1/2.5-Inch CMOS Digital Image Sensor

MT9E001

For the latest MT9E001 data sheet, refer to Micron's Web site: www.mircon.com/imaging

Features

- DigitalClarity® CMOS imaging technology
- Superior low-light performance
- Low dark current
- Simple two-wire serial interface
- Auto black level calibration
- Support for external mechanical shutter
- Support for external LED or Xenon flash
- High frame rate preview mode with arbitrary downsize scaling from maximum resolution
- Programmable controls: gain, frame size/rate, exposure, left-right and top-bottom image reversal, window size, and panning
- Data interface: parallel
- On-chip phase-locked loop (PLL)
- Bayer pattern down-size scaler
- Four channel shading correction (SC)

Applications

- Digital still cameras
- Cellular phones

Ordering Information

Table 1: Available Part Numbers

Part Number	Description
MT9E001I12STC	48-pin iLCC

Table 2: Key Performance Parameters

Parameter		Value
Optical format		1/2.5-inch (4:3)
Full resolution		3264 x 2448 pixels
Pixel size		1.75 x 1.75µm
Chief ray angle		10.19 maximum
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS) with global reset release (GRR)
Input clock frequency		6–48 MHz
Maximum data rate/master clock		96 Mbps
Frame rate	Full resolution	11 fps
	Video mode	30 fps
Supply voltage	Analog	2.4–3.1V (2.8V nominal)
	Digital	1.7–1.9V (1.8V nominal)
	I/O	1.8 or 2.8V
	PLL	2.4–3.1V (2.8V nominal)
ADC resolution		12-bit
Responsivity		0.3 V/lux-sec (at 550nm) (preliminary)
Dynamic range		70dB (preliminary)
SNR_{MAX}		38.9dB (preliminary)
Power consumption	Full resolution	500mW[1] (typical, 8Mp, 11 fps at 96 MHz)
	Video mode	300mW[1] (typical, VGA resolution by subsampling[2]), 30 fps at 48 MHz
	Standby	45µW (typical, EXTCLK disabled)
Operating temperature		–30°C to +70°C (at junction)
Package		48-pin iLCC

Notes: 1. Output pads power not included, Tj = 55°C.
2. By skip2bin2 subsampling, maintaining full FOV as full resolution mode.

Table of Contents

List of Figures

List of Tables

General Description

The Micron® Imaging MT9E001 is a 1/2.5-inch format CMOS active-pixel digital image sensor with a pixel array of 3264H x 2448V. It incorporates sophisticated on-chip camera functions such as windowing, mirroring, binning and skip modes, and snapshot mode. It is programmable through a simple two-wire serial interface and has very low power consumption.

The MT9E001 digital image sensor features DigitalClarity® technology—Micron's break-through low-noise CMOS imaging technology that achieves near CCD image quality (based on signal-to-noise ratio and low-light sensitivity) while maintaining the inherent size, cost, power consumption, and integration advantages of CMOS.

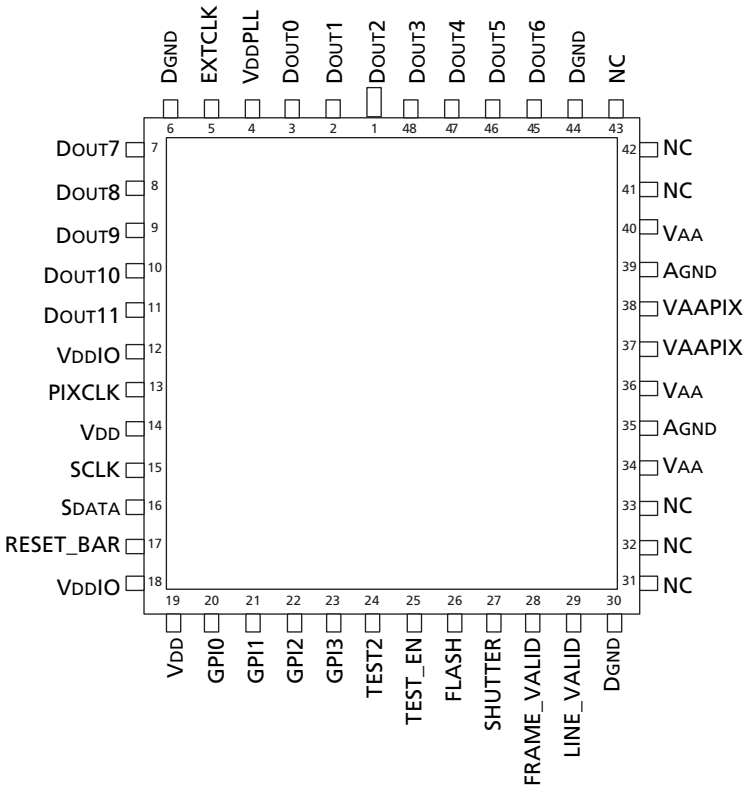
Signal Description

Table 3 provides the signal descriptions for the MT9E001.

Table 3: Signal Description

Name	Type	Description
SCLK	Input	Serial clock for access to control and status registers.
TEST2	Input	Reserved for factory use. Tie to digital ground during normal operation.
RESET_BAR	Input	Asynchronous active LOW reset. When asserted, data output stops and all internal register are restored to their factory default settings.
EXTCLK	Input	Master clock input; PLL input clock, 6–48 MHz.
TEST_EN	Input	Reserved for factory use. Tie to digital ground during normal operation.
GPI[3:0]	Input	General purpose inputs. After reset, these pads are powered down by default (it is not necessary to bond to these pads). Any of these pads can be configured for hardware control of SADDR, output enable, and shutter trigger functions.
SDATA	I/O	Serial data.
PIXCLK	Output	Pixel clock. Used to qualify the LINE_VALID, FRAME_VALID, and DOUT[11:0] outputs.
FRAME_VALID	Output	FRAME_VALID output. Qualified by PIXCLK.
LINE_VALID	Output	LINE_VALID output. Qualified by PIXCLK.
SHUTTER	Output	Control for external mechanical shutter.
FLASH	Output	Flash output. Synchronization pulse for external light source.
DOUT[11:0]	Output	Twelve-bit image data output.
VDD	Supply	Digital power (1.8V).
VAAPIX	Supply	Pixel array power (2.8V).
VAA	Supply	Analog power (2.8V).
VDDPLL	Supply	PLL power (2.8V).
VDDIO	Supply	I/O power supply (1.8 or 2.8V).
DGND	Supply	Digital, I/O, and PLL ground.
AGND	Supply	Analog ground.

Figure 1: 48-Pin ILCC 10 x 10 Package Pinout Diagram (Top View)

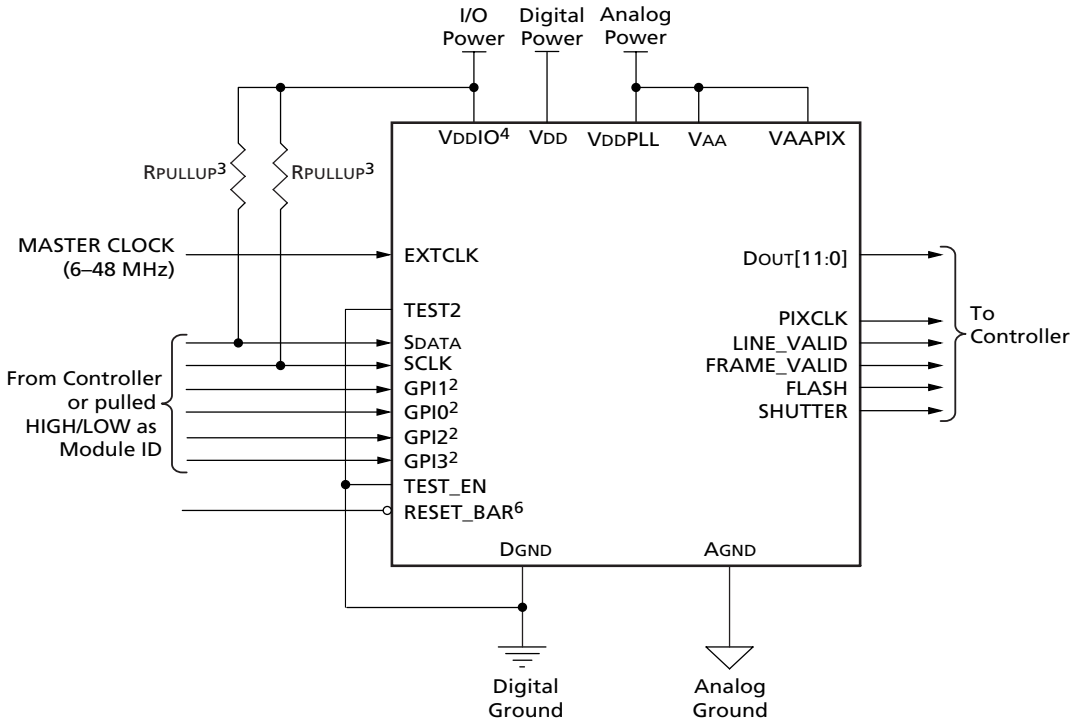


Typical Connections

Figure 2 shows typical MT9E001 device connections. For low-noise operation, the MT9E001 requires separate power supplies for analog and digital. Incoming digital and analog ground conductors can be tied together next to the die. Both power supply rails should be decoupled to ground using capacitors as close as possible to the die. The use of inductance filters is not recommended on the power supplies or output signals.

The MT9E001 also supports different digital core (VDD/DGND) and I/O power (VDDIO/DGND) power domains that can be at different voltages. The PLL requires a clean power source (VDDPLL).

Figure 2: Typical Configuration (connection)

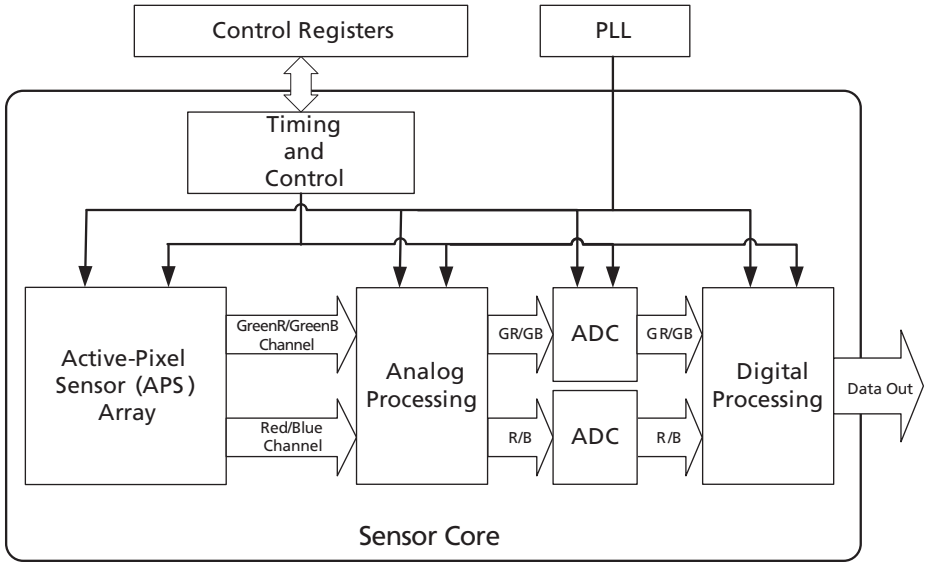


Notes:
1. Connection diagram shows only one of many possible variations for this sensor.
2. The GPI pads can configure multiple features for the sensor.
3. Recommended resistor value is 1.5KΩ for the two-wire serial interface RPULL-UP; however, greater value may be used for slower transmission speed.
4. All inputs must be configured with VDDIO.
5. VAA and VAAPIX must be tied together.
6. TEST_EN and TEST2 must be connected to DGND for normal device operation.

Architecture Overview

The MT9E001 is a progressive-scan sensor that generates a stream of pixel data at a constant frame rate. It uses an on-chip PLL to generate all internal clocks from a single master input clock running between 6 and 48 MHz. The maximum pixel rate is 96 Mbps, corresponding to a physical pixel clock rate of 96 MHz. Figure 3 shows a block diagram of the sensor.

Figure 3: Block Diagram



Sensor Core Description

The core of the sensor is an active-pixel array. The timing and control circuitry sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and reading that row, the pixels in the row integrate incident light. The exposure is controlled by varying the integration. Once a row has been read, the data from the columns is sequenced through an analog signal chain (providing offset correction and gain), and then through an ADC. The output from the ADC is a 12-bit value for each pixel in the array. The ADC output passes through a digital processing signal chain (providing further data path corrections and applying digital gain). The digital block contains SMIA functionality including scaler, compression, embedded data, limiter, and FIFO.

The pixel array contains optically active and light-shielded (black) pixels. The black pixels are used to provide data for on-chip offset correction algorithms (black level control).

The sensor contains a set of control and status registers that can be used to control many aspects of the sensor behavior including the frame size, exposure, and gain setting. These registers can be accessed through a two-wire serial interface.

The output from the sensor is a Bayer pattern: alternate rows are a sequence of either green/red pixels or blue/green pixels. The offset and gain stages of the analog signal chain provide per-color control of the pixel data.

The control registers, timing and control and digital processing functions shown in Figure 3 on page 10 are partitioned into two logical parts:

- A sensor core which provides array control and data path corrections. The output of the sensor core is 12-bit parallel pixel data stream qualified by an output data clock (PIXCLK), together with LINE_VALID and FRAME_VALID signals.
- Additional functionality is included to support the SMIA standard. This includes a horizontal and vertical image scaler, a limiter, a data compressor, an output FIFO.

A flash output strobe is provided to allow an external Xenon or LED light source to synchronize with the sensor exposure time. Additional I/O signals support the provision of an external mechanical shutter.

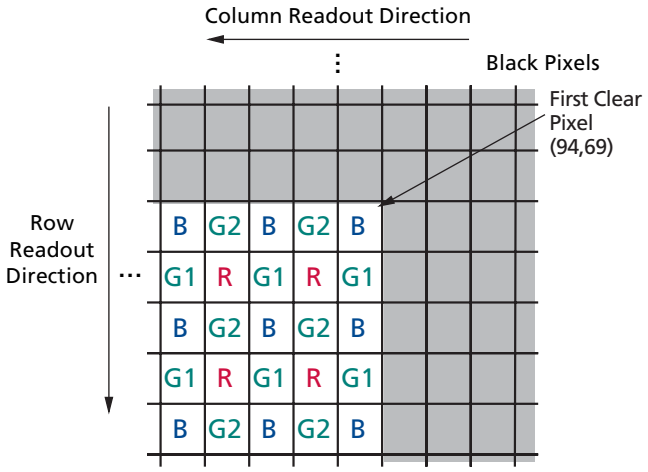
Pixel Array

The MT9E001 image sensor array consists of a 3,382-column by 2,540-row matrix of pixels addressed by column and row. The address (column 0, row 0) represents the upper-left corner of the entire array as oriented in the output image, which is the upper-right pixel, when looking at the chip.

The active region in the center of the array consists of a 3,264-columns by 2,448-rows representing the default output image. It is surrounded by a boundary region (also active), and a border of shielded dark pixels. The boundary region can be used to avoid edge effects when doing color processing to achieve a 3264 x 2448 result image.

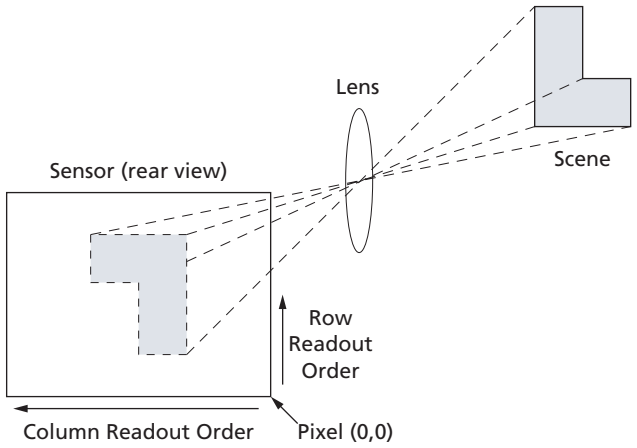
The 4-pixel border on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end and y_addr_end registers.

Figure 4: Pixel Color Pattern Detail (Top Right Corner)



Default Readout Order

Figure 5: Imaging a Scene



Timing and Control

Analog Gain Options

The MT9E001 provides two mechanisms for setting the analog gain. The first uses the SMIA gain model; the second uses the traditional Micron Imaging gain model. The following sections describe both models, the mapping between the models, and the operation of the per-color and global gain control. Use of high gains can result in reduced image quality by introducing noise and by amplifying image defects or artifacts.

Using Per-color or Global Gain Control

The read-only analogue_gain_capability register returns a value of "1," indicating that the MT9E001 provides per-color gain control. However, the MT9E001 also provides the option of global gain control. Per-color and global gain control can be used interchangeably. A write to a global gain register is aliased as a write of the same data to the four associated color-dependent gain registers. A read from a global gain register is aliased to a read of the associated color-dependent gain registers.

The read/write gain_mode register required by SMIA has no defined function in the SMIA specification. In the MT9E001 this register has no side effects on the operation of the gain; per-color and global gain control can be used interchangeably regardless of the state of the gain_mode register.

SMIA Gain Model

The SMIA gain model uses the following registers to set the analog gain:

- analogue_gain_code_global
- analogue_gain_code_greenR
- analogue_gain_code_red
- analogue_gain_code_blue
- analogue_gain_code_greenB

The SMIA gain model requires a uniform step size between all gain settings. The analog gain is given by:

$$gain = \frac{analogue\_gain\_m0 \;\; x \;\; analogue\_gain\_code}{analogue\_gain\_c1} = \frac{analogue\_gain\_code\_<color>}{8} \quad \text{(EQ 1)}$$

Micron Imaging Gain Model

The Micron Imaging gain model uses the following registers to set the analog gain:

- global_gain
- greenR_gain
- red_gain
- blue_gain
- greenB_gain

This gain model maps directly to the control settings applied to the gain stages of the analog signal chain. This provides a 7-bit gain stage and two 2X gain stages. As a result, the step size varies depending upon whether the 2X gain stages are enabled. The analog gain is given by:

$$gain = (<color>\_gain[8]+1)\times(<color>\_gain[7]+1)\times\frac{<color>\_gain[6:0]}{32} \quad \text{(EQ 2)}$$

As a result of the 2X gain stages, many of the possible gain settings can be achieved in two different ways. For example, red_gain = 0x02A0 provides the same gain as red_gain = 0x0240 and red_gain = 0x0320. The first example uses the first 2X gain stage, the second example uses no 2X gain stage and the third example uses the second 2X gain stage. In all cases, the preferred setting is the setting that enables the first 2X gain stage and not the last 2X gain stage, since this will result in lower noise. The recommended sequence is shown in Table 4.

Table 4: Recommended Gain Settings

Desired Gain	Recommended Gain Register Setting
1–1.969	0x0220–0x023F
2–7.9375	0x02A0–0x02FF
8–15.875	0x03C0–0x03FF

Gain Code Mapping

The Micron Imaging gain model maps directly to the underlying structure of the gain stages in the analog signal chain. When the SMIA gain model is used, gain codes are translated into equivalent settings in the Micron Imaging gain model.

When the SMIA gain model is in use and values have been written to the analogue_gain_code_<color> registers, the associated value in the Micron Imaging gain model can be read from the SMIA associated <color>_gain register. In cases where there is more than one possible mapping, the recommended gain register setting is followed, in order to provide the mapping with the lowest noise.

When the Micron Imaging gain model is in use and values have been written to the gain_<color> registers, data read from the associated analogue_gain_code_<color> register is UNDEFINED. The reason for this is that many of the gain codes available in the Micron Imaging gain model have no corresponding value in the SMIA gain model.

The result of this is that the two gain models can be used interchangeably but, having written gains through one set of registers, those gains should be read back through the same set of registers.

Digital Gain

Integer digital gains in the range 1–7 can be programmed.

As gain is increased, image quality degrades due to the amplification of image defects.

Pedestal

This block adds the value from R0x301E (data_pedestal_) to the incoming pixel value. The data_pedestal register is read-only by default but can be configured to be read/write by clearing the lock_reg bit in R0x301A–B. The only way to disable the effect of the pedestal is to set the register to “0.”

Integration Time

The integration (exposure) time of the sensor is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

(EQ 3)

$$fine\_integration\_time\_min < fine\_integration\_time < (line\_length\_pck - fine\_integration\_time\_max\_margin)$$

The limits for coarse time are defined by:

(EQ 4)

The actual integration time is given by:

coarse_integration_time_min< *coarse_integration_time* <(*frame_length_lines* - *coarse_integration_time_max_margin*)

(EQ 5)

$$integration\_time = \frac{((coarse\_integration\_time \times line\_length\_pck) + fine\_integration\_time)}{vt\_pix\_clk\_freq\_mhz/10^6}$$

PLL

The sensor contains a PLL for timing generation and control. The PLL contains a prescaler to divide the input clock applied on EXTCLK, a VCO to multiply the prescaler output, and a set of dividers to generate the output clocks. The clocking structure is shown in Figure 6 on page 15.

Figure 6: Clocking Structure

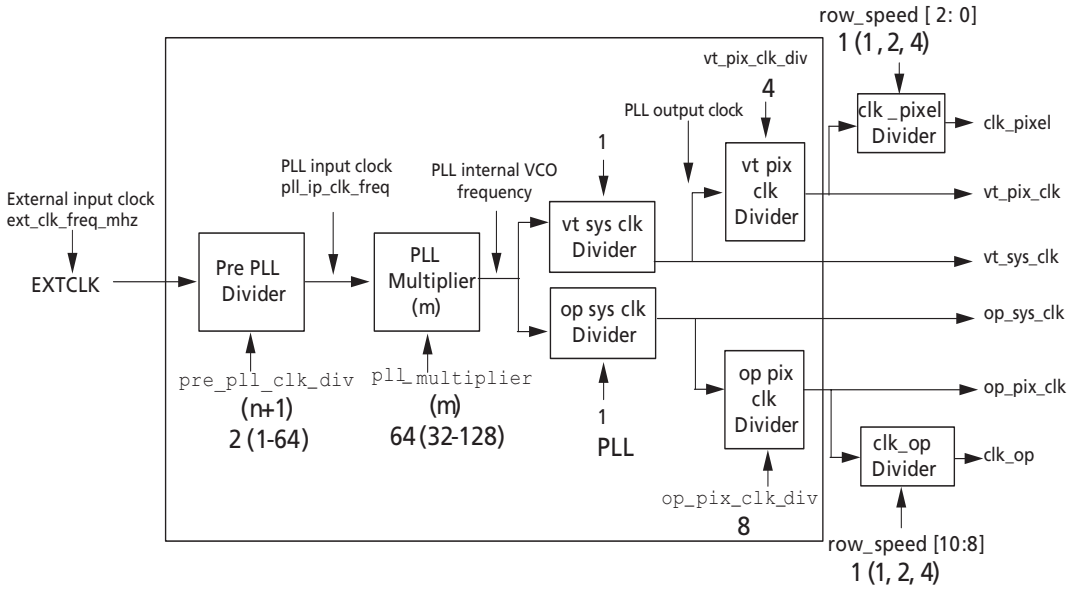


Figure 6 shows the different clocks and the register names (in `courier` font). It also shows the default setting for each divider/multiplier control register, and the range of legal values for each divider/multiplier control register. The vt and op sys clk Divider is hardwired in the design.

From the diagram, the clock frequencies can be calculated as follows:

Internal pixel clock used to readout the pixel array:

(EQ 6)

External pixel clock used to output the data:

(EQ 7)

$$clk\_pixel\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times vt\_pix\_clk\_div \times row\_speed\ [2:0]} = \frac{24\ MHz \times 64}{2 \times 4 \times 1} = 192\ MHz$$

$$clk\_op\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times op\_pix\_clk\_div \times row\_speed\ [10:8]} = \frac{24\ MHz \times 64}{2 \times 8 \times 1} = 96\ MHz$$

Internal master clock:

$$vt\_pix\_clk\_freq\_mhz/2 \quad \text{(EQ 8)}$$

The parameter limit register space contains registers that declare the minimum and maximum allowable values for:

- The frequency allowable on each clock.
- The divisors that are used to control each clock.

The following factors determine what are valid values, or combinations of valid values, for the divider/multiplier control registers:

- The minimum/maximum frequency limits for the associated clock must be met.

  pll_ip_clk_freq must be in the range 2–24 MHz. Higher frequencies are preferred.

  PLL internal VCO frequency must be in the range 384–768 MHz.
- The minimum/maximum value for the divider/multiplier must be met.

  Range for m: 32–128.

