ROI integration time in CLK_CTRL x *t_int_clk_mult_factor*
14	analog_gain	ROI analog gain
15	dig_gain_glob	ROI Global digital gain
16	dig_gain_b	Blue digital gain
17	dig_gain_gb	Green blue digital gain
18	dig_gain_gr	Green red digital gain
19	dig_gain_r	Red digital gain
20	fb_ana_offset	Analog offset
21	fb_dig_offset	Digital offset
22	‘0’ &	
	fb_flag_dir_cor &	
	fb_error_time_overflow &	
	fb_error_corrupted_video &	
	fb_error_ll_vs_xfer &	
	fb_error_ll_vs_conv &	
	fb_error_t_int_big &	
	fb_error_t_int_small	
23	t_frame_period_actual[15:8]	Frame period (MSB)
24	t_frame_period_actual [7:0]	Frame period (LSB)
25, …	"00..0"	line is filled with extra 00

Table 11: Footer content

Word count	Name	Description
0	'0'	
	fb_flag_dir_cor &	
	fb_error_time_overflow &	
	fb_error_corrupted_video &	
	fb_error_ll_vs_xfer &	
	fb_error_ll_vs_conv &	
	fb_error_t_int_big &	
	fb_error_t_int_small	
1	low_sat_nb[15:8]	Number of pixels at 0 value (MSB)
2	low_sat_nb[7:0]	Number of pixels at 0 value (LSB)
3	high_sat_nb[15:8]	Number of pixels at 1023 value (MSB)
4	high_sat_nb[7:0]	Number of pixels at 1023 value (LSB)
5, …	"00..0"	line is filled with extra 00

16. MUX OUT

This block multiplexes the different signals to the output: video, context and histograms.

17. TIMING GENERATOR AND POWER MANAGEMENT

Under SPI control, the timing generator provides the necessary timing to the sensor. It manages the different read modes depending on the global states programmed by the application. It times the reading of the matrix to follow the ROI, Binning and sub sampling functions.

18. CLOCK GENERATOR

The application should provide 1 or 2 clocks to the sensor:

- The reference clock (CLK_REF),
- A second stable clock (CLK_FIX), to dither CLK_REF to improve EMC performance, for example.

Two other clocks are available in the sensor:

- CLK_OSC which is generated by an internal oscillator
- CLK_PLL which is output by the PLL with CLK_REF as the reference clock.

These four clocks are the sensor input clocks.

The sensor needs three different clocks for three separate domains (See Figure 2: Block diagram):

- One for the ADC (CLK_ADC).
- One for the Timing control (CLK_CTRL)
- One for the digital chain (CLK_CHAIN).

Figure 22: Clock management

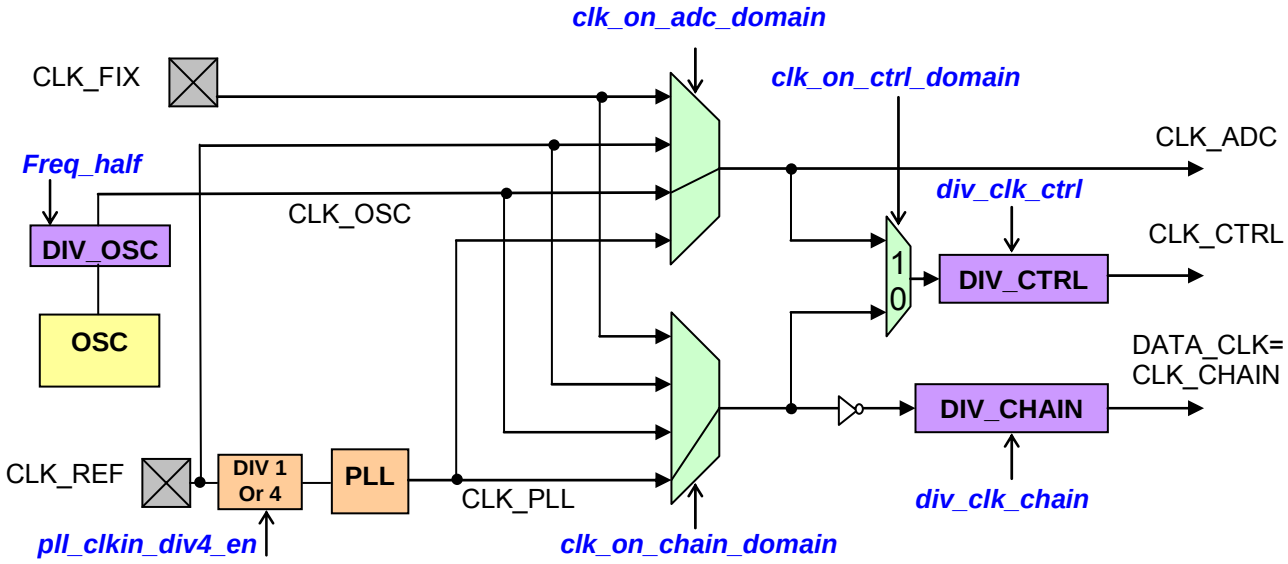
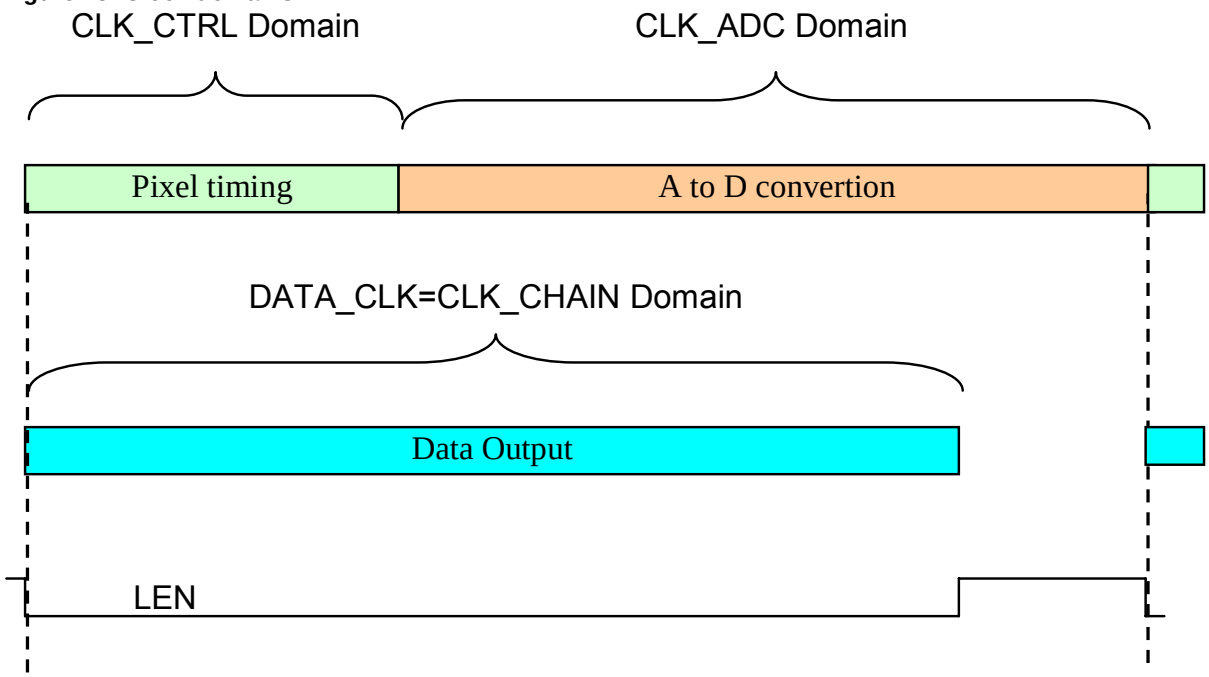


Figure 23: Clock domains.



Notes:

- CLK_ADC & CLK_CTRL must be stable. No dithered clock allowed.
- If needed for EMC constrains the CLK_CHAIN may be dithered.
- Pixel_timing duration is given at §
- A to D conversion and data output durations are computed in §

18.1 PLL

A Phase-Locked Loop block (PLL) is embedded to provide an output frequency (CLK_PLL) from a reference frequency (CLK_REF). (See **Figure 22: Clock management**)

If the PLL is not used, the block is in power down mode.

18.1.1 Register used

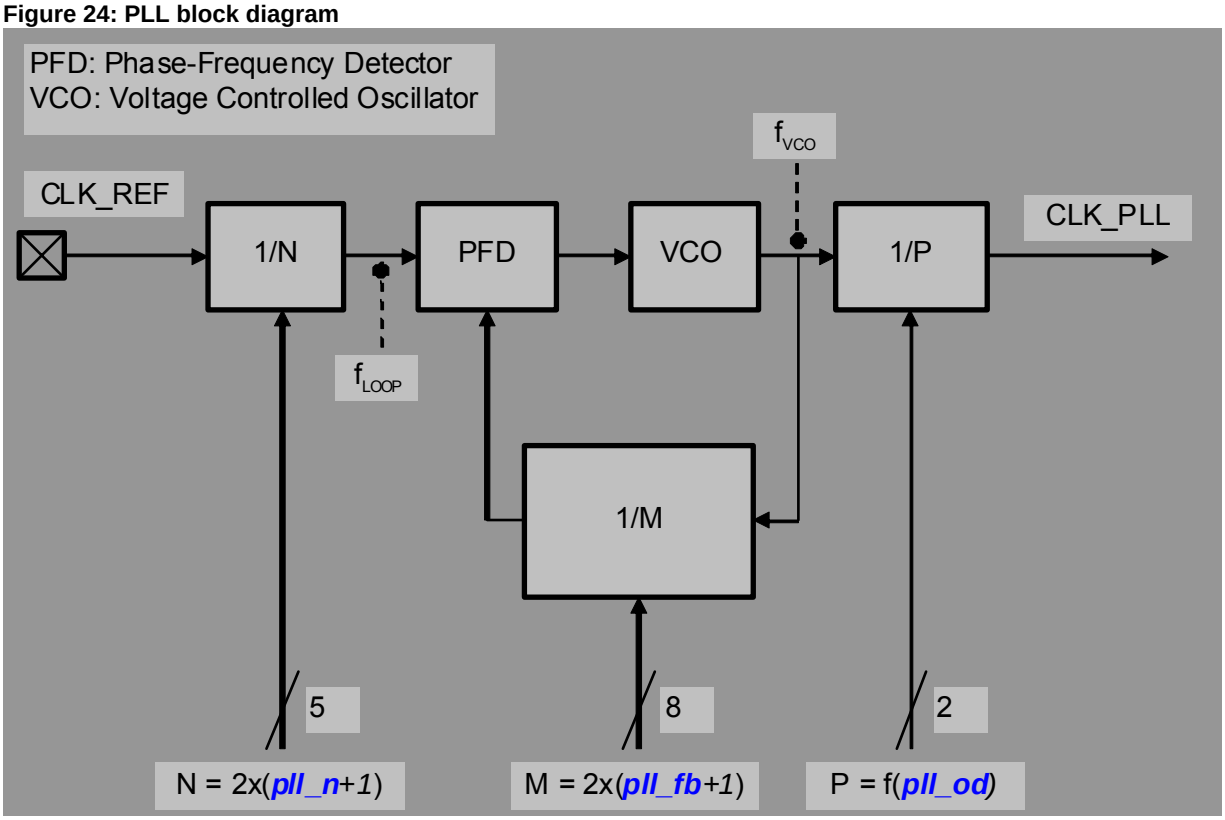
The registers used are *pll_od*, *pll_n* and *pll_fb* in <Reg_pll_cfg> see § **20.3.6**.

The PLL output frequency CLK_PLL is given by the equation:

$$CLK\_PLL = \frac{M}{N \times P} \times CLK\_REF$$

With:
4 < M =2x(pll_fb + 1)< 512,
2 < N =2x(pll_n + 1) < 20,
P (pll_od) = 4,
5 MHz < CLK_REF < 50 MHz

Figure 24: PLL block diagram



18.1.2 Limits and Conditions

The following conditions and limits must be respected to allow the PLL to operate efficiently:

- $325MHz < F_{VCO} = CLK\_PLL \times P < 480MHz$
- $2.5MHz < F_{LOOP}$
- $81.25MHz < CLK\_PLL < 120MHz$ (if directly used internally)

18.1.3 PLL settings calculations

For a given input frequency (CLK_REF) and the desired output frequency (CLK_PLL), follow these steps to calculate the pll_fb, pll_n and pll_od parameters.

1. Calculation of P:

- If CLK_PLL < 175 MHz → P = 4
- If 175 MHz < CLK_PLL < 351 MHz → P = 2
- If CLK_PLL > 350 MHz → P = 1

2. Calculation of N:

$$pll\_n = IntegerPart\left(\frac{CLK\_REF}{5}\right) - 1$$

3. Calculation of M:

$$pll\_fb = 2 \times RoundedUp\left(\frac{(pll\_n + 1) \times CLK\_PLL \times pll\_od}{CLK\_REF}\right) - 1$$

4. Calculation of the real CLK_PLL

The above formulas can be used to calculate the PLL output frequency (CLK_PLL).

The following table gives the some frequency calculation examples showing the pll_fb, pll_n and pll_od parameter settings used to obtain a 114 MHz system frequency with different input reference frequencies:

Table 12: example of PLL parameter settings for a 114MHz PLL output frequency

Parameter setting	CLK_REF input frequency		
	12 MHz	24 MHz	48 MHz
P	4	4	4
pll_od	h03	h03	h03
N	4	8	18
pll_n	h01	h03	h08
M	152	152	172
pll_fb	h4B	h4B	h55

18.2 Internal oscillator

The internal oscillator has to be calibrated by the application. During the calibration procedure the sensor counts the number of CLK_OSC cycles during the calibration reference period ***calib_count_ref***. The length of ***calib_count_ref*** is defined by the user as a number of CLK_REF cycles. The number of CLK_OSC cycles can be read in the ***fb_calib_count_osc*** register when the ***flag_reg_calib_count_ref*** flag goes back to low level.

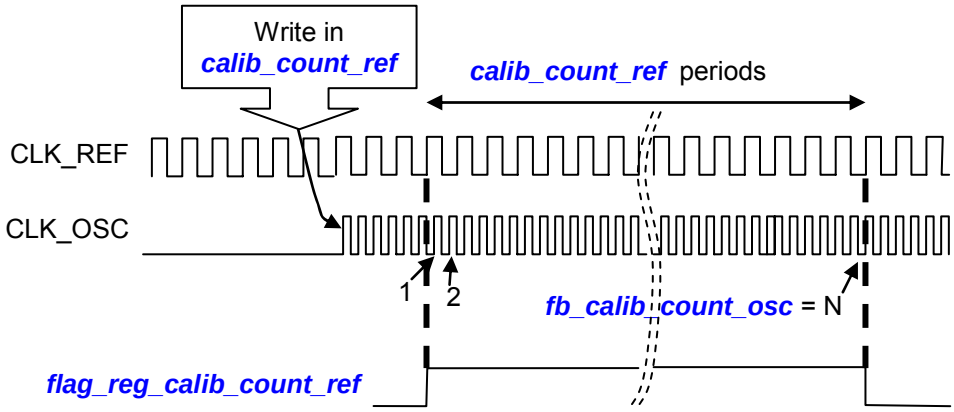
If needed the oscillator frequency can be adjusted using *prg_osc_freq_adjust* in **<reg_prg_osc>** see **20.3.19**

<**freq_half**> may be used to divide the internal oscillator frequency by 2.

The internal oscillator frequency can be computed using the formula below, where R_{EXT} is the ADC_REF external resistor:

$$Frequency = \frac{1}{\left[\frac{R_{EXT} \times 316(10^{-13})}{(36 + prg\_osc\_freq\_adjust)}\right] + 3.4(10^{-9})}$$

Figure 25: Oscillator calibration.



18.3 Nominal clock configurations

CLK_OSC is used for A to D conversion (CLK_ADC) and pixel timing (CLK_CTRL) with a DIV_CTRL = 2.
CLK_PLL is used for the digital chain (CLK_CHAIN).

The typical clock configuration is as follows:
clk_on_adc_domain = h2 in <**reg_clk_cfg**> see § **20.3.5**
clk_on_ctrl_domain = h1 in <**reg_clk_cfg**> see § **20.3.5**
clk_on_chain_domain = h3 in <**reg_clk_cfg**> see § **20.3.5**
div_clk_chain = h2 in <**reg_clk_cfg**> see § **20.3.5**

With this configuration a dithered clock can be used as CLK_REF for the PLL.

To allow the maximum frame rate, CLK_OSC must be above 114 MHz.

19. TEST PATTERN GENERATOR

A test pattern allows the signal processing to be checked. It generates repeated slope from 0 to 1023 with a 1 LSB step.

The timing and image size used in this mode uses the ROI and timing configuration.

The block generates 3 different patterns.

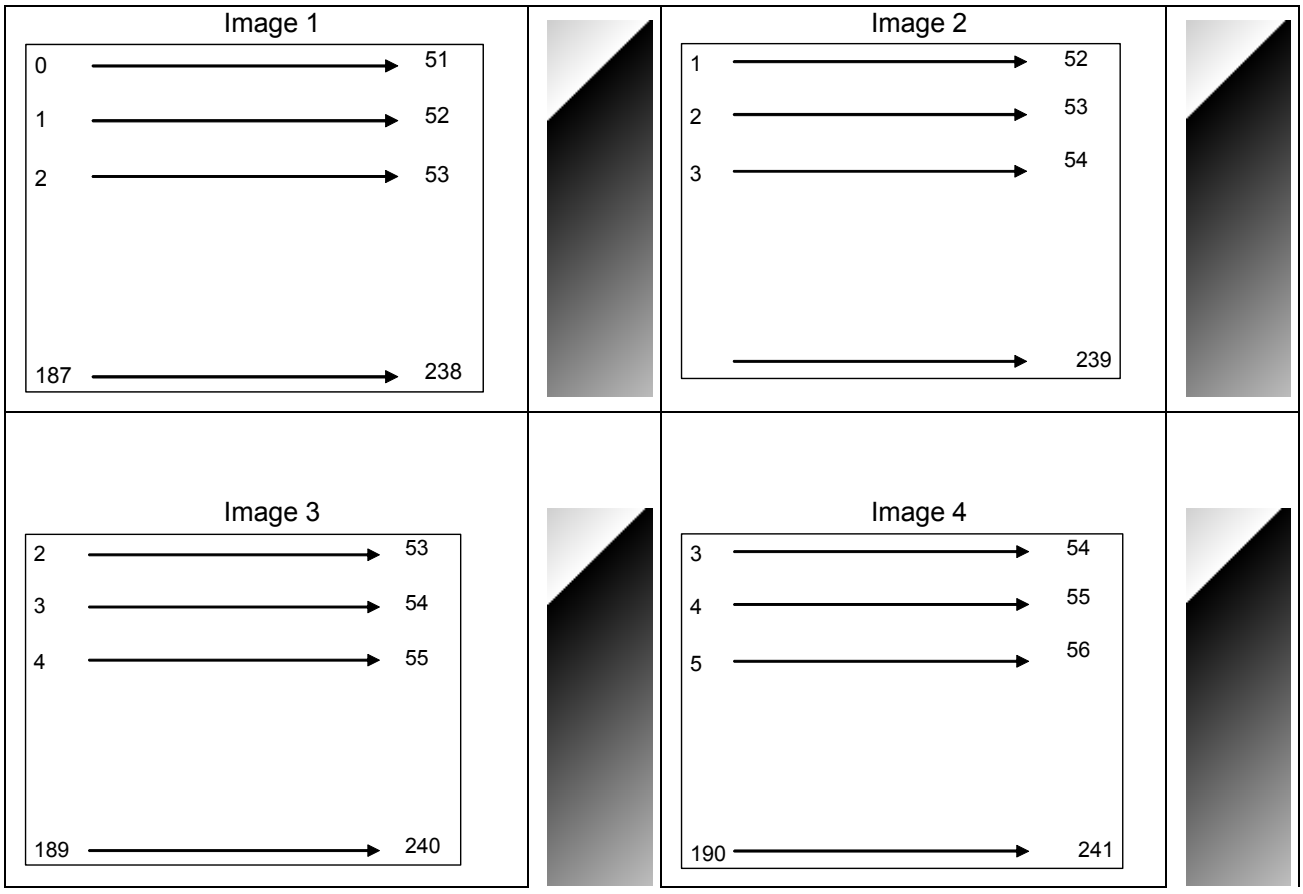
19.1 Moving test pattern

pattern_ena = 01

In this mode, the test pattern changes from line to line and from frame to frame.