  Range for n: 0–63. Range for (n + 1): 1–64.
- The op_pix_clk must never run faster than the vt_pix_clk to ensure that the output data stream is contiguous.
- Given the maximum programmed line length, the minimum blanking time, the maximum image width, the available PLL divisor/multiplier values, and the requirement that the output line time (including the necessary blanking) must be output in a time equal to or less than the time defined by line_length_pck.

Although the PLL VCO input frequency range is advertised as 6 MHz–48 MHz, superior performance is obtained by keeping the VCO input frequency as high as possible.

The usage of the output clocks is shown below:

- clk_pixel is used by the sensor core to control the timing of the pixel array. The sensor core produces one 12-bit pixel each vt_pix_clk period. The line length (line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of the clk_pixel period.
- clk_op is used to load parallel pixel data from the output FIFO. The output FIFO generates one pixel each op_pix_clk period.

PLL Generated Master Clock

PLL Setup

The PLL divisors should be programmed while the sensor is in the software standby state. The PLL is enabled by entering the STREAMING state. STREAMING state will be entered after the VCO lock time.

The VCO lock time is 100µs (typical), 1ms (maximum).

The effect of programming the PLL divisors whilst the sensor is in the streaming state is UNDEFINED.

Table 5: Frequency Parameters

Frequency	Equation	Min (MHz)	Max (MHz)
^{f}IN	–	6	48
^{f}PFD	^{f}extclk /(pll_n+1)	2	24
^{f}VCO	^{f}extclk * pll_m/(pll_n+1)	384	768

Readout Options

The sensor core supports different readout options to modify the output image. The readout can be limited to a specific window of the original pixel array.

For preview modes, the sensor core supports both skipping and pixel binning in x and y directions.

By changing the readout direction, the image can be flipped in the vertical and/or mirrored in the horizontal direction.

Window Size

The sequencing of the pixel array is controlled by the x_addr_start, y_addr_start, x_addr_end, and y_addr_end registers. The image output from the sensor core data path is controlled by these registers. The output image size is controlled by the x_output_size and y_output_size registers.

Pixel Border

The default settings of the sensor provide a 3264 x 2448 image. A border of up to 4 pixels on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end, and y_addr_end registers and then adjusting the x_output_size and y_output_size registers accordingly.

Column Readout Limitation

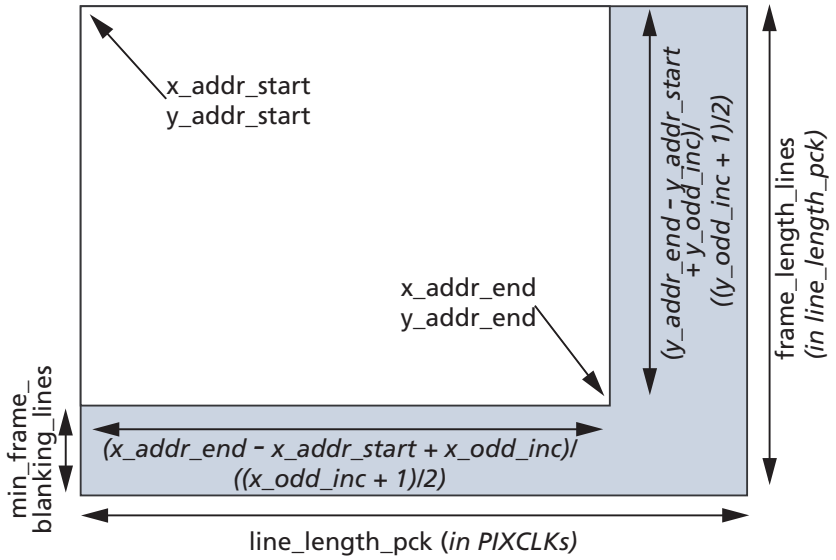
The MT9E001 has limitations on the allowed values of x_addr_start and x_addr_end.

The x_addr_start needs to be a multiple of 8 in normal mode, 16 in 2X skip or binning mode and 32 in 4X skip or binning mode. Similarly x_addr_end needs to be set so the width of the window read out after taking subsampling mode into account is a multiple of 8.

SMIA Windowing

The array windowing is programmed using the SMIA standard. The row time is defined by the line_length_pck which includes pixel clock times for both the active pixels in the row along with the line blanking time. The frame_length_lines similarly defines the number of active rows included with the frame as well as the vertical blanking lines in the frame.

Figure 7: SMIA Windowing



T_{FRAME} = line_length_pck * frame_length_lines * T_{PCK}

Equation 9 and Equation 10 provide the minimum timing for both the line_length_pck and frame_length_lines.

(EQ 9)

$$\text{line\_length\_pck}_{min} = \max\left[\begin{array}{l}\text{min\_line\_length\_pck} \\ \left(\frac{\text{x\_addr\_end} - \text{x\_addr\_start} + \text{x\_odd\_inc}}{(\text{x\_odd\_inc} + 1)/2} + \text{min\_line\_blanking\_pck}\right) \\ \left(\frac{\text{clk\_pixel}}{\text{clk\_op}}\right)(\text{x\_output\_size}) + 30\end{array}\right]$$

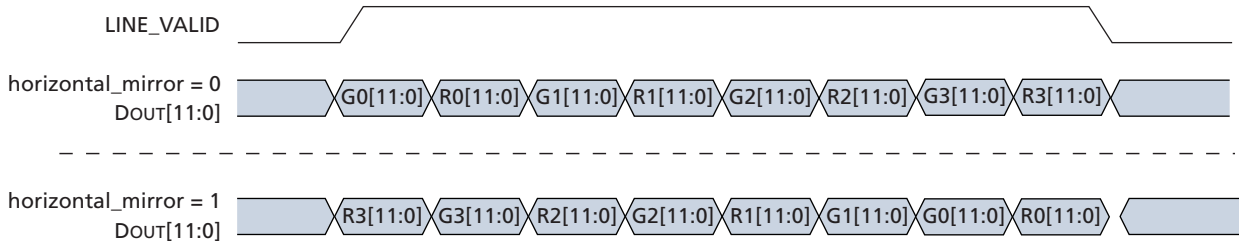
$$\text{frame\_length\_lines}_{min} = \frac{\text{y\_addr\_end - y\_addr\_start} + y\_\text{odd\_inc}}{(y\_\text{odd\_inc} + 1)/2} + \text{min\_frame\_blanking\_lines} \quad \text{(EQ 10)}$$

Readout Modes

Horizontal Mirror

When the horizontal_mirror bit (R0x3040[0]) is set in the image_orientation_ register, the order of pixel readout within a row is reversed, so that readout starts from x_addr_end and ends at x_addr_start. Figure 8 shows a sequence of 8 pixels being read out with horizontal_mirror = 0 and horizontal_mirror = 1. Changing horizontal_mirror causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.

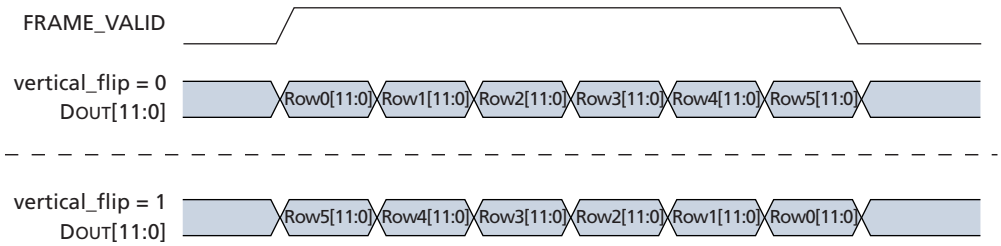
Figure 8: 8 Pixels in Normal and Column Mirror Readout Modes



Vertical Flip

When the vertical_flip bit is set in the image_orientation register, the order in which pixel rows are read out is reversed, so that row readout starts from y_addr_end and ends at y_addr_start. Figure 9 shows a sequence of six rows being read out with vertical_flip= 0 and vertical_flip = 1. Changing vertical_flip causes the Bayer order of the output image to change; the new order is reflected in the value of the pixel_order register.

Figure 9: 6 Rows in Normal and Row Mirror Readout Modes



Column and Row Skip

The sensor supports subsampling. Subsampling reduces the amount of data processed by the analogue signal chain in the sensor and thereby allows the frame rate to be increased. Subsampling is enabled by changing x_odd_inc and/or y_odd_inc. Values of 1, 3, and 7 are supported. Setting both of these variables to 3 reduces the amount of row and column data processed and is equivalent to the skip2 readout mode provided by earlier Micron Imaging sensors. Figure 10 shows a sequence of 8 columns being read out with x_odd_inc = 3 and y_odd_inc = 1.

Figure 10: Effect of x_odd_inc = 3 on Readout Sequence

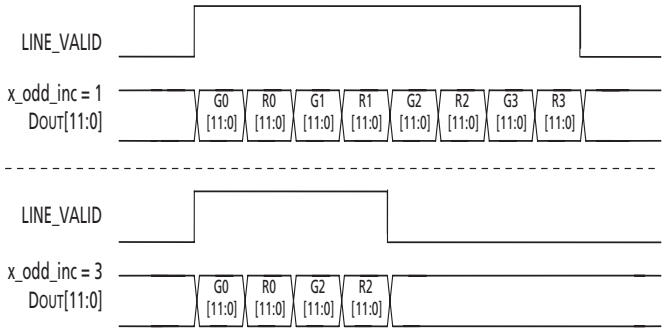
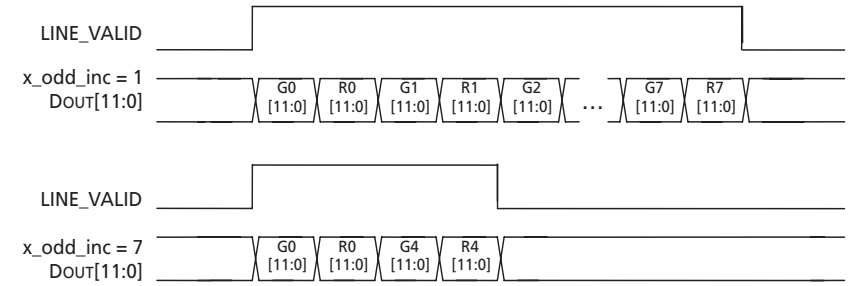


Figure 11: Effect of x_odd_inc = 7 on Readout Sequence



A 1/16 reduction in resolution is achieved by setting both x_odd_inc and y_odd_inc to 7. This is equivalent to skip4 readout mode provided by earlier Micron Imaging sensors. Figure 11 shows a sequence of 16 columns being read out with x_odd_inc = 7 and y_odd_inc = 1.
The effect of the different subsampling settings on the pixel array readout is shown in Figures 12 through Figure 14 on page 21.

Figure 12: Pixel Readout (no skipping, x_odd_inc = 1, y_odd_inc = 1)

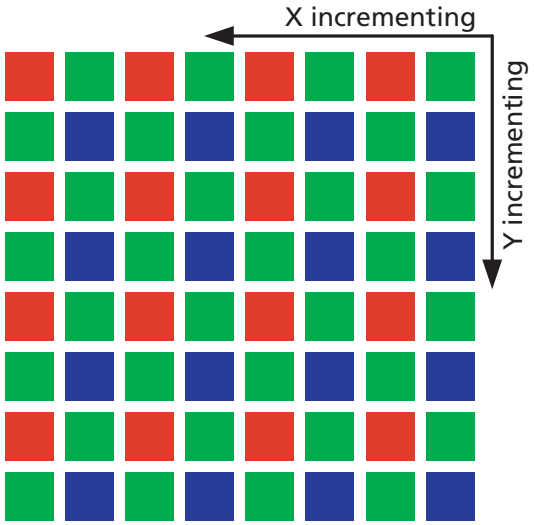


Figure 13: Pixel Readout (x_odd_inc = 3, y_odd_inc = 1)

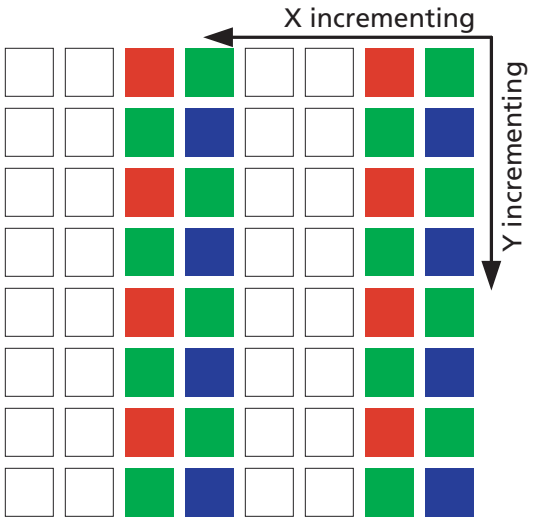


Figure 14: Pixel Readout (x_odd_inc = 1, y_odd_inc = 3)

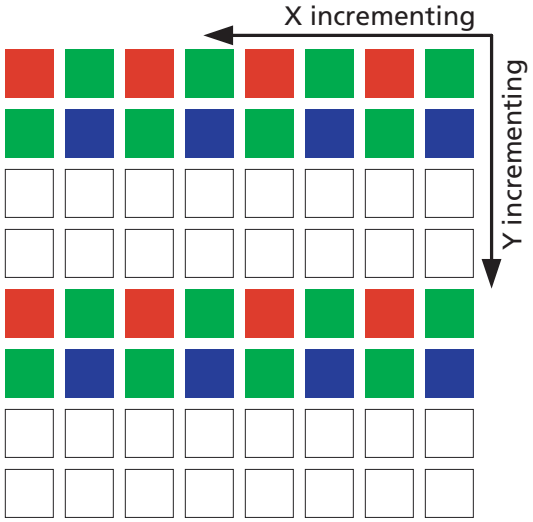
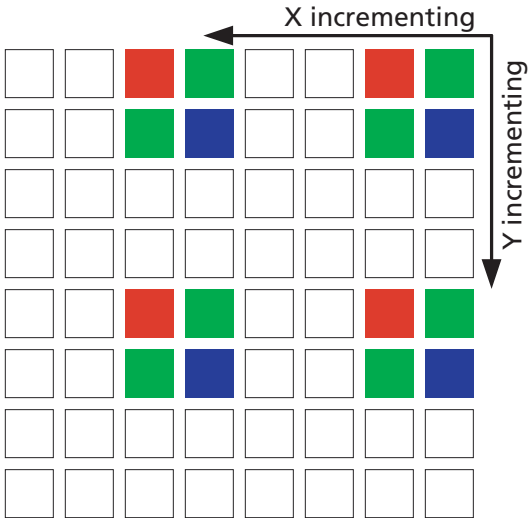


Figure 15: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)



Programming Restrictions when Subsampling

When subsampling is enabled as a viewfinder mode, and the sensor is switched back and forth between full resolution and subsampling, it is recommended that line_length_pck be kept constant between the two modes. This allows the same integration times to be used in each mode to maintain the same brightness.

When subsampling is enabled, it may be necessary to adjust the x_addr_end, x_addr_start, and y_addr_end settings: the values for these registers are required to correspond with rows/columns that form part of the subsampling sequence. The adjustment should be made in accordance with the following rules:

x_skip_factor = (x_odd_inc + 1) / 2

y_skip_factor = (y_odd_inc + 1) / 2

- x_addr_start should be a multiple of x_skip_factor*8
- (x_addr_end - x_addr_start + x_odd_inc) should be a multiple of x_skip_factor*8
- (y_addr_end - y_addr_start + y_odd_inc) should be a multiple of y_skip_factor*8

The number of columns/rows read out with subsampling can be found from the equation below:

- columns/rows = (addr_end - addr_start + odd_inc) / skip_factor

Example:

The sensor is set up to give out a 640 x 480 image:

- x_addr_start = 8
- x_addr_end = 647
- y_addr_start = 8
- y_addr_end = 487

To half the resolution in each direction the registers need to be reprogrammed as follows:

- x_addr_start = 0 (8 is not read out in subsampling mode)
- x_addr_end = 637 (adjust for new start address and end requirement)
- y_addr_start = 8 (no restrictions on row starting address)
- y_addr_end = 485 (adjust for end requirement)

To quarter the resolution in each direction the registers need to be reprogrammed as follows:

- x_addr_start = 0
- x_addr_end = 633 (adjust for new start address and end requirement)
- y_addr_start = 8 (no restrictions on row starting address)
- y_addr_end = 481 (adjust for end requirement)

Table 6 shows the row address sequencing for normal and subsampled readout. The same sequencing applies to column addresses for subsampled readout. There are two possible subsampling sequences for the rows (because the subsampling sequence only read half of the rows) depending upon the alignment of the start address. The row address sequencing during binning is also shown. Due to the restrictions in column readout, only one subsampling sequence that meets the required x_addr_start is supported. This corresponds to the columns for start = 0 in Table 6.

Table 6: Row Address Sequencing

odd_inc = 1	odd_inc = 3				odd_inc = 7			
Normal	Normal		Binned		Normal		Binned	
start = 0	start = 0	start = 2	start = 0	start = 2	start = 0	start = 2	start = 0	start = 2
0	0		0, 2		0		0, 2	
1	1		1, 3		1		1,	
2		2		2, 4		2		2, 4
3		3		3, 5		3		3, 5
4	4		4, 6					
5	5		5, 7					
6		6		6, 8				
7		7		7, 9				
8	8		8, 10		8		8, 10	
9	9		9, 11		9		9, 11	
10		10		10, 12		10		10, 12
11		11		11, 13		11		11, 13
12	12		12, 14					
13	13		13, 15					
14		14		14, 16				
15		15		15, 17				

Binning

The sensor supports 2 x 1 and 2 x 2 analog binning which includes column binning (x-binning), and row/column binning (xy-binning). Binning has many of the same characteristics as subsampling, however:

- It gathers image data from all pixels in the active window (rather than a subset of them).
- It achieves superior image quality.
- It avoids the aliasing artifacts that can be a characteristic side effect of subsampling.

Binning is enabled by selecting the appropriate subsampling settings (x_odd_inc = 3 and y_odd_inc = 1 for x-binning, x_odd_inc = 3 and y_odd_inc = 3 for xy-binning) and setting the appropriate binning bit in read_mode (R0x3040–1). In subsampling, x_addr_end and y_addr_end may require adjustment when binning is enabled. It is the first of the two columns/rows binned together that should be the end column/row in binning, so the requirements for the end address is exactly the same as in nonbinning subsampling mode.

Binning can also be enabled when the 4X subsampling mode is enabled (x_odd_inc = 7 and y_odd_inc = 1 for x-binning, x_odd_inc = 7 and y_odd_inc = 7 for xy-binning). In this mode, however, not all pixels will be used so this is not a 4X binning implementation. An implementation providing a combination of skip2 and bin2 is used to achieve 4X subsampling with better image quality.

The effect of the different subsampling settings is shown in Figure 16 and Figure 17 on page 25.

Figure 16: Pixel Readout (x_odd_inc = 3, y_odd_inc = 1, x_bin = 1)

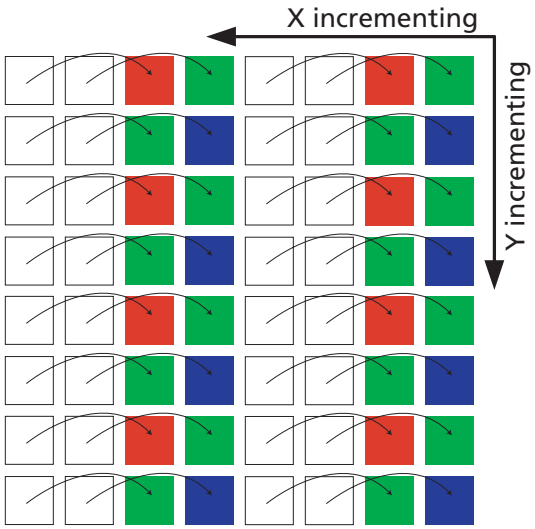


Figure 17: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3, x_ybin = 1)

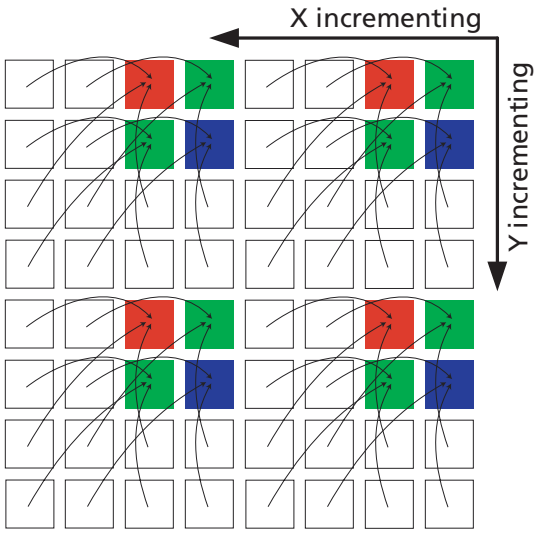
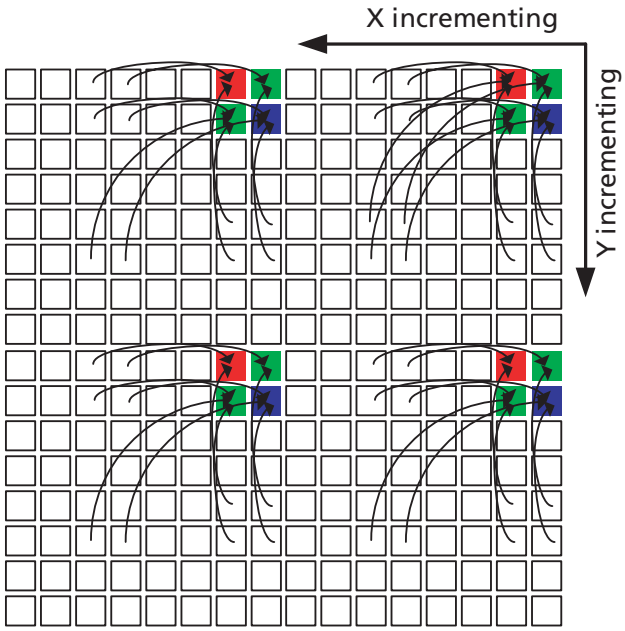


Figure 18: Pixel Readout (x_odd_inc = 7, y_odd_inc = 7, x_ybin = 1)



Binning Limitations

Binning requires different sequencing of the pixel array and imposes different timing limits on the operation of the sensor. In particular, xy-binning requires two read operations from the pixel array for each line of output data, which has the effect of increasing the minimum line blanking time. The SMIA specification cannot accommodate this variation because its parameter limit registers are defined as being static.

As a result, when xy-binning is enabled, some of the programming limits declared in the parameter limit registers are no longer valid. In addition, the default values for some of the manufacturer specific registers need to be reprogrammed. The recommended settings are shown in Table 7. None of these adjustments are required for x-binning.

Table 7: Register Adjustments Required for Binning Mode

Register	Type	Default (Normal Readout)	Recommended Setting During Binning	Notes
min_line_blanking_pck	Read-only	0x06AC	0x0C40	Read-only register for control software; does not affect operation of sensor.
min_line_length_pck	Read-only	0x0914	0x1200	Read-only register for control software; does not affect operation of sensor.
fine_integration_time_min	Read-only	0x056A	0x0B1A	Read-only register for control software; does not affect operation of sensor.
fine_integration_time_max_margin	Read-only	0x03AA	0x06E6	Read-only register for control software; does not affect operation of sensor.
fine_correction	Read/write	0x0100	0x0238	Affects operation of sensor
fine_integration_time	Read/write	0x056A	0x0B1A	Normal default is minimum value

Since binning also requires subsampling to be enabled, the same restrictions apply to the setting of x_addr_end and y_addr_end ("Programming Restrictions when Subsampling" on page 22).

A given row *n* will always be binned with row *n* + 2 for 2X subsampling mode and row n+4 for 4X subsampling mode. Therefore, there are two candidate rows that a row can be binned with, depending upon the alignment of y_addr_start.

For a given column *n*, there is only one other column, n_bin, that is can be binned with. Since the x_addr_start is restricted to multiple of 8 a column *n* will also always we binned with column *n* + 2 for 2X subsampling mode and column *n* + 4 for 4X subsampling mode.

Scaler

The MT9E001 sensor includes scaling capabilities. This allows the user to generate full field of view, low resolution images. Scaling is advantageous because it uses all pixel values to calculate the output image which helps to avoid aliasing. It is also more convenient than binning because the scale factor varies smoothly and the user is not limited to certain ratios of size resolution.

The scaling factor is programmable in 1/16th steps.

(EQ 11)

$$ScaleFactor = \frac{scale\_n}{scale\_m} = \frac{16}{scale\_m}$$

n is fixed at 16.

m is adjustable with R0x0404 (scale_m)

Legal values for *m* are 16 through 256. The user has the ability to scale from 1:1 (m = 16) to 1:8 (m = 256).