Figure 26 gives examples for a ROI (52 x 188 pixels). If the ROI width or height is larger than 1024 the test pattern counter will create additional ramp pulses in both directions.

Figure 26: Moving test pattern

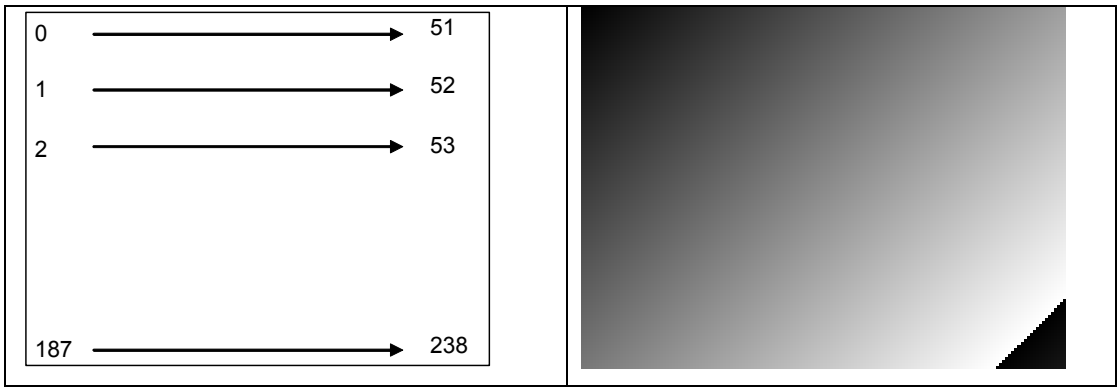


19.2 Fixed test pattern

pattern_ena = 10

Figure 27 shows this pattern, using the same resolution as the previous example. The test pattern ramp generator will always have the same starting point at 0, at the first pixel of the first line.

Figure 27: Fixed test pattern example



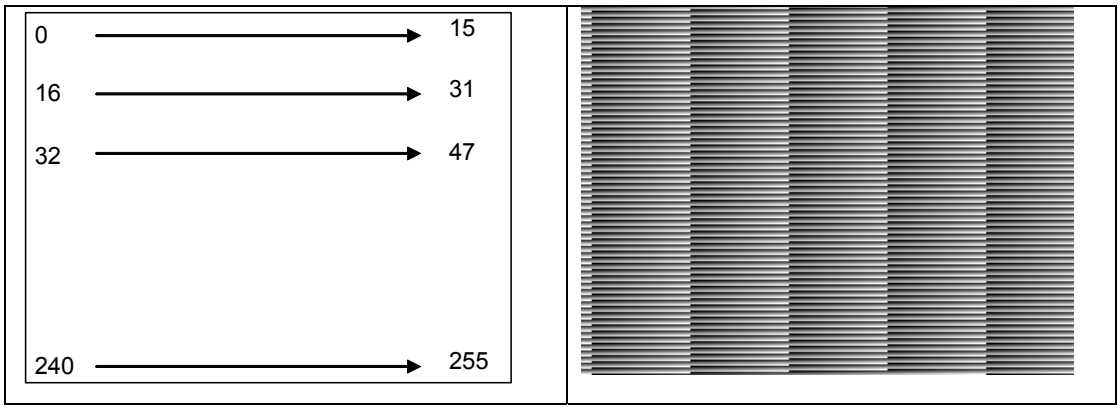
19.3 Functional test pattern

pattern_ena = 11

This test pattern allows all output values to occur in the smallest possible image. The test pattern counter counts only during active FEN & LEN. The first pixel of the first line is at 0

Figure 25 gives an example of a 16 x 16 image:

Figure 28: Functional test pattern



20. SPI

The SPI communication interface allows the sensor be controlled by an external device.

Most of built-in functions are configurable via SPI registers. We can distinguish 6 types of registers:

- **Dynamic registers (D)** are read/write registers. They are refreshed only one time per frame at the beginning of the readout or on matrix reset, depending on the selected readout mode. A lock mechanism allows the refresh to be disabled. This is useful for insuring that several register changes are taken into account in the same frame,
- **Mailbox registers (MBX)** are read/write registers. They are used to send abort requests or read calibration status information,
- **Static registers (S)** are read/write registers. Any change in their value is taken into account immediately,
- **Restricted static registers (RS)** are like static registers but they must be modified only in STANDBY or IDLE state. Any change in their value during an acquisition sequence may have an unpredictable effect,
- **Feedback registers (F)** are read only registers. They are used to report the current state of the sensor,
- **Reset registers (RST)** are read/write registers. They are used to perform a soft reset of the device.

20.1 Register summary tables

Table 13 : 8-bit registers

Addr (hex)	Register Name	Type	Width	Bit	Content	Reference section	
0000	reg0	rs	8	7:0	Burst mode	20.2.1	20.4
0001	reg_soft_reset	rst	8	7:0	Soft reset global command	20.2.2	
0002	calib_mbx	mbx	1	0	flag_reg_calib_count_ref	20.2.3	
0003	abort_mbx	mbx	1	0	flag_abort_mbx	20.2.4	

Table 14 : 16-bit registers

Addr (hex)	Register Name	Type	Width	Bit	Content	Reference section	
0004	reg_line_cfg	rs	4	15:12	extra_line_nb	20.3.1	
			1	11	Reserved		
			11	10:0	line_length		
0005	reg_flash_delay	rs	8	15:8	t_flash_del_off	20.3.2	22.6
			8	7 :0	t_flash_del_on		
0006	reg_miscel1	rs	8	15:8	max_offset	20.3.3	
			8	7:0	range_coeff		14
0007	reg_miscel2	rs	1	15	Reserved	20.3.4	
			1	14	sync_flo_inv		22.6
			1	13	sync_len_inv		
			1	12	sync_fen_inv		
			1	11	mask_idle_data		
			1	10	color_en		
			1	9	clamp_auto_en		9
			1	8	roi_expanded		
			1	7	roi_flip_h		7.5.1
			1	6	roi_flip_v		
			2	5:4	pattern_type		19
			4	3:0	vlr_ph_ctrl		
0008	reg_clk_cfg	rs	1	15	clk_chain_low_pwr	20.3.5	

			1	14	clk_out_inv		
			1	13	freq_half		
			1	12	clk_on_ctrl_domain		
			2	11:10	clk_on_adc_domain		
			2	9:8	clk_on_chain_domain		18
			4	7:4	div_clk_ctrl		
			4	3:0	div_clk_chain		
0009	reg_pll_cfg	rs	1	15	pll_clk_div4_en	20.3.6	18.1
			2	14:13	pll_od		
			5	12:8	pll_n		
			8	7:0	pll_fb		
000A	reg_chain_cfg	d	2	15:14	t_int_clk_mult_factor	20.3.7	
			2	13:12	roi_max_id		7.5.3
			2	11:10	hist_bin_nb		13
			2	9:8	binning_div_factor		12
			1	7	roi_context_out_en		15
			1	6	roi_histo_en		
			1	5	roi_ddc_en		11
			1	4	range_en		14
			1	3	roi4_binning_en		7.5.2.1
			1	2	roi3_binning_en		
			1	1	roi2_binning_en		
			1	0	roi1_binning_en		
000B	reg_ctrl_cfg	rs	1	13	dum_stdby_en	20.3.8	
			1	12	dum_pwrup_en		
			1	11	dum_img_out_en		
			1	10	lock_dyn_reg		
			1	9	trig_pad_inv		
			1	8	trig_pad_sel		
			2	7:6	roi_flash_mode		22.6
			2	5:4	roi_readout_mode		22.2
			1	3	roi_video_en		21.2.5
			1	2	roi_overlap_en		22.2
			1	1	trig_rqst		
			1	0	stdby_rqst		21.1.2
000C	reg_t_frame_period	d	16	15:0	t_frame_period	20.3.9	
000D	reg_t_wait	d	16	15:0	t_wait	0	
000E	reg_roi1_t_int_ll	d	16	15:0	roi1_t_int_ll	20.3.11	7.5.3
000F	reg_roi1_rep_nb_t_int_clk	d	8	15:8	roi1_rep_nb		
			8	7:0	roi1_t_int_clk		
0010	reg_roi1_t_wait_ext	d	11	10:0	roi1_t_wait_ext		
0011	reg_roi1_gain	d	3	10:8	roi1_ana_gain		
			8	7:0	roi1_dig_gain		
0012	reg_roi1_0l_1	d	11	10:0	roi1_0l_1		
0013	reg_roi1_h_1	d	12	11	roi1_subs_v_mult16		
				10:0	roi1_h_1		

0014	reg_roi1_0c_1	d	11	10:0	roi1_0c_1		
0015	reg_roi1_w_1	d	11	10:0	roi1_w_1		
0016	reg_roi1_0l_2	d	11	10:0	roi1_0l_2		
0017	reg_roi1_h_2	d	11	10:0	roi1_h_2		
0018	reg_roi1_0c_2	d	11	10:0	roi1_0c_2		
0019	reg_roi1_w_2	d	11	10:0	roi1_w_2		
001A	reg_roi1_subs	d	8	15:8	roi1_subs_v		7.5.2.1
			8	7:0	roi1_subs_h		
001B	reg_roi2_t_int_ll	d	16	15:0	roi2_t_int_ll		
001C	reg_roi2_rep_nb_t_int_clk	d	8	15:8	roi2_rep_nb		
			8	7:0	roi2_t_int_clk		
001D	reg_roi2_t_wait_ext	d	11	10:0	roi2_t_wait_ext		
001E	reg_roi2_gain	d	3	10:8	roi2_ana_gain		
			8	7:0	roi2_dig_gain		7.5.3
001F	reg_roi2_0l_1	d	11	10:0	roi2_0l_1	20.3.12	
0020	reg_roi2_h_1	d	12	11	roi2_subs_v_mult16		
				10:0	roi2_h_1		
0021	reg_roi2_0c_1	d	11	10:0	roi2_0c_1		
0022	reg_roi2_w_1	d	11	10:0	roi2_w_1		
0023	reg_roi2_subs	d	8	15:8	roi2_subs_v		7.5.2.1
			8	7:0	roi2_subs_h		
0024	reg_roi3_t_int_ll	d	16	15:0	roi3_t_int_ll		
0025	reg_roi3_rep_nb_t_int_clk	d	8	15:8	roi3_rep_nb		
			8	7:0	roi3_t_int_clk		
0026	reg_roi3_t_wait_ext	d	11	10:0	roi3_t_wait_ext		
0027	reg_roi3_gain	d	3	10:8	roi3_ana_gain		
			8	7:0	roi3_dig_gain		7.5.3
0028	reg_roi3_0l_1	d	11	10:0	roi3_0l_1	20.3.13	
0029	reg_roi3_h_1	d	12	11	roi3_subs_v_mult16		
				10:0	roi3_h_1		
002A	reg_roi3_0c_1	d	11	10:0	roi3_0c_1		
002B	reg_roi3_w_1	d	11	10:0	roi3_w_1		
002C	reg_roi3_subs	d	8	15:8	roi3_subs_v		7.5.2.1
			8	7:0	roi3_subs_h		
002D	reg_roi4_t_int_ll	d	16	15:0	roi4_t_int_ll	20.3.14	
002E	reg_roi4_rep_nb_t_int_clk	d	8	15:8	roi4_rep_nb		
			8	7:0	roi4_t_int_clk		
002F	reg_roi4_t_wait_ext	d	11	10:0	roi4_t_wait_ext		
0030	reg_roi4_gain	d	3	10:8	roi4_ana_gain		
			8	7:0	roi4_dig_gain		7.5.3
0031	reg_roi4_0l_1	d	11	10:0	roi4_0l_1		
0032	reg_roi4_h_1	d	12	11	Roi4_subs_v_mult16		
				10:0	roi4_h_1		
0033	reg_roi4_0c_1	d	11	10:0	roi4_0c_1		
0034	reg_roi4_w_1	d	11	10:0	roi4_w_1		
0035	reg_roi4_subs	d	8	15:8	roi4_subs_v		7.5.2.1

			8	7:0	roi4_subs_h		
0036	reg_dig_gain_gb_gr	d	8	15:8	gb_dig_gain	20.3.15	10
			8	7:0	gr_dig_gain		
0037	reg_dig_gain_b_r	d	8	15:8	b_dig_gain	20.3.16	
			8	7:0	r_dig_gain		
0038	reg_clamp_offset	d	8	15:8	clamp_add_offset	20.3.17	
			8	7:0	clamp_manual_offset		
0039	reg_clamp_cfg	rs	3	14:12	init_line_nb	20.3.18	
			4	11:8	clamp_lock_th		
			1	7	clamp_lock_en		
			1	6	dig_cor_en		
			6	5:0	v0_gain		
003A	reg_prg_osc	s	7	15:9	prg_osc_vsat_adjust	20.3.19	
			2	8:7	prg_osc_vsat_select		
			7	6:0	prg_osc_freq_adjust		
003B	reg_calib_count_ref	s	16	15:0	calib_count_ref	20.3.20	
003C	fb_calib_osc_count	f	16	15:0	fb_calib_count_osc	20.3.21	
003D	fb_clamp	f	8	15:8	fb_ana_offset	20.3.22	
			8	7:0	fb_dig_offset		
003E	fb_status	f	1	8	flag_dig_cor	20.3.23	
			2	7:6	fb_state_main_global		
			1	5	error_time_overflow		
			1	4	error_corrupted_video		
			1	3	error_ll_vs_xfer		
			1	2	error_ll_vs_conv		
			1	1	error_t_int_big		
			1	0	error_t_int_small		
0044	clk_out_low_pwr	rs	1	13	clk_out_low_pwr		
0049	pixtime_read_width	rs	8	15:8	pixtime_read_5t_width		
			8	7:0	pixtime_read_4t_width		
007F	chip_id	f	16	15:0	chip_id		

20.2 8 bit register descriptions

20.2.1 Register 0

Name	reg0
Address	h00
Type	Restricted Static
Default	h01

Def Val	Bitfield name	Description
0000 0001	reg0_0[7:0]	Burst mode 0→ Normal mode active (no burst) **1→ Burst mode active**

20.2.2 Soft reset register

Name	reg_soft_reset
Address	h01
Type	soft reset
Default	h00

Def Val	Bitfield name	Description
0000 0000	soft_reset[7:0]	Soft reset Writing or reading in SPI address h01 resets the whole chip, except the SPI state machine

20.2.3 Calibration Mailbox

Name	calib_mbx
Address	h02
Type	Mailbox
Default	h00

Def Val	Bitfield name	Description
....0	flag_reg_calib_count_ref	Oscillator calibration status 0 → Calibration sequence has ended (or not requested) 1 → Request was recorded. Calibration is ongoing

20.2.4 Abort mailbox

Name	abort_mbx
Address	h03
Type	Mailbox
Default	h00

Def Val	Bitfield name	Description
....0	flag_abort_mbx	Abort Status Write to this address to send an abort request. 0 → Abort has ended (or not requested) 1 → Request was recorded. Current sequence should stop within one line duration.

20.3 16 bit register descriptions

20.3.1 Line configuration

Name	reg_line_cfg
Address	h04
Type	Restricted static
Default	h886E

Default Value		Bitfield name	Description
1000 ----	---- ----	extra_line_nb[15:12]	Number of extra lines Defines the number of extra lines added after ROI readout Min = 0 → 1 line added **Default = h8** → 9 lines added Max = hF → d16 lines added See formula below.
---- 1---	---- ----	reserved	
---- -000	0110 1110	line_length[10:0]	Line length Defines the line length specified in *CLK_CTRL* cycles multiplied by 8 with CLK_CTRL=57MHz Min =0 **Default = h6E** → 15.44 µs Max = h7FF → 287 µs

There is a minimum *extra_line_nb* to be respected, depending on *reg_chain_cfg* see § **20.3.7:**
MIN extra_line_nb = (2 x roi<i>_binning_en + roi_histo_en + roi_context_out_en) x 2^(roi<i>_binning_en)

20.3.2 Flash delay

Name	reg_flash_delay
Address	h05
Type	Restricted static
Default	h0000

Default Value		Signal Name	Description
0000 0000	---- ----	t_flash_del_off[7:0]	Flash off delay Delay between end of active FLO and end of integration in number of lines **Min = 0 → No delay added** Max = hFF → 255 lines delay • *t_flash_del_off* must be lower than *roi<i>_t_int_ll* (ex: @ h0E for ROI1).
---- ----	0000 0000	t_flash_del_on[7:0]	Flash on delay Delay between start of active FLO and start of integration in number of lines **Min = 0 → No delay added** Max = hFF → 255 lines delay ☞ *t_flash_del_on* increases the frame period if *roi_overlap_en* (@ h0B) = 0. ☞ If *roi_readout_mode* (@ h0B) = 4T ERS, the applied delay will be the programmed delay + 2 lines.