Shading Correction (SC)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing fixed pattern signal gradients in images captured by image sensors. The cumulative result of all these factors is known as image shading. The MT9E001 has an embedded shading correction module that can be programmed to counter the shading effects on each individual Red, GreenR, GreenB, and Blue color signal.

The Correction Function

Color dependent solutions are calibrated using the sensor, lens system, and an image of an evenly illuminated, featureless gray calibration field. From the resulting image the color correction functions can be derived.

The correction functions can then be applied to each pixel value to equalize the response across the image as follows:

$$P_{corrected}(row,col) = P_{sensor}(row,col) * f(row,col) \quad \text{(EQ 12)}$$

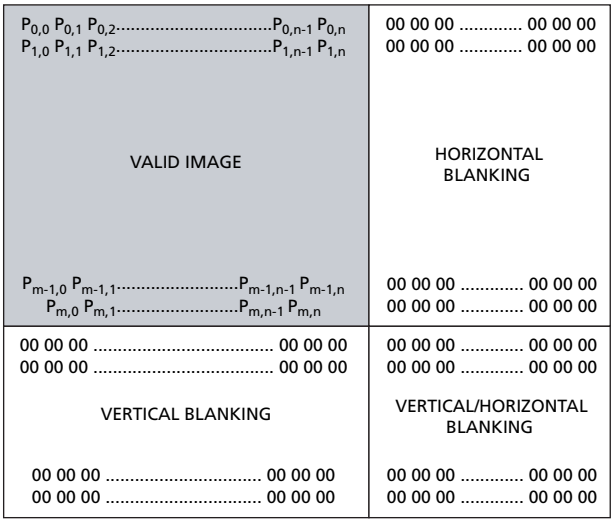
where *P* are the pixel values and *f* is the color dependent correction functions for each color channel.

Each function includes a set of color dependent coefficients defined by registers R0x3600–3726. The function's origin is the center point of the function used in the calculation of the coefficients. Using an origin near the central point of symmetry of the sensor response provides the best results. The center point of the function is determined by ORIGIN_C (R0x3782) and ORIGIN_R (R0x3784) and can be used to counter an offset in the system lens from the center of the sensor array.

Output Data Format (Parallel Pixel Data Interface)

The sensor image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking as shown in Figure 19. The amount of horizontal blanking and vertical blanking is programmable. LINE_VALID is HIGH during the shaded region of the figure. FRAME_VALID timing is described in the next section.

Figure 19: Pixel Data Timing Example



Output Data Timing (Parallel Pixel Data Interface)

The sensor core output data is synchronized with the PIXCLK output. When LINE_VALID is HIGH, one pixel data is output on the 12-bit DOUT output every PIXCLK period. By default, the sensor master input clock (vt_pix_clk_mhz) is set up as the 192 MHz clock. Hence, the output clock (op_pix_clk_mhz) is set up as half the sensor master input clock (vt_pix_clk_mhz). The rising edges on the PIXCLK signal occur one-half of a pixel clock period after transitions on LINE_VALID, FRAME_VALID, and DOUT (Figure 20). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking periods. The sensor can be programmed to delay the PIXCLK edge relative to the DOUT transitions. This can be achieved by programming the corresponding bits in the row_speed register. The parameters P, A, and Q in Figure 21 on page 29 are defined inTable 8 on page 29.

Figure 20: Pixel Data Timing Example

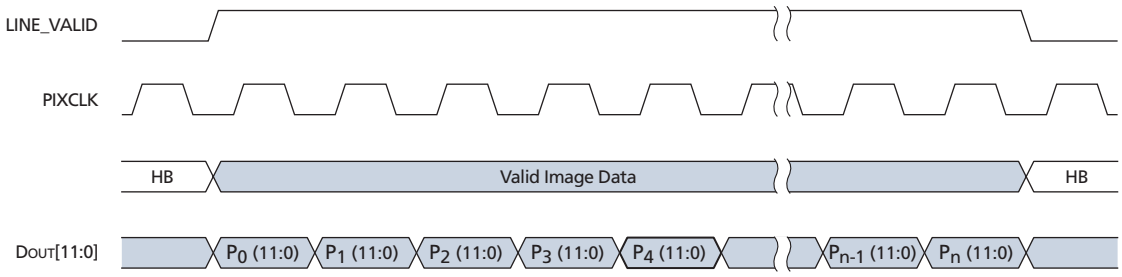


Figure 21: Row Timing and FRAME_VALID/LINE_VALID Signals

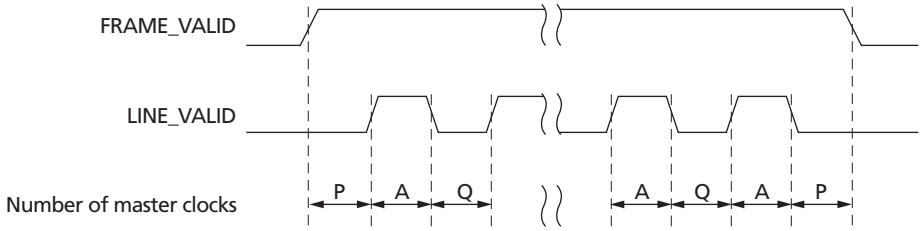


Table 8: Row Timing Parameters

Parameter	Name	Equation	Default Timing
PIXCLK_PERIOD	Pixel clock period	R0x3016-7[2:0] / vt_pix_clk_freq_mhz	1 pixel clock = 5.2ns
S	Skip (subsampling) factor	x_odd_inc = y_odd_inc = 3, S = 2 x_odd_inc = y_odd_inc = 7, S = 4 otherwise, S = 1	1
A	Active data time	(x_addr_end - x_addr_start + x_odd_inc) * PIXCLK_PERIOD/S	3264 pixel clocks = 17.0µs
P	Frame start/end blanking	12 * PIXCLK_PERIOD	12 pixel clocks = 62.5ns
Q	Horizontal blanking	(line_length_pck - A) * PIXCLK_PERIOD	6558 - 3264 pixel clocks = 17.16µs
A + Q	Row time	line_length_pck * PIXCLK_PERIOD	6558 pixel clocks = 34.16µs
N	Number of rows	(y_addr_end - y_addr_start + y_odd_inc)/S	2448 rows
V	Vertical blanking	((frame_length_lines - N) * (A+Q)) + Q - (2*P)	737,766 pixel clocks = 3.84ms
N * (A+Q)	Frame valid time	(N * (A + Q)) - Q + (2*P)	16,050,714 pixel clocks = 83.60ms
F	Total frame time	line_length_pck * frame_length_lines * PIXCLK_PERIOD	16,788,480 pixel clocks = 87.44ms

Notes: 1. This table if for internal frame timing.What is seen on the output will be determined by the op_pix_clk and x_output_size, y_output_size, and the scaling factor. This table is only true when no scaling, op_pix_clk = vt_pix_clk/2 and x/y_output_size set to caputre the read out image.

The sensor timing (Table 8) is shown in terms of pixel clock and master clock cycles (Figure 19 on page 28). The default settings for the on-chip PLL generate a 9 6MHz master input clock and pixel clock given a 24 MHz input clock to the sensor.

Table 9: Row Timing Constants

	No Row Binning			Row Binning			
row_speed[2:0]	1	2	4	1	2	4	
min_line_blanking_pck	6ac	3ea	289	c40	6a4	3d6	(hex)
min_line_length_pck	910	488	244	1200	900	480	(hex)
fine_integration_time_min	56a	2c6	c2	b1a	59e	178	(hex)
fine_int_time_max_magin	3aa	1ea	8a	95a	58e	140	(hex)
fine_correction	100	7a	37	238	116	85	(hex)
min_frame_blanking_lines	55			53			(hex)
Output overhead	1e						(hex)

Table 10: Frame Rate Calculations

Resolution	Frame Rate	scale_m (ROx 0404)	x_addr_start (ROx 3004)	x_addr_end (ROx 3008)	y_addr_start (ROx 3002)	y_addr_end (ROx 3006)	coarse_integration_time (ROx 3012)	fine_integration_time_ (ROx3014)	xybin (ROx3040[10])	x_odd_inc (ROx 3040 [7:5])	y_odd_inc (ROx 3040 [4:2])	line_length_pck (ROx 300C)	frame_length_lines (ROx 300A)	x_output_size (ROx 034C)	y_output_size (ROx 034E)
Full (3264x2448)	11.4	16	0	3263	8	2455	16	1386	1	1	0	6558	2560	3264	2448
6MP (2592x1944)	11.4	20	0	3263	8	2455	16	1386	1	1	0	6558	2560	2592	1944
xMP (2048x1536)	11.4	25	0	3263	8	2455	16	1386	1	1	0	6558	2560	2048	1536
xMP (1600x1200)	11.4	32	0	3263	8	2455	16	1386	1	1	0	6558	2560	1600	1200
Video															
1280x960	30.8	20	0	3257	8	2449	568	2842	1	3	3	4768	1307	1280	960
1024x768	30.8	24	0	3257	8	2449	568	2842	1	3	3	4608	695	1024	768
640x480	60	20	0	3257	8	2449	556	2842	1	7	7	4768	1307	640	480
720p	30.8	20	0	3257	8	2449	200	2842	1	3	3	4768	1307	1280	720

General Purpose Inputs

The sensor provides four general purpose inputs; before reset they are in an unknown state. After reset, the input pads associated with these signals are powered-down by default, allowing the pads to be left disconnected/floating.

The general purpose inputs are enabled by setting reset_register[8] (0x301A[8]). Once enabled, all four inputs must be driven to valid logic levels by external signals. The state of the general purpose inputs can be read through gpi_status (0x3026[3:0]).

In addition, each of the following functions can be associated with none, one or more of the general purpose inputs so that the function can be directly controlled by a hardware input:

- output enable (see "Output Enable Control" on page 31)
- SADDR (selects device address for the two-wire serial interface)
- trigger (see the sections below)
- standby functions (see the sections below)

The gpi_status register (0x3026) is used to associate a function with a general purpose input.

Parallel Pixel Data Interface

The parallel pixel data interface uses the following output-only signals:

- FRAME_VALID
- LINE_VALID
- PIXCLK
- DOUT[11:0]

The parallel pixel data interface is disabled by default at power-up and after reset. It can be enabled by programming R0x301A.

Output Enable Control

When the parallel pixel data interface is enabled, its signals can be switched asynchronously between the driven and High-Z—this is controlled either by pin or register control, as shown in Table 11.

Table 11: Output Enable Control

GPI Configured OE_N Pin	Drive Signals R0x301A–B[6]	Description
disabled	0	Interface High-Z
disabled	1	Interface driven
1	0	Interface High-Z
X	1	Interface driven
0	X	Interface driven

Trigger Control

When the global reset feature is in use, the trigger for the sequence can be initiated either under pin or register control, as shown in Table 12 on page 32.

Table 12: Trigger Control

GPI Configured TRIGGER Pin	Global Trigger R0x3160–1[0]	Description
Disabled	0	Idle
Disabled	1	Trigger
0	0	Idle
X	1	Trigger
1	X	Trigger

Streaming/Standby Control

The sensor can be switched between its soft standby and streaming states under pin or register control, as shown in the Table 13 above. Selection of a pin to use for the STANDBY function is described in "General Purpose Inputs" on page 31. The state diagram for transitions between soft standby and streaming states is shown in the Figure 34 on page 87.

Table 13: Streaming/STANDBY

GPI Configured STANDBY Pin	Streaming R0x301A–B[2]	Description
Disabled	0	Soft standby
Disabled	1	Streaming
X	0	Soft standby
0	1	Streaming
1	X	Soft standby

Operational Modes

Snapshot and Flash

The sensor supports both Xenon and LED flash through the FLASH output signal. The timing of the FLASH signal with the default settings is shown in Figure 22 on page 33 through Figure 24 on page 33. The flash and flash_count registers allow the timing of the flash to be changed. The flash can be programmed to fire only once, to be delayed by a few frames when asserted, and (for Xenon flash) to program the flash duration.

Enabling the LED flash will cause one bad frame, where several of the rows only have the flash on for part of their integration time. This can be avoided by forcing a restart of the frame (write reset_register[1] = 1) immediately after enabling the flash; the first bad frame will then be masked out as shown in Figure 24 on page 33. Read-only bit flash[14] is set during frames that are correctly integrated; the state of this bit is shown in Figures 22 through Figure 24 on page 33.

Figure 22: Xenon Flash Enabled

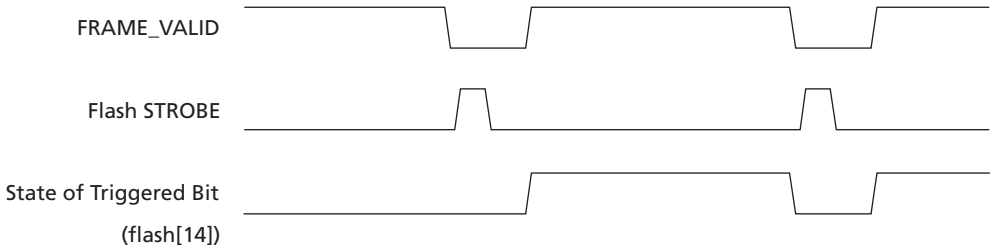


Figure 23: LED Flash Enabled

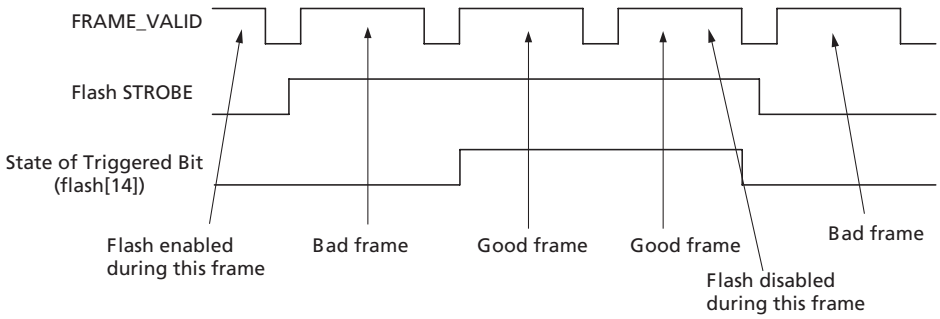
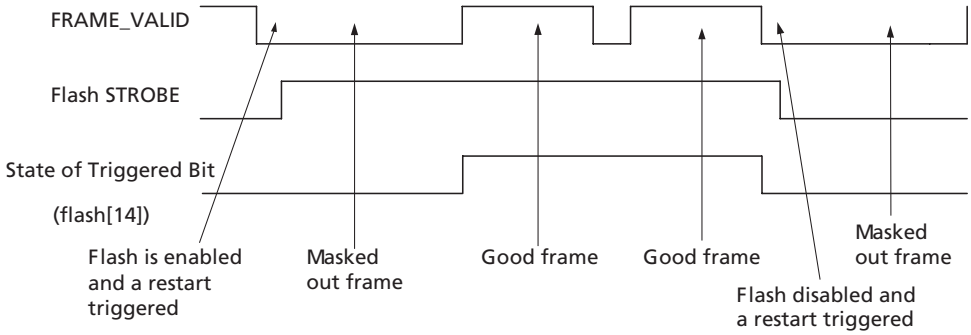


Figure 24: LED Flash Enabled Following Forced Restart



Low Power Mode

The sensor supports a low-power mode by programming register bit read_mode[9]. Setting this bit will result in the following:

- Double the value of pc_speed[2:0] internally. This means halving the internal pixel clock frequency.

The slower pixel clock provides more time for settling in the analog domain, thus, the low power DAC values can be approximately half the full power DAC values.

Enabling the low power mode will not put the sensor in subsampling mode; this has to be programmed separately as described earlier in this document. Low power is independent of the readout mode, and can also be enabled in full resolution mode. However, since the pixel clock speed is halved, the frame rates that can be achieved with low power mode are lower than in full power mode.

Only internal pixel clock speeds of 1, 2, and 4 are supported; therefore, low power mode combined with pc_speed[2:0] = 4 is an illegal combination.

Any limitations related to changing the internal pixel clock speed will also apply to low power mode since it automatically changes the pixel clock speed. Therefore, SMIA parameter limit registers need to be reprogrammed to match the new internal pixel clock frequency.

Test Patterns

For test purposes, pixel data can be replaced with a fixed image generated by a special test module in the pipeline. The module provides a selection of test patterns sufficient for basic testing of the signal chain.

Test patterns are accessible using R0x0600 and are shown in Table 14.

Table 14: Test Patterns

Test Pattern	Register Value
Normal Operation: no test pattern	0
Flat Field	1
Color Bar	2
Fade-to-Gray Color Bar	3
Marching 1s	256

Two-Wire Serial Interface

The two-wire serial interface bus enables read/write access to control and status registers within the sensor. This interface is designed to be compatible with the "SMIA 1.0 Part2: CCP2 Specification Camera Control Interface (CCI)," that uses the electrical characteristics and transfer protocols of the two-wire serial interface specification.

The interface protocol uses a master/slave model in which a master controls one or more slave devices. The sensor acts as a slave device. The master generates a clock (SCLK) that is an input to the sensor and used to synchronize transfers. Data is transferred between the master and the slave on a bidirectional signal (SDATA). SDATA is pulled up to VDD off-chip by a 1.5KΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The protocols described in the two-wire serial interface specification allow the slave device to drive SCLK LOW; the sensor uses SCLK as an input only; therefore, never drives it LOW.

Protocol

Data transfers on the two-wire serial interface bus are performed by a sequence of low-level protocol elements, as follows:

- a (repeated) start condition
- a slave address/data direction byte
- an (a no) acknowledge bit
- a message byte
- a stop condition

The bus is idle when both SCLK and SDATA are HIGH. Control of the bus is initiated with a start condition, and the bus is released with a stop condition. Only the master can generate the start and stop conditions.

Start Condition

A start condition is defined as a HIGH-to-LOW transition on SDATA while SCLK is HIGH. At the end of a transfer, the master can generate a start condition without previously generating a stop condition; this is known as a repeated start or restart condition.

Stop Condition

A stop condition is defined as a LOW -to-HIGH transition on SDATA while SCLK is HIGH.

Data Transfer

Data is transferred serially, 8 bits at a time, with the MSB transmitted first. Each byte of data is followed by an acknowledge bit or a no-acknowledge bit. This data transfer mechanism is used for the slave address/data direction byte and for message bytes.

One data bit is transferred during each SCLK clock period. SDATA can change when SCLK is LOW and must be stable while SCLK is HIGH.

Slave Address/Data Direction Byte

Bits [7:1] of this byte represent the device slave address and bit [0] indicates the data transfer direction. A "0" in bit [0] indicates a write, and a "1" indicates a read. The default slave addresses used by the sensor are 0x20 (write address) and 0x21 (read address), in accordance with the SMIA specification. Alternate slave addresses of 0x30 (write

address) and 0x31 (read address) can be selected. The GPI pins can be configured for SADDR functionality through register bit fields 0x3026[6:4], and enabled by setting 0x301A[8].

These default slave addresses are also fully programmable through the I^2C address registers (0x31FC). Before this register can be written to, it needs to be unlocked through reset_register 0x301A[3].

Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data. The protocol used is outside the scope of the two-wire serial interface specification and is defined as part of the SMIA CCI.

Acknowledge Bit

Each 8-bit data transfer is followed by an acknowledge bit or a no-acknowledge bit in the SCLK clock period following the data transfer. The transmitter (which is the master when writing, or the slave when reading) releases SDATA. The receiver indicates an acknowledge bit by driving SDATA LOW. As for data transfers, SDATA can change when SCLK is LOW and must be stable while SCLK is HIGH.

No-Acknowledge Bit

The no-acknowledge bit is generated when the receiver does not drive SDATA LOW during the SCLK clock period following a data transfer. A no-acknowledge bit is used to terminate a read sequence.

Typical Serial Transfer

A typical read or write sequence begins by the master generating a start condition on the bus. After the start condition, the master sends the 8-bit slave address/data direction byte. The last bit indicates whether the request is for a read or a write, where a “0” indicates a write and a “1” indicates a read. If the address matches the address of the slave device, the slave device acknowledges receipt of the address by generating an acknowledge bit on the bus.

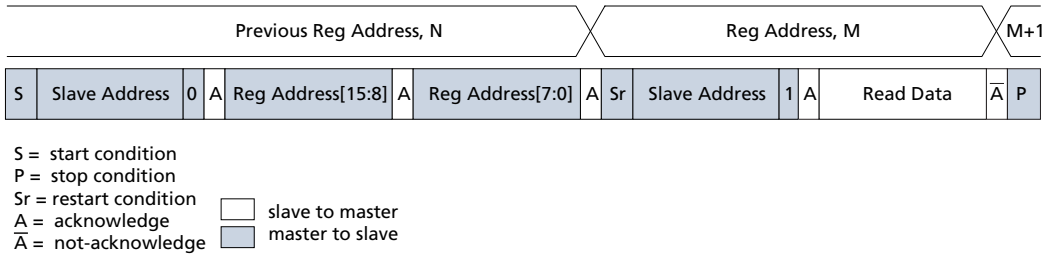
If the request was a write, the master then transfers the 16-bit register address to which the write should take place. This transfer takes place as two, 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the data as an 8-bit sequence; the slave sends an acknowledge bit at the end of the sequence. After 8 bits have been transferred, the slave’s internal register address is incremented automatically, so that the next 8 bits are written to the next register address. The master stops writing by generating a (re)start or stop condition.

If the request was a read, the master sends the 8-bit write slave address/data direction byte and 16-bit register address, the same way as with a write request. The master then generates a (re)start condition and the 8-bit read slave address/data direction byte, and clocks out the register data, 8 bits at a time. The master generates an acknowledge bit after each 8-bit transfer. The slave’s internal register address is auto-incremented after every 8 bits are transferred. The data transfer is stopped when the master sends a no-acknowledge bit.

Single READ from Random Location

This sequence (Figure 25) starts with a dummy WRITE to the 16-bit address that is to be used for the READ. The master terminates the WRITE by generating a restart condition. The master then sends the 8-bit read slave address/data direction byte and clocks out one byte of register data. The master terminates the READ by generating a no-acknowledge bit followed by a stop condition. Figure 25 shows how the internal register address maintained by the sensor is loaded and incremented as the sequence proceeds.

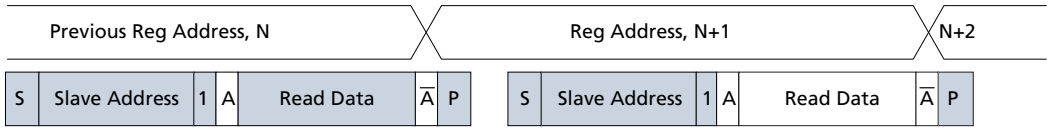
Figure 25: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 26) performs a read using the current value of the sensor internal register address. The master terminates the READ by generating a no-acknowledge bit followed by a stop condition. The figure shows two independent READ sequences.

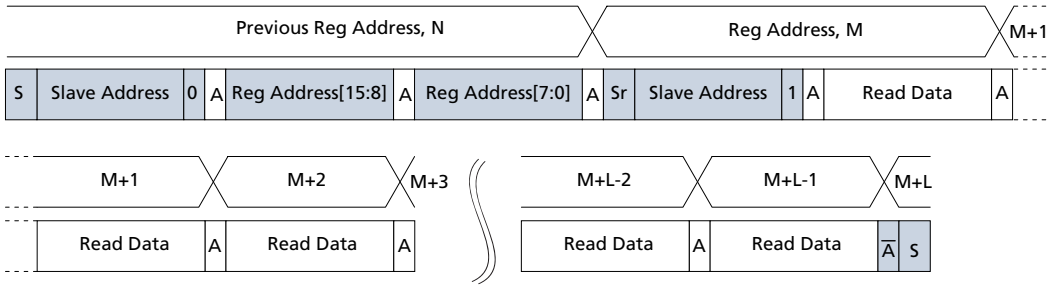
Figure 26: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 27) starts in the same way as the single READ from random location (Figure 25). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until "L" bytes have been read.

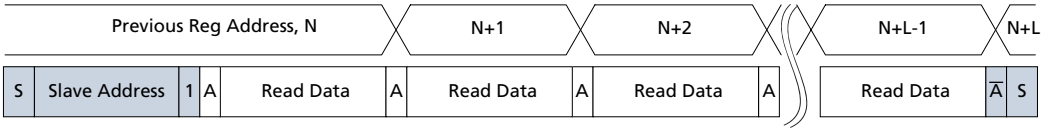
Figure 27: Sequential READ, Start from Random Location



Sequential READ, Start from Current Location

This sequence (Figure 28) starts in the same way as the single READ from current location (Figure 26 on page 37). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until "L" bytes have been read.