- Both flash delays are automatically set to 0 if *roi_flash_mode* (@ h0B) = 0 (= FLASH_OFF).
- *t_flash_del_off* is ignored if *roi_flash_mode* (@ h0B) = 3 (= FLASH_ON).
- *t_flash_del_off* should be set to 0 if *roi_readout_mode* (@ h0B) = 4T+ERS and *roi_overlap_en* (@ h0B) = 1. (If not, there is a risk of finding “holes” in FLO)

20.3.3 Miscellaneous register 1

Name	reg_miscel1
Address	h06
Type	Restricted static
Default	h345A

Default Value		Bitfield name	Description
0011 0100	---- ----	max_offset[7:0]	ADC max offset Maximum offset that can be applied to ADC column (analogically) MIN = 0 **Default → h34** : Offset max 208 LSB @gain1 MAX → hFF ☞ See *dig_cor_en* (@ h39) to enable digital offset beyond this offset, and see *fb_ana_offset* / *fb_dig_offset* (@ h3D) to see how the offset is effectively split ☞ This register influences *line_length* (@ h04)
---- ----	0101 1010	range_coeff[7:0]	10 to 8 bit knee point Define the knee Kn for 10-bit to 8-bit compression Min = h00 → gain 1/4 **Default = h5A** Max = h92 → function with only 2 slopes G=4 and G= 1/2 ☞ This register is used only if *range_en* (@ h0A) = 1

20.3.4 Miscellaneous register 2

Name	reg_miscel2
Address	h07
Type	Restricted static
Default	h0A01

Default Value		Bitfield name	Description
0--- ----	---- ----	reserved	
-0-- ----	---- ----	sync_flo_inv	FLO signal polarity Inversion of polarity for *FLO* output signal. *FLO* is normally active high. **0 → *FLO* is not inverted (active high: *FLO* = 1 means that light may be turned on)** 1 → *FLO* is inverted (active low)
--0- ----	---- ----	sync_len_inv	LEN signal polarity Inversion of polarity for *LEN* output signal : *LEN* is normally active low **0 → *LEN* is not inverted (active low: *LEN* = 0 means that pixels are being output)** 1 → LEN is inverted (active high)
---0 ----	---- ----	sync_fen_inv	FEN signal polarity Inversion of polarity for *FEN* output signal : *FEN* is normally active low **0 → *FEN* is not inverted (active low: F*EN* = 0 means that pixels are being output)** 1 → FEN is inverted (active high)

---- 1---	---- ----	mask_idle_data	Enables a mask which sets output data to 0, when not useful 0 → *D0..D9* output data may change, whatever LEN value **1 → if *LEN* is at inactive level then *D0..D9* are set to 0**
---- -0--	---- ----	color_en	Color mode selection **0 → B&W mode** 1 → Color mode (with Bayer pattern) ☞ This bit is used for default correction, binning algorithms and for ROI size calculation. It must be set to 1 when using a color sensor.
---- --1-	---- ----	clamp_auto_en	Auto clamp mode Enables automatic black level adjustment 0 → Black level adjustment has to be done manually **1 → Enables automatic black level adjustment**
---- ---0	---- ----	roi_expanded	ROI expanded mode This mode allows the ROI to include the entire matrix including the masked reference lines : **0→ Programmed ROI has its origin (0,0) in the first illuminated pixel of the physical matrix** 1→ Programmed ROI has its origin (0,0) in the first pixel of the physical matrix, including dark pixels (The 19 first black lines can be read at the beginning of frame) If *roi_expanded*= 1, *clamp_auto_en* (bit 9) must be set at 0.
---- ----	0--- ----	roi_flip_h	Horizontal flip enable **0 → No horizontal flip** 1 → Horizontal flip
---- ----	-0-- ----	roi_flip_v	Vertical flip enable **0 → No vertical flip** 1 → Vertical flip
---- ----	--00 ----	pattern_type[1:0]	Test pattern type selection **00→ Video output** 01→ not used 10→ Diagonal grey scale pattern, still image with first pixel = 0 11→ Counting pattern, with continuously incrementing pixel
---- ----	---- 0001	vlr_ph_ctrl[3:0]	Photodiode antiblooming control (low level of reset gate) h0→ NOT allowed **h1→ linear response** h2→ hF control the knee point between linear response and log response.

20.3.5 Clock configuration

Name	reg_clk_cfg
Address	h08
Type	Restricted static
Default	hDF21

Default Value		Bitfield name	Description
1--- ----	---- ----	clk_chain_low_pwr	CLK_CHAIN low power mode Defines *CLK_CHAIN* activity (digital chain clock) 0 → *CLK_CHAIN* active during whole acquisition (integration and readout) **1 → *CLK_CHAIN* active only for data readout**
-1-- ----	---- ----	clkout_inv	Clock output polarity Phase of output clock DATA_CLK / CLK_CHAIN: 0 → rising edge is simultaneous with output data change **1 → falling edge is simultaneous with output data change**

Default Value		Bitfield name	Description
--0- ----	---- ----	freq_half	Oscillator frequency divider Controls the frequency generated by the internal oscillator : **0 → Oscillator frequency not divided** 1 → Oscillator frequency is divided by 2 ☞ If *freq_half*=1, the *fb_calib_count_osc* (@ h3C) counts the divided period.
---1 ----	---- ----	clk_adc_on_ctrl_domain	CLK_CTRL clock source selection Selects the clock source for CLK_CTRL (before division): 0 → CLK_CHAIN **1 → CLK_ADC**
---- 11--	---- ----	clk_on_adc_domain[1:0]	CLK_ADC clock source selection Selects the clock source for CLK_ADC: 00 → clock from CLK_FIX pad 01 → clock from CLK_REF pad 10 → clock from internal oscillator **11 → clock from PLL**
---- --11	---- ----	clk_on_chain_domain[1:0]	CLK_CHAIN clock source selection S selects the clock source for CLK_CHAIN: 00 → clock from CLK_FIX pad 01 → clock from CLK_REF pad 10 → clock from internal oscillator **11 → clock from PLL**
---- ----	0010 ----	div_clk_ctrl[3:0]	CLK_CTRL frequency divider Defines the clock division ratio applied to *CLK_CTRL*. Min = h0 & h1 → *CLK_CTRL* divided by DIV_CTRL = 1 **Default h2 → *CLK_CTRL* divided by DIV_CTRL = 2** Max hF → *CLK_CTRL* divided by DIV_CTRL = 15
---- ----	---- 0001	div_clk_chain[3:0]	CLK_CHAIN frequency divider Defines the clock division ratio applied to *CLK_CHAIN*. Min = h0 / h1 → *CLK_CHAIN* divided by DIV_CHAIN = 1 **Default h1 → *CLK_CHAIN* divided by DIV_CHAIN = 1** Max hF → *CLK_CHAIN* divided by DIV_CHAIN = 15

20.3.6 PLL configuration

Name	reg_pll_cfg
Address	h09
Type	Restricted static
Default	h6125

Default Value		Bitfield name	Description
0--- ----	---- ----	pll_clkin_div4_en	Division factor applied on the input reference clock (allows inputting up to 200MHz on the PLL); This value has to be taken into account when PLL parameters are computed. **0 → clock divided by 1** 1 → clock divided by 4
.11- ----	---- ----	pll_od[1:0]	PLL P parameter 00 → P= 1 01 → P= 2 10 → forbidden value **11 → P= 4**
.--0 0001	---- ----	pll_n[4:0]	PLL N parameter N = 2 x (*pll_n* + 1) MIN → h00 : N = d2 **Default → h01 : N = d4** MAX → h09 : N = d20

Default Value		Bitfield name	Description
.--- ----	0010 0101	pll_fb[7:0]	PLL M parameter M = 2 x (*pll_fb* + 1) MIN → h01 : M = d4 **Default → h25 : M = d76** MAX → hFF : M = d512

20.3.7 Chain configuration

Name	reg_chain_cfg
Address	h0A
Type	Dynamic
Default	h0200

Default Value		Bitfield name	Description
00-- ----	---- ----	t_int_clk_mult_factor[1:0]	Integration time multiplication factor Multiplication factor of the decimal part of integration time. **00 → x 8** 01 → x 16 10 → x 32 11 → x 64 ☞ it influences each *roi<i>_t_int_clk* (@h0F / @h1C …)
..00 ----	---- ----	roi_max_id[1:0]	Number of ROIs Defines the maximum number of ROIs to read in MIMR mode. **00 → 1 ROI : ROI1** 01 → 2 ROIs : ROI1 & ROI2 10 → 3 ROIs : ROI1, ROI2 & ROI3 11 → 4 ROIs : ROI1, ROI2, ROI3 & ROI4
..-- 00--	---- ----	hist_bin_nb[1:0]	Number of histogram bins Defines the maximum number of histogram bins: **00 → 64 / 1 = 64 bins** 01 → 64 / 2 = 32 bins 10 → 64 / 4 = 16 bins 11 → 64 / 8 = 8 bins (Warning, possible overflow) ☞ used only if r*oi_histo_en* ON, in the same register
..-- --10	---- ----	binning_div_factor[1:0]	Binning division factor Division factor of 4-pixel sum in binning mode 00 → 4-pixel sum divided by 1 01 → 4-pixel sum divided by 2 **10 → 4-pixel sum divided by 4** ☞ used only if *roi<i>_binning_en* is ON, in the same register
..-- ----	0--- ----	roi_context_out_en	Context output enable Enables context (header and footer) on output data **0 → No context available on output** 1 → Enables context on output This parameter influences *extra_line_nb* (@ h04)
..-- ----	-0-- ----	roi_histo_en	Histogram calculation enable Enables histogram calculation on data stream **0 → No histogram calculation requested** 1 → Enables histogram calculation This parameter influences *extra_line_nb* (@ h04)
..-- ----	--0- ----	roi_ddc_en	Defect correction enable Enables defect correction on data stream **0 → No defect correction requested** 1 → Enables defect correction

..-- ----	---0 ----	range_en	Range compression enable Enables range compression on data stream **0 → No range compression requested** 1 → Enables range compression ☞ See *range_coef* (@ h06) to control range compression
..-- ----	---- 0---	roi4_binning_en	ROI4 binning **0 → No binning requested on ROI4** 1 → Enables binning on ROI4
..-- ----	---- -0--	roi3_binning_en	ROI3 binning **0 → No binning requested on ROI3** 1→ Enables binning on ROI3
..-- ----	---- --0-	roi2_binning_en	RIO2 binning **0 → No binning requested on ROI2** 1 → Enables binning on ROI2
..-- ----	---- ---0	roi1_binning_en	ROI1 binning **0 → No binning requested on ROI1** 1 → Enables binning on ROI1 These parameters influence *extra_line_nb* (@ h04)

20.3.8 Control configuration

Name	reg_ctrl_cfg
Address	h0B
Type	Mixed: Static (s) and Restricted Static (rs)
Default	h0005

Default Value		Bitfield name	Description
..0- ----	---- ----	rs dum_stdby_en	Reserved, must be kept at 0
..-0 ----	---- ----	rs dum_pwrup_en	Reserved, must be kept at 0
..-- 0---	---- ----	rs dum_img_out_en	Reserved, must be kept at 0
..-- -0--	---- ----	s lock_dyn_reg	Lock dynamic registers **0 → Dynamic registers are not locked. Changes to dynamic registers are applied at the end of current frame** 1 → Dynamic registers are locked. . All changes are memorized but are not taken into account. They are applied only when *lock_dyn_reg* is set to 0, at the end of the current frame. ☞ Depending on *overlap_en* (on same register), there might be a delay of 1 frame to apply changes to dynamic registers.
..-- --0-	---- ----	s trig_pad_inv	TRIG pin polarity **0 → *TRIG* pin is active high** 1 → *TRIG* pin is active low
..-- ---0	---- ----	s trig_pad_sel	TRIG pin enable **0 → *TRIG* pin is disabled**. 1 → *TRIG* pin is enabled
..-- ----	00-- ----	rs roi_flash_mode[7:6]	ROI Flash mode selection: **00 → Flash OFF FLO = 0** 01 → Flash ON: FLO = 1 during integration time 10 → Flash ON: FLO = 1 during integration time + readout 11 → Flash ON: FLO = 1 at any time when running
..-- ----	--00 ----	rs roi_readout_mode[5:4]	ROI readout mode selection **00 → 5T Global Shutter** 01 → 4T + Global Reset 10 → 4T + ERS 11 → Multi-integration (Global Shutter)

Default Value		Bitfield name	Description
..-- ----	---- 0---	rs roi_video_en	Video mode enable : **0 → video mode disabled** 1 → Acquisitions are done in video mode, with a constant frame period. See *t_frame_period* (@ h0C).
..-- ----	---- -1--	rs roi_overlap_en	Overlap mode enable 0 → No overlap mode enabled **1 → Acquisitions are done in overlap mode, not used if readout_mode = 4T+GR**
..-- ----	---- --0-	s trig_rqst	SPI trigger enable **0 → SPI trigger inactive.** 1 → SPI trigger call for an acquisition
..-- ----	---- ---1	s stdby_rqst	STANDBY request : 0 → The chip is exiting STANDBY state **1 → The device will re-enter STANDBY state after the end of the frame that has started to integrate**. ☞ This means that, if *overlap_en* = 1, then STANDBY state is entered only after the end of next frame.

Caution: This register is not a simple static (s) register; it contains some restricted static (rs) bitfields. Take care not to change any 'rs' bitfields (ex: *overlap_en*), while changing an 's' bitfield (ex: *trig_rqst*), when the device is not in IDLE or STANDBY state.

20.3.9 Frame period

Name	reg_t_frame_period
Address	h0C
Type	Dynamic
Default	h0000

Default Value		Bitfield name	Description
0000 0000	0000 0000	t_frame_period[15:0]	Frame period length Defines the frame period in number of lines This frame period is used in video mode if video mode is enabled. See *roi_video_en* (@ h0B) **Min = h0000 → not take into account (see below)** Max = hFFFE → Frame period of 65534 lines = 1s if CLK_CTRL @57 MHz and *line_length* = h70 ☞ If the requested frame period is lower than the automatic internal frame_period, then *error_corrupted_video* (@ h3E) is set. The current *frame_period* setting can be read in the context line (header)

20.3.10 Wait time

Name	reg_t_wait
Address	h0D
Type	Dynamic
Default	h0000

Default Value		Bitfield name	Description
0000 0000	0000 0000	roi_t_wait[15:0]	ROI wait time Defined the wait time after the end of each read image, programmed in numbers of lines. **MIN = h0000 → wait time = 0 line** MAX = hFFF0 → d65520 lines ≈ 1s if CLK_CTRL @57 MHz and *line_length* = h6E ☞ At the end of each ROI<i> cycle, this wait time is added with a specific roi<i>_t_wait_ext (@h10 /@h1D /…) ☞ Check *error_time_overflow* (@ h3E) to see if *roi_t_wait* is too long. See frame period calculation in 22.2.2 for details. .

20.3.11 ROI 1 control

Those registers define all ROI1 parameters

Group Name	reg_roi1*
Group Addresses	h0E to h1A
Type	Dynamic

Address	Default Value		Bitfield name	Description
h0E	h0200		roi1_t_int_ll[15:0]	Integer part of ROI1 integration time Defines the integer part of the integration time in number of lines. Min = h0 → Integer part of integration time is null. **Default = h200 → d512 lines ≈ 7.90 ms with CLK_CTRL @57 MHz and *line_length* = h6E** Max = hFFFE → d65534 lines ≈ 1.01s ☞ This integration time is added to the fractional part *roi1_t_int_clk* (@ h0F) ☞ Check *error_time_overflow* (@ h3E) to see if this parameter is too big. See frame period calculation in 22.2.2 for details.
h0F	h00	--	roi1_rep_nb[7:0]	Number of ROI1 cycle repetitions Defines the number of ROI1 cycles that are read out = *roi1_rep_nb* +1 **Min h00 → 1 ROI1 is read out**. Max hFF → 256 ROI1 are read out.
	--	h00	roi1_t_int_clk[7:0]	Fractional part of ROI1 integration time Defines the fractional part of the integration time in *CLK_CTRL* cycles x *t_int_clk_mult_factor* (@ h0A) **Min= h00 → fractional part of integration time is null** Max= it is recommended to take *line_length* / *t_int_clk_mult_factor* as a maximum. ☞ If *overlap_en* (@ h0B) = 1, then take care to check both *error_t_tint_big* and error*_t_tint_small* (@ h3E).