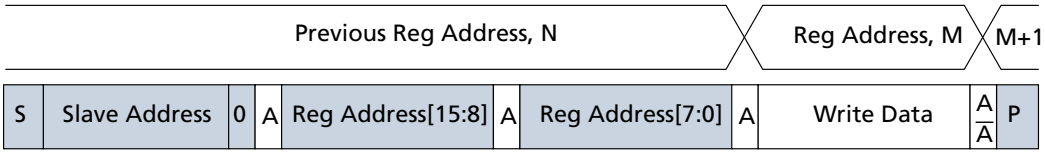
Figure 28: Sequential READ, Start from Current Location



Single WRITE to Random Location

This sequence (Figure 29) begins with the master generating a start condition. The slave address/data direction byte signals a WRITE and is followed by the HIGH then LOW bytes of the register address that is to be written. The master follows this with the byte of write data. The WRITE is terminated by the master generating a stop condition.

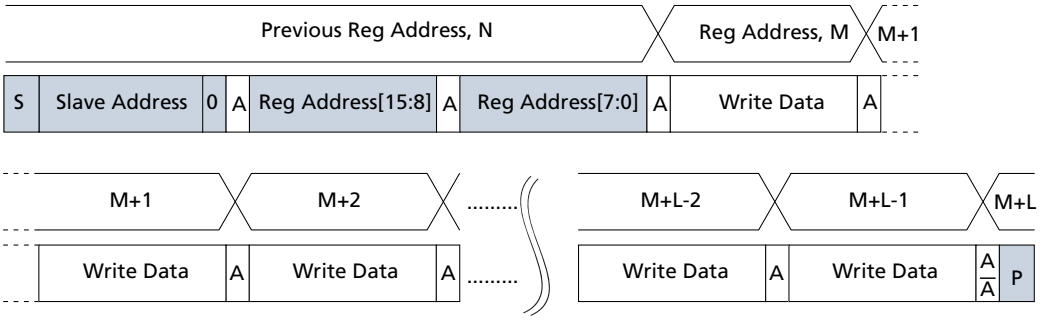
Figure 29: Single WRITE to Random Location



Sequential WRITE, Start at Random Location

This sequence (Figure 30) starts in the same way as the single WRITE to random location (Figure 29). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte WRITEs until "L" bytes have been written. The WRITE is terminated by the master generating a stop condition.

Figure 30: Sequential WRITE, Start at Random Location



Registers

The sensor provides a 16-bit register address space accessed through a serial interface. Each register location is 8 bits in size.

The address space is divided into the five major regions shown in Table 15.

Table 15: Address Space Regions

Address Regions	Description
0x0000–0x0FFF	Configuration registers (Read-only and read-write dynamic registers)
0x1000–0x1FFF	Parameter limit registers (Read-only static registers)
0x2000–0x2FFF	Reserved (Undefined)
0x3000–0x3FFF	Manufacturer specific registers (Read-only and read-write dynamic registers)
0x4000–0xFFFF	Reserved (Undefined)

Register Notation

The underlying mechanism for reading and writing registers provides byte write capability. However, it is convenient to consider some registers as multiple adjacent bytes. The sensor uses 8-bit, 16-bit, and 32-bit registers; all implemented as 1 or more bytes at naturally aligned, contiguous locations in the address space.

Registers are described either by address or by name. When registers are described by address, the size of the registers is explicit. For example, R0x3024 is an 8-bit register at address 0x3024, and R0x3000–1 is a 16-bit register at address 0x3000–0x3001. When registers are described by name, the size of the register is not implicit. For example, it is necessary to refer to the register table to determine that model_id is a 16-bit register.

Register Aliases

A consequence of the internal architecture of the sensor is that some registers are decoded at multiple addresses: some registers in configuration space are also decoded in manufacturing specific space. In order to provide unique names for all registers, the name of the register within manufacturer specific register space has a trailing underscore. For example, R0x0000–1 is model_id, and R0x3000–1 is model_id_ (see the register tables for more examples). The effect of reading or writing a register to itself or through any of its aliases is identical.

Bit Fields

Some registers provide control of several different pieces of related functionality and this makes it necessary to refer to bit fields within registers. As an example of the notation used for this, the least significant 4 bits of the model_id register are referred to as model_id[3:0] or R0x0000–1[3:0].

Bit Field Aliases

In addition to the register aliases described above, some register fields are aliased in multiple places. For example, R0x0100 (mode_select) only has one operational bit: R0x0100[0]. This bit is aliased to R0x3001A–B[2]. The effect of reading or writing a bit field through any of its aliases is identical.

Byte Ordering

Registers that occupy more than one byte of address space are shown with the lowest address in the highest-order byte lane, to match the byte-ordering on the SMIA bus. For example, the model_id register is R0x0000–1. In the register table its default value is shown as 0x2B00. This means that a read from address 0x0000 would return 0x2B and a read from address 0x0001 would return 0x00. When reading this register as two 8-bit transfers on the serial interface, the 0x2B will appear on the serial interface first, followed by the 0x00.

Address Alignment

All register addresses are naturally-aligned: registers that occupy two bytes of address space are aligned to even 16-bit addresses, and registers that occupy four bytes of address space are aligned to 16-bit addresses that are an integer multiple of 4.

Bit Representation

For clarity, 32-bit hex numbers are shown with an underscore between the upper and lower sixteen bits. For example: 0x3000_01AB.

Data Format

Most registers represent an unsigned binary value or set of bit fields. For all other register formats, the format is stated explicitly at the start of the register description. The notation for these formats is shown in Table 16.

Table 16: Data Formats

Name	Description
FIX16	Signed fixed-point 16-bit number: two's complement number, 8 fractional bits. Examples: 0x0100 = 1.0, 0x8000 = -128, 0xFFFF = -0.0039065
UFIX16	Unsigned fixed-point 16-bit number: 8.8 format. Examples: 0x0100 = 1.0, 0x280 = 2.5
FLP32	Example: 0x4280_0000 = 64.0

Register Behavior

Registers vary from "read-only," "read/write," and "read, write-1-to-clear."

Double-Buffered Registers

Some sensor settings cannot be changed during frame readout. For example, changing R0x0344–5 (x_addr_start) partway through frame readout would result in inconsistent row lengths within a frame. To avoid this, the sensor double buffers many registers by implementing a "pending" and a "live" version. Reads and writes access the pending register. The live register controls the sensor operation.

The value in the pending register is transferred to a live register at a fixed point in the frame timing, called frame start. Frame start is defined as the point at which the first dark row is read out internally to the sensor. By default, this occurs 82 row times before FRAME_VALID goes HIGH. R0x3044–5 enables the dark rows to be shown in the image, but this has no effect on the position of frame-start. In the register tables the "Frame Synced" field shows which registers or register fields are double-buffered in this way.

Using grouped_parameter_hold

The grouped_parameter_hold (R0x0104) can be used to inhibit transfers from the pending to the live registers. When the sensor is in streaming mode, this register should be written to “1” before making changes to any multi-byte registers or any group of registers where a set of changes is required to take effect simultaneously. When this register is written to “0,” all transfers from pending to live registers take place on the next frame start.

An example of the consequences of failing to set this bit follows:

The coarse integration time is controlled by a 16-bit register. If the integration time is changed from 0x00FF to 0x0100 and the writes of 0x01, 0x00 (the two bytes used to set the new integration time) occur during a frame start, the first byte could be seen and transferred to the live register one frame sooner than the second byte. Instead of seeing successive frames integrated at 0x00FF, 0x0100, 0x0100, 0x0100, successive frames would be integrated at 0x00FF, 0x01FF, 0x0100, 0x0100.

Bad Frames

A bad frame is defined as a frame where all rows do not have the same integration time, or where offsets to the pixel values have changed during the frame.

Many changes to the sensor register settings can cause a bad frame. For example, when line_length_pck (R0x0342–3) is changed, the new register value does not affect sensor behavior until the next frame start. However, the frame that would be read out at that frame start will have been integrated using the old row width, so reading it out using the new row width would result in a frame with an incorrect integration time.

By default, bad frames are not masked. However, when bad frames are masked (0x301A[9]), LINE_VALID and FRAME_VALID are inhibited for these frames so that the vertical blanking time between frames is extended by the frame time.

In the register tables, the "Cause Bad Frame" field shows where changing a register or register field will cause a bad frame. The following notation is used:

- False:Changing the register value will not produce a bad frame.
- True:Changing the register value might produce a bad frame.
- Dropped: As true, but the bad frame will be masked out when mask_corrupted_frames (R0x0105) is set to "1."

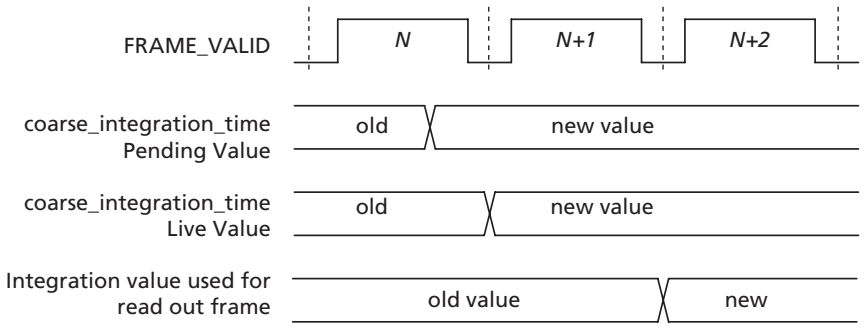
Changes to Integration Time

If the integration time is changed while FRAME_VALID is asserted for frame *n*, the first frame output using the new integration time is frame *(n + 2)*. The sequence is as follows:

1. During frame *n*, the new integration time is held in the pending register.
2. At the start of frame *(n + 1)*, the new integration time is transferred to the live register. Integration for each row of frame *(n + 1)* has been completed using the old integration time.
3. The earliest time that a row can start integrating using the new integration time is immediately after that row has been read for frame *(n + 1)*. The actual time that rows start integrating using the new integration time is dependent upon the new value of the integration time.
4. When frame *(n + 1)* is read out, the next frame will have been integrated using the new integration time.

If the integration time is changed on successive frames, each value written will be applied for a single frame; the latency between writing a value and it affecting the frame readout remains at two frames.

Figure 31: Changes to Integration Time



Changes to Gain Settings

Usually, when the gain settings are changed, the gain is updated on the next frame start as is shown in Figure 32. When the integration time and the gain are changed at the same time, the gain update is held off by one frame so that the first frame output with the new integration time also has the new gain applied (Figure 33 on page 43).

If the gain and integration time are both changed on successive frames, some gain values will be overwritten without ever being applied, while each integration time will be used for one single frame.

Figure 32: Changes to Gain

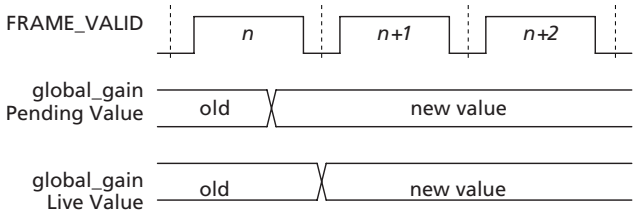
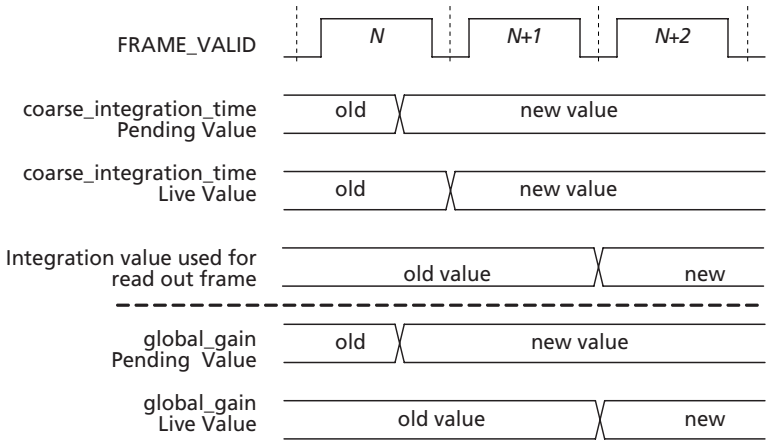


Figure 33: Changes to Gain and Integration Time



Embedded Data

The current values of implemented registers in the address range 0x0000–0x0FFF can be generated as part of the pixel data. This embedded data is enabled by default.

The current value of a register is the value that was used for the image data in that frame. In general, this is the live value of the register. The exceptions are:

- The integration time is delayed by one further frame, so that the value corresponds to the integration time used for the image data in the frame. See "Changes to Integration Time" on page 42.
- The PLL timing registers are not double-buffered, since the result of changing them in streaming mode is UNDEFINED. Therefore, the pending and live values for these registers are equivalent.

Register List and Default Value

Table 17: SMIA Configuration

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R0(R0x0000)	model_id	dddd dddd dddd dddd	11008 (0x2B00)
R2(R0x0002)	revision_number	dddd dddd	0 (0x0000)
R3(R0x0003)	manufacturer_id	???? ????	6 (0x0006)
R4(R0x0004)	smia_version	???? ????	10 (0x000A)
R5(R0x0005)	frame_count	???? ????	255 (0x00FF)
R6(R0x0006)	pixel_order	0000 00??	0 (0x0000)
R8(R0x0008)	data_pedestal	0000 dddd dddd dddd	168 (0x00A8)
R64(R0x0040)	frame_format_model_type	???? ????	1 (0x0001)
R65(R0x0041)	frame_format_model_subtype	???? ????	18 (0x0012)
R66(R0x0042)	frame_format_descriptor_0	???? ???? ???? ????	23744 (0x5CC0)
R68(R0x0044)	frame_format_descriptor_1	???? ???? ???? ????	4098 (0x1002)
R70(R0x0046)	frame_format_descriptor_2	???? ???? ???? ????	22928 (0x5990)
R72(R0x0048)	frame_format_descriptor_3	???? ???? ???? ????	0 (0x0000)
R74(R0x004A)	frame_format_descriptor_4	???? ???? ???? ????	0 (0x0000)
R76(R0x004C)	frame_format_descriptor_5	???? ???? ???? ????	0 (0x0000)
R78(R0x004E)	frame_format_descriptor_6	???? ???? ???? ????	0 (0x0000)
R80(R0x0050)	frame_format_descriptor_7	???? ???? ???? ????	0 (0x0000)
R82(R0x0052)	frame_format_descriptor_8	???? ???? ???? ????	0 (0x0000)
R84(R0x0054)	frame_format_descriptor_9	???? ???? ???? ????	0 (0x0000)
R86(R0x0056)	frame_format_descriptor_10	???? ???? ???? ????	0 (0x0000)
R88(R0x0058)	frame_format_descriptor_11	???? ???? ???? ????	0 (0x0000)
R90(R0x005A)	frame_format_descriptor_12	???? ???? ???? ????	0 (0x0000)
R92(R0x005C)	frame_format_descriptor_13	???? ???? ???? ????	0 (0x0000)
R94(R0x005E)	frame_format_descriptor_14	???? ???? ???? ????	0 (0x0000)
R128(R0x0080)	analogue_gain_capability	???? ???? ???? ????	1 (0x0001)
R132(R0x0084)	analogue_gain_code_min	???? ???? ???? ????	8 (0x0008)
R134(R0x0086)	analogue_gain_code_max	???? ???? ???? ????	127 (0x007F)
R136(R0x0088)	analogue_gain_code_step	???? ???? ???? ????	1 (0x0001)
R138(R0x008A)	analogue_gain_type	???? ???? ???? ????	0 (0x0000)
R140(R0x008C)	analogue_gain_m0	???? ???? ???? ????	1 (0x0001)
R142(R0x008E)	analogue_gain_c0	???? ???? ???? ????	0 (0x0000)
R144(R0x0090)	analogue_gain_m1	???? ???? ???? ????	0 (0x0000)
R146(R0x0092)	analogue_gain_c1	???? ???? ???? ????	8 (0x0008)
R192(R0x00C0)	data_format_model_type	???? ????	1 (0x0001)
R193(R0x00C1)	data_format_model_subtype	???? ????	5 (0x0005)
R194(R0x00C2)	data_format_descriptor_0	???? ???? ???? ????	2570 (0x0A0A)
R196(R0x00C4)	data_format_descriptor_1	???? ???? ???? ????	2056 (0x0808)

Table 17: SMIA Configuration (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R198(R0x00C6)	data_format_descriptor_2	???? ???? ???? ????	2568 (0x0A08)
R200(R0x00C8)	data_format_descriptor_3	???? ???? ???? ????	3084 (0x0C0C)
R202(R0x00CA)	data_format_descriptor_4	???? ???? ???? ????	3080 (0x0C08)
R204(R0x00CC)	data_format_descriptor_5	???? ???? ???? ????	0 (0x0000)
R206(R0x00CE)	data_format_descriptor_6	???? ???? ???? ????	0 (0x0000)
R256(R0x0100)	mode_select	0000 000d	0 (0x0000)
R257(R0x0101)	image_orientation	0000 00dd	0 (0x0000)
R259(R0x0103)	software_reset	0000 000d	0 (0x0000)
R260(R0x0104)	grouped_parameter_hold	0000 000d	0 (0x0000)
R261(R0x0105)	mask_corrupted_frames	0000 000d	0 (0x0000)
R272(R0x0110)	Reserved	0000 0ddd	0 (0x0000)
R273(R0x0111)	Reserved	0000 000d	0 (0x0000)
R274(R0x0112)	ccp_data_format	0000 ddd0 0000 ddd0	3084 (0x0C0C)
R288(R0x0120)	gain_mode	0000 000d	0 (0x0000)
R512(R0x0200)	fine_integration_time	dddd dddd dddd ddd0	1386 (0x056A)
R514(R0x0202)	coarse_integration_time	dddd dddd dddd dddd	16 (0x0010)
R516(R0x0204)	analogue_gain_code_global	0000 0000 0ddd dddd	13 (0x000D)
R518(R0x0206)	analogue_gain_code_greenR	0000 0000 0ddd dddd	13 (0x000D)
R520(R0x0208)	analogue_gain_code_red	0000 0000 0ddd dddd	13 (0x000D)
R522(R0x020A)	analogue_gain_code_blue	0000 0000 0ddd dddd	13 (0x000D)
R524(R0x020C)	analogue_gain_code_greenB	0000 0000 0ddd dddd	13 (0x000D)
R526(R0x020E)	digital_gain_greenR	0000 0ddd 0000 0000	256 (0x0100)
R528(R0x0210)	digital_gain_red	0000 0ddd 0000 0000	256 (0x0100)
R530(R0x0212)	digital_gain_blue	0000 0ddd 0000 0000	256 (0x0100)
R532(R0x0214)	digital_gain_greenB	0000 0ddd 0000 0000	256 (0x0100)
R768(R0x0300)	vt_pix_clk_div	0000 0000 0000 dddd	4 (0x0004)
R770(R0x0302)	vt_sys_clk_div	0000 0000 000d dddd	1 (0x0001)
R772(R0x0304)	pre_pll_clk_div	0000 0000 00dd dddd	2 (0x0002)
R774(R0x0306)	pll_multiplier	0000 0000 dddd dddd	64 (0x0040)
R776(R0x0308)	op_pix_clk_div	0000 0000 000d dddd	8 (0x0008)
R778(R0x030A)	op_sys_clk_div	0000 0000 000d dddd	1 (0x0001)
R832(R0x0340)	frame_length_lines	dddd dddd dddd dddd	2560 (0x0A00)
R834(R0x0342)	line_length_pck	dddd dddd dddd ddd0	6558 (0x199E)
R836(R0x0344)	x_addr_start	0000 dddd dddd dddd	0 (0x0000)
R838(R0x0346)	y_addr_start	0000 dddd dddd dddd	8 (0x0008)
R840(R0x0348)	x_addr_end	0000 dddd dddd dddd	3263 (0x0CBF)
R842(R0x034A)	y_addr_end	0000 dddd dddd dddd	2455 (0x0997)
R844(R0x034C)	x_output_size	0000 dddd dddd ddd0	3264 (0x0CC0)
R846(R0x034E)	y_output_size	0000 dddd dddd ddd0	2448 (0x0990)

Table 17: SMIA Configuration (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R896(R0x0380)	x_even_inc	0000 0000 0000 000?	1 (0x0001)
R898(R0x0382)	x_odd_inc	0000 0000 0000 0ddd	1 (0x0001)
R900(R0x0384)	y_even_inc	0000 0000 0000 000?	1 (0x0001)
R902(R0x0386)	y_odd_inc	0000 0000 0000 0ddd	1 (0x0001)
R1024(R0x0400)	scaling_mode	0000 0000 0000 00dd	0 (0x0000)
R1026(R0x0402)	spatial_sampling	0000 0000 0000 000d	0 (0x0000)
R1028(R0x0404)	scale_m	0000 0000 dddd dddd	16 (0x0010)
R1030(R0x0406)	scale_n	0000 0000 ???? ????	16 (0x0010)
R1280(R0x0500)	compression_mode	0000 0000 0000 000?	1 (0x0001)
R1536(R0x0600)	test_pattern_mode	0000 000d dddd dddd	0 (0x0000)
R1538(R0x0602)	test_data_red	0000 dddd dddd dddd	0 (0x0000)
R1540(R0x0604)	test_data_greenR	0000 dddd dddd dddd	0 (0x0000)
R1542(R0x0606)	test_data_blue	0000 dddd dddd dddd	0 (0x0000)
R1544(R0x0608)	test_data_greenB	0000 dddd dddd dddd	0 (0x0000)
R1546(R0x060A)	horizontal_cursor_width	0000 dddd dddd dddd	0 (0x0000)
R1548(R0x060C)	horizontal_cursor_position	0000 dddd dddd dddd	0 (0x0000)
R1550(R0x060E)	vertical_cursor_width	0000 dddd dddd dddd	0 (0x0000)
R1552(R0x0610)	vertical_cursor_position	0000 dddd dddd dddd	0 (0x0000)

Table 18: 1: SMIA Parameter Limits

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R4096(R0x1000)	integration_time_capability	???? ???? ???? ????	1 (0x0001)
R4100(R0x1004)	coarse_integration_time_min	dddd dddd dddd dddd	0 (0x0000)
R4102(R0x1006)	coarse_integration_time_max_margin	dddd dddd dddd dddd	1 (0x0001)
R4104(R0x1008)	fine_integration_time_min	dddd dddd dddd dddd	1386 (0x056A)
R4106(R0x100A)	fine_integration_time_max_margin	dddd dddd dddd dddd	938 (0x03AA)
R4224(R0x1080)	digital_gain_capability	???? ???? ???? ????	1 (0x0001)
R4228(R0x1084)	digital_gain_min	???? ???? ???? ????	256 (0x0100)
R4230(R0x1086)	digital_gain_max	???? ???? ???? ????	1792 (0x0700)
R4232(R0x1088)	digital_gain_step_size	???? ???? ???? ????	256 (0x0100)
R4352(R0x1100)	min_ext_clk_freq_mhz_1	???? ???? ???? ????	16576 (0x40C0)
R4354(R0x1102)	min_ext_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4356(R0x1104)	max_ext_clk_freq_mhz_1	???? ???? ???? ????	16960 (0x4240)
R4358(R0x1106)	max_ext_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4360(R0x1108)	min_pre_pll_clk_div	???? ???? ???? ????	1 (0x0001)
R4362(R0x110A)	max_pre_pll_clk_div	???? ???? ???? ????	64 (0x0040)