<table>
<tr><td>h10</td><td colspan="2">h0000</td><td>roi1_t_wait_ext[10:0]</td><td>ROI1 extended wait time
Defines an additional wait time after the end of the ROI1 cycle (last repetition of ROI1), to be added to t_wait, in number of lines
Min= h000 → 0 line added
Max= h7FF → d2047 lines added on twait ≈ 31.60 ms if CLK_CTRL @57 MHz and line_length = h6E</td></tr>
<tr><td rowspan="3">h11</td><td>h00</td><td>--</td><td>roi1_ana_gain[2:0]</td><td>Analog gain applied on ROI1<table><tr><td>h0</td><td>h1</td><td>h2</td><td>h3</td><td>h4</td><td>h5</td><td>h6</td><td>h7</td></tr><tr><td>1</td><td>1.5</td><td>2</td><td>3</td><td>4</td><td>6</td><td>8</td><td>8</td></tr></table></td></tr>
<tr><td>--</td><td>h00</td><td>roi1_dig_gain[7:0]</td><td>Global digital gain applied on ROI1
Min= h00 → x1
Max= hFF → 15.875</td></tr>
<tr><td>h12</td><td colspan="2">h0006</td><td>roi1_0l_1[10:0]</td><td>1st line of 1st SIMR horizontal band
Min = 0
Default = h06
Max: [roi1_0l_1 + roi1_h_1] < d1212</td></tr>
<tr><td rowspan="2">h13</td><td colspan="2">h0</td><td>roi1_subs_v_mult16 [11]</td><td>Subsapmling of ROI1:
0 → subsampling factor multiplied by 1
Subs. Factor = 8 / (roi1_subs_v + 8)
1 → subsampling factor multiplied by 16
Subs. Factor = 8 / (16 x roi1_subs_v + 8)</td></tr>
<tr><td colspan="2">h04B0</td><td>roi1_h_1[10:0]</td><td>Height of 1st SIMR horizontal band
Min = 1
Default = h4B0 → d1200 pixels
Max: [roi1_0l_1 + roi1_h_1] < d1212</td></tr>
<tr><td>h14</td><td colspan="2">h0006</td><td>roi1_0c_1[10:0]</td><td>1st column of 1st SIMR vertical band
Min = 0
Default = h006
Max : [roi1_0c_1 + roi1_w_1] < d1612</td></tr>
<tr><td>h15</td><td colspan="2">h0640</td><td>roi1_w_1[10:0]</td><td>Width of 1st SIMR vertical band
Min = 1
Default h0640 → d1600 pixels
Max : [roi1_0c_1 + roi1_w_1] < d1612</td></tr>
<tr><td>h16</td><td colspan="2">h0000</td><td>roi1_0l_2[10:0]</td><td>1st line of 2nd SIMR horizontal band
Min: [roi1_0l_1 + roi1_h_1] < roi1_0l_2
Default = h00
Max: [roi1_0l_2 + roi1_h_2] < d1212
☞ used only if roi1_h_2 (@ h17) > 0</td></tr>
<tr><td>h17</td><td colspan="2">h0000</td><td>roi1_h_2[10:0]</td><td>Height of 2nd SIMR horizontal band
Min = 0 → no second horizontal band
Max : [roi1_0l_2 + roi1_h_2] < d1212</td></tr>
<tr><td>h18</td><td colspan="2">h0000</td><td>roi1_0c_2[10:0]</td><td>1st column of 2nd SIMR vertical band
Min : [roi1_0c_1 + roi1_w_1] < roi1_0c_2
Default = h00
Max: [roi1_0c_2 + roi1_w_2] < d1612
☞ used only if roi1_w_2 (@ h19) > 0</td></tr>
<tr><td>h19</td><td colspan="2">h0000</td><td>roi1_w_2[10:0]</td><td>Width of 2nd SIMR vertical band
Min 0 → no second vertical band
Max: [roi1_0c_2 + roi1_w_2] < d1612</td></tr>
<tr><td rowspan="2">h1A</td><td>h00</td><td>--</td><td>roi1_subs_v[7:0]</td><td>Vertical sub-sampling on ROI1
= 8/(roi1_subs_v + 8)
Min h00 → sub-sampling factor 1/1
Max hFF → sub-sampling factor 1/32.875</td></tr>
<tr><td>--</td><td>h00</td><td>roi1_subs_h[7:0]</td><td>Horizontal sub-sampling on ROI1
= 8/(roi1_subs_h + 8)
Min h00 → sub-sampling factor 1/1
Max hFF → sub-sampling factor 1/32.875</td></tr>
</table>

20.3.12 ROI 2 control

Those registers define all ROI2 cycle parameters

Group Name	reg_roi2*
Group Addresses	h1B to h23
Type	Dynamic

Address	Default Value		Bitfield name	Description
h1B	h0200		roi2_t_int_ll[15:0]	Integer part of ROI2 integration time Defines the integer part of the integration time in number of lines. Min = h0 → Integer part of integration time is null. **Default = h200 → d512 lines ≈ 7.90 ms with CLK_CTRL @57 MHz and *line_length* = h6E** Max = hFFFE → d65534 lines ≈ 1.01s ☞ This integration time is added to the fractional part *roi2_t_int_clk* (@ h1C) ☞ Check *error_time_overflow* (@ h3E) to see if this parameter is too big. See frame period calculation in 22.2.2 for details
h1C	h00	--	roi2_rep_nb[7:0]	Number of ROI2 cycle repetitions Defines the number of ROI2 cycles that are read out = *roi2_rep_nb* +1 **Min h00 → 1 ROI2 is read out.** Max hFF → 256 ROI2 are read out.
	--	h00	roi2_t_int_clk[7:0]	Fractional part of ROI2 integration time Defines the fractional part of the integration time in *CLK_CTRL* cycles x *t_int_clk_mult_factor* (@ h0A) **Min= h00 → fractional part of integration time is null** Max= it is recommended to take *line_length* / *t_int_clk_mult_factor* as a maximum. ☞ If *overlap_en* (@ h0B) = 1, then take care to check both *error_t_tint_big* and *error_t_tint_small* (@ h3E).
h1D	h0000		roi2_t_wait_ext[10:0]	ROI2 extended wait time Defines an additional wait time after the end of the ROI2 cycle (last repetition of ROI2), to be added to *t_wait*, in number of lines **Min= h000 → 0 line added** Max= h7FF → d2047 lines added on twait ≈ 31.60 ms if CLK_CTRL @57 MHz and *line_length* = h6E
h1E	h00	--	roi2_ana_gain[2:0]	Analog gain applied on ROI2 **h0** | h1 | h2 | h3 | h4 | h5 | h6 | h7 **1** | 1.5 | 2 | 3 | 4 | 6 | 8 | 8
	--	h00	roi2_dig_gain[7:0]	Global digital gain applied on ROI2 **Min= h00 → x1** Max= hFF → 15.875
h1F	h0006		roi2_0l_1[10:0]	1st line of ROI2 Min = 0 **Default = h06** Max: [roi2_0l_1 + roi2_h_1] < d1212

h20	h0		roi2_subs_v_mult16 [11]	Subsapmling of ROI2: **0 → subsampling factor multiplied by 1** Subs. Factor = 8 / (roi1_subs_v + 8) 1 → subsampling factor multiplied by 16 Subs. Factor = 8 / (16 x roi1_subs_v + 8)
	h04B0		roi2_h_1[10:0]	Height of 2nd SIMR horizontal band Min = 1 **Default = h4B0 → d1200 pixels** Max: [roi2_0l_1 + roi2_h_1] < d1212
h21	h0006		roi2_0c_1[10:0]	1st column of ROI2 Min = 0 **Default = h006** Max : [roi2_0c_1 + roi2_w_1] < d1612
h22	h0640		roi2_w_1[10:0]	Width of ROI2 Min = 1 **Default h0640 → d1600 pixels** Max : [roi2_0c_1 + roi2_w_1] < d1612
h23	h00	--	roi2_subs_v[7:0]	Vertical sub-sampling on ROI2 = 8/(roi2_subs_v + 8) **Min h00 → sub-sampling factor 1/1** Max hFF → sub-sampling factor 1/32.875
	--	h00	roi2_subs_h[7:0]	Horizontal sub-sampling on ROI2 = 8/(roi2_subs_h + 8) **Min h00 → sub-sampling factor 1/1** Max hFF → sub-sampling factor 1/32.875

20.3.13 ROI 3 control

Those registers define all ROI3 cycle parameters

Group Name	reg_roi3*
Group Addresses	h24 to h2C
Type	Dynamic

<table>
<tr><th>Address</th><th colspan="2">Default Value</th><th>Bitfield name</th><th>Description</th></tr>
<tr><td>h24</td><td colspan="2">h0200</td><td>roi3_t_int_ll[15:0]</td><td>Integer part of ROI3 integration time
Defines the integer part of the integration time in number of lines.
Min = h0 → Integer part of integration time is null.
Default = h200 → d512 lines ≈ 7.90 ms with CLK_CTRL @57MHz and line_length = h6E
Max = hFFFE → d65534 lines ≈ 1.01s
☞ This integration time is added to the fractional part roi3_t_int_clk (@ h25)
☞ Check error_time_overflow (@ h3E) to see if this parameter is too big. See frame period calculation in 22.2.2 for details</td></tr>
<tr><td rowspan="2">h25</td><td>h00</td><td>--</td><td>roi3_rep_nb[7:0]</td><td>Number of ROI3 cycle repetitions
Defines the number of ROI3 cycles that are read out = roi3_rep_nb +1
Min h00 → 1 ROI3 is read out.
Max hFF → 256 ROI3 are read out.
☞ Check roi_max_id (@ h0A) to see if this ROI cycle is run</td></tr>
<tr><td>--</td><td>h00</td><td>roi3_t_int_clk[7:0]</td><td>Fractional part of ROI3 integration time
Defines the fractional part of the integration time in CLK_CTRL cycles x t_int_clk_mult_factor (@ h0A) for ROI3
Min= h00 → fractional part of integration time is null
Max= it is recommended to take line_length / t_int_clk_mult_factor as a maximum.
☞ If overlap_en (@ h0B) = 1, then take care to check both error_t_tint_big and error_t_tint_small (@ h3E).</td></tr>
<tr><td>h26</td><td colspan="2">h0000</td><td>roi3_t_wait_ext[10:0]</td><td>ROI3 extended wait time
Defines an additional wait time after the end of ROI3 cycle (last repetition of ROI3), to be added to t_wait, in number of lines
Min= h000 → 0 line added
Max= h7FF → 2047 lines added on twait = 31.60 ms if CLK_CTRL @57 MHz and line_length = h6E</td></tr>
<tr><td rowspan="3">h27</td><td rowspan="2">h00</td><td rowspan="2">--</td><td rowspan="2">roi3_ana_gain[2:0]</td><td>Analog gain applied on ROI3
<table>
<tr><td>h0</td><td>h1</td><td>h2</td><td>h3</td><td>h4</td><td>h5</td><td>h6</td><td>h7</td></tr>
<tr><td>1</td><td>1.5</td><td>2</td><td>3</td><td>4</td><td>6</td><td>8</td><td>8</td></tr>
</table>
</td></tr>
<tr><td>--</td><td>h00</td><td>roi3_dig_gain[7:0]</td><td>Global digital gain applied on ROI3
Min= h00 → x1
Max= hFF → 15.875</td></tr>
<tr><td>h28</td><td colspan="2">h0006</td><td>roi3_0l_1[10:0]</td><td>1st line of ROI3
Min = 0
Default = h06
Max: [roi3_0l_1 + roi3_h_1] < d1212</td></tr>
</table>

h29	h0		roi3_subs_v_mult16 [11]	Subsapmling of ROI3: **0 → subsampling factor multiplied by 1** Subs. Factor = 8 / (roi1_subs_v + 8) 1 → subsampling factor multiplied by 16 Subs. Factor = 8 / (16 x roi1_subs_v + 8)
	h04B0		roi3_h_1[10:0]	Height of 3rd SIMR horizontal band Min = 1 **Default = h4B0 → d1200 pixels** Max: [roi3_0l_1 + roi3_h_1] < d1212
h2A	h0006		roi3_0c_1[10:0]	1st column of ROI3 Min = 0 **Default = h006** Max : [roi3_0c_1 + roi3_w_1] < d1612
h2B	h0640		roi3_w_1[10:0]	Width of ROI3 Min = 1 **Default h0640 → d1600 pixels** Max : [roi3_0c_1 + roi3_w_1] < d1612
h2C	h00	--	roi3_subs_v[7:0]	Vertical sub-sampling on ROI3 = 8/(roi3_subs_v + 8) **Min = h00 → sub-sampling factor 1/1** Max hFF → sub-sampling factor 1/32.875
	--	h00	roi3_subs_h[7:0]	Horizontal sub-sampling on ROI3 = 8/(roi3_subs_h + 8) **Min = h00 → sub-sampling factor 1/1** Max hFF → sub-sampling factor 1/32.875

20.3.14 ROI 4 control

Those registers define all ROI4 cycle parameters

Group Name	reg_roi4*
Group Addresses	h2D to h35
Type	Dynamic

<table>
<tr><th>Address</th><th colspan="2">Default Value</th><th>Bitfield name</th><th>Description</th></tr>
<tr><td>h2D</td><td colspan="2">h0200</td><td>roi4_t_int_ll[15:0]</td><td>Integer part of ROI4 integration time
Defines the integer part of the integration time in number of lines.
Min = h0 → Integer part of integration time is null.
Default = h200 → d512 lines ≈ 7.90 ms with CLK_CTRL @57 MHz and line_length = h6E
Max = hFFFE → d65534 lines ≈ 1.01s
☞ This integration time is added to the fractional part roi4_t_int_clk (@ h2E)
☞ Check error_time_overflow (@ h3E) to see if this parameter is too big. See frame period calculation in 22.2.2 for details.</td></tr>
<tr><td rowspan="2">H2E</td><td>h00</td><td>--</td><td>roi4_rep_nb[7:0]</td><td>Number of ROI4 cycle repetitions
Defines the number of ROI4 cycles that are read out = roi4_rep_nb +1
Min = h00 → 1 ROI4 is read out.
Max = hFF → 256 ROI4 are read out.</td></tr>
<tr><td>--</td><td>h00</td><td>roi4_t_int_clk[7:0]</td><td>Fractional part of ROI4 integration time
Defines the fractional part of the integration time in CLK_CTRL cycles x t_int_clk_mult_factor (@ h0A) for ROI4
Min = h00 → fractional part of integration time is null
Max= it is recommended to take line_length / t_int_clk_mult_factor as a maximum.
☞ If overlap_en (@ h0B) = 1, then take care to check both error_t_tint_big and error_t_tint_small (@ h3E).</td></tr>
<tr><td>h2F</td><td colspan="2">h000</td><td>roi4_t_wait_ext[10:0]</td><td>ROI4 extended wait time
Defines an additional wait time after the end of the ROI4 cycle (last repetition of ROI4), to be added to t_wait, in number of lines
Min = h000 → 0 line added
Max = h7FF → 2047 lines added on twait ≈ 31.6 ms if CLK_CTRL @57 MHz and line_length = h6E</td></tr>
<tr><td rowspan="3">h30</td><td rowspan="2">h00</td><td rowspan="2">--</td><td rowspan="2">roi4_ana_gain[10:8]</td><td>Analog gain applied on ROI4
<table>
<tr><td>h0</td><td>h1</td><td>h2</td><td>h3</td><td>h4</td><td>h5</td><td>h6</td><td>h7</td></tr>
<tr><td>1</td><td>1.5</td><td>2</td><td>3</td><td>4</td><td>6</td><td>8</td><td>8</td></tr>
</table></td></tr>
<tr><td>--</td><td>h00</td><td>roi4_dig_gain[7:0]</td><td>Global digital gain applied on ROI4
Min = h00 → x1
Max = hFF → 15.875</td></tr>
<tr><td>h31</td><td colspan="2">h0006</td><td>roi4_0l_1[10:0]</td><td>1st line of ROI4
Min = 0
Default = h06
Max: [roi4_0l_1 + roi4_h_1] < d1212</td></tr>
</table>

h32	h0		roi4_subs_v_mult16 [11]	Subsapmling of ROI4: **0 → subsampling factor multiplied by 1** Subs. Factor = 8 / (roi1_subs_v + 8) 1 → subsampling factor multiplied by 16 Subs. Factor = 8 / (16 x roi1_subs_v + 8)
	h04B0		roi4_h_1[10:0]	Height of 4th SIMR horizontal band Min = 1 **Default = h4B0 → d1200 pixels** Max: [roi4_0l_1 + roi4_h_1] < d1212
h33	h0006		roi4_0c_1[10:0]	1st column of ROI4 Min = 0 **Default = h006** Max: [roi4_0c_1 + roi4_w_1] < d1612
h34	h0640		roi4_w_1[10:0]	Width of ROI4 Min = 1 **Default h0640 → d1600 pixels** Max: [roi4_0c_1 + roi4_w_1] < d1612
h35	h00	--	roi4_subs_v[7:0]	Vertical sub-sampling on ROI4 = 8/(roi4_subs_v + 8) **Min = h00 → sub-sampling factor 1/1** Max = hFF → sub-sampling factor 1/32.875
	--	h00	roi4_subs_h[7:0]	Horizontal sub-sampling on ROI4 = 8/(roi4_subs_h + 8) **Min = h00 → sub-sampling factor 1/1** Max = hFF → sub-sampling factor 1/32.875

20.3.15 Blue and red gains control

Name	reg_dig_gain_gb_gr
Address	h36
Type	Dynamic
Default	h8080

Default Value		Bitfield name	Description
1000 0000	---- ----	gb_dig_gain[7:0]	Green blue digital gain in color version Min = h00 → x0.25 **Default = h80 → x1** Max = hFF → x3.97 ☞ Used only if *color_en* (@ h07)= 1
---- ----	1000 0000	gr_dig_gain[7:0]	Green red digital gain in color version Min = h00 → x0.25 **Default = h80 → x1** Max = hFF → x3.97 ☞ Used only if *color_en* (@ h07)= 1

20.3.16 Green gain control

Name	reg_dig_gain_b_r
Address	h37
Type	Dynamic
Default	h8080

Default Value		Bitfield name	Description
1000 0000	---- ----	b_dig_gain[7:0]	Blue digital gain Defines the blue digital gain in color version, and general digital gain in B&W version. Min = h00 → x0.25 **Default = h80 → x1** Max = hFF → x3.97 ☞ This parameter is be used whatever the value of *color_en* (@ h07)
---- ----	1000 0000	r_dig_gain[7:0]	Red digital gain in color version Min = h00 → x0.25 **Default = h80 → 1** Max = hFF → x3.97 ☞ This parameter is used only if *color_en* (@ h07)= 1

20.3.17 Clamp & offset adjustments

Name	reg_clamp_offset
Address	h38
Type	Dynamic
Default	h0000

Default Value		Signal Name	Description
0000 0000	---- ----	clamp_add_offset[7:0]	Additional clamp offset Defines a signed additional (2's-complement) offset in LSB: Min = h80 → -128 hFF → - 1 **Default = h00 → 0** h01 → 1 Max = h7F → 127 ☞ Used only if *clamp_auto_en* (@ h07)= 1
---- ----	0000 0000	clamp_manual_offset[7:0]	Manual clamp offset Applied offset in LSB if clamp_auto_en (@ h07)= '0' **Min = h00→ 0** Max = hFF→ 255

20.3.18 Clamp configuration

Name	reg_clamp_cfg
Address	h39
Type	Restricted static
Default	h3880

Default Value		Bitfield name	Description
.011 ----	---- ----	init_line_nb[2:0]	Number of init lines Number of init lines (V0) before reading matrix Min = h0 **Default = h3 → 3 init lines** Max = h7
.--- 1000	---- ----	clamp_lock_th[3:0]	Clamp lock mechanism threshold Defines the threshold for the lock mechanism (0 to 15 LSB) : Min = h0 **Default = h8 → 8 LSB threshold** Max = hF
.--- ----	1--- ----	clamp_lock_en	Clamp lock mechanism enable : 0→ Disables lock mechanism **1→ Enables lock mechanism during automatic black level adjustment**
.--- ----	-0-- ----	dig_cor_en	Digital correction enable : **0→ Digital correction is not allowed** 1→ Allows digital correction
.--- ----	--00 0000	v0_gain[5:0]	Clamp digital V0 correction enable : **Min = h00 → Bypass V0 correction** h01 → Apply a V0 ratio 1/64 Max = h3F→ Apply a V0 ratio 63/64

20.3.19 Oscillator programming

Name	reg_prg_osc
Address	h3A
Type	Restricted static
Default	h80C0

Default Value		Bitfield name	Description
1000 000-	---- ----	prg_osc_vsat_adjust[6:0]	Adjust the ADC saturation to leave room for the clamp dark signal compensation. Min = h00 → nominal value -64% **Default = h40 → nominal value of Vsat ADC = 850mV (see Rext calculation)** Max = h7F → nominal value +64%
---- ---0	1--- ----	prg_osc_vsat_select[1:0]	**01 → reserved, must be kept at 01**
---- ----	-100 0000	prg_osc_freq_adjust[6:0]	Adjustment of the internal oscillator frequency @ 137MHz. Min = h00 **Default = h40 → default value given by Rext** Max = h7F

The oscillator is used only if selected as clock source by *clk_on_adc_domain* or *clk_on_chain_domain* (@ h08, or if calibration is requested (see *calib_count_ref* @ h3B).

20.3.20 Calibration count for oscillator calibration

Name	reg_calib_count_ref
Address	h3B
Type	Restricted Static
Default	h0000

Default Value		Bitfield name	Description
0000 0000	0000 0000	calib_count_ref[15:0]	Oscillator calibration reference count This register has two different uses: It sets the number of *CLK_REF* clock cycles to count for oscillator calibration: Min = h0001 → 1 clock cycle for calibration phase Max = hFFFF → 65535 clock cycles. 💣 This parameter must not be too big, because *fb_calib_count_osc* (@ h3C) could overflow, depending on both frequency and ratio… ☞ This is NOT the number of clock cycles of the whole calibration sequence. The wake up phase for the oscillator (see *t_wakeup_osc* @ h43) and a few extra cycles will be added. Writing into this register starts a calibration using the previously written parameters. You can check the calibration progress in *calib_mbx* (@ h02), and retrieve the result in *fb_calib_osc_count* (@ h3C)

20.3.21 Oscillator calibration feedback

Name	fb_calib_osc_count
Address	h3C
Type	Feedback
Default	h0000

Used bits		Bitfield name	Description
xxxx xxxx	xxxx xxxx	fb_calib_count_osc[15:0]	Oscillator calibration result The calibration result is given as the number of CLK_OSC clock cycles counted. No saturation mechanism on the counter.

20.3.22 Clamp feedback

Name	fb_clamp
Address	h3D
Type	Feedback

Used bits		Bitfield name	Description
xxxx xxxx	xx-- ----	fb_ana_offset[7:0]	Analog offset Offset applied on ADC column in LSB (0 to 1023) (no dynamic loss)
---- ----	--xx xxxx	fb_dig_offset[7:0]	Digital offset Digital offset applied on pixel value after ADC conversion in LSB (0 to 63)

20.3.23 Sensor status feedback

Name	fb_status
Address	h3E
Type	Feedback

Used bits		Bitfield name	Description
....X	---- ----	flag_dig_cor	Digital Correction flag 0→ Offset correction is done entirely in analog 1→ Digital correction is needed ☞ Used only if *dig_cor_en* (@h39) = 1
....-	XX-- ----	fb_state_main_global[1:0]	Main global device state 00 → Device is in STANDBY 01 → Device is in WAKEUP (going to IDLE) 10 → Device is in IDLE (waiting for new trig) 11 → Device is in ACQUISITION
....-	--X- ----	error_time_overflow	Timing configuration overflow error 0 → OK 1 → Computed frame period is greater than hFFFE
....-	---X ----	error_corrupted_video	Corrupted video error An error occurred on the applied frame_period: 0 → OK 1 → Computed frame period is greater than configured *t_frame_period* (@ h0C), in video mode.
....-	---- X---	error_ll_vs_xfer	Line length error reading pixels 0 → OK 1 → The programmed line_length (@ h04) is too small, and pixels are still being read into matrix at end of line
....-	---- -X--	error_ll_vs_conv	Line length error during conversion 0 → OK 1 → The programmed line_length (@ h04) is too small, and conversion is still running at end of line
....-	---- --X-	error_t_int_big	Integration time error on next image An integration time error of max 1.5 µs occurred. 0 → OK 1 → An error has been made on the fractional part of the NEXT image integration time (the biggest one if we are in high dynamic configuration)
....-	---- ---X	error_t_int_small	Integration time error on current image An integration time error of max 1.5 µs occurred.0 → OK 1 → An error has been made on the fractional part of the CURRENT image integration time (the smallest one if we are in high dynamic configuration)

20.3.24 DATA_CLK activity

Name	DATA_CLK Activity
Address	h44
Type	Restricted Static

Default Value		Signal Name	Description
--1- ----	---- ----	clk_out_low_pwr [13]	Defines DATA_CLK activity 0 → DATA_CLK is always active except in Stand By state **1 → DATA_CLK is active during the whole acquisition (integration + readout + wait)**

20.3.25 Pixtime_read_width

Name	pixtime_read_width
Address	h49
Type	Restricted Static
Default	h7A6F

Default Value		Signal Name	Description
0111 1010	---- ----	pixtime_read_5t_width[7:0]	Width of read part of timing pixel for 5T mode (adc starts after) defined by step of CLK_CTRLx2 **default = 122 x 2 = 244 clocks = 4.3 us @57MHz**
---- ----	0110 11111	pixtime_read_4t_width[7:0]	Width of read part of timing pixel for 4T mode (adc starts after) **default = 111 x 2 = 222 clocks = 3.9 us @57MHz**

20.3.26 Chip ID

Name	chip_id
Address	h7F
Type	Restricted Static
Default	h0900

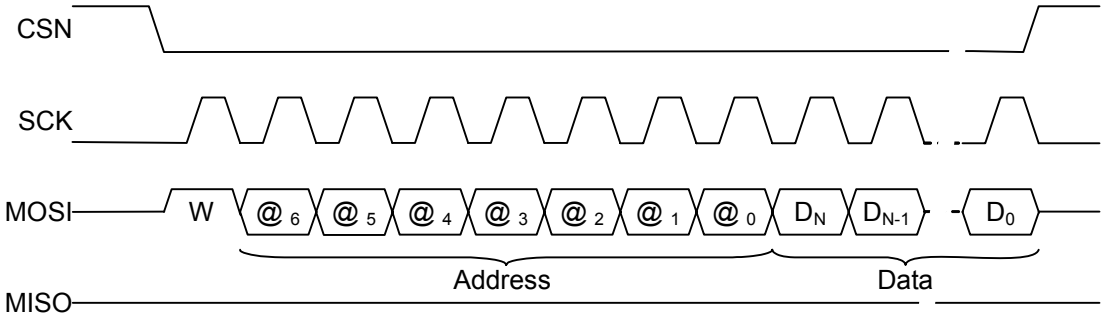
Default Value		Signal Name	Description
0000 1001	0000 ----	chip_id [15:4]	Gives the sensor silicon version

20.4 SPI timing

First, the SPI interface of the EV76C570 has to receive the first bit from the master on the on MOSI line which indicates if it is a read or write command. This first bit is followed by 7 bits giving the d1 to d127 addresses.

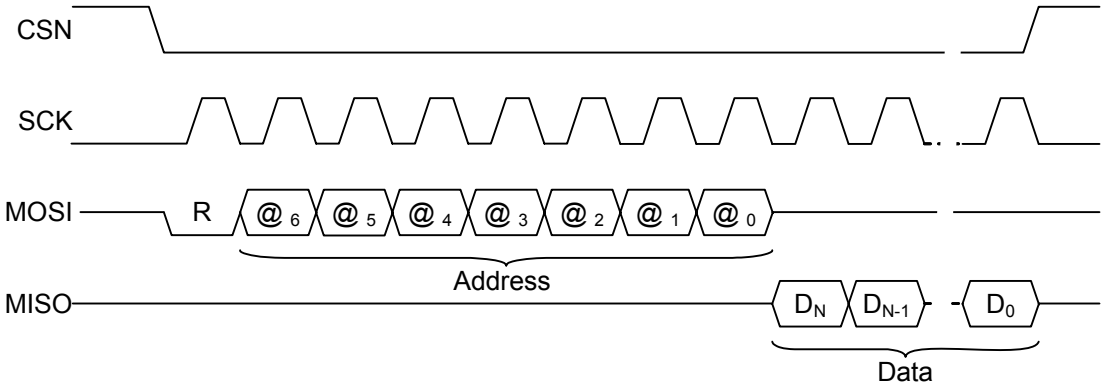
In a write sequence (see Figure 29) first bit must be at high level. After having sent the 7 address bit - MSB first - the data are sent on the MOSI line (also MSB first).

Figure 29: One register write sequence



In a read sequence (see Figure 30) first bit must be at low level. After having sent the 7 address bit MSB first, the data are read on the MISO line (also MSB first).

Figure 30: One register read sequence



The master can also use a burst sequence (see Figure 31) to read or write several adjacent registers.

The end of burst sequence occurs when the CSN Chip Select line is put back into inactive state at high level.

In burst mode the internal address is automatically incremented at the end of each data read/write phase.

For example, to read three16-bit registers starting at address h10:

Figure 31: Burst sequence

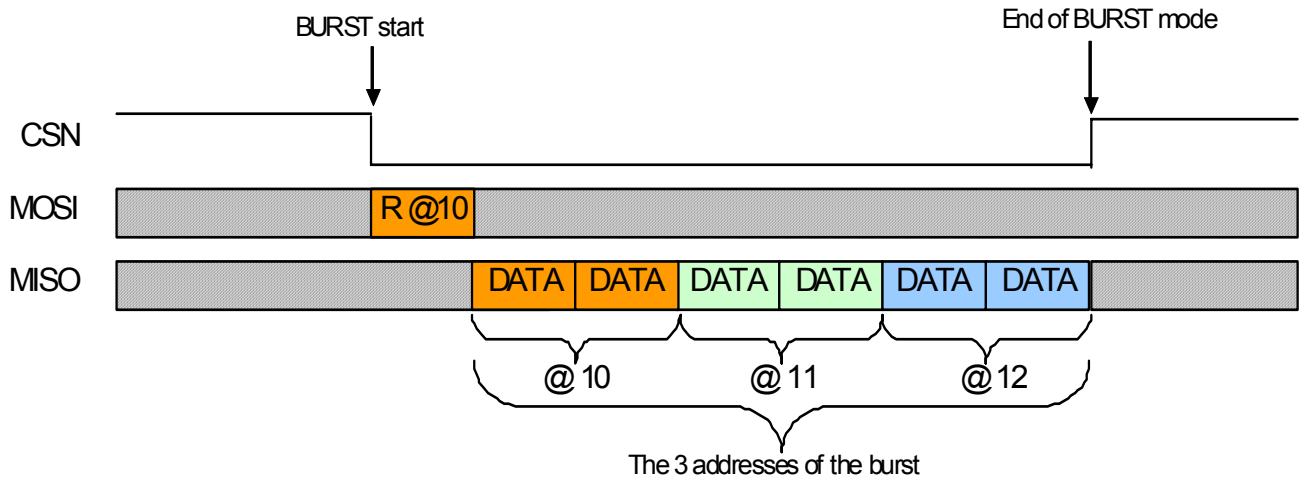
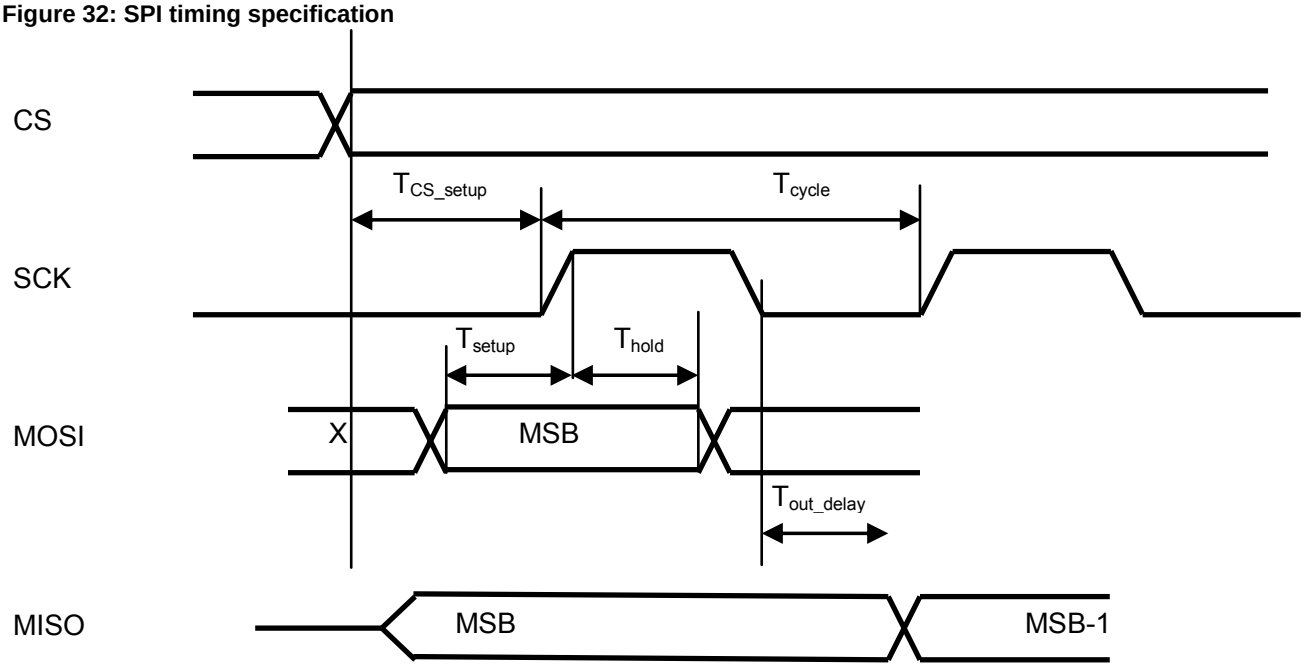


Figure 32: SPI timing specification



These timings depend on the process, current load, and post layout. The values given here should be considered only as general guidelines.

Table 15: SPI timing specification

Symbol	Typical timing
T_{cycle}	50 ns
T_{setup}	5 ns
T_{hold}	2 ns
T_{cs_setup}	5 ns
T_{out_delay}	5 ns depending on the current load

21. SENSOR STATES

21.1 Static states

At start-up, the sensor state is controlled by internal registers and by the RESETN and TRIG external signals.

The registers are in a known state after a device reset.

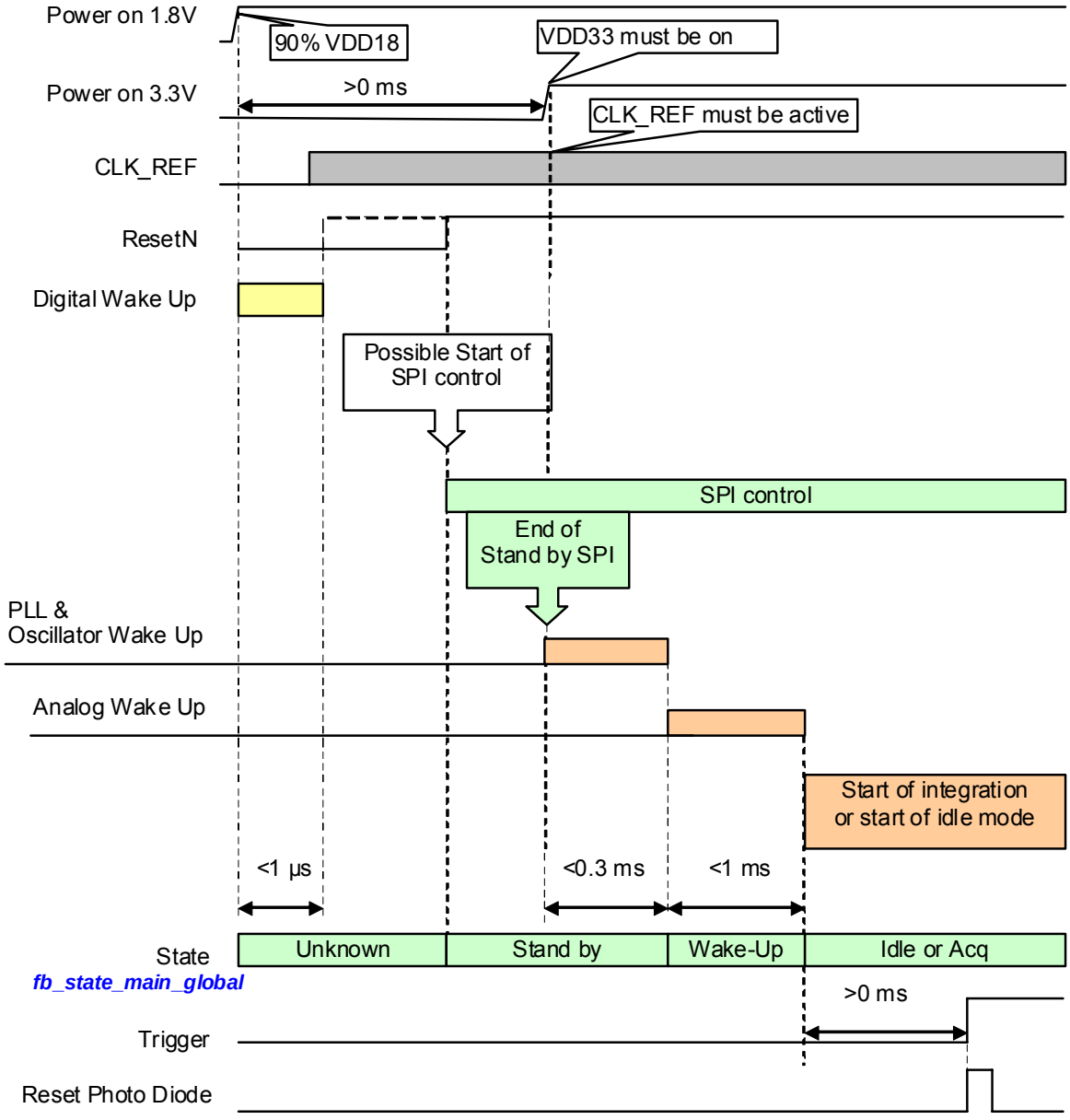
The RESETN must be pulled up and the TRIG signal can toggle.

The sensor state is indicated in the internal status registers and from the FLO external signal.

21.1.1 Power-On sequence

The following timing diagram shows the power up sequence initiated by a rising edge on 3.3 V and 1.8 V power supplies.

Figure 33: Power up sequence



21.1.2 STANDBY

This is the lowest power consumption mode.

At power-up the sensor is in STANDBY state.

During STANDBY state the SPI registers may be read or written.

Transition from this state to IDLE state or beginning of integration has duration of less than 1 ms and is under SPI control with ***stdby_rqst*** in **<reg_ctrl_cfg>** see § **20.3.8**

21.1.3 IDLE

In IDLE state, the device is "ready to start".

During IDLE state, SPI registers can be read or written.