Table 18: 1: SMIA Parameter Limits (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R4364(R0x110C)	min_pll_ip_freq_mhz_1	???? ???? ???? ????	16384 (0x4000)
R4366(R0x110E)	min_pll_ip_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4368(R0x1110)	max_pll_ip_freq_mhz_1	???? ???? ???? ????	16832 (0x41C0)
R4370(R0x1112)	max_pll_ip_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4372(R0x1114)	min_pll_multiplier	???? ???? ???? ????	32 (0x0020)
R4374(R0x1116)	max_pll_multiplier	???? ???? ???? ????	256 (0x0100)
R4376(R0x1118)	min_pll_op_freq_mhz_1	???? ???? ???? ????	17344 (0x43C0)
R4378(R0x111A)	min_pll_op_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4380(R0x111C)	max_pll_op_freq_mhz_1	???? ???? ???? ????	17472 (0x4440)
R4382(R0x111E)	max_pll_op_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4384(R0x1120)	min_vt_sys_clk_div	???? ???? ???? ????	1 (0x0001)
R4386(R0x1122)	max_vt_sys_clk_div	???? ???? ???? ????	1 (0x0001)
R4388(R0x1124)	min_vt_sys_clk_freq_mhz_1	???? ???? ???? ????	17344 (0x43C0)
R4390(R0x1126)	min_vt_sys_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4392(R0x1128)	max_vt_sys_clk_freq_mhz_1	???? ???? ???? ????	17472 (0x4440)
R4394(R0x112A)	max_vt_sys_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4396(R0x112C)	min_vt_pix_clk_freq_mhz_1	???? ???? ???? ????	17088 (0x42C0)
R4398(R0x112E)	min_vt_pix_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4400(R0x1130)	max_vt_pix_clk_freq_mhz_1	???? ???? ???? ????	17216 (0x4340)
R4402(R0x1132)	max_vt_pix_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4404(R0x1134)	min_vt_pix_clk_div	???? ???? ???? ????	4 (0x0004)
R4406(R0x1136)	max_vt_pix_clk_div	???? ???? ???? ????	4 (0x0004)
R4416(R0x1140)	min_frame_length_lines	dddd dddd dddd dddd	87 (0x0057)
R4418(R0x1142)	max_frame_length_lines	dddd dddd dddd dddd	65535 (0xFFFF)
R4420(R0x1144)	min_line_length_pck	dddd dddd dddd dddd	2324 (0x0914)
R4422(R0x1146)	max_line_length_pck	dddd dddd dddd dddd	65534 (0xFFFE)
R4424(R0x1148)	min_line_blanking_pck	dddd dddd dddd dddd	1708 (0x06AC)
R4426(R0x114A)	min_frame_blanking_lines	dddd dddd dddd dddd	85 (0x0055)
R4448(R0x1160)	min_op_sys_clk_div	???? ???? ???? ????	1 (0x0001)
R4450(R0x1162)	max_op_sys_clk_div	???? ???? ???? ????	1 (0x0001)
R4452(R0x1164)	min_op_sys_clk_freq_mhz_1	???? ???? ???? ????	17344 (0x43C0)
R4454(R0x1166)	min_op_sys_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4456(R0x1168)	max_op_sys_clk_freq_mhz_1	???? ???? ???? ????	17472 (0x4440)
R4458(R0x116A)	max_op_sys_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4460(R0x116C)	min_op_pix_clk_div	???? ???? ???? ????	8 (0x0008)
R4462(R0x116E)	max_op_pix_clk_div	???? ???? ???? ????	8 (0x0008)
R4464(R0x1170)	min_op_pix_clk_freq_mhz_1	???? ???? ???? ????	16960 (0x4240)
R4466(R0x1172)	min_op_pix_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4468(R0x1174)	max_op_pix_clk_freq_mhz_1	???? ???? ???? ????	17088 (0x42C0)

Table 18: 1: SMIA Parameter Limits (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R4470(R0x1176)	max_op_pix_clk_freq_mhz_2	???? ???? ???? ????	0 (0x0000)
R4480(R0x1180)	x_addr_min	???? ???? ???? ????	0 (0x0000)
R4482(R0x1182)	y_addr_min	???? ???? ???? ????	0 (0x0000)
R4484(R0x1184)	x_addr_max	???? ???? ???? ????	3279 (0x0CCF)
R4486(R0x1186)	y_addr_max	???? ???? ???? ????	2463 (0x099F)
R4544(R0x11C0)	min_even_inc	???? ???? ???? ????	1 (0x0001)
R4546(R0x11C2)	max_even_inc	???? ???? ???? ????	1 (0x0001)
R4548(R0x11C4)	min_odd_inc	???? ???? ???? ????	1 (0x0001)
R4550(R0x11C6)	max_odd_inc	???? ???? ???? ????	3 (0x0003)
R4608(R0x1200)	scaling_capability	???? ???? ???? ????	2 (0x0002)
R4612(R0x1204)	scaler_m_min	???? ???? ???? ????	16 (0x0010)
R4614(R0x1206)	scaler_m_max	???? ???? ???? ????	128 (0x0080)
R4616(R0x1208)	scaler_n_min	???? ???? ???? ????	16 (0x0010)
R4618(R0x120A)	scaler_n_max	???? ???? ???? ????	16 (0x0010)
R4864(R0x1300)	compression_capability	???? ???? ???? ????	1 (0x0001)
R5120(R0x1400)	matrix_element_RedInRed	dddd dddd dddd dddd	578 (0x0242)
R5122(R0x1402)	matrix_element_GreenInRed	dddd dddd dddd dddd	65280 (0xFF00)
R5124(R0x1404)	matrix_element_BlueInRed	dddd dddd dddd dddd	65470 (0xFFBE)
R5126(R0x1406)	matrix_element_RedInGreen	dddd dddd dddd dddd	65460 (0xFFB4)
R5128(R0x1408)	matrix_element_GreenInGreen	dddd dddd dddd dddd	512 (0x0200)
R5130(R0x140A)	matrix_element_BlueInGreen	dddd dddd dddd dddd	65357 (0xFF4D)
R5132(R0x140C)	matrix_element_RedInBlue	dddd dddd dddd dddd	65521 (0xFFF1)
R5134(R0x140E)	matrix_element_GreenInBlue	dddd dddd dddd dddd	65332 (0xFF34)
R5136(R0x1410)	matrix_element_BlueInBlue	dddd dddd dddd dddd	476 (0x01DC)

Table 19: 3: Manufacturer Specific

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12288(R0x3000)	model_id_	dddd dddd dddd dddd	11008 (0x2B00)
R12290(R0x3002)	y_addr_start_	0000 dddd dddd dddd	8 (0x0008)
R12292(R0x3004)	x_addr_start_	0000 dddd dddd dddd	0 (0x0000)
R12294(R0x3006)	y_addr_end_	0000 dddd dddd dddd	2455 (0x0997)
R12296(R0x3008)	x_addr_end_	0000 dddd dddd dddd	3263 (0x0CBF)
R12298(R0x300A)	frame_length_lines_	dddd dddd dddd dddd	2560 (0x0A00)
R12300(R0x300C)	line_length_pck_	dddd dddd dddd ddd0	6558 (0x199E)
R12304(R0x3010)	fine_correction	0ddd dddd dddd dddd	256 (0x0100)
R12306(R0x3012)	coarse_integration_time_	dddd dddd dddd dddd	16 (0x0010)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12308(R0x3014)	fine_integration_time_	dddd dddd dddd ddd0	1386 (0x056A)
R12310(R0x3016)	row_speed	0000 0ddd 0ddd 0ddd	273 (0x0111)
R12312(R0x3018)	extra_delay	dddd dddd dddd ddd0	0 (0x0000)
R12314(R0x301A)	reset_register	d00d 0ddd dd0d dddd	88 (0x0058)
R12316(R0x301C)	mode_select_	0000 000d	0 (0x0000)
R12317(R0x301D)	image_orientation_	0000 00dd	0 (0x0000)
R12318(R0x301E)	data_pedestal_	0000 dddd dddd dddd	168 (0x00A8)
R12321(R0x3021)	software_reset_	0000 000d	0 (0x0000)
R12322(R0x3022)	grouped_parameter_hold_	0000 000d	0 (0x0000)
R12323(R0x3023)	mask_corrupted_frames_	0000 000d	0 (0x0000)
R12324(R0x3024)	pixel_order_	0000 00??	0 (0x0000)
R12326(R0x3026)	gpi_status	dddd dddd dddd ????	65535 (0xFFFF)
R12328(R0x3028)	analogue_gain_code_global_	0000 0000 0ddd dddd	13 (0x000D)
R12330(R0x302A)	analogue_gain_code_greenR_	0000 0000 0ddd dddd	13 (0x000D)
R12332(R0x302C)	analogue_gain_code_red_	0000 0000 0ddd dddd	13 (0x000D)
R12334(R0x302E)	analogue_gain_code_blue_	0000 0000 0ddd dddd	13 (0x000D)
R12336(R0x3030)	analogue_gain_code_greenB_	0000 0000 0ddd dddd	13 (0x000D)
R12338(R0x3032)	digital_gain_greenR_	0000 0ddd 0000 0000	256 (0x0100)
R12340(R0x3034)	digital_gain_red_	0000 0ddd 0000 0000	256 (0x0100)
R12342(R0x3036)	digital_gain_blue_	0000 0ddd 0000 0000	256 (0x0100)
R12344(R0x3038)	digital_gain_greenB_	0000 0ddd 0000 0000	256 (0x0100)
R12346(R0x303A)	smia_version_	???? ????	10 (0x000A)
R12347(R0x303B)	frame_count_	???? ????	255 (0x00FF)
R12348(R0x303C)	frame_status	0000 0000 0000 00??	0 (0x0000)
R12352(R0x3040)	read_mode	dd0d ddd0 dddd dddd	36 (0x0024)
R12356(R0x3044)	Reserved	–	34112 (0x8540)
R12358(R0x3046)	flash	??dd dddd 0000 0000	1536 (0x0600)
R12360(R0x3048)	flash_count	0000 00dd dddd dddd	8 (0x0008)
R12374(R0x3056)	green1_gain	0000 dddd dddd dddd	564 (0x0234)
R12376(R0x3058)	blue_gain	0000 dddd dddd dddd	564 (0x0234)
R12378(R0x305A)	red_gain	0000 dddd dddd dddd	564 (0x0234)
R12380(R0x305C)	green2_gain	0000 dddd dddd dddd	564 (0x0234)
R12382(R0x305E)	global_gain	0000 dddd dddd dddd	564 (0x0234)
R12384(R0x3060)	Reserved	–	5376 (0x1500)
R12386(R0x3062)	Reserved	–	0 (0x0000)
R12388(R0x3064)	Reserved	–	261 (0x0105)
R12390(R0x3066)	Reserved	–	0 (0x0000)
R12392(R0x3068)	Reserved	–	2730 (0x0AAA)
R12394(R0x306A)	datapath_status	0000 0000 000d dddd	0 (0x0000)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12396(R0x306C)	Reserved	–	32768 (0x8000)
R12398(R0x306E)	datapath_select	dddd dd00 d00d 0000	36992 (0x9080)
R12400(R0x3070)	test_pattern_mode_	0000 000d 0000 0ddd	0 (0x0000)
R12402(R0x3072)	test_data_red_	0000 dddd dddd dddd	0 (0x0000)
R12404(R0x3074)	test_data_greenR_	0000 dddd dddd dddd	0 (0x0000)
R12406(R0x3076)	test_data_blue_	0000 dddd dddd dddd	0 (0x0000)
R12408(R0x3078)	test_data_greenB_	0000 dddd dddd dddd	0 (0x0000)
R12416(R0x3080)	Reserved	–	164 (0x00A4)
R12418(R0x3082)	Reserved	–	4369 (0x1111)
R12420(R0x3084)	Reserved	–	9252 (0x2424)
R12422(R0x3086)	Reserved	–	9321 (0x2469)
R12424(R0x3088)	Reserved	–	26096 (0x65F0)
R12426(R0x308A)	Reserved	–	25700 (0x6464)
R12428(R0x308C)	Reserved	–	14483 (0x3893)
R12430(R0x308E)	Reserved	–	5654 (0x1616)
R12432(R0x3090)	Reserved	–	22102 (0x5656)
R12434(R0x3092)	Reserved	–	2660 (0x0A64)
R12436(R0x3094)	Reserved	–	25700 (0x6464)
R12438(R0x3096)	Reserved	–	25700 (0x6464)
R12440(R0x3098)	Reserved	–	27684 (0x6C24)
R12442(R0x309A)	Reserved	–	44288 (0xAD00)
R12444(R0x309C)	Reserved	–	6400 (0x1900)
R12446(R0x309E)	Reserved	–	25856 (0x6500)
R12448(R0x30A0)	x_even_inc_	0000 0000 0000 000?	1 (0x0001)
R12450(R0x30A2)	x_odd_inc_	0000 0000 0000 0ddd	1 (0x0001)
R12452(R0x30A4)	y_even_inc_	0000 0000 0000 000?	1 (0x0001)
R12454(R0x30A6)	y_odd_inc_	0000 0000 0000 0ddd	1 (0x0001)
R12490(R0x30CA)	Reserved	–	4 (0x0004)
R12492(R0x30CC)	Reserved	–	0 (0x0000)
R12494(R0x30CE)	Reserved	–	0 (0x0000)
R12496(R0x30D0)	Reserved	–	0 (0x0000)
R12498(R0x30D2)	Reserved	–	0 (0x0000)
R12500(R0x30D4)	Reserved	–	32896 (0x8080)
R12502(R0x30D6)	Reserved	–	2048 (0x0800)
R12504(R0x30D8)	Reserved	–	0 (0x0000)
R12506(R0x30DA)	Reserved	–	0 (0x0000)
R12510(R0x30DE)	Reserved	–	17 (0x0011)
R12512(R0x30E0)	Reserved	–	46594 (0xB602)
R12514(R0x30E2)	Reserved	–	37475 (0x9263)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12516(R0x30E4)	Reserved	–	46226 (0xB492)
R12518(R0x30E6)	Reserved	–	46083 (0xB403)
R12520(R0x30E8)	Reserved	–	255 (0x00FF)
R12522(R0x30EA)	Reserved	–	13059 (0x3303)
R12524(R0x30EC)	Reserved	–	25395 (0x6333)
R12526(R0x30EE)	Reserved	–	255 (0x00FF)
R12528(R0x30F0)	Reserved	–	255 (0x00FF)
R12530(R0x30F2)	Reserved	–	46083 (0xB403)
R12532(R0x30F4)	Reserved	–	255 (0x00FF)
R12534(R0x30F6)	Reserved	–	2048 (0x0800)
R12536(R0x30F8)	Reserved	–	2048 (0x0800)
R12538(R0x30FA)	Reserved	–	2048 (0x0800)
R12540(R0x30FC)	Reserved	–	2048 (0x0800)
R12542(R0x30FE)	Reserved	–	28761 (0x7059)
R12544(R0x3100)	Reserved	–	28761 (0x7059)
R12546(R0x3102)	Reserved	–	37233 (0x9171)
R12548(R0x3104)	Reserved	–	45203 (0xB093)
R12550(R0x3106)	Reserved	–	12809 (0x3209)
R12552(R0x3108)	Reserved	–	22579 (0x5833)
R12554(R0x310A)	Reserved	–	46082 (0xB402)
R12556(R0x310C)	Reserved	–	46082 (0xB402)
R12558(R0x310E)	Reserved	–	46089 (0xB409)
R12560(R0x3110)	Reserved	–	46337 (0xB501)
R12562(R0x3112)	Reserved	–	46081 (0xB401)
R12564(R0x3114)	Reserved	–	1795 (0x0703)
R12566(R0x3116)	Reserved	–	46338 (0xB502)
R12568(R0x3118)	Reserved	–	0 (0x0000)
R12570(R0x311A)	Reserved	–	28772 (0x7064)
R12572(R0x311C)	Reserved	–	0 (0x0000)
R12574(R0x311E)	Reserved	–	2048 (0x0800)
R12576(R0x3120)	Reserved	–	0 (0x0000)
R12578(R0x3122)	Reserved	–	2048 (0x0800)
R12580(R0x3124)	Reserved	–	255 (0x00FF)
R12582(R0x3126)	Reserved	–	46594 (0xB602)
R12584(R0x3128)	Reserved	–	46088 (0xB408)
R12586(R0x312A)	Reserved	–	255 (0x00FF)
R12588(R0x312C)	Reserved	–	176 (0x00B0)
R12590(R0x312E)	Reserved	–	13494 (0x34B6)
R12608(R0x3140)	Reserved	–	13313 (0x3401)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12610(R0x3142)	Reserved	–	13058 (0x3302)
R12612(R0x3144)	Reserved	–	12803 (0x3203)
R12614(R0x3146)	Reserved	–	11524 (0x2D04)
R12616(R0x3148)	Reserved	–	11269 (0x2C05)
R12618(R0x314A)	Reserved	–	11524 (0x2D04)
R12620(R0x314C)	Reserved	–	0 (0x0000)
R12622(R0x314E)	Reserved	–	13058 (0x3302)
R12624(R0x3150)	Reserved	–	0 (0x0000)
R12628(R0x3154)	Reserved	–	5249 (0x1481)
R12630(R0x3156)	Reserved	–	7297 (0x1C81)
R12632(R0x3158)	Reserved	–	0 (0x0000)
R12634(R0x315A)	Reserved	–	0 (0x0000)
R12640(R0x3160)	global_seq_trigger	0000 00?? 0000 0ddd	0 (0x0000)
R12642(R0x3162)	global_rst_end	dddd dddd dddd dddd	80 (0x0050)
R12644(R0x3164)	global_shutter_start	dddd dddd dddd dddd	120 (0x0078)
R12646(R0x3166)	global_read_start	dddd dddd dddd dddd	160 (0x00A0)
R12652(R0x316C)	Reserved	–	17424 (0x4410)
R12654(R0x316E)	Reserved	–	1024 (0x0400)
R12656(R0x3170)	Reserved	–	11686 (0x2DA6)
R12660(R0x3174)	Reserved	–	4626 (0x1212)
R12662(R0x3176)	Reserved	–	4626 (0x1212)
R12664(R0x3178)	Reserved	–	4626 (0x1212)
R12672(R0x3180)	Reserved	–	36863 (0x8FFF)
R12674(R0x3182)	Reserved	–	0 (0x0000)
R12676(R0x3184)	Reserved	–	0 (0x0000)
R12678(R0x3186)	Reserved	–	0 (0x0000)
R12680(R0x3188)	Reserved	–	0 (0x0000)
R12682(R0x318A)	Reserved	–	0 (0x0000)
R12684(R0x318C)	Reserved	–	0 (0x0000)
R12686(R0x318E)	Reserved	–	0 (0x0000)
R12688(R0x3190)	Reserved	–	0 (0x0000)
R12690(R0x3192)	Reserved	–	0 (0x0000)
R12692(R0x3194)	Reserved	–	0 (0x0000)
R12694(R0x3196)	Reserved	–	0 (0x0000)
R12696(R0x3198)	Reserved	–	0 (0x0000)
R12776(R0x31E8)	horizontal_cursor_position_	0000 dddd dddd dddd	0 (0x0000)
R12778(R0x31EA)	vertical_cursor_position_	0000 dddd dddd dddd	0 (0x0000)
R12780(R0x31EC)	horizontal_cursor_width_	0000 dddd dddd dddd	0 (0x0000)
R12782(R0x31EE)	vertical_cursor_width_	0000 dddd dddd dddd	0 (0x0000)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R12788(R0x31F4)	Reserved	–	291 (0x0123)
R12790(R0x31F6)	Reserved	–	17767 (0x4567)
R12792(R0x31F8)	Reserved	–	35243 (0x89AB)
R12794(R0x31FA)	Reserved	–	52719 (0xCDEF)
R12796(R0x31FC)	1^2C Addresses	–	12320 (0x3020)
R12798(R0x31FE)	Reserved	–	0 (0x0000)
R13824(R0x3600)	P_GR_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R13826(R0x3602)	P_GR_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R13828(R0x3604)	P_GR_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R13830(R0x3606)	P_GR_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R13832(R0x3608)	P_GR_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R13834(R0x360A)	P_RD_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R13836(R0x360C)	P_RD_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R13838(R0x360E)	P_RD_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R13840(R0x3610)	P_RD_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R13842(R0x3612)	P_RD_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R13844(R0x3614)	P_BL_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R13846(R0x3616)	P_BL_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R13848(R0x3618)	P_BL_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R13850(R0x361A)	P_BL_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R13852(R0x361C)	P_BL_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R13854(R0x361E)	P_GB_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R13856(R0x3620)	P_GB_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R13858(R0x3622)	P_GB_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R13860(R0x3624)	P_GB_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R13862(R0x3626)	P_GB_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R13888(R0x3640)	P_GR_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R13890(R0x3642)	P_GR_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R13892(R0x3644)	P_GR_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R13894(R0x3646)	P_GR_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R13896(R0x3648)	P_GR_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R13898(R0x364A)	P_RD_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R13900(R0x364C)	P_RD_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R13902(R0x364E)	P_RD_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R13904(R0x3650)	P_RD_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R13906(R0x3652)	P_RD_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R13908(R0x3654)	P_BL_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R13910(R0x3656)	P_BL_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R13912(R0x3658)	P_BL_P1Q2	dddd dddd dddd dddd	0 (0x0000)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R13914(R0x365A)	P_BL_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R13916(R0x365C)	P_BL_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R13918(R0x365E)	P_GB_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R13920(R0x3660)	P_GB_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R13922(R0x3662)	P_GB_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R13924(R0x3664)	P_GB_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R13926(R0x3666)	P_GB_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R13952(R0x3680)	P_GR_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R13954(R0x3682)	P_GR_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R13956(R0x3684)	P_GR_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R13958(R0x3686)	P_GR_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R13960(R0x3688)	P_GR_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R13962(R0x368A)	P_RD_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R13964(R0x368C)	P_RD_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R13966(R0x368E)	P_RD_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R13968(R0x3690)	P_RD_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R13970(R0x3692)	P_RD_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R13972(R0x3694)	P_BL_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R13974(R0x3696)	P_BL_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R13976(R0x3698)	P_BL_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R13978(R0x369A)	P_BL_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R13980(R0x369C)	P_BL_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R13982(R0x369E)	P_GB_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R13984(R0x36A0)	P_GB_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R13986(R0x36A2)	P_GB_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R13988(R0x36A4)	P_GB_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R13990(R0x36A6)	P_GB_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R14016(R0x36C0)	P_GR_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R14018(R0x36C2)	P_GR_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R14020(R0x36C4)	P_GR_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R14022(R0x36C6)	P_GR_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R14024(R0x36C8)	P_GR_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R14026(R0x36CA)	P_RD_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R14028(R0x36CC)	P_RD_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R14030(R0x36CE)	P_RD_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R14032(R0x36D0)	P_RD_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R14034(R0x36D2)	P_RD_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R14036(R0x36D4)	P_BL_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R14038(R0x36D6)	P_BL_P3Q1	dddd dddd dddd dddd	0 (0x0000)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R14040(R0x36D8)	P_BL_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R14042(R0x36DA)	P_BL_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R14044(R0x36DC)	P_BL_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R14046(R0x36DE)	P_GB_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R14048(R0x36E0)	P_GB_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R14050(R0x36E2)	P_GB_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R14052(R0x36E4)	P_GB_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R14054(R0x36E6)	P_GB_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R14080(R0x3700)	P_GR_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R14082(R0x3702)	P_GR_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R14084(R0x3704)	P_GR_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R14086(R0x3706)	P_GR_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R14088(R0x3708)	P_GR_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R14090(R0x370A)	P_RD_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R14092(R0x370C)	P_RD_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R14094(R0x370E)	P_RD_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R14096(R0x3710)	P_RD_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R14098(R0x3712)	P_RD_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R14100(R0x3714)	P_BL_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R14102(R0x3716)	P_BL_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R14104(R0x3718)	P_BL_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R14106(R0x371A)	P_BL_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R14108(R0x371C)	P_BL_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R14110(R0x371E)	P_GB_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R14112(R0x3720)	P_GB_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R14114(R0x3722)	P_GB_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R14116(R0x3724)	P_GB_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R14118(R0x3726)	P_GB_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R14144(R0x3740)	Reserved	–	0 (0x0000)
R14146(R0x3742)	Reserved	–	0 (0x0000)
R14148(R0x3744)	Reserved	–	0 (0x0000)
R14150(R0x3746)	Reserved	–	0 (0x0000)
R14152(R0x3748)	Reserved	–	0 (0x0000)
R14160(R0x3750)	Reserved	–	0 (0x0000)
R14162(R0x3752)	Reserved	–	0 (0x0000)
R14164(R0x3754)	Reserved	–	0 (0x0000)
R14166(R0x3756)	Reserved	–	0 (0x0000)
R14168(R0x3758)	Reserved	–	0 (0x0000)
R14208(R0x3780)	SC_ENABLE	d000 0000 0000 0000	0 (0x0000)

Table 19: 3: Manufacturer Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable;? = read-only, dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
R14210(R0x3782)	ORIGIN_C	0000 dddd dddd dddd	0 (0x0000)
R14212(R0x3784)	ORIGIN_R	0000 dddd dddd dddd	0 (0x0000)
R15872(R0x3E00)	Reserved	–	0 (0x0000)
R16128(R0x3F00)	Reserved	–	0 (0x0000)

Register Description

Table 20: 0: SMIA Configuration

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0 R0x0000	**15:0**	**0x2B00**	**model_id (RW)**	N	N
	This register is an alias of R0x3000–1. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R2 R0x0002	**7:0**	**0x0000**	**revision_number (RW)**	N	N
	Micron Imaging assigned revision number. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R3 R0x0003	**7:0**	**0x0006**	**manufacturer_id (RO)**	N	N
	Manufacturer ID assigned to Micron Imaging. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4 R0x0004	**7:0**	**0x000A**	**smia_version (RO)**	N	N
	This register is an alias of R0x303A. Read-only.				
R5 R0x0005	**7:0**	**0x00FF**	**frame_count (RO)**	Y	N
	This register is an alias of R0x303B. Read-only.				
R6 R0x0006	**7:0**	**0x0000**	**pixel_order (RO)**	N	N
	This register is an alias of R0x3024. Read-only.				
R8 R0x0008	**15:0**	**0x00A8**	**data_pedestal (RW)**	N	Y
	This register is an alias of R0x301E–F. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R64 R0x0040	**7:0**	**0x0001**	**frame_format_model_type (RO)**	N	N
	Type 1. 2-byte Generic Frame Format Description. Read-only.				
R65 R0x0041	**7:0**	**0x0012**	**frame_format_model_subtype (RO)**	N	N
	Number of descriptors: 1 X (column) descriptor and two Y (row) descriptors. Read-only.				
R66 R0x0042	**15:0**	**0x5CC0**	**frame_format_descriptor_0 (RO)**	Y	N
	X descriptor: Bits[11:0] of this register reflect the current value of x_output_size[11:0]. Upper 4 bits is the pixel code; 5 = Visible Pixel Data. Read-only, dynamic.				
R68 R0x0044	**15:0**	**0x1002**	**frame_format_descriptor_1 (RO)**	Y	N
	Y descriptor: In normal operation, returns 0x1002 to indicate that 2 rows of embedded data are present in the output image. If embedded data is disabled (by selecting the PN9 test pattern using R0x3070–1) this register will return 0x1000. Read-only.				
R70 R0x0046	**15:0**	**0x5990**	**frame_format_descriptor_2 (RO)**	Y	N
	Y descriptor: Bits[11:0] of this register reflect the current value of y_output_size[11:0]. Upper 4 bits is the pixel code; 5 = Visible Pixel Data. Read-only, dynamic.				
R72 R0x0048	**15:0**	**0x0000**	**frame_format_descriptor_3 (RO)**	N	N
	Read-only.				
R74 R0x004A	**15:0**	**0x0000**	**frame_format_descriptor_4 (RO)**	N	N
	Read-only.				
R76 R0x004C	**15:0**	**0x0000**	**frame_format_descriptor_5 (RO)**	N	N
	Read-only.				
R78 R0x004E	**15:0**	**0x0000**	**frame_format_descriptor_6 (RO)**	N	N
	Read-only.				
R80 R0x0050	**15:0**	**0x0000**	**frame_format_descriptor_7 (RO)**	N	N
	Read-only.				
R82 R0x0052	**15:0**	**0x0000**	**frame_format_descriptor_8 (RO)**	N	N
	Read-only.				