The sensor can start integration from this state in less than 10 µs, with an SPI command ***trig_rqst*** in **<reg_ctrl_cfg>** or with a hardware trigger on the TRIG pin if enabled by ***trig_pad_sel***. in **<reg_ctrl_cfg>** see § **20.3.8**

Transition from this state to STANDBY state is under SPI control with ***stdby_rqst*** in **<reg_ctrl_cfg>** see § **20.3.8**

21.2 Active States

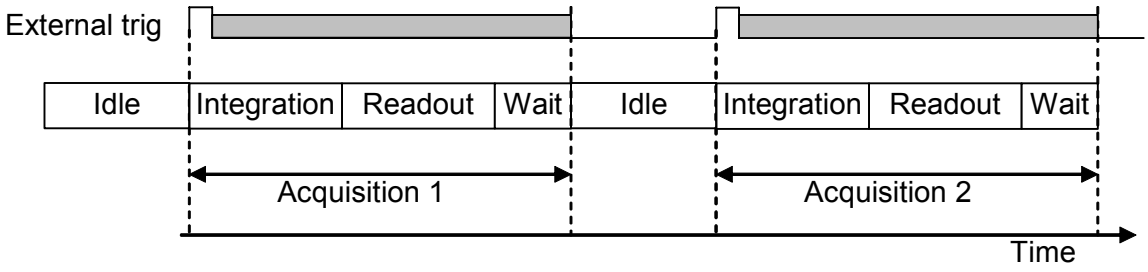
Active state defines a state of the sensor during which it runs in integration or readout or is waiting for the end of an acquisition task.

A typical acquisition sequence includes 3 states:
- Integration,
- Readout,
- Wait.

When the sensor is waiting for a trigger, it is put in IDLE state.

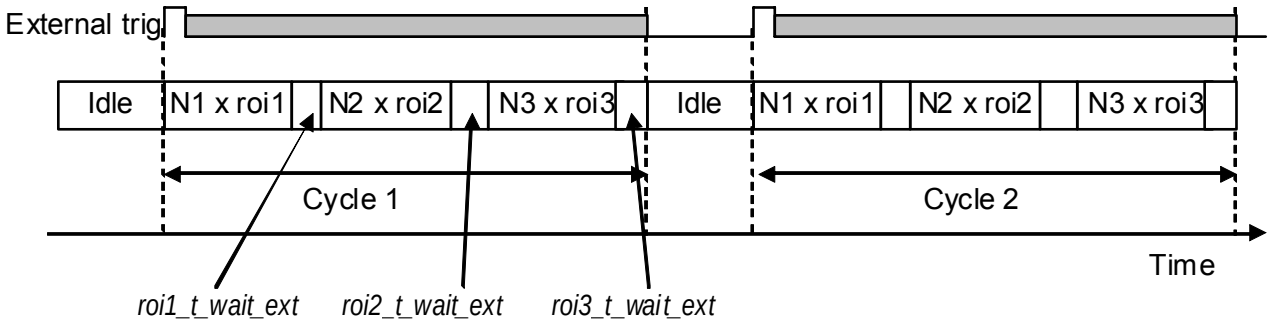
A short hardware or software trigger pulse starts the configured acquisition cycle. The example in Figure 34 is for only 1 ROI with a repetition number of 1.

Figure 34: Acquisition sequence example 1



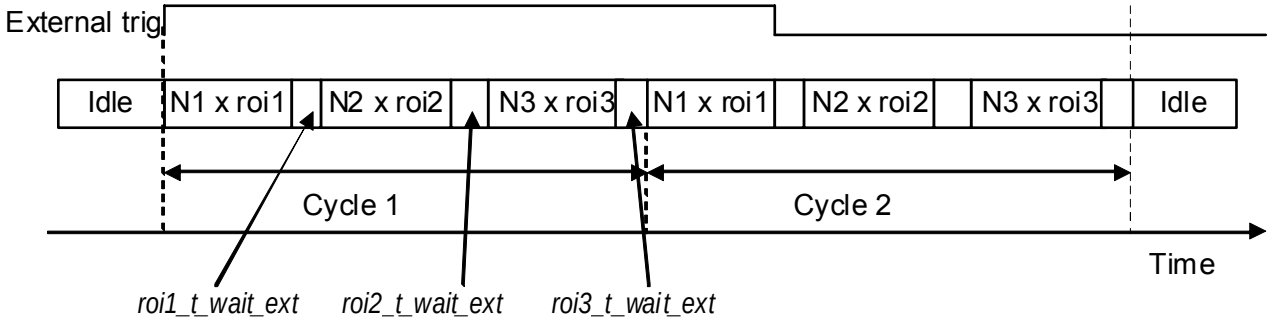
The example in Figure 35 uses the cycle principle for 3 ROIs (***roi_max_id*** = h2) (N1, N2 and N3 are configured by the ***roi1_rep_nb***, ***roi2_rep_nb*** & ***roi3_rep_nb*** registers respectively). The Wait time (T_{WAIT}) is defined in ***<reg_t_wait>*** and is added at the end of each acquisition.

Figure 35: Acquisition sequence example 2



The example in Figure 36 shows the behavior with the trig signal kept at high level.

Figure 36: Acquisition sequence example 3



Notes:

- The grey area indicates that any TRIG or trig_rqst pulse during this period is not taken into account.
- If the trig_rqst bit or the TRIG pin is deactivated during a cycle sequence, the sensor waits the end of the cycle before entering IDLE state.

The sensor provides four capture modes selectable by ***roi_readout_mode*** in <**reg_ctrl_cfg**> see § **20.3.8**

- Global Shutter : ***roi_readout_mode*** = h0
- 4T + Global Reset : ***roi_readout_mode*** = h1
- 4T + Electonic Rolling Shutter : ***roi_readout_mode*** = h2
- Multi-integration (Global Shutter) : ***roi_readout_mode*** = h3

With these 4 modes, there are different possible operating sequences.
These are described in detail in the following paragraphs, except the multi-integration mode to be addressed in a separate application note.

21.2.1 Global Shutter mode

A GS capture sequence includes the following stages:

- Global reset of all photodiodes
- Integration simultaneously in all photodiodes
- Global transfer of all photodiode signals in sensing nodes
- Readout line by line
- Wait state (option)

Figure 37: Global shutter

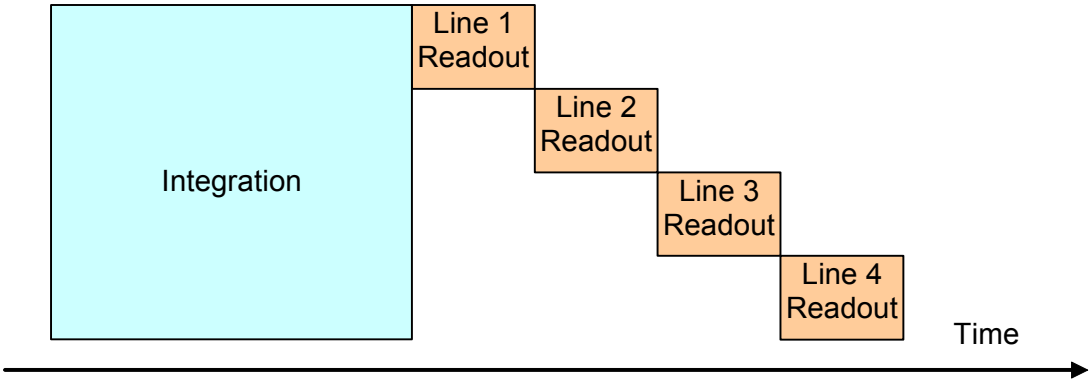
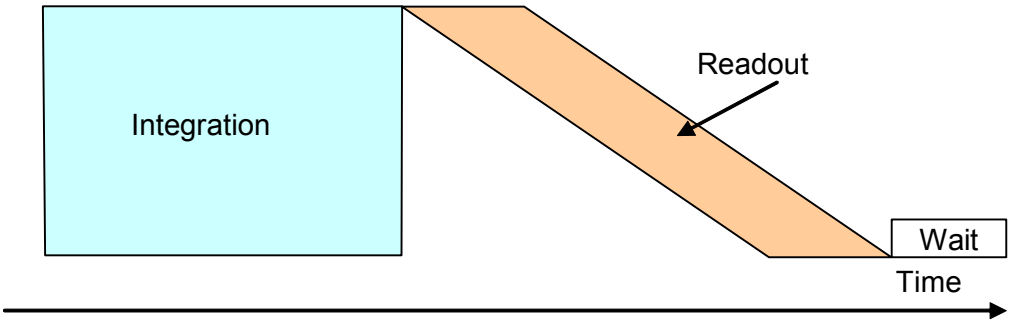


Figure 38: Global shutter symbolization



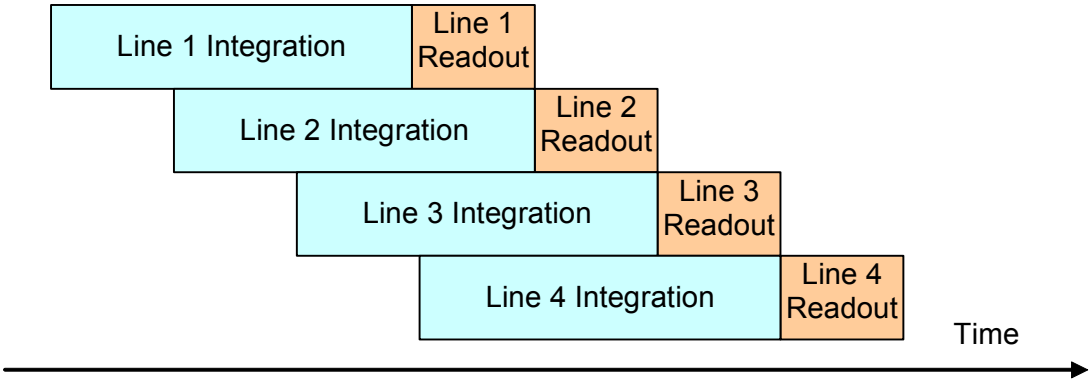
Using a 5T pixel, the global shutter mode does not allow a true CDS during pixel readout.

21.2.2 ERS mode

Electronic Rolling Shutter (ERS) mode can perform true CDS timing that removes kTC (reset) noise. It offers better performance in terms of SNR and dynamic, but it is sensitive to the relative movement between camera and scene (rolling shutter distortion).

In this mode, every line has the same integration time duration but not at the same time. Refer to Figure 39.

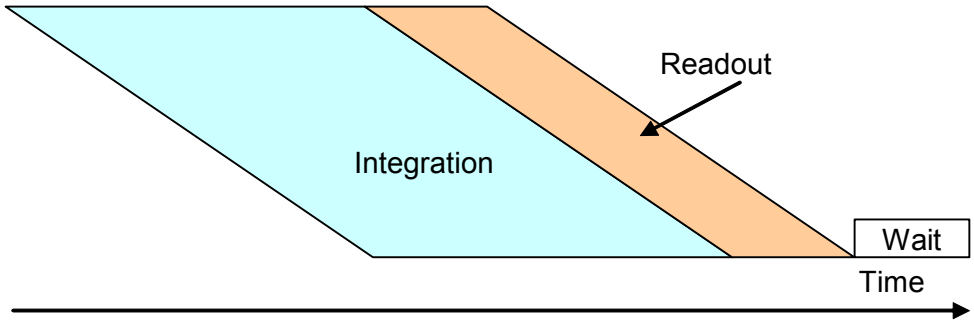
Figure 39: Rolling shutter principle



An ERS acquisition sequence includes the following states:

- Line by line integration state
- Line by line transfer and readout (this state starts at the end of the integration of the first line)
- Wait state (option)

Figure 40: Rolling shutter symbolization



21.2.3 Overlap option definition

For ERS and GS modes, an overlap option is selectable by SPI. When it is selected the integration state of an image starts as soon as possible, before the end of the previous image.
Figure 41 shows ERS mode where the integration is done line by line
Figure 42 shows GS mode where the integration is done simultaneously in all photodiodes.
Twait is adjusted under SPI control

Figure 41: Overlap / Non-overlap in ERS mode

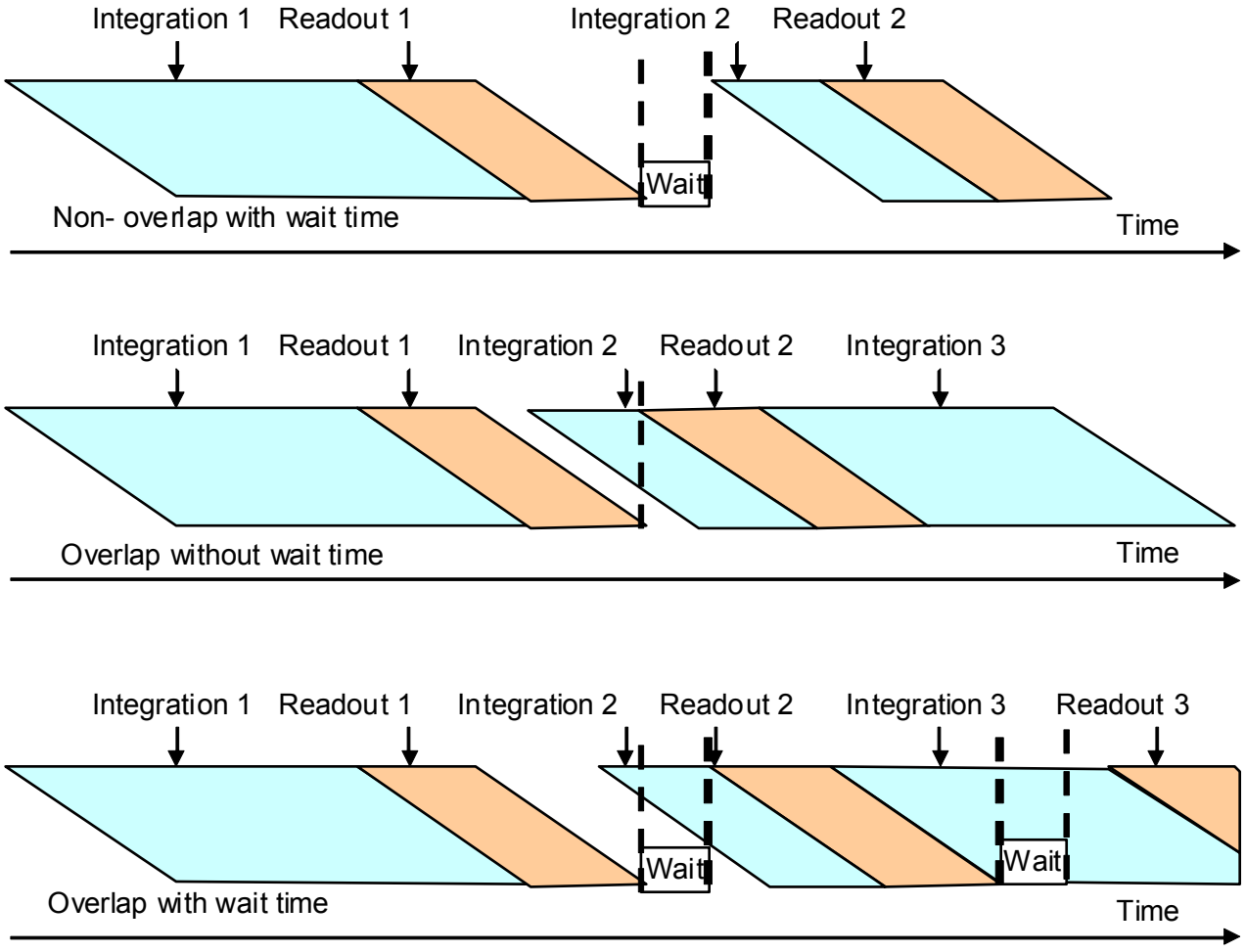
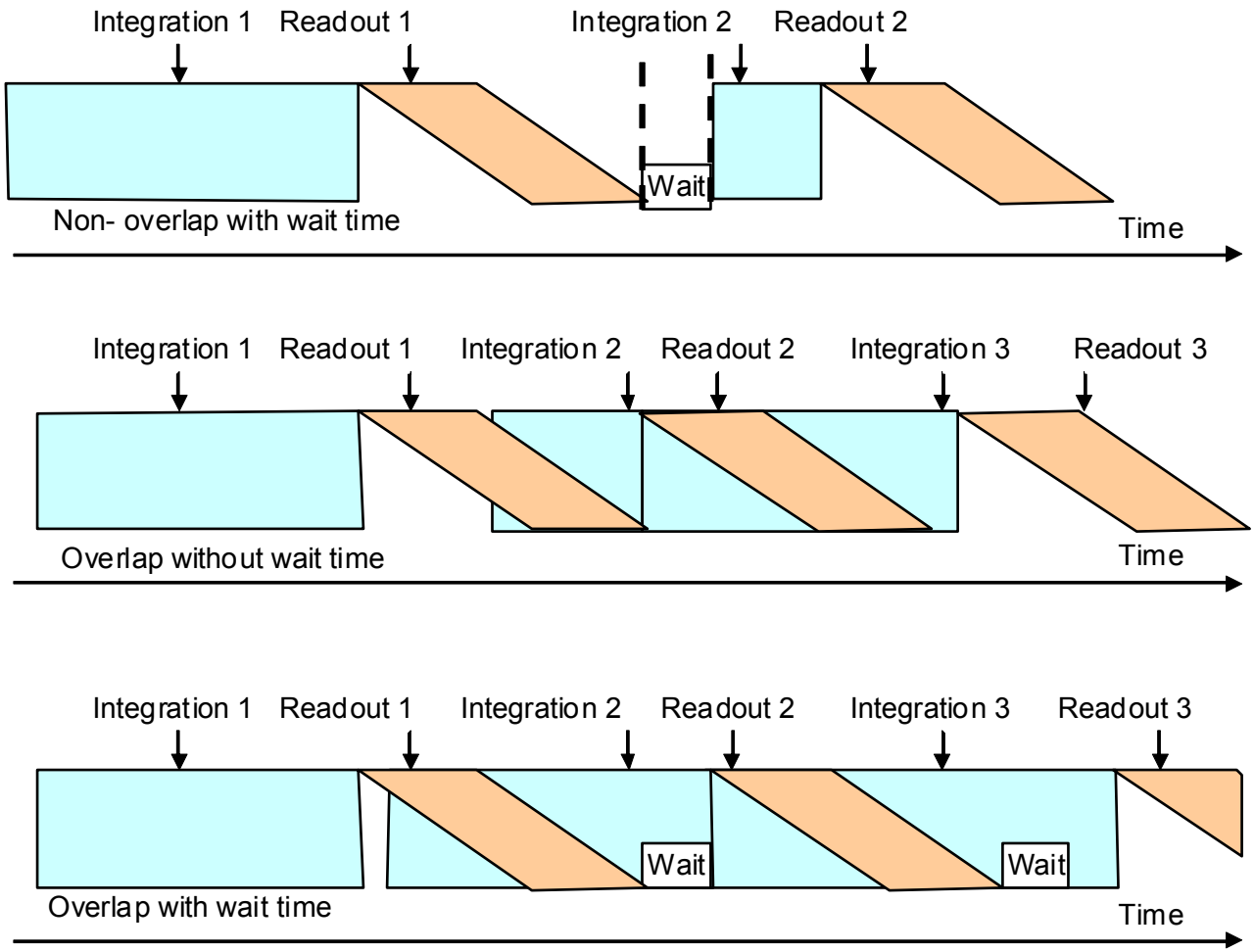


Figure 42: Overlap / Non-overlap in GS mode



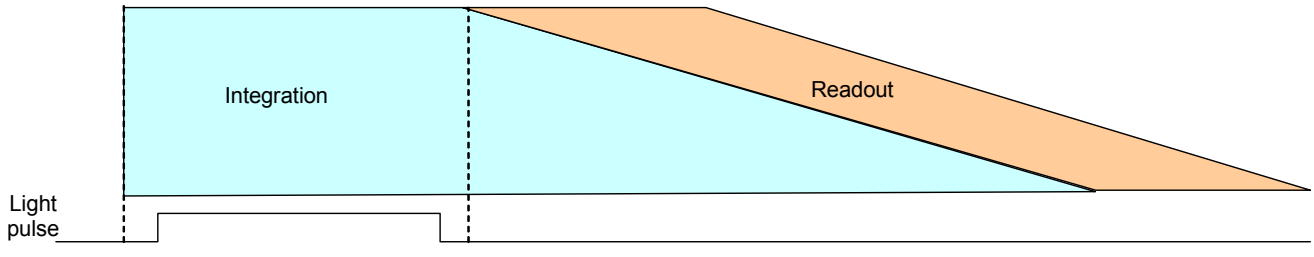
In these figures, the integration time is changed for each image to illustrate most of the different cases.