Table 20: 0: SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R84 R0x0054	15:0	0x0000	frame_format_descriptor_9 (RO)	N	N
	Read-only.				
R86 R0x0056	15:0	0x0000	frame_format_descriptor_10 (RO)	N	N
	Read-only.				
R88 R0x0058	15:0	0x0000	frame_format_descriptor_11 (RO)	N	N
	Read-only.				
R90 R0x005A	15:0	0x0000	frame_format_descriptor_12 (RO)	N	N
	Read-only.				
R92 R0x005C	15:0	0x0000	frame_format_descriptor_13 (RO)	N	N
	Read-only.				
R94 R0x005E	15:0	0x0000	frame_format_descriptor_14 (RO)	N	N
	Read-only.				
R128 R0x0080	15:0	0x0001	analogue_gain_capability (RO)	N	N
	Indicates the provision of separate (per-color) analog gain control. The sensor supports both global and separate (per-color) analog gain control. Read-only.				
R132 R0x0084	15:0	0x0008	analogue_gain_code_min (RO)	N	N
	Minimum gain code. Read-only.				
R134 R0x0086	15:0	0x007F	analogue_gain_code_max (RO)	N	N
	Maximum gain code. Read-only.				
R136 R0x0088	15:0	0x0001	analogue_gain_code_step (RO)	N	N
	Gain code step size. Read-only.				
R138 R0x008A	15:0	0x0000	analogue_gain_type (RO)	N	N
	Indicates support for analog gain coding type 0 (baseline SMIA). Read-only.				
R140 R0x008C	15:0	0x0001	analogue_gain_m0 (RO)	N	N
	Constants for the gain equation. Read-only.				
R142 R0x008E	15:0	0x0000	analogue_gain_c0 (RO)	N	N
	Constants for the gain equation. Read-only.				
R144 R0x0090	15:0	0x0000	analogue_gain_m1 (RO)	N	N
	Constants for the gain equation. Read-only.				
R146 R0x0092	15:0	0x0008	analogue_gain_c1 (RO)	N	N
	Constants for the gain equation. Read-only.				
R192 R0x00C0	7:0	0x0001	data_format_model_type (RO)	N	N
	Indicates the use of 2-byte data format. Read-only.				
R193 R0x00C1	7:0	0x0005	data_format_model_subtype (RO)	N	N
	Indicates the provision of 5 data format descriptors. Read-only.				
R194 R0x00C2	15:0	0x0A0A	data_format_descriptor_0 (RO)	N	N
	Indicates support for RAW10 data format in which the two LSB of each 12-bit pixel data value are discarded. Read-only.				
R196 R0x00C4	15:0	0x0808	data_format_descriptor_1 (RO)	N	N
	Indicates support for RAW8 data format in which the four LSB of each 12-bit pixel data value are discarded. Read-only.				

Table 20: 0: SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R198 R0x00C6	**15:0**	**0x0A08**	**data_format_descriptor_2 (RO)**	N	N
	Indicates support for RAW8 data format in which each truncated 10-bit pixel data value is compressed to an 8-bit value. Read-only.				
R200 R0x00C8	**15:0**	**0x0C0C**	**data_format_descriptor_3 (RO)**	N	N
	Indicates support for RAW12, uncompressed data format. Read-only.				
R202 R0x00CA	**15:0**	**0x0C08**	**data_format_descriptor_4 (RO)**	N	N
	Indicates support for RAW8 data format in which each 12-bit pixel data value is compressed to an 8-bit value. Read-only.				
R204 R0x00CC	**15:0**	**0x0000**	**data_format_descriptor_5 (RO)**	N	N
	Read-only.				
R206 R0x00CE	**15:0**	**0x0000**	**data_format_descriptor_6 (RO)**	N	N
	Read-only.				
R256 R0x0100	**7:0**	**0x0000**	**mode_select (RW)**	Y	N
	This register field is an alias of R0x301A[2].				
R257 R0x0101	**7:0**	**0x0000**	**image_orientation (RW)**		
	7:2	X	Reserved		
	1	0x0000	Vertical Flip This register field is an alias of R0x3040–1[1].	Y	YM
	0	0x0000	Horizontal Mirror This register field is an alias of R0x3040–1[0].	Y	YM
R259 R0x0103	**7:0**	**0x0000**	**software_reset (RW)**	N	Y
	This register field is an alias of R0x301A–B[0].				
R260 R0x0104	**7:0**	**0x0000**	**grouped_parameter_hold (RW)**	N	N
	This register field is an alias of R0x301A–B[15].				
R261 R0x0105	**7:0**	**0x0000**	**mask_corrupted_frames (RW)**	N	Y
	This register field is an alias of R0x301A–B[9].				
R272 R0x0110	**7:0**	**0x0000**	**Reserved (RW)**	Y	N
	Not used.				
R273 R0x0111	**7:0**	**0x0000**	**Reserved (RW)**	Y	N
	Not used.				
R274 R0x0112	**15:0**	**0x0C0C**	**ccp_data_format (RW)**	Y	N
	[7:0] = The bit-width of the compressed pixel data [15:8] = The bit-width of the uncompressed pixel data The value in this register must match one of the valid data_format_descriptor registers (R0x00C2–R0x00C7).				
R288 R0x0120	**7:0**	**0x0000**	**gain_mode (RW)**	N	N
	This read/write bit has no function.				
R512 R0x0200	**15:0**	**0x056A**	**fine_integration_time (RW)**	Y	N
	Integration time programmed in units of pck. This register is an alias of R0x3014–5.				
R514 R0x0202	**15:0**	**0x0010**	**coarse_integration_time (RW)**	Y	N
	Integration time programmed in units of line_length_pck. This register is an alias of R0x3012–3.				

Table 20: 0: SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R516 R0x0204	**15:0**	**0x000D**	**analogue_gain_code_global (RW)**	Y	N
	This register is an alias of R0x3028–9.				
R518 R0x0206	**15:0**	**0x000D**	**analogue_gain_code_green1 (RW)**	Y	N
	This register is an alias of R0x302A–B.				
R520 R0x0208	**15:0**	**0x000D**	**analogue_gain_code_red (RW)**	Y	N
	This register is an alias of R0x302C–D.				
R522 R0x020A	**15:0**	**0x000D**	**analogue_gain_code_blue (RW)**	Y	N
	This register is an alias of R0x302E–F.				
R524 R0x020C	**15:0**	**0x000D**	**analogue_gain_code_green2 (RW)**	Y	N
	This register is an alias of R0x3030–1.				
R526 R0x020E	**15:0**	**0x0100**	**digital_gain_green1 (RW)**	Y	N
	This register is an alias of R0x3032–3.				
R528 R0x0210	**15:0**	**0x0100**	**digital_gain_red (RW)**	Y	N
	This register is an alias of R0x3034–5.				
R530 R0x0212	**15:0**	**0x0100**	**digital_gain_blue (RW)**	Y	N
	This register is an alias of R0x3036–7.				
R532 R0x0214	**15:0**	**0x0100**	**digital_gain_green2 (RW)**	Y	N
	This register is an alias of R0x3038–9.				
R768 R0x0300	**15:0**	**0x0004**	**vt_pix_clk_div (RW)**	N	Y
	Not in use. Use pc_speed[2:0] to change vt_pix_clk_mhz instead.				
R770 R0x0302	**15:0**	**0x0001**	**vt_sys_clk_div (RW)**	N	N
	Clock divisor applied to PLL output clock to generate video timing system clock. Read-only.				
R772 R0x0304	**15:0**	**0x0002**	**pre_pll_clk_div (RW)**	N	Y
	Clock divisor applied to EXTCLK to generate PLL input clock.				
R774 R0x0306	**15:0**	**0x0040**	**pll_multiplier (RW)**	N	Y
	Clock multiplier applied to PLL input clock.				
R776 R0x0308	**15:0**	**0x0008**	**op_pix_clk_div (RW)**	N	Y
	Clock divisor applied to the output system clock to generate the output pixel clock. Legal values are 1, 2 and 4.				
R778 R0x030A	**15:0**	**0x0001**	**op_sys_clk_div (RW)**	N	Y
	Clock divisor applied to PLL output clock to generate output system clock. Read-only.				
R832 R0x0340	**15:0**	**0x0A00**	**frame_length_lines (RW)**	Y	YM
	This register is an alias of R0x300A–B.				
R834 R0x0342	**15:0**	**0x199E**	**line_length_pck (RW)**	Y	YM
	This register is an alias of R0x300C–D.				
R836 R0x0344	**15:0**	**0x0000**	**x_addr_start (RW)**	Y	N
	This register is an alias of R0x3004–5.				
R838 R0x0346	**15:0**	**0x0008**	**y_addr_start (RW)**	Y	YM
	This register is an alias of R0x3002–5.				
R840 R0x0348	**15:0**	**0x0CBF**	**x_addr_end (RW)**	Y	N
	This register is an alias of R0x3008–9.				
R842 R0x034A	**15:0**	**0x0997**	**y_addr_end (RW)**	Y	YM
	This register is an alias of R0x3006–7.				

Table 20: 0: SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R844 R0x034C	**15:0**	**0x0CC0**	**x_output_size (RW)**	Y	N
	Set X output size of displayed image. Bit[0] is read-only 0. The default value of this register is set to be consistent with the default values of x_addr_end and x_addr_start.				
R846 R0x034E	**15:0**	**0x0990**	**y_output_size (RW)**	Y	N
	Set Y output size of the displayed image. Bit[0] is read-only 0. The default value of this registers set to be consistent with the default values of y_addr_end and y_addr_start. The output image will have two additional rows containing embedded data, in accordance with the frame format descriptors.				
R896 R0x0380	**15:0**	**0x0001**	**x_even_inc (RO)**	N	N
	Read-only. The fixed value of "1" constrains subsampling operation to use adjacent pixels of a pixel quad.				
R898 R0x0382	**15:0**	**0x0001**	**x_odd_inc (RW)**	Y	YM
	This register field is an alias of R0x3040–1[7:5].				
R900 R0x0384	**15:0**	**0x0001**	**y_even_inc (RO)**	N	N
	Read-only. The fixed value of "1" constrains subsampling operation to use adjacent pixels of a pixel quad.				
R902 R0x0386	**15:0**	**0x0001**	**y_odd_inc (RW)**	Y	YM
	This register field is an alias of R0x3040–1[4:2].				
R1024 R0x0400	**15:0**	**0x0000**	**scaling_mode (RW)**	Y	N
	0 = Disable scaler 1 = Enable horizontal scaling 2 = Enable horizontal and vertical scaling 3 = Reserved				
R1026 R0x0402	**15:0**	**0x0000**	**spatial_sampling (RW)**	Y	N
	0 = Bayer sampling 1 = Co-sited sampling				
R1028 R0x0404	**15:0**	**0x0010**	**scale_m (RW)**	Y	N
	Scale factor M.				
R1030 R0x0406	**15:0**	**0x0010**	**scale_n (RO)**	N	N
	Scale factor N. Read-only.				
R1280 R0x0500	**15:0**	**0x0001**	**compression_mode (RO)**	N	Y
	0x0001 = 10-bit to 8-bit and 12-bit to 8-bit compression uses the DPCM/PCM Simple Predictor algorithm. Read-only. This register controls the algorithm that is to be used for compression. The sensor only supports a single algorithm and therefore this register is read-only. This register does not control whether data compression is enabled; that is controlled by the ccp_data_format register (R0x0012–3).				
R1536 R0x0600	**15:0**	**0x0000**	**test_pattern_mode (RW)**	N	Y
	This register is an alias of R0x3070–1.				
R1538 R0x0602	**15:0**	**0x0000**	**test_data_red (RW)**	N	Y
	This register is an alias of R0x3072–3.				
R1540 R0x0604	**15:0**	**0x0000**	**test_data_green1 (RW)**	N	Y
	This register is an alias of R0x3074–5.				
R1542 R0x0606	**15:0**	**0x0000**	**test_data_blue (RW)**	N	Y
	This register is an alias of R0x3076–7.				
R1544 R0x0608	**15:0**	**0x0000**	**test_data_green2 (RW)**	N	Y
	This register is an alias of R0x3078–8.				

Table 20: 0: SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R1546 R0x060A	15:0	0x0000	horizontal_cursor_width (RW)	N	N
	This register is an alias of R0x31EC–D.				
R1548 R0x060C	15:0	0x0000	horizontal_cursor_position (RW)	N	N
	This register is an alias of R0x31E8–9.				
R1550 R0x060E	15:0	0x0000	vertical_cursor_width (RW)	N	N
	This register is an alias of R0x31EE–F.				
R1552 R0x0610	15:0	0x0000	vertical_cursor_position (RW)	N	N
	This register is an alias of R0x31EA–B.				

Table 21: 1: SMIA Parameter Limits

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R4096 R0x1000	15:0	0x0001	integration_time_capability (RO)	N	N
	Indicates the provision of coarse and fine integration time control. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4100 R0x1004	15:0	0x0000	coarse_integration_time_min (RW)	N	N
	The minimum coarse integration time. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4102 R0x1006	15:0	0x0001	coarse_integration_time_max_margin (RW)	N	N
	The maximum coarse integration time is (frame_length_lines - coarse_integration_time_max_margin). Read-only. Can be made read/write by clearing R0x301A–B[3]. In the sensor, this limit can be broken. The result will be a graceful degradation of frame rate.				
R4104 R0x1008	15:0	0x056A	fine_integration_time_min (RW)	N	N
	The minimum fine integration time. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4106 R0x100A	15:0	0x03AA	fine_integration_time_max_margin (RW)	N	N
	The minimum fine integration time is (line_length_pck - fine_integration_time_max_margin). Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4224 R0x1080	15:0	0x0001	digital_gain_capability (RO)	N	N
	Indicates the provision of separate (per-color) digital gain control. Read-only.				
R4228 R0x1084	15:0	0x0100	digital_gain_min (RO)	N	N
	UFIX16. Minimum value of digital gain is 1.0. Read-only.				
R4230 R0x1086	15:0	0x0700	digital_gain_max (RO)	N	N
	UFIX16. Maximum value of digital gain is 4.0. Read-only.				
R4232 R0x1088	15:0	0x0100	digital_gain_step_size (RO)	N	N
	UFIX16. Step size for digital gain is 1.0. Read-only.				
R4352 R0x1100	15:0	0x40C0	min_ext_clk_freq_mhz_1 (RO)	N	N
	FLP32. Minimum external clock frequency into PLL is 6 MHz. Read-only.				
R4354 R0x1102	15:0	0x0000	min_ext_clk_freq_mhz_2 (RO)	N	N
	FLP32. Minimum external clock frequency into PLL is 6 MHz. Read-only.				

Table 21: 1: SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R4356 R0x1104	15:0	0x4240	max_ext_clk_freq_mhz_1 (RO)	N	N
	FLP32. Maximum external clock frequency into PLL is 48 MHz. Read-only.				
R4358 R0x1106	15:0	0x0000	max_ext_clk_freq_mhz_2 (RO)	N	N
	FLP32. Maximum external clock frequency into PLL is 48 MHz. Read-only.				
R4360 R0x1108	15:0	0x0001	min_pre_pll_clk_div (RO)	N	N
	Minimum clock divisor applied to PLL input clock. Read-only.				
R4362 R0x110A	15:0	0x0040	max_pre_pll_clk_div (RO)	N	N
	Maximum clock divisor applied to PLL input clock. Read-only.				
R4364 R0x110C	15:0	0x4000	min_pll_ip_freq_mhz_1 (RO)	N	N
	FLP32. Minimum clock frequency into the PFD of the PLL is 2 MHz. Read-only.				
R4366 R0x110E	15:0	0x0000	min_pll_ip_freq_mhz_2 (RO)	N	N
	FLP32. Minimum clock frequency into the PFD of the PLL is 2 MHz. Read-only.				
R4368 R0x1110	15:0	0x41C0	max_pll_ip_freq_mhz_1 (RO)	N	N
	FLP32. Maximum clock frequency into the PFD of the PLL is 22.5 MHz. Read-only.				
R4370 R0x1112	15:0	0x0000	max_pll_ip_freq_mhz_2 (RO)	N	N
	FLP32. Maximum clock frequency into the PFD of the PLL is 22.5 MHz. Read-only.				
R4372 R0x1114	15:0	0x0020	min_pll_multiplier (RO)	N	N
	Minimum multiplier applied by PLL. Read-only.				
R4374 R0x1116	15:0	0x0100	max_pll_multiplier (RO)	N	N
	Maximum multiplier applied by PLL. Read-only.				
R4376 R0x1118	15:0	0x43C0	min_pll_op_freq_mhz_1 (RO)	N	N
	FLP32. Minimum output frequency supported by the PLL is 160 MHz. Read-only.				
R4378 R0x111A	15:0	0x0000	min_pll_op_freq_mhz_2 (RO)	N	N
	FLP32. Minimum output frequency supported by the PLL is 160 MHz. Read-only.				
R4380 R0x111C	15:0	0x4440	max_pll_op_freq_mhz_1 (RO)	N	N
	FLP32. Maximum output frequency supported by the PLL is 768 MHz. Read-only.				
R4382 R0x111E	15:0	0x0000	max_pll_op_freq_mhz_2 (RO)	N	N
	FLP32. Maximum output frequency supported by the PLL is 768 MHz. Read-only.				
R4384 R0x1120	15:0	0x0001	min_vt_sys_clk_div (RO)	N	N
	The video timing sys_clk has a fixed divisor. Read-only.				
R4386 R0x1122	15:0	0x0001	max_vt_sys_clk_div (RO)	N	N
	The video timing sys_clk has a fixed divisor. Read-only.				
R4388 R0x1124	15:0	0x43C0	min_vt_sys_clk_freq_mhz_1 (RO)	N	N
	FLP32. Minimum frequency for the video timing sys_clk is 40 MHz.				
R4390 R0x1126	15:0	0x0000	min_vt_sys_clk_freq_mhz_2 (RO)	N	N
	FLP32. Minimum frequency for the video timing sys_clk is 40 MHz.				
R4392 R0x1128	15:0	0x4440	max_vt_sys_clk_freq_mhz_1 (RO)	N	N
	Maximum frequency for the video timing sys_clk is 192 MHz. Read-only.				
R4394 R0x112A	15:0	0x0000	max_vt_sys_clk_freq_mhz_2 (RO)	N	N
	Maximum frequency for the video timing sys_clk is 192 MHz. Read-only.				
R4396 R0x112C	15:0	0x42C0	min_vt_pix_clk_freq_mhz_1 (RO)	N	N
	FLP32. Minimum frequency for video timing pix_clk is 10 MHz. Read-only.				

Table 21: 1: SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R4398 R0x112E	15:0	0x0000	min_vt_pix_clk_freq_mhz_2 (RO)	N	N
	FLP32. Minimum frequency for video timing pix_clk is 10 MHz. Read-only.				
R4400 R0x1130	15:0	0x4340	max_vt_pix_clk_freq_mhz_1 (RO)	N	N
	FLP32. Maximum frequency for video timing pix_clk is 192 MHz. Read-only.				
R4402 R0x1132	15:0	0x0000	max_vt_pix_clk_freq_mhz_2 (RO)	N	N
	FLP32. Maximum frequency for video timing pix_clk is 192 MHz. Read-only.				
R4404 R0x1134	15:0	0x0004	min_vt_pix_clk_div (RO)	N	N
	Minimum divisor for the video timing pix_clk. Read-only.				
R4406 R0x1136	15:0	0x0004	max_vt_pix_clk_div (RO)	N	N
	Maximum divisor for the video timing pix_clk. Read-only.				
R4416 R0x1140	15:0	0x0057	min_frame_length_lines (RW)	N	N
	Minimum frame length. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4418 R0x1142	15:0	0xFFFF	max_frame_length_lines (RW)	N	N
	Maximum frame length. The maximum frame length is only constrained by the size of the read/write field in the frame_length_lines register (16-bits). Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4420 R0x1144	15:0	0x0914	min_line_length_pck (RW)	N	N
	Minimum line length. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4422 R0x1146	15:0	0xFFFE	max_line_length_pck (RW)	N	N
	Maximum line length. The maximum line length is only constrained by the size of the read/write field in the line_length_pck register (16 bits). Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4424 R0x1148	15:0	0x06AC	min_line_blanking_pck (RW)	N	N
	Minimum line blanking time. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4426 R0x114A	15:0	0x0055	min_frame_blanking_lines (RW)	N	N
	Minimum frame blanking time. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R4448 R0x1160	15:0	0x0001	min_op_sys_clk_div (RO)	N	N
	Minimum divisor for the output sys_clk. Read-only.				
R4450 R0x1162	15:0	0x0001	max_op_sys_clk_div (RO)	N	N
	Maximum divisor for the output sys_clk. Read-only.				
R4452 R0x1164	15:0	0x43C0	min_op_sys_clk_freq_mhz_1 (RO)	N	N
	FLP32. Minimum frequency for output sys_clk is 10 MHz. Read-only.				
R4454 R0x1166	15:0	0x0000	min_op_sys_clk_freq_mhz_2 (RO)	N	N
	FLP32. Minimum frequency for output sys_clk is 10 MHz. Read-only.				
R4456 R0x1168	15:0	0x4440	max_op_sys_clk_freq_mhz_1 (RO)	N	N
	FLP32. Maximum frequency for output sys_clk is 92 MHz. Read-only.				
R4458 R0x116A	15:0	0x0000	max_op_sys_clk_freq_mhz_2 (RO)	N	N
	FLP32. Maximum frequency for output sys_clk is 92 MHz. Read-only.				
R4460 R0x116C	15:0	0x0008	min_op_pix_clk_div (RO)	N	N
	Minimum divisor for output pix_clk. Read-only. Legal values for op_pix_clk_div are 0x01, 0x02 and 0x04.				
R4462 R0x116E	15:0	0x0008	max_op_pix_clk_div (RO)	N	N
	Maximum divisor for output pix_clk. Read-only. Legal values for op_pix_clk_div are 0x01, 0x02 and 0x04.				
R4464 R0x1170	15:0	0x4240	min_op_pix_clk_freq_mhz_1 (RO)	N	N
	FLP32. Minimum frequency for output pix_clk is 5 MHz. Read-only.				