21.2.4 ERS + GR mode

ERS + GR mode is a combination of ERS and GS modes. It allows the use of true CDS during pixel readout. It can be used for example if a synchronized light pulse is provided by the application. The moving effect may be negligible if the signal without light pulse is only negligibly different from the signal with light pulse.

The overlap option is not possible in ERS+GR mode.

Figure 43: ERS with Global Reset



An ERS + GS acquisition sequence includes the following states:

- Global reset of all photodiodes and sensing nodes
- Integration stage
- Transfer, conversion and readout line by line (this state starts at the end of the integration of the first line)
- Wait state (option).

•

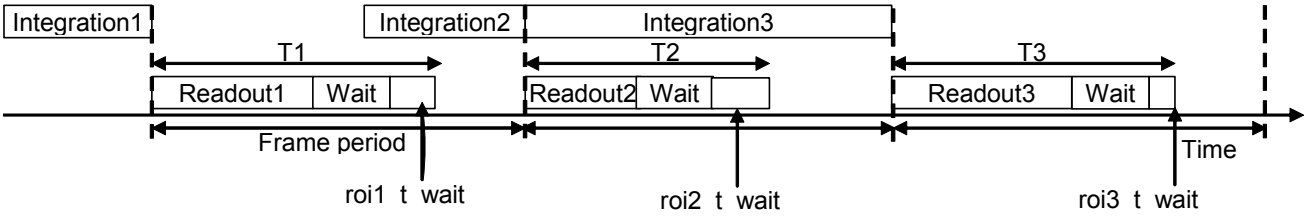
21.2.5 Video option definition

The Video Option can be defined for all capture modes. It is selected using the SPI. When it is selected the frame period is programmed by SPI and it constrains the integration time to a value less than the frame period.

The overlap option can be used with the video option.

Figure 44 gives a timing diagram showing the principle of the video option for GS readout mode with overlap option and 3 ROI's configured with only one repetition.

Figure 44: Video mode option



Notes:

- For each new frame, the device computes the minimum frame period value needed to correctly apply the integration time, the ROI readout and the wait time (in all frame and mode configurations). If the SPI frame period value is smaller than this minimum value, then the applied frame period is set to the computed value, and the ***error_corrupted_video*** bit is set to inform the user that the configured value is too small. The actually applied frame_period can be read in the header.

21.3 Interrupt functions

21.3.1 Abort function

This function allows the application to abort the current acquisition sequence. It has no effect if the sensor is in Standby or IDLE state.

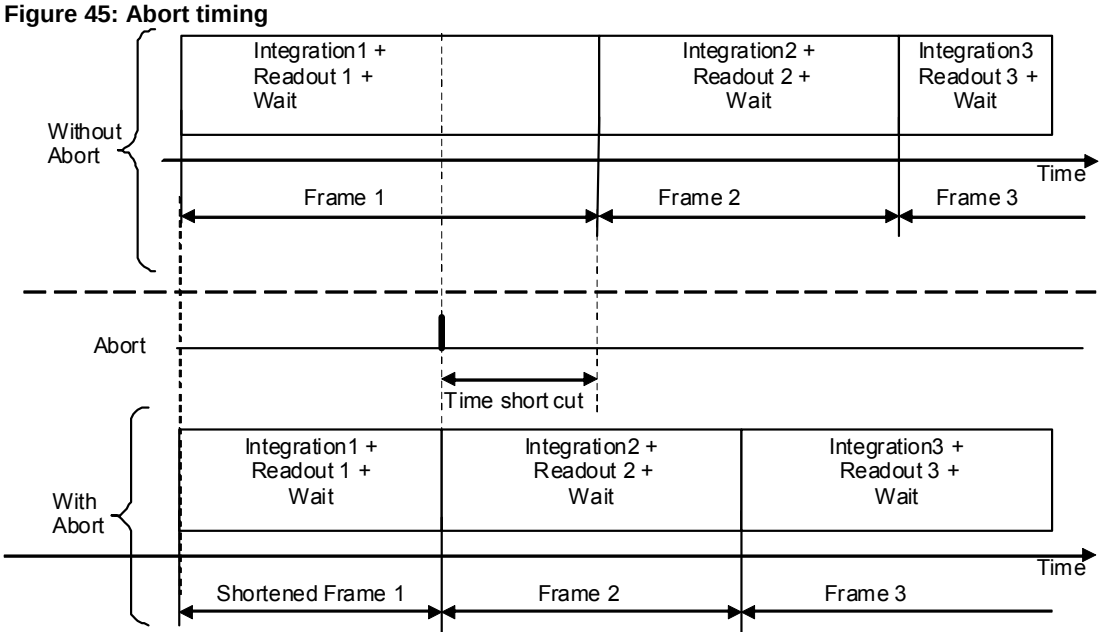
The abort is taken into account:

- Immediately when the abort occurs during the integration or wait stage.
- At the end of current line, when the abort occurs during the readout stage.

The abort sequence is cleared by writing any value in the ***flag_abort_mbx*** register.

When an abort occurs, all the register settings are preserved, so a new acquisition can be started immediately afterward. Depending on the used mode some artefacts may occur on the next image.

Figure 45: Abort timing



If an abort occurs during a MIMR sequence, then the sensor is ready to start the first integration of ROI1.

21.3.2 Reset function

The device can be reset either by:

- Writing or reading the ***soft_reset*** see § **20.2.2**
- Applying a low pulse on the RESETN pin (minimum pulse duration is 20 ns).

After a reset, the device returns immediately to the default SPI register configuration. All registers to be configured with other values must be written again.

22. SYNCHRONIZATION PULSE AND TIMINGS

The FEN and LEN synchronization signals may be inverted by programming ***sync_len_inv*** and ***sync_fen_inv*** in **<reg_miscel2>**. The DATA_CLK may be inverted by using ***clk_out_inv*** in **<reg_miscel2>.** See Section 17.3.4.

22.1 Clocks limits

Table 16: Frequency limits for 50fps

Parameter	Value			
	Min	Typ	Max	Unit
CLK_REF input for PLL	5	24	50	MHz
CLK_REF input for direct use	5		120	MHz
CLK_FIX input (if used)	5		120	MHz
Duty cycle on CLK_REF and CLK_FIX	40	50	60	%
DATA-CLK , CMOS output (to be able to reach 60 fps)	114	114	120	MHz
Duty cycle on DATA-CLK		50		%

22.2 Vertical timings

22.2.1 Timing diagram

Figure 46: Vertical timing graph

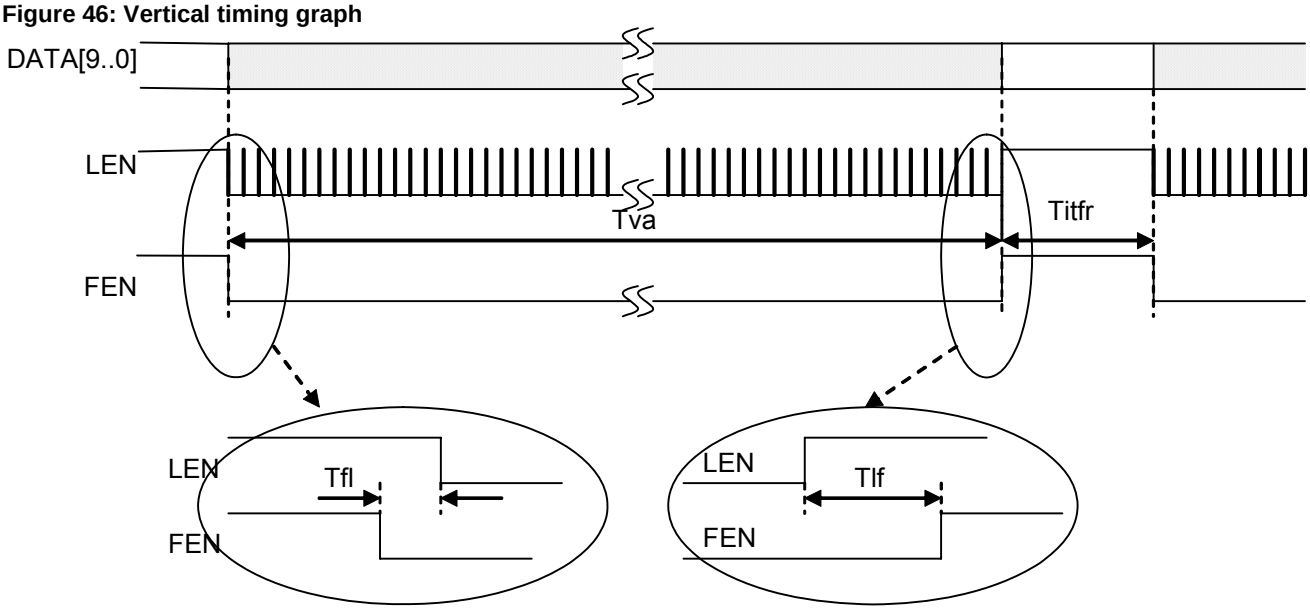


Table 17: vertical timing specification

Parameter	Symbol	Nominal	Unit
Vertical valid data (1)	Tva	1200	Line period
FEN falling to LEN falling	Tfl	2 minimum	DATA_CLK
LEN rising to FEN rising	Tlf	2 minimum	DATA_CLK
Inter-frame time(2)	Titfr	Configurable	Line period

(1): Depends on ROI and Sub sampling: Tva = (roi_height + 2 x context_en + histo_en)
(2): Titfr = t_frame_period – Tva

22.2.2 Frame period calculation

The sensor runs properly in video mode if the frame period is well programmed. The following paragraph gives the user a procedure to calculate the minimum frame period value to program (in number of lines) when ***roi_video_en***=1 (in register h0B).

If the frame period is not correct, two flags can warn the user:

- ***error_corrupt_video*** flag in the register h3E. A bad frame period will set this flag,
- ***error_corrupt_overflow*** flag in the register h3E: *reg_frame_period* exceeds 65534 (hFFFE) and this saturation value is applied.

Table 18 : Useful registers for the frame period calculation

Entries	Register Address	Comments
reg_t_frame_period	h0C	
roiN_t_int_ll	h0E, h1B, h24, h2D	Depends on the ROI used
extra_line_nb	h04	Bits (12:15)
init_line_nb	h39	Bits (12:14)
roi_expanded	h07	Bit 8
roiN_binning_en	h0A	Bit 0,1,2 or 3
t_wait	h0D	
roiN_t_wait_ext	h10, h1D, h26, h2F	Depends on the ROI used
roi_histo_en	h0A	Bit 6
roi_context_out_en	h0A	Bit 7
roi_overlap_en	h0B	Bit 2
Roi_height		Calculated in paragraph 7.5.2.1

For the Frame period computation (in number of lines), the user may follow the steps above:

1- Program the minimum number of extra-lines (register ***extra_line_nb*** @ h04)

$$extra\_line\_nb = (2 \times roiN\_binning\_en + roi\_histo\_en + roi\_context\_out\_en) \times 2^{roiN\_binning\_en}$$

2- Readout time

$$Treadout = 2 + init\_line\_nb + \left(6 \times \overline{roi\_\exp anded}\right) + roi\_height + extra\_line\_nb + 1$$

3- Minimum frame period

If ***overlap_en*** = 0, then:

$$reg\_t\_frame\_period = roiN\_t\_\text{int}\_ll + Treadout + t\_wait + roiN\_t\_wait\_ext + t\_flash\_on$$

If ***overlap_en*** = 1, then:

$$reg\_t\_frame\_period = MAX\left(roiN\_t\_\text{int}\_ll + 1 + t\_flash\_on; Treadout + t\_wait + roiN\_t\_wait\_ext\right)$$

The result of the frame period calculation can be read from the context data, depending on the video mode:

- if ***video_en***=1, and *reg_t_frame_period* ≥ t_frame_period_min:
  then Actual_frame_period = *t_frame_period,*
- if ***video_en***=0, then Actual_frame_period = t_frame_period_min

22.2.3 Typical frame rate

Table 11 gives the possible frame rates in overlap mode with a DATA_CLK of 114 MHz.
In non-overlap mode, the integration time must be added to T_{READ}.

Table 19: frame rate example

Format	Number of columns	Number of lines	Line Length(1)	T_{READ} (ms)	Frame Rate (fps)
2 MP	1600	1200	1792	18.7	53.6
1 MP	1600	600	1792	9.3	107
1.3 MP	1280	1024	1792	15.9	62.8
0.3 MP	640	480	1792	7.5	134

(1) in DATA_CLK periods.

Using sub-sampling or windowing reduces the readout time only by the reduced number of lines.

22.3 Horizontal timings

Figure 47: Horizontal timing graph

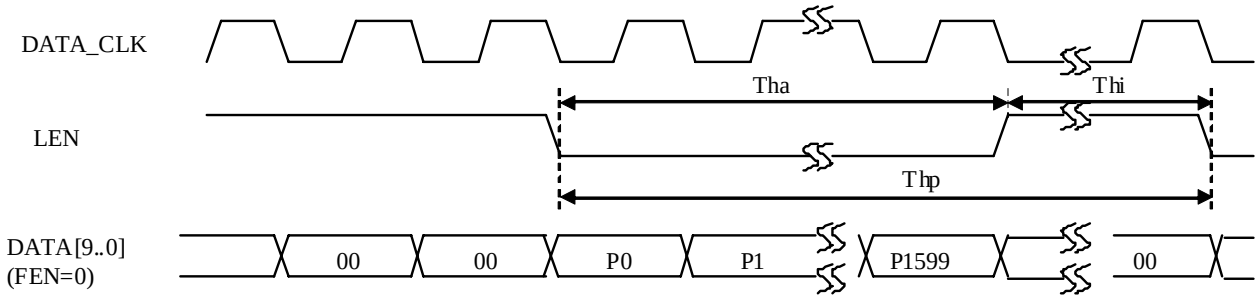


Table 20: horizontal timing specification

Parameter	Symbol	Default ROI	Unit
Horizontal active pixel(1)	Tha	1600	DATA_CLK
Horizontal period(2)	Thp	880	CLK_CTRL
Horizontal inactive pixel(2)	Thi	= Thp - Tha	DATA_CLK

(1) Depends on ROI and Sub sampling.
(2) Depends on line length configuration.

22.4 Line_length calculation

The sensor runs properly if the line length is well programmed. The following paragraph gives the user a procedure to calculate the minimum line length value to program for a correct sensor operation. Line length is calculated in number of CLK_CTRL periods.

If the line length is not correct, two flags can warn the user:
the ***error_ll_vs_conv*** flag in the register h3E. A too long conversion time will set this flag.
the ***error_ll_vs_xfer*** flag in the register h3E. A too long data readout time will set this flag.

The user must verify both flags to avoid any line length programming errors.

Table 21 : Useful registers for Line Length calculation

Entries	Register Address	Comments
CLK_CTRL (MHz)	-	
CLK_ADC (MHz)	-	
CLK_CHAIN (MHz)	-	
Pixtime_read_5T_width	h49	Range [0;255]
Pixtime_read_4T_width	h49	Range [0;255]
Max_offset	h06	Range [0;255]
Roi_width		Calculated in paragraph 7.5.2.1

The minimum line length value to be programmed in the SPI register (@ h04) is given by the following formula:

$$Line\_length\_\min = \frac{\max\left[Line\_length\_conv; Line\_length\_roi\right]}{8}$$

For a 4T pixel timing:

$$Line\_length\_conv = 4 + 2 \times pixtime\_read\_4T\_width + \frac{\left(\max\_offset + 2^{10} + 40\right) \times CLK\_CTRL}{CLK\_ADC}$$

$$Line\_length\_roi = \frac{Line\_width \times CLK\_CTRL}{CLK\_CHAIN} + \frac{40 \times CLK\_CTRL}{CLK\_ADC}$$

For a 5T pixel timing:

$$Line\_length\_conv = 4 + 2 \times pixtime\_read\_5T\_width + \frac{\left(\max\_offset + 2^{10} + 40\right) \times CLK\_CTRL}{CLK\_ADC}$$

$$Line\_length\_roi = \frac{Line\_width \times CLK\_CTRL}{CLK\_CHAIN} + \frac{40 \times CLK\_CTRL}{CLK\_ADC}$$

Both with:
Line_width = max [ROI_Width ; 1028]

22.5 Pixel timings

Figure 48: Data and sync timing

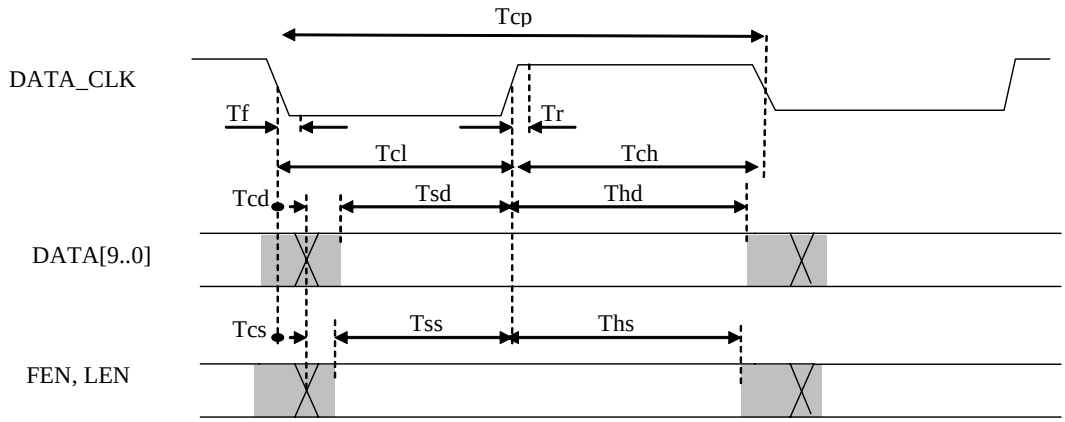


Table 22 : Data and Sync parameters

Parameter Definition	Symbol	Min	Typ	Max	Unit
Clock period	Tcp	8.33	8.77	200	ns
Clock low time(1)	Tcl	3.7			ns
Clock high time(1)	Tch	2			ns
DATA_CLK to data	Tcd	-0.9	-0.2	+0.7	ns
DATA_CLK to synch FEN or LEN	Tcs	-1.4	-0.6	+0.2	ns
Falling & rising edges on all signals with 10 pF load	Tr / Tf	0.8	1.5	2.6	ns

(1) Including the clock input duty cycle 50 +/- 10% & frequency precision.
Setup times: Tsd = Tcl – Tcd max – Tr / Tss = Tcl – Tcs max – Tr,
Hold times: Thd = Tch – Tcd min – Tr / Ths = Tch – Tcs min – Tr.

22.6 FLO (FLash Output)

This signal can be used to control the light source. Several SPI registers are used to define this signal.

This signal may be inverted *sync_flo_inv* in *<reg_miscel2>* see § **20.3.4** (in the timings FLO is shown non inverted)

The FLO control mode can be selected using *roi_flash_mode* in <reg_ctrl_cfg> see § **20.3.8**

- FLO signal may be stuck at 1 or at 0 *roi_flash_mode* = h3 or h0 respectively.
- FLO1 can be calculated based on integration time only *roi_flash_mode* = h1
- FLO2 can be calculated based on integration time plus readout time *roi_flash_mode* = h2

The timings can be adjusted using ***t_flash_del_off*** & ***t_flash_del_on*** in <**reg_flash_delay**> see § **0.**
Programming the FLO depends on the selected mode.

22.6.1 FLO in GS mode

In this mode use only the FLO based on integration time only (FLO1).

Figure 49: FLO timing, serial mode GS

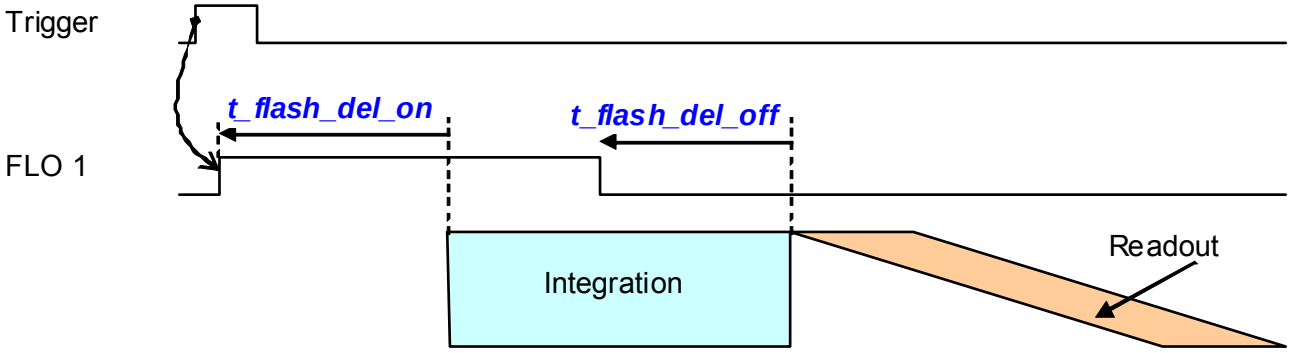
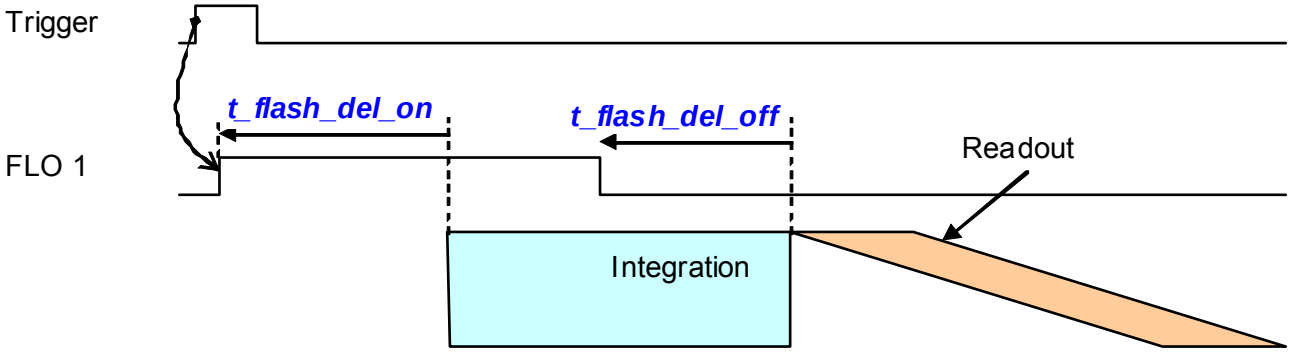


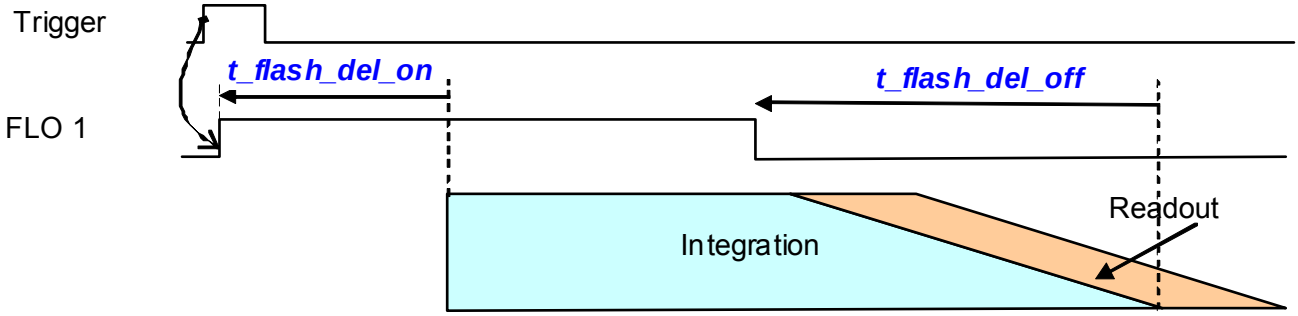
Figure 50: FLO timing, overlap mode GS



22.6.2 FLO in 4T + GR mode

In this mode the FLO based on integration time only should be used. (FLO1). Flash should be switched off before readout.

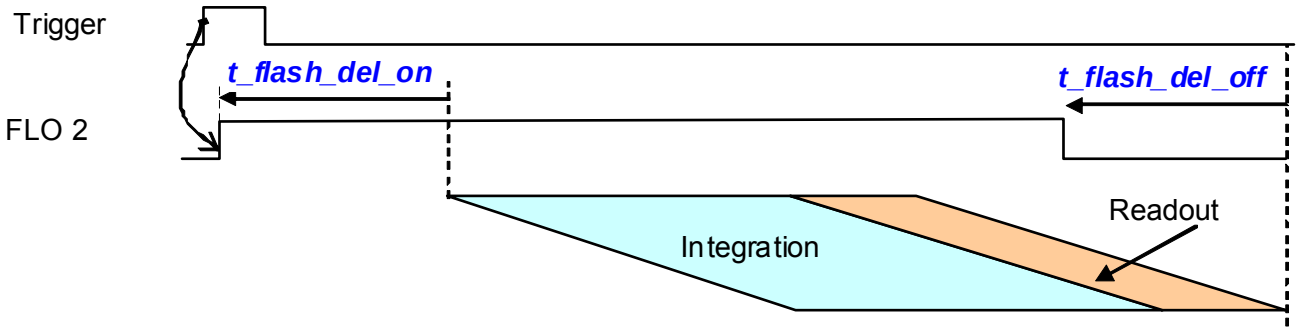
Figure 51: FLO timing, 4T + GR mode



22.6.3 FLO in 4T ERS mode

In this mode the FLO based on integration time + readout should be used. (FLO2). Using *t_flash_del_off* at 0 allows all overlap integration conditions.

Figure 52: FLO timing, serial mode ERS



23. PACKAGE SPECIFICATION

Figure 53 CLCC 48 Package Drawing

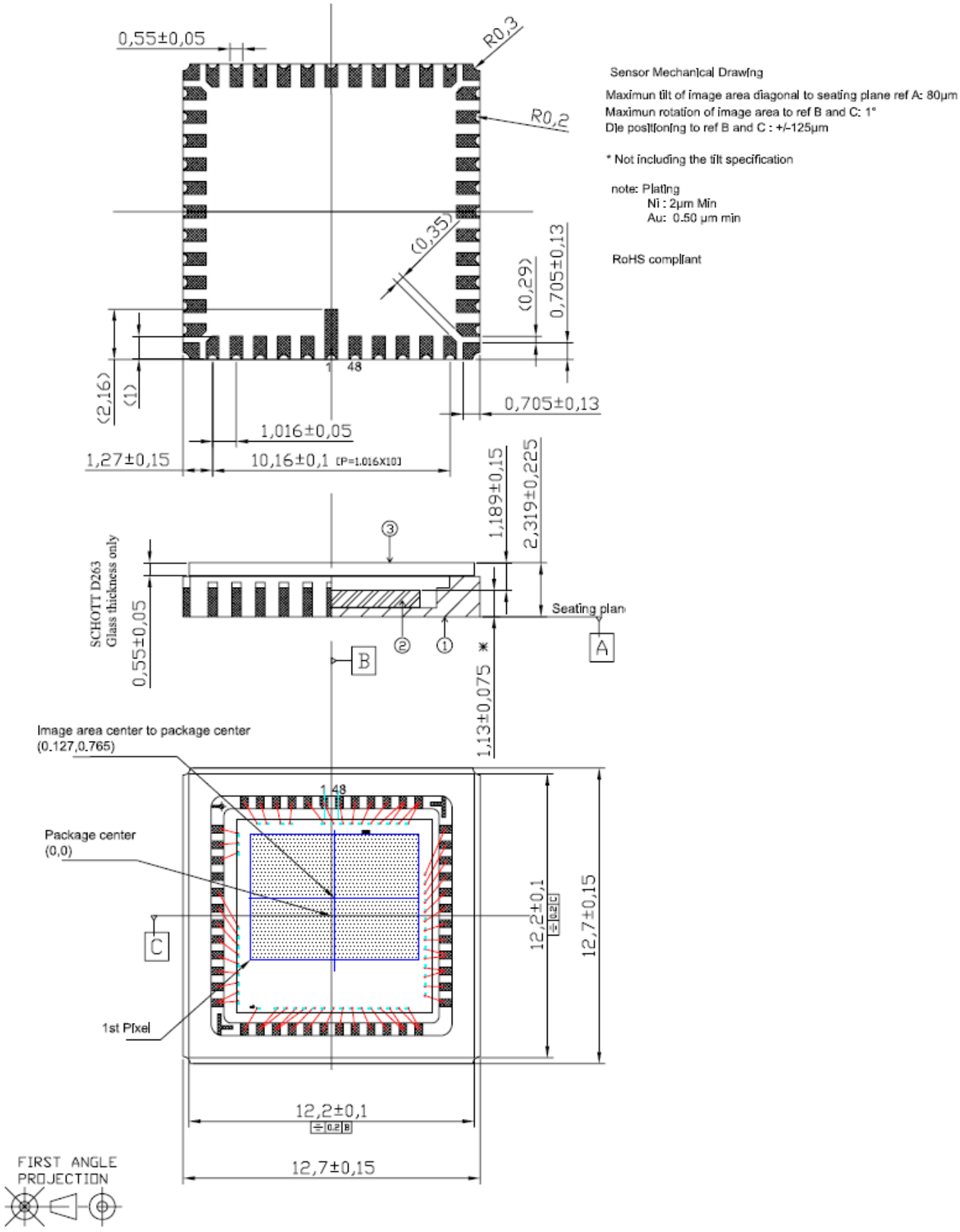
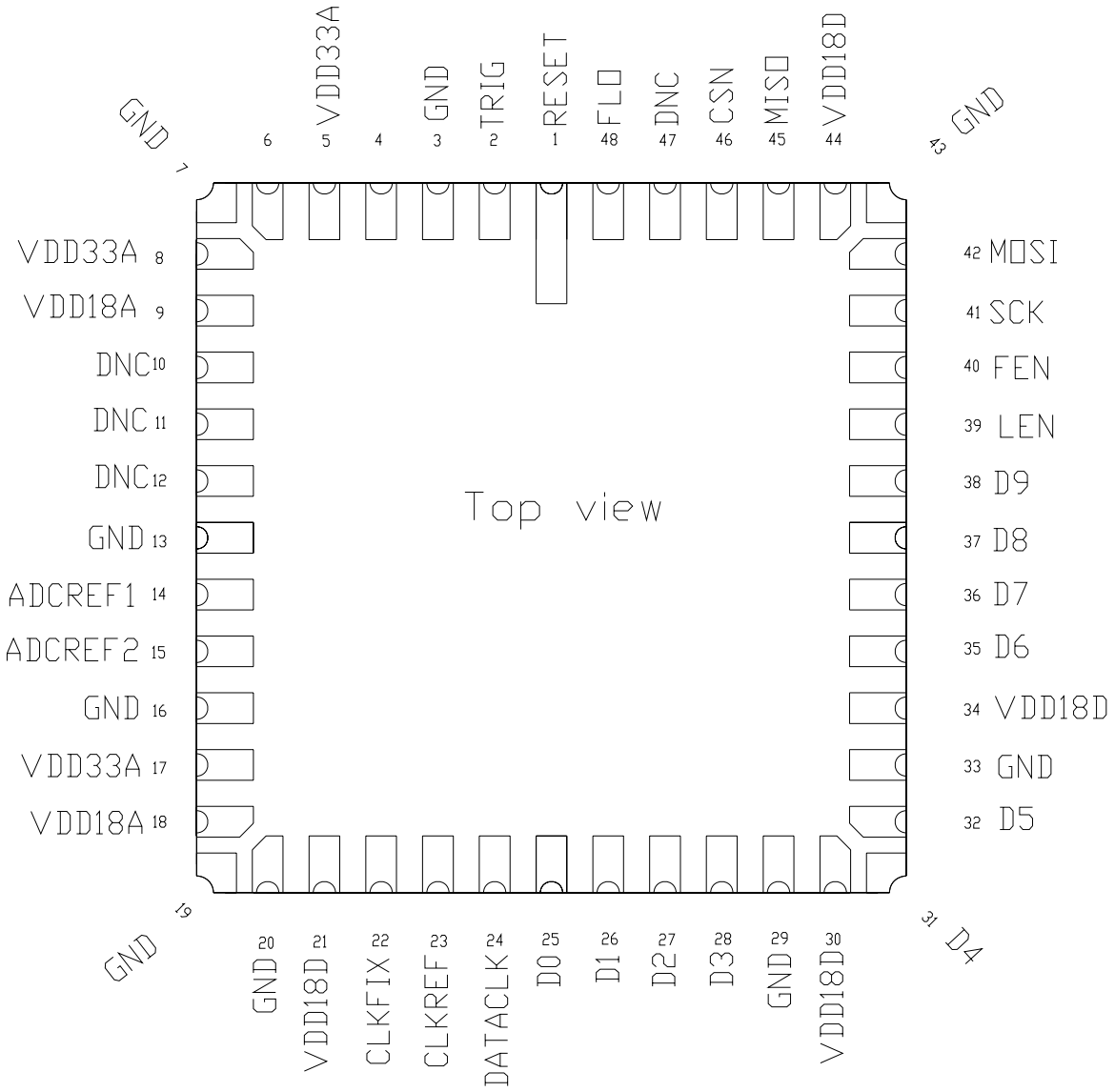


Table 23 : Window Characteristics

Parameters	Specification
Window material	SCHOTT D263
Window thickness	0.55 +/-0.05 mm
Window index	ne = 1.5255

Figure 54 CLCC 48 Package Pinout drawing



24. INPUT/OUTPUT LIST

Table 24 : I/O list

Name	Function / Description	I/O	Pin N°
VDD33A	3.3 V supply voltage for analog domain	POWER(1)	5, 8, 17
VDD18A	1.8 V Analog power, decoupling	POWER(1)	9, 18
VDD18D	1.8 V supply voltage for digital domain	POWER(1)	21, 30, 34, 44
GND	Grounds	POWER(2)	3, 7, 13, 16, 19, 20, 29, 33, 43
Test	Test pins	DNC(3)	10, 11, 12, 47
RESETN	Reset control	IN	1
TRIG	Trigger input with pull-down	IN	2
VREFP_1	VREFP supply for line decoder	IN/OUT	4
VREFP_2	VREFP supply for matrix	IN/OUT	6
ADC_REF_1	Adjusts ADC range by inserting a resistor between these two pins.	IN/OUT	14
ADC_REF_2		IN/OUT	15
CLK_FIX	Clock input fixed	IN	22
CLK_REF	Reference Clock input	IN	23
DATA_CLK	Data output clock	OUT	24
DATA 0	Data 0	OUT	25
DATA 1	Data 1	OUT	26
DATA 2	Data 2	OUT	27
DATA 3	Data 3	OUT	28
DATA 4	Data 4	OUT	31
DATA 5	Data 5	OUT	32
DATA 6	Data 6	OUT	35
DATA 7	Data 7	OUT	36
DATA 8	Data 8	OUT	37
DATA 9	Data 9	OUT	38
LEN	Line ENable	OUT	39
FEN	Frame ENable	OUT	40
SCK	SPI Clock input	IN	41
MOSI	SPI Data Input in slave mode,	IN/OUT	42
MISO	SPI Data Output in slave mode,	IN/OUT	45
CSN	SPI Chip Select Enable	IN	46
FLO	Flash Output	OUT	48

(1) All power pins with the same name must be connected to the same power supply.
(2) All grounds must be connected.
(3) DNC stands for Do Not Connect

25. ORDERING CODES

25.1 Standard version

- EV76C570ABT-TRAY Monochrome product,
- EV76C570ACT-TRAY Color Bayer product.

25.2 60fps version

- EV76C570ABT6-TRAY Monochrome product,
- EV76C570ACT6-TRAY Color Bayer product.

25.3 60fps version with protective foil

- EV76C570ABT6F-TRAY Monochrome product,
- EV76C570ACT6F-TRAY Color Bayer product.

For other packaging or other CFA please contact e2v
The sensors are delivered in Jedec trays of 119 units each.