Table 21: 1: SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R4466 R0x1172	15:0	0x0000	**min_op_pix_clk_freq_mhz_2 (RO)**	N	N
	FLP32. Minimum frequency for output pix_clk is 5 MHz. Read-only.				
R4468 R0x1174	15:0	0x42C0	**max_op_pix_clk_freq_mhz_1 (RO)**	N	N
	FLP32. Maximum frequency for output pix_clk is 92 MHz. Read-only.				
R4470 R0x1176	15:0	0x0000	**max_op_pix_clk_freq_mhz_2 (RO)**	N	N
	FLP32. Maximum frequency for output pix_clk is 92 MHz. Read-only.				
R4480 R0x1180	15:0	0x0000	**x_addr_min (RO)**	N	N
	Minimum value for x_addr_start, x_addr_end. Read-only.				
R4482 R0x1182	15:0	0x0000	**y_addr_min (RO)**	N	N
	Minimum value for y_addr_start, y_addr_end. Read-only.				
R4484 R0x1184	15:0	0x0CCF	**x_addr_max (RO)**	N	N
	Maximum value for x_addr_start, x_addr_end. Read-only.				
R4486 R0x1186	15:0	0x099F	**y_addr_max (RO)**	N	N
	Maximum value for y_addr_start, y_addr_end. Read-only.				
R4544 R0x11C0	15:0	0x0001	**min_even_inc (RO)**	N	N
	Minimum value for increment of even X/Y addresses when subsampling is enabled. Read-only.				
R4546 R0x11C2	15:0	0x0001	**max_even_inc (RO)**	N	N
	Maximum value for increment of even X/Y addresses when subsampling is enabled. Read-only.				
R4548 R0x11C4	15:0	0x0001	**min_odd_inc (RO)**	N	N
	Minimum value for increment of odd X/Y addresses when subsampling is enabled. Read-only.				
R4550 R0x11C6	15:0	0x0003	**max_odd_inc (RO)**	N	N
	Maximum value for increment of odd X/Y addresses when subsampling is enabled. Read-only. This set of 4 registers declares the capability for the subsampling mode that was called "skip2" and "skip4" on earlier Micron Imaging sensors. Note that this value should have been 3, since only values of 1, 3 and 7 are supported.				
R4608 R0x1200	15:0	0x0002	**scaling_capability (RO)**	N	N
	Indicates the provision of a full (horizontal and vertical) scaler. Read-only.				
	Indicates the minimum M value for the scaler. Read-only.				
R4614 R0x1206	15:0	0x0080	**scaler_m_max (RO)**	N	N
	Indicates the maximum M value for the scaler. Read-only.				
R4616 R0x1208	15:0	0x0010	**scaler_n_min (RO)**	N	N
	Indicates the minimum N value for the scaler. Read-only.				
R4618 R0x120A	15:0	0x0010	**scaler_n_max (RO)**	N	N
	Indicates the maximum N value for the scaler. Read-only.				
R4864 R0x1300	15:0	0x0001	**compression_capability (RO)**	N	N
	Indicates the capability for performing 12/10-bit to 8-bit pixel data compression. Read-only.				
R5120 R0x1400	15:0	0x0242	**matrix_element_RedInRed (RW)**	N	N
	Read-only. Can be made read/write by clearing R0x301A-B[3].				
R5122 R0x1402	15:0	0xFF00	**matrix_element_GreenInRed (RW)**	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5124 R0x1404	15:0	0xFFBE	**matrix_element_BlueInRed (RW)**	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				

Table 21: 1: SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R5126 R0x1406	15:0	0xFFB4	matrix_element_RedInGreen (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5128 R0x1408	15:0	0x0200	matrix_element_GreenInGreen (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5130 R0x140A	15:0	0xFF4D	matrix_element_BlueInGreen (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5132 R0x140C	15:0	0xFFF1	matrix_element_RedInBlue (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5134 R0x140E	15:0	0xFF34	matrix_element_GreenInBlue (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R5136 R0x1410	15:0	0x01DC	matrix_element_BlueInBlue (RW)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A–B[3].				

Table 22: 3: Manufacturer Specific

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12288 R0x3000	15:0	0x2B00	model_id_ (RW)	N	N
	Model ID. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R12290 R0x3002	15:0	0x0008	y_addr_start_ (RW)	Y	YM
	The first row of visible pixels to be read out (not counting any dark rows that may be read). To move the image window, set this register to the starting Y value.				
R12292 R0x3004	15:0	0x0000	x_addr_start_ (RW)	Y	N
	The first column of visible pixels to be read out (not counting any dark columns that may be read). To move the image window, set this register to the starting X value.				
R12294 R0x3006	15:0	0x0997	y_addr_end_ (RW)	Y	YM
	The last row of visible pixels to be read out.				
R12296 R0x3008	15:0	0x0CBF	x_addr_end_ (RW)	Y	N
	The last column of visible pixels to be read out.				
R12298 R0x300A	15:0	0x0A00	frame_length_lines_ (RW)	Y	YM
	The number of complete lines (rows) in the output frame. This includes visible lines and vertical blanking lines.				
R12300 R0x300C	15:0	0x199E	line_length_pck_ (RW)	Y	YM
	The number of pixel clock periods in one line (row) time. This includes visible pixels and horizontal blanking time.				
R12304 R0x3010	15:0	0x0100	fine_correction (RW)	N	Y
	Fine integration time correction factor. This is an offset that is applied to the programmed value of fine_integration_time such that the actual integration time matches the integration time equation. This register should not be modified under normal operation, but must be modified when binning is enabled or the internal pixel clock divider (pc_speed)[2:0]) is used.				
R12306 R0x3012	15:0	0x0010	coarse_integration_time_ (RW)	Y	N
	Integration time specified in multiples of line_length_pck_.				
R12308 R0x3014	15:0	0x056A	fine_integration_time_ (RW)	Y	N
	Integration time specified as a number of pixel clocks.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12310 R0x3016	**15:0**	**0x0111**	**row_speed (RW)**		
	15:11	X	Reserved		
	10:8	0x0001	Output Clock Speed Slows down the output pixel clock frequency relative to the system clock frequency. A programmed value of N gives a output pixel clock period of N system clocks. Only values 1, 2 and 4 are supported. A value of 0 is illegal: it causes the clock to stop.	N	N
	7	X	Reserved		
	6:4	0x0001	Output Clock Delay Number of half-system-clock-cycle increments to delay the rising edge of PIXCLK relative to transitions on FRAME_VALID, LINE_VALID, and DOUT.	N	N
	3	X	Reserved		
	2:0	0x0001	Pixel Clock Speed Slows down the pixel clock frequency relative to the system clock frequency. A programmed value of N gives a pixel clock period of N system clocks. Only values 1, 2 and 4 are supported. A value of 0 is illegal: it causes the clock to stop.	Y	YM
R12312 R0x3018	**15:0**	**0x0000**	**extra_delay (RW)**	Y	N
	Extra blanking inserted between frames. A programmed value of N increases the vertical blanking time by N pixel clock periods. Can be used to get a more exact frame rate. May affect the integration times of parts of the image when the integration time is less than 1 frame.				
R12314 R0x301A	**15:0**	**0x0058**	**reset_register (RW)**		
	15	0x0000	grouped parameter hold 0 = Update of many of the registers is synchronized to frame start. 1 = Inhibit register updates; register changes will remain pending until this bit is returned to 0. When this bit is returned to 0, all pending register updates will be made on the next frame start.	N	N
	14:13	X	Reserved		
	12	0x0000	Reserved Not used.	N	N
	11	X	Reserved		

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12314 R0x301A	10	0x0000	Restart Bad Frames 1 = a restart is forced any time a bad frame is detected. This can shorten the delay when waiting for a good frame, since the delay for masking out a bad frame will be the integration time rather than the full-frame time.	N	N
	9	0x0000	Mask Bad Frames 0 = The sensor will produce bad (corrupted) frames as a result of some register changes. 1 = Bad (corrupted) frames are masked within the sensor by extending the vertical blanking time for the duration of the bad frame.	N	N
	8	0x0000	GPI Enable 0 = the primary input buffers associated with the GPI0, GPI1, GPI2, GPI3 inputs are powered down and the GPI cannot be used. 1 = the input buffers are enabled and can be read through R0x3026-7.	N	N
	7	0x0000	Parallel Enable 0 = The parallel data interface (DOUT[11:0], LINE_VALID, FRAME_VALID, and PIXCLK) is disabled and the outputs are placed in a HIGH-Z. 1 = The parallel data interface is enabled. The output signals can be switched between a driven and a High-Z using output-enable control.	N	N
	6	0x0001	Drive Pins 0 = The parallel data interface (DOUT[11:0], LINE_VALID, FRAME_VALID, and PIXCLK) may enter a High-Z (depending upon the configuration of R0x3026). 1 = The parallel data interface is driven. This bit is "Don't Care" unless bit[7]=1.	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12314 R0x301A	5	X	Reserved		
	4	0x0001	Standby EOF 0 = Transition to standby is synchronized to the end of a sensor row readout (held-off until LINE_VALID has fallen). 1 = Transition to standby is synchronized to the end of a frame.	N	Y
	3	0x0001	Lock Reg Many SMIA registers that are specified as read-only are actually implemented as read/write registers. Clearing this bit allows such registers to be written.	N	N
	2	0x0000	Stream Setting this bit places the sensor in streaming mode. Clearing this bit places the sensor in a low power mode. The result of clearing this bit depends upon the operating mode of the sensor. Entry and exit from streaming mode can also be controlled from the signal interface.	Y	N
	1	0x0000	Restart This bit always reads as "0." Setting this bit causes the sensor to truncate the current frame at the end of the current row and start resetting (integrating) the first row. The delay before the first valid frame is read out is equal to the integration time.	N	Y
	0	0x0000	Reset This bit always reads as "0." Setting this bit initiates a reset sequence: the frame being generated will be truncated.	N	Y
R12316 R0x301C	**7:0**	**0x0000**	**mode_select_ (RW)**	Y	N
	This bit is an alias of R0x301A–B[2].				
R12317 R0x301D	**7:0**	**0x0000**	**image_orientation_ (RW)**		
	7:2	X	Reserved		
	1	0x0000	Vertical Flip This bit is an alias of R0x3040[1].	Y	YM
	0	0x0000	Horizontal Mirror This bit is an alias of R0x3040[0].	Y	YM

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12318 R0x301E	**15:0**	**0x00A8**	**data_pedestal_ (RW)**	N	Y
	Constant offset that is added to the ADC output for all visible pixels in order to set the black level to a value greater than 0. Read-only. Can be made read/write by clearing R0x301A–B[3].				
R12321 R0x3021	**7:0**	**0x0000**	**software_reset_ (RW)**	N	Y
	This bit is an alias of R0x301A–B[0].				
R12322 R0x3022	**7:0**	**0x0000**	**grouped_parameter_hold_ (RW)**	N	N
	This bit is an alias of R0x301A–B[15].				
R12323 R0x3023	**7:0**	**0x0000**	**mask_corrupted_frames_ (RW)**	N	N
	This bit is an alias of R0x301A–B[9].				
R12324 R0x3024	**7:0**	**0x0000**	**pixel_order_ (RO)**	N	N
	00 = First row is GreenR/Red, first pixel is GreenR 01 = First row is GreenR/Red, first pixel is Red 02 = First row is Blue/GreenB, first pixel is Blue 03 = First row is Blue/GreenB, first pixel is GreenB The value in this register changes as a function of R0x3040[1:0].				
R12326 R0x3026	**15:0**	**0xFFFF**	**gpi_status (RW)**		
	15:13	0x0007	Standby Pin Select Associate the standby function with an active-high input pin 0 = associate with GPI0 1 = associate with GPI1 2 = associate with GPI2 3 = associate with GPI3 4–6 = RESERVED 7 = standby function cannot be controlled by any pin Must be set to 7 if reset[8] = 0.	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12326 R0x3026	12:10	0x0007	OE_N Pin Select Associate the output-enable function with an active-low input pin 0 = associate with GPI0 1 = associate with GPI1 2 = associate with GPI2 3 = associate with GPI3 4–6 = RESERVED 7 = output-enable function is not controlled by any pin Must be set to 7 if reset[8] = 0. S	N	N
	9:7	0x0007	Trigger Pin Select Associate the trigger function with an active-high input pin 0 = associate with GPI0 1 = associate with GPI1 2 = associate with GPI2 3 = associate with GPI3 4–6 = RESERVED 7 = Trigger function is not controlled by any pin Must be set to 7 if R0x301A–B[8] = 0.	N	N
	6:4	0x0007	SADDR Pin Select Associate the SADDR function with an active-high input pin 0 = associate with GPI0 1 = associate with GPI1 2 = associate with GPI2 3 = associate with GPI3 4–6 = RESERVED 7 = SADDR function is not controlled by any pin Must be set to 7 if R0x301A-B[8] = 0.	N	N
	3	RO	GPI3 Read-only. Return the current state of the GPI3 input pin. Invalid if R0x301A–B[8] = 0.	N	N
	2	RO	GPI2 Read-only. Return the current state of the GPI2 input pin. Invalid if R0x301A–B[8] = 0.	N	N
	1	RO	GPI1 Read-only. Return the current state of the GPI1 input pin. Invalid if R0x301A–B[8] = 0.	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12326 R0x3026	0	RO	GPI0 Read-only. Return the current state of the GPI0 input pin. Invalid if R0x301A–B[8] = 0.	N	N
R12328 R0x3028	**15:0**	**0x000D**	**analogue_gain_code_global_ (RW)**	Y	N
	Writing a gain code to this register is equivalent to writing that code to each of the 4 color-specific gain code registers. Reading from this register returns the value most recently written to the analogue_gain_code_greenR register.				
R12330 R0x302A	**15:0**	**0x000D**	**analogue_gain_code_greenR_ (RW)**	Y	N
	The gain code written to this register sets the gain for green pixels on red/green rows of the pixel array.				
R12332 R0x302C	**15:0**	**0x000D**	**analogue_gain_code_red_ (RW)**	Y	N
	The gain code written to this register sets the gain for red pixels.				
R12334 R0x302E	**15:0**	**0x000D**	**analogue_gain_code_blue_ (RW)**	Y	N
	The gain code written to this register sets the gain for blue pixels.				
R12336 R0x3030	**15:0**	**0x000D**	**analogue_gain_code_greenB_ (RW)**	Y	N
	The gain code written to this register sets the gain for green pixels on blue/green rows of the pixel array.				
R12338 R0x3032	**15:0**	**0x0100**	**digital_gain_greenR_ (RW)**	Y	N
	Digital gain applied to green pixels on red/green rows of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x3056[11:9].				
R12340 R0x3034	**15:0**	**0x0100**	**digital_gain_red_ (RW)**	Y	N
	Digital gain applied to red pixels of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x305A[11:9].				
R12342 R0x3036	**15:0**	**0x0100**	**digital_gain_blue_ (RW)**	Y	N
	Digital gain applied to blue pixels of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x3058[11:9].				
R12344 R0x3038	**15:0**	**0x0100**	**digital_gain_greenB_ (RW)**	Y	N
	Digital gain applied to green pixels on blue/green rows of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x305C[11:9].				
R12346 R0x303A	**7:0**	**0x000A**	**smia_version_ (RO)**	N	N
	Return the value 10 to indicate an implementation of revision 1.0 of the SMIA specification. Read-only.				
R12347 R0x303B	**7:0**	**0x00FF**	**frame_count_ (RO)**	Y	N
	In the soft standby state this counter is set to 0xFF. In the streaming state this counter increments by 1 (modulo 255) at the start of each frame. The counter is incremented for both good frames and bad (corrupted) frames - its behavior is not affected by the state of R0x301A–B[9] (mask_corrupted_frames). After entry to the streaming state, the first frame will show a frame count of 0x01 in its embedded data. Read-only.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12348 R0x303C	**15:0**	**0x0000**	**frame_status (RO)**		
	15:2	X	Reserved		
	1	RO	Standby status This bit tells you whether the sensor is in standby state. Can be polled after standby is entered to see when the real low-power state is entered; which can happen at the end of row or frame depending on bit 0x301A[4]. The bit actually reflects the internal signal standby_gated.	N	N
	0	RO	Framesync Set on register write and reset on framesync. Acts as debug flag to verify that register writes completed before last framesync.	N	N
R12352 R0x3040	**15:0**	**0x0024**	**read_mode (RW)**		
	15:14	0x0000	Special LINE_VALID This feature is not working. Keep setting at 00. 00 = Normal behavior of LINE_VALID 01 = LINE_VALID is driven continuously (continue generating LINE_VALID during vertical blanking) 10 = LINE_VALID is driven continuously as LINE_VALID XOR FRAME_VALID.	N	N
	13	X	Reserved		
	12	0x0000	Binning summing Enable summing mode for binning.	Y	N
	11	0x0000	x bin enable Enable analogue binning in X (column) direction. When set, x_odd_inc must be set to 3 or 7 and y_odd_inc must be set to 1, along with other register changes.	Y	N
	10	0x0000	xy bin enable Enable analogue binning in X and Y (column and row) directions. When set, x_odd_inc and y_odd_inc must be set to 3 or 7, along with other register changes.	Y	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12352 R0x3040	9	0x0000	Low power mode Enables low power mode. This will automatically half the pixel clock speed. Can not be used when pc_speed[2:0] = 4.	Y	YM
	8	X	Reserved		
	7:5	0x0001	X odd increment Increment applied to odd addresses in X (column) direction. 1= Normal readout 3 = Read out alternate pixel pairs to halve the amount of horizontal data in a frame. 7 = Read out 1 of 4 pixel pairs to reduce the amount of horizontal data in a frame by 4.	Y	YM
	4:2	0x0001	Y odd increment Increment applied to odd addresses in Y (row) direction. 1= Normal readout 3 = Read out alternate pixel pairs to halve the amount of vertical data in a frame. 7 = Read out 1 of 4 pixel pairs to reduce the amount of vertical data in a frame by 4.	Y	YM
	1	0x0000	Vertical Flip 0 = Normal readout 1 = Readout is flipped (mirrored) vertically so that the row specified by y_addr_end_ is read out of the sensor first. Setting this bit will change the Bayer pixel order (see R0x3024).	Y	YM
	0	0x0000	Horizontal Mirror 0 = Normal readout 1 = Readout is mirrored horizontally so that the column specified by x_addr_end_ is read out of the sensor first. Setting this bit will change the Bayer pixel order (see R0x3024).	Y	YM

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12358 R0x3046	**15:0**	**0x0600**	**flash (RW)**		
	15	RO	Strobe Reflects the current state of the FLASH output signal. Read-only.	N	N
	14	RO	Triggered Indicates that the FLASH output signal was asserted for the current frame. Read-only.	N	N
	13	0x0000	Xenon Flash Enable Xenon flash. When set, the FLASH output signal will assert for the programmed period (bits [7:0]) during vertical blanking. This is achieved by keeping the integration time equal to one frame, and the pulse width less than the vertical blanking time.	Y	N
	12:11	0x0000	Frame Delay Flash pulse delay measured in frames.	N	N
	10	0x0001	End of Reset 1 = In Xenon mode, the flash is triggered after resetting a frame. 0 = In Xenon mode, the flash is triggered after a frame readout.	N	N
	9	0x0001	Every Frame 1 = Flash should be enabled every frame. 0 = Flash should be enabled for 1 frame only.	N	N
	8	0x0000	LED Flash Enable LED flash. When set, the FLASH output signal will assert prior to the start of the resetting of a frame and will remain asserted until the end of the frame readout.	Y	Y
	7:0	X	Reserved		
R12360 R0x3048	**15:0**	**0x0008**	**flash_count (RW)**	N	N
	Length of flash pulse when Xenon flash is enabled. The value specifies the length in units of 256 x PIXCLK cycle increments (by default, PIXCLK = system_clock). When the Xenon count is set to its maximum value (0x3FF), the flash pulse will automatically be truncated prior to the readout of the first row, giving the longest pulse possible.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12374 R0x3056	**15:0**	**0x0234**	**green1_gain (RW)**		
	15:12	X	Reserved		
	11:9	0x0001	Digital Gain Digital Gain. Legal values 1-7.	Y	N
	8:7	0x0000	Analog Gain Analog gain = (bit [8] + 1) * (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0034	Initial Gain Initial gain = bits [6:0] * 1/32.	Y	N
R12376 R0x3058	**15:0**	**0x0234**	**blue_gain (RW)**		
	15:12	X	Reserved		
	11:9	0x0001	Digital Gain Digital Gain. Legal values 1-7.	Y	N
	8:7	0x0000	Analog Gain Analog gain = (bit [8] + 1) * (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0034	Initial Gain Initial gain = bits [6:0] * 1/32.	Y	N
R12378 R0x305A	**15:0**	**0x0234**	**red_gain (RW)**		
	15:12	X	Reserved		
	11:9	0x0001	Digital Gain Digital Gain. Legal values 1-7.	Y	N
	8:7	0x0000	Analog Gain Analog gain = (bit [8] + 1) * (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0034	Initial Gain Initial gain = bits [6:0] * 1/32.	Y	N
R12380 R0x305C	**15:0**	**0x0234**	**green2_gain (RW)**		
	15:12	X	Reserved		
	11:9	0x0001	Digital Gain Digital Gain. Legal values 1-7.	Y	N
	8:7	0x0000	Analog Gain Analog gain = (bit [8] + 1) * (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0034	Initial Gain Initial gain = bits [6:0] * 1/32.	Y	N
R12382 R0x305E	**15:0**	**0x0234**	**global_gain (RW)**	Y	N
	Writing a gain to this register is equivalent to writing that code to each of the 4 color-specific gain registers. Reading from this register returns the value most recently written to the green1_gain register.				
R12394 R0x306A	**15:0**	**0x0000**	**datapath_status (RW)**		
	15:5	X	Reserved		
	4	0x0000	Reserved	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12394 R0x306A	3	0x0000	Frame time exceeded If the y_output_size is so large that the total number of lines output on the odp exceeds the total number of lines allowed by frame_length_lines, the "frame time exceeded" error will be flagged. The general solution to this error is to reduce y_output_size to match the size of the frame being generated by the sensor_core. Once this bit is set, the condition that caused the error must be cleared, then write a “1” to this bit position to clear it.	N	N
	2	0x0000	Line time exceeded If the odp clock rate and x_output_size do not allow an output line to be generated within the time allowed by line_length_pck, the "line time exceeded" error will be flagged. The general solution to this error is first to reduce x_output_size to match the size of the frame being generated by the sensor_core and then (if necessary) increase line_length_pck to allow time for the output line. Once this bit is set, the condition that caused the error must be cleared, then write a “1” to this bit position to clear it.	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12394 R0x306A	1	0x0000	FIFO overflow If the odp data rate is lower than the sensor_core data rate and x_output_size is large enough, the output buffer can overflow, and the "FIFO overflow" error will be flagged. The FIFO is sized to accommodate a full-sizeframe from the sensor core, so this error can only occur when x_output_size is unnecessarily large. The general solution to this error is to reduce x_output_size. Once this bit is set, the condition that caused the error must be cleared, then write a "1" to this bit position to clear it.	N	N
	0	0x0000	FIFO underflow If the output buffer underflows, the "FIFO underflow" error will be flagged. There is no known setup scenario that will stimulate this error. Once this bit is set, you must clear the condition that caused the error then write a "1" to this bit position to clear it.	N	N
R12398 R0x306E	**15:0**	**0x9080**	**datapath_select (RW)**		
	15:13	0x0004	Slew-rate control Parallel Interface Selects the slew (edge) rate for the DOUT[11:0], SHUTTER, FRAME_VALID, LINE_VALID and FLASH outputs. Only affects SHUTTER and FLASH outputs when parallel data output is disabled. The value 7 results in the fastest edge rates on these signals. Slowing down the edge rate can reduce ringing and electro-magnetic emissions.	N	N
	12:10	0x0004	Slew-rate control PIXCLK Selects the slew (edge) rate for the PIXCLK output. Has no effect when parallel data output is disabled. The value 7 results in the fastest edge rates on this signal. Slowing down the edge rate can reduce ringing and electromagnetic emissions.	N	N

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12398 R0x306E	9:8	X	Reserved		
	7	0x0001	Profile SMIA profile mode. Should only be changed in standby, and with attention to other clock settings. 0 = Profile 0 1 = Profile 1/2.	N	Y
	6:5	X	Reserved		
	4	0x0000	True Bayer mode Enables true Bayer scaling mode.	N	N
	3:0	X	Reserved		
R12400 R0x3070	**15:0**	**0x0000**	**test_pattern_mode_ (RW)**	N	Y
	0 = Normal operation: Generate output data from pixel array 1 = Solid color test pattern. 2 = 100% color bar test pattern 3 = Fade to gray color bar test pattern 4 = PN9 Link integrity test pattern 256 = Marching 1s test pattern other = Reserved.				
R12402 R0x3072	**15:0**	**0x0000**	**test_data_red_ (RW)**	N	Y
	The value for red pixels in the Bayer data used for the solid color test pattern and the test cursors.				
R12404 R0x3074	**15:0**	**0x0000**	**test_data_green1_ (RW)**	N	Y
	The value for green pixels in red/green rows of the Bayer data used for the solid color test pattern and the test cursors.				
R12406 R0x3076	**15:0**	**0x0000**	**test_data_blue_ (RW)**	N	Y
	The value for blue pixels in the Bayer data used for the solid color test pattern and the test cursors.				
R12408 R0x3078	**15:0**	**0x0000**	**test_data_green2 (RW)**	N	Y
	The value for green pixels in blue/green rows of the Bayer data used for the solid color test pattern and the test cursors.				
R12448 R0x30A0	**15:0**	**0x0001**	**x_even_inc_ (RO)**	N	N
	Read-only.				
R12450 R0x30A2	**15:0**	**0x0001**	**x_odd_inc_ (RW)**	Y	YM
	This register field is an alias of R0x3040[7:5]				
R12452 R0x30A4	**15:0**	**0x0001**	**y_even_inc_ (RO)**	N	N
	Read-only.				
R12454 R0x30A6	**15:0**	**0x0001**	**y_odd_inc_ (RW)**	Y	YM
	This register field is an alias of R0x3040[4:2]				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12640 R0x3160	**15:0**	**0x0000**	**global_seq_trigger (RW)**		
	15:10	X	Reserved		
	9	RO	Grst Rd Read-Only. Global reset read sequence indicator.	N	N
	8	RO	Grst Sequence Read-only. Global reset sequence indicator.	N	N
	7:3	X	Reserved		
	2	0x0000	Global Flash 0 = When a global reset sequence is triggered, the FLASH output will remain negated. 1 = When a global reset sequence is triggered, the FLASH output will pulse during the integration phase.	N	Y
	1	0x0000	Global Bulb 0 = Shutter open is triggered from bit[0] and shutter close is timed from the trigger point. 1 = Shutter open and close are triggered from bit[0]. This corresponds to the shutter "B" setting on a traditional camera, where "B" originally stood for "Bulb" (the shutter setting used for synchronization with a magnesium foil flash bulb) and was later considered to stand for "Brief" (an exposure that was longer than the shutter could automatically accommodate).	N	Y
	0	0x0000	Global Trigger When bit[1]=0, a 0-to-1 transition of this bit initiates (triggers) a global reset sequence. When bit[1]=1, a 0-to-1 transition of this bit initiates a global reset sequence, and leaves the shutter open; a 1-to-0 transition of this bit closes the shutter. These operations can also be controlled from the signal interface by enabling one of the GPI[3:0] signals as a trigger input.	N	Y
R12642 R0x3162	**15:0**	**0x0050**	**global_rst_end (RW)**	N	N
	Controls the duration of the global reset row reset phase. A value of N gives a duration of N * 512 / vt_pix_clk_freq_mhz.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12644 R0x3164	15:0	0x0078	global_shutter_start (RW)	N	N
	Controls the delay before the assertion of the SHUTTER output during a global reset sequence. A value of *N* gives an assertion time of N * 512 / vt_pix_clk_freq_mhz timed from the end of row that was in progress when the global reset sequence was triggered.				
R12646 R0x3166	15:0	0x00A0	global_read_start (RW)	N	N
	Controls the delay before the start of the global reset readout phase (equivalent to the end of global reset integration phase). A value of *N* gives a delay of N * 512 / vt_pix_clk_freq_mhz. The integration time is given by (global_read_start - global_rst_end) * 512 / vt_pix_clk_freq_mhz.				
R12776 R0x31E8	15:0	0x0000	horizontal_cursor_position_ (RW)	N	N
	Specify the start column for the test cursor.				
R12778 R0x31EA	15:0	0x0000	vertical_cursor_position_ (RW)	N	N
	Specify the start column for the test cursor.				
R12780 R0x31EC	15:0	0x0000	horizontal_cursor_width_ (RW)	N	N
	Specify the width, in rows, of the horizontal test cursor. A width of 0 disables the cursor.				
R12782 R0x31EE	15:0	0x0000	vertical_cursor_width_ (RW)	N	N
	Specify the width, in columns, of the vertical test cursor. A width of 0 disables the cursor.				
R12796 R0x31FC	15:0	0x03020	i2c_ids	N	N
	I2C address registers.				
R13824 R0x3600	15:0	0x0000	P_GR_P0Q0 (RW)	N	N
	P0 coefficient for Q0 for greenR.				
R13826 R0x3602	15:0	0x0000	P_GR_P0Q1 (RW)	N	N
R13828 R0x3604	15:0	0x0000	P_GR_P0Q2 (RW)	N	N
	P0 coefficient for Q2 for greenR.				
R13830 R0x3606	15:0	0x0000	P_GR_P0Q3 (RW)	N	N
	P0 coefficient for Q3 for greenR.				
R13832 R0x3608	15:0	0x0000	P_GR_P0Q4 (RW)	N	N
	P0 coefficient for Q4 for greenR.				
R13834 R0x360A	15:0	0x0000	P_RD_P0Q0 (RW)	N	N
	P0 coefficient for Q0 for red.				
R13836 R0x360C	15:0	0x0000	P_RD_P0Q1 (RW)	N	N
	P0 coefficient for Q1 for red.				
R13838 R0x360E	15:0	0x0000	P_RD_P0Q2 (RW)	N	N
	P0 coefficient for Q2 for red.				
R13840 R0x3610	15:0	0x0000	P_RD_P0Q3 (RW)	N	N
	P0 coefficient for Q3 for red.				
R13842 R0x3612	15:0	0x0000	P_RD_P0Q4 (RW)	N	N
	P0 coefficient for Q4 for red.				
R13844 R0x3614	15:0	0x0000	P_BL_P0Q0 (RW)	N	N
	P0 coefficient for Q0 for blue.				
R13846 R0x3616	15:0	0x0000	P_BL_P0Q1 (RW)	N	N
	P0 coefficient for Q1 for blue.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R13848 R0x3618	15:0	0x0000	P_BL_P0Q2 (RW)	N	N
	P0 coefficient for Q2 for blue.				
R13850 R0x361A	15:0	0x0000	P_BL_P0Q3 (RW)	N	N
	P0 coefficient for Q3 for blue.				
R13852 R0x361C	15:0	0x0000	P_BL_P0Q4 (RW)	N	N
	P0 coefficient for Q4 for blue.				
R13854 R0x361E	15:0	0x0000	P_GB_P0Q0 (RW)	N	N
	P0 coefficient for Q0 for greenB				
R13856 R0x3620	15:0	0x0000	P_GB_P0Q1 (RW)	N	N
	P0 coefficient for Q1 for greenB.				
R13858 R0x3622	15:0	0x0000	P_GB_P0Q2 (RW)	N	N
	P0 coefficient for Q2 for greenB.				
R13860 R0x3624	15:0	0x0000	P_GB_P0Q3 (RW)	N	N
	P0 coefficient for Q3 for greenB.				
R13862 R0x3626	15:0	0x0000	P_GB_P0Q4 (RW)	N	N
	P0 coefficient for Q4 for greenB.				
R13888 R0x3640	15:0	0x0000	P_GR_P1Q0 (RW)	N	N
	P1 coefficient for Q0 for greenR.				
R13890 R0x3642	15:0	0x0000	P_GR_P1Q1 (RW)	N	N
	P1 coefficient for Q1 for greenR.				
R13892 R0x3644	15:0	0x0000	P_GR_P1Q2 (RW)	N	N
	P1 coefficient for Q2 for greenR.				
R13894 R0x3646	15:0	0x0000	P_GR_P1Q3 (RW)	N	N
	P1 coefficient for Q3 for greenR.				
R13896 R0x3648	15:0	0x0000	P_GR_P1Q4 (RW)	N	N
	P1 coefficient for Q4 for greenR.				
R13898 R0x364A	15:0	0x0000	P_RD_P1Q0 (RW)	N	N
	P1 coefficient for Q0 for red.				
R13900 R0x364C	15:0	0x0000	P_RD_P1Q1 (RW)	N	N
	P1 coefficient for Q1 for red.				
R13902 R0x364E	15:0	0x0000	P_RD_P1Q2 (RW)	N	N
	P1 coefficient for Q2 for red.				
R13904 R0x3650	15:0	0x0000	P_RD_P1Q3 (RW)	N	N
	P1 coefficient for Q3 for red.				
R13906 R0x3652	15:0	0x0000	P_RD_P1Q4 (RW)	N	N
	P1 coefficient for Q4 for red.				
R13908 R0x3654	15:0	0x0000	P_BL_P1Q0 (RW)	N	N
	P1 coefficient for Q0 for blue.				
R13910 R0x3656	15:0	0x0000	P_BL_P1Q1 (RW)	N	N
	P1 coefficient for Q1 for blue.				
R13912 R0x3658	15:0	0x0000	P_BL_P1Q2 (RW)	N	N
	P1 coefficient for Q2 for blue.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R13914 R0x365A	15:0	0x0000	P_BL_P1Q3 (RW)	N	N
	P1 coefficient for Q3 for blue.				
R13916 R0x365C	15:0	0x0000	P_BL_P1Q4 (RW)	N	N
	P1 coefficient for Q4 for blue.				
R13918 R0x365E	15:0	0x0000	P_GB_P1Q0 (RW)	N	N
	P1 coefficient for Q0 for greenB.				
R13920 R0x3660	15:0	0x0000	P_GB_P1Q1 (RW)	N	N
	P1 coefficient for Q1 for greenB.				
R13922 R0x3662	15:0	0x0000	P_GB_P1Q2 (RW)	N	N
	P1 coefficient for Q2 for greenB.				
R13924 R0x3664	15:0	0x0000	P_GB_P1Q3 (RW)	N	N
	P1 coefficient for Q3 for greenB.				
R13926 R0x3666	15:0	0x0000	P_GB_P1Q4 (RW)	N	N
	P1 coefficient for Q4 for greenB.				
R13952 R0x3680	15:0	0x0000	P_GR_P2Q0 (RW)	N	N
	P2 coefficient for Q0 for greenR.				
R13954 R0x3682	15:0	0x0000	P_GR_P2Q1 (RW)	N	N
	P2 coefficient for Q1 for greenR.				
R13956 R0x3684	15:0	0x0000	P_GR_P2Q2 (RW)	N	N
	P2 coefficient for Q2 for greenR.				
R13958 R0x3686	15:0	0x0000	P_GR_P2Q3 (RW)	N	N
	P2 coefficient for Q3 for greenR.				
R13960 R0x3688	15:0	0x0000	P_GR_P2Q4 (RW)	N	N
	P2 coefficient for Q4 for greenR.				
R13962 R0x368A	15:0	0x0000	P_RD_P2Q0 (RW)	N	N
	P2 coefficient for Q0 for red.				
R13964 R0x368C	15:0	0x0000	P_RD_P2Q1 (RW)	N	N
	P2 coefficient for Q1 for red.				
R13966 R0x368E	15:0	0x0000	P_RD_P2Q2 (RW)	N	N
	P2 coefficient for Q2 for red.				
R13968 R0x3690	15:0	0x0000	P_RD_P2Q3 (RW)	N	N
	P2 coefficient for Q3 for red.				
R13970 R0x3692	15:0	0x0000	P_RD_P2Q4 (RW)	N	N
	P2 coefficient for Q4 for red.				
R13972 R0x3694	15:0	0x0000	P_BL_P2Q0 (RW)	N	N
	P2 coefficient for Q0 for blue.				
R13974 R0x3696	15:0	0x0000	P_BL_P2Q1 (RW)	N	N
	P2 coefficient for Q1 for blue.				
R13976 R0x3698	15:0	0x0000	P_BL_P2Q2 (RW)	N	N
	P2 coefficient for Q2 for blue.				
R13978 R0x369A	15:0	0x0000	P_BL_P2Q3 (RW)	N	N
	P2 coefficient for Q3 for blue.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R13980 R0x369C	15:0	0x0000	P_BL_P2Q4 (RW)	N	N
	P2 coefficient for Q4 for blue.				
R13982 R0x369E	15:0	0x0000	P_GB_P2Q0 (RW)	N	N
	P2 coefficient for Q0 for greenB.				
R13984 R0x36A0	15:0	0x0000	P_GB_P2Q1 (RW)	N	N
	P2 coefficient for Q1 for greenB.				
R13986 R0x36A2	15:0	0x0000	P_GB_P2Q2 (RW)	N	N
	P2 coefficient for Q2 for greenB.				
R13988 R0x36A4	15:0	0x0000	P_GB_P2Q3 (RW)	N	N
	P2 coefficient for Q3 for greenB.				
R13990 R0x36A6	15:0	0x0000	P_GB_P2Q4 (RW)	N	N
	P2 coefficient for Q4 for greenB.				
R14016 R0x36C0	15:0	0x0000	P_GR_P3Q0 (RW)	N	N
	P3 coefficient for Q0 for greenR.				
R14018 R0x36C2	15:0	0x0000	P_GR_P3Q1 (RW)	N	N
	P3 coefficient for Q1 for greenR.				
R14020 R0x36C4	15:0	0x0000	P_GR_P3Q2 (RW)	N	N
	P3 coefficient for Q2 for greenR.				
R14022 R0x36C6	15:0	0x0000	P_GR_P3Q3 (RW)	N	N
	P3 coefficient for Q3 for greenR.				
R14024 R0x36C8	15:0	0x0000	P_GR_P3Q4 (RW)	N	N
	P3 coefficient for Q4 for greenR.				
R14026 R0x36CA	15:0	0x0000	P_RD_P3Q0 (RW)	N	N
	P3 coefficient for Q0 for red.				
R14028 R0x36CC	15:0	0x0000	P_RD_P3Q1 (RW)	N	N
	P3 coefficient for Q1 for red.				
R14030 R0x36CE	15:0	0x0000	P_RD_P3Q2 (RW)	N	N
	P3 coefficient for Q2 for red.				
R14032 R0x36D0	15:0	0x0000	P_RD_P3Q3 (RW)	N	N
	P3 coefficient for Q3 for red.				
R14034 R0x36D2	15:0	0x0000	P_RD_P3Q4 (RW)	N	N
	P3 coefficient for Q4 for red.				
R14036 R0x36D4	15:0	0x0000	P_BL_P3Q0 (RW)	N	N
	P3 coefficient for Q0 for blue.				
R14038 R0x36D6	15:0	0x0000	P_BL_P3Q1 (RW)	N	N
	P3 coefficient for Q1 for blue.				
R14040 R0x36D8	15:0	0x0000	P_BL_P3Q2 (RW)	N	N
	P3 coefficient for Q2 forblue.				
R14042 R0x36DA	15:0	0x0000	P_BL_P3Q3 (RW)	N	N
	P3 coefficient for Q3 for blue.				
R14044 R0x36DC	15:0	0x0000	P_BL_P3Q4 (RW)	N	N
	P3 coefficient for Q4 for blue.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R14046 R0x36DE	15:0	0x0000	P_GB_P3Q0 (RW)	N	N
	P3 coefficient for Q0 for greenB.				
R14048 R0x36E0	15:0	0x0000	P_GB_P3Q1 (RW)	N	N
	P3 coefficient for Q1 for greenB.				
R14050 R0x36E2	15:0	0x0000	P_GB_P3Q2 (RW)	N	N
	P3 coefficient for Q2 for greenB.				
R14052 R0x36E4	15:0	0x0000	P_GB_P3Q3 (RW)	N	N
	P3 coefficient for Q3 for greenB.				
R14054 R0x36E6	15:0	0x0000	P_GB_P3Q4 (RW)	N	N
	P3 coefficient for Q4 for greenB.				
R14080 R0x3700	15:0	0x0000	P_GR_P4Q0 (RW)	N	N
	P4 coefficient for Q0 for greenR.				
R14082 R0x3702	15:0	0x0000	P_GR_P4Q1 (RW)	N	N
	P4 coefficient for Q1 for greenR.				
R14084 R0x3704	15:0	0x0000	P_GR_P4Q2 (RW)	N	N
	P4 coefficient for Q2 for greenR.				
R14086 R0x3706	15:0	0x0000	P_GR_P4Q3 (RW)	N	N
	P4 coefficient for Q3 for greenR.				
R14088 R0x3708	15:0	0x0000	P_GR_P4Q4 (RW)	N	N
	P4 coefficient for Q4 for greenR.				
R14090 R0x370A	15:0	0x0000	P_RD_P4Q0 (RW)	N	N
	P4 coefficient for Q0 for red.				
R14092 R0x370C	15:0	0x0000	P_RD_P4Q1 (RW)	N	N
	P4 coefficient for Q1 for red.				
R14094 R0x370E	15:0	0x0000	P_RD_P4Q2 (RW)	N	N
	P4 coefficient for Q2 for red.				
R14096 R0x3710	15:0	0x0000	P_RD_P4Q3 (RW)	N	N
	P4 coefficient for Q3 for red.				
R14098 R0x3712	15:0	0x0000	P_RD_P4Q4 (RW)	N	N
	P4 coefficient for Q4 for red.				
R14100 R0x3714	15:0	0x0000	P_BL_P4Q0 (RW)	N	N
	P4 coefficient for Q0 for blue.				
R14102 R0x3716	15:0	0x0000	P_BL_P4Q1 (RW)	N	N
	P4 coefficient for Q1 for blue.				
R14104 R0x3718	15:0	0x0000	P_BL_P4Q2 (RW)	N	N
	P4 coefficient for Q2 for blue.				
R14106 R0x371A	15:0	0x0000	P_BL_P4Q3 (RW)	N	N
	P4 coefficient for Q3 for blue.				
R14108 R0x371C	15:0	0x0000	P_BL_P4Q4 (RW)	N	N
	P4 coefficient for Q4 for blue.				
R14110 R0x371E	15:0	0x0000	P_GB_P4Q0 (RW)	N	N
	P4 coefficient for Q0 for greenB.				

Table 22: 3: Manufacturer Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R14112 R0x3720	**15:0**	**0x0000**	**P_GB_P4Q1 (RW)**	N	N
	P4 coefficient for Q1 for greenB.				
R14114 R0x3722	**15:0**	**0x0000**	**P_GB_P4Q2 (RW)**	N	N
	P4 coefficient for Q2 for greenB.				
R14116 R0x3724	**15:0**	**0x0000**	**P_GB_P4Q3 (RW)**	N	N
	P4 coefficient for Q3 for greenB.				
R14118 R0x3726	**15:0**	**0x0000**	**P_GB_P4Q4 (RW)**	N	N
	P4 coefficient for Q4 for greenB.				
R14208 R0x3780	**15:0**	**0x0000**	**SC_ENABLE (WO)**		
	15	0x0000	Enable lens shading correction	N	N
	14:0	X	Reserved		
	When SC_ENABLE bit is set, _sc will generate function and correct stream of pixels. When not set, _sc will bypass data.				
R14210 R0x3782	**15:0**	**0x0000**	**ORIGIN_C (RW)**	N	N
	Origin of function: Applied as offset to X (col) coordinate of pixel.				
R14212 R0x3784	**15:0**	**0x0000**	**ORIGIN_R (RW)**	N	N
	Origin of function: Applied as offset to Y (row) coordinate of pixel.				

System States

The system states of the sensor are represented as a state diagram below and described in subsequent sections. The effect of RESET_BAR on the system state and the configuration of the PLL in the different states are shown in Figure 34.

Figure 34: Sensor System States

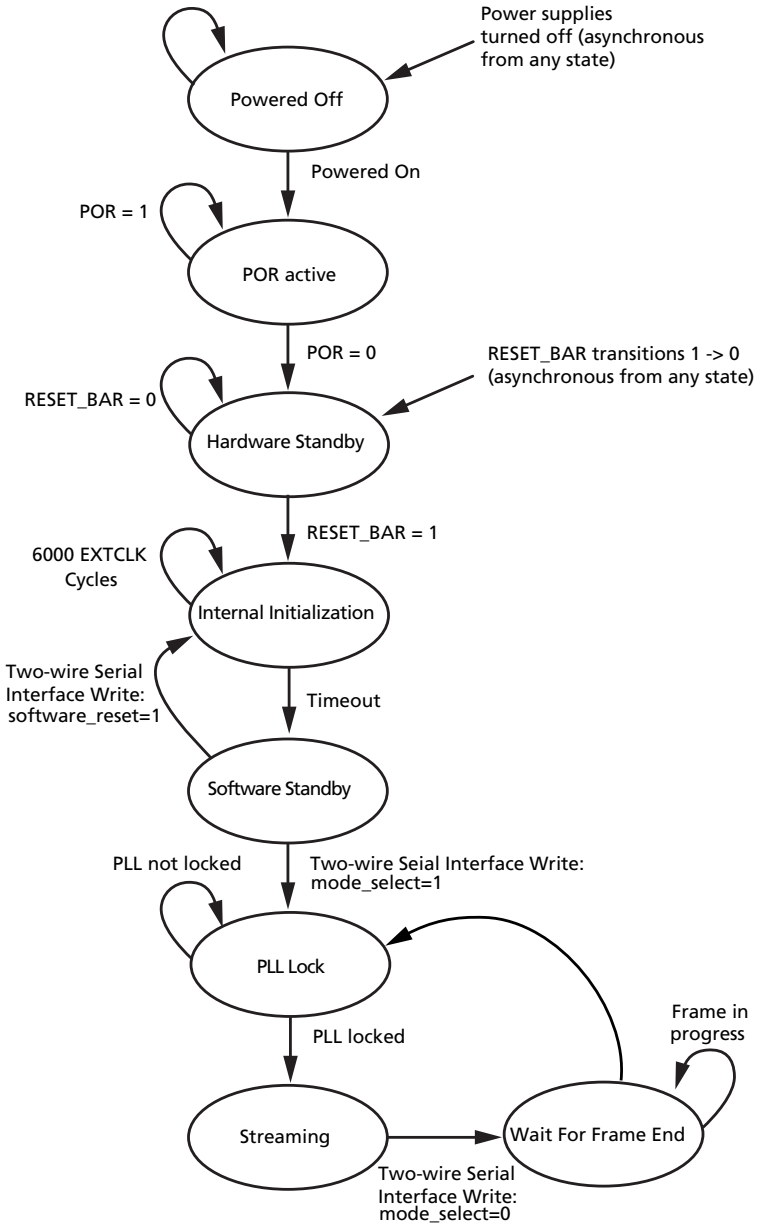
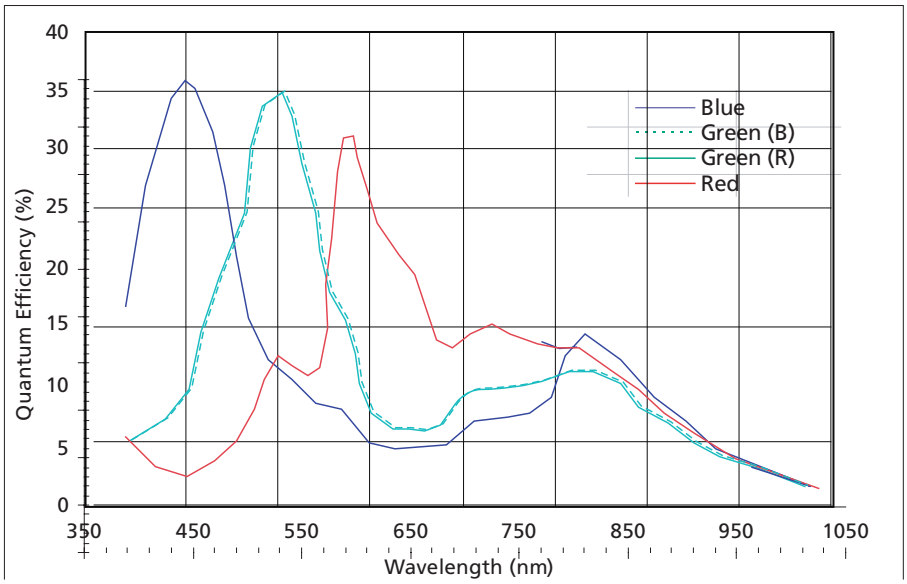


Table 23: RESET_BAR and PLL in System States

State	RESET_BAR	PLL
Powered off	x	VCO Powered-down
Hardware standby	0	
Internal Initialization	1	
Software standby		
PLL Lock		VCO powering up and locking, PLL output bypassed
Streaming		VCO running, PLL clock outputs active
Wait for frame end		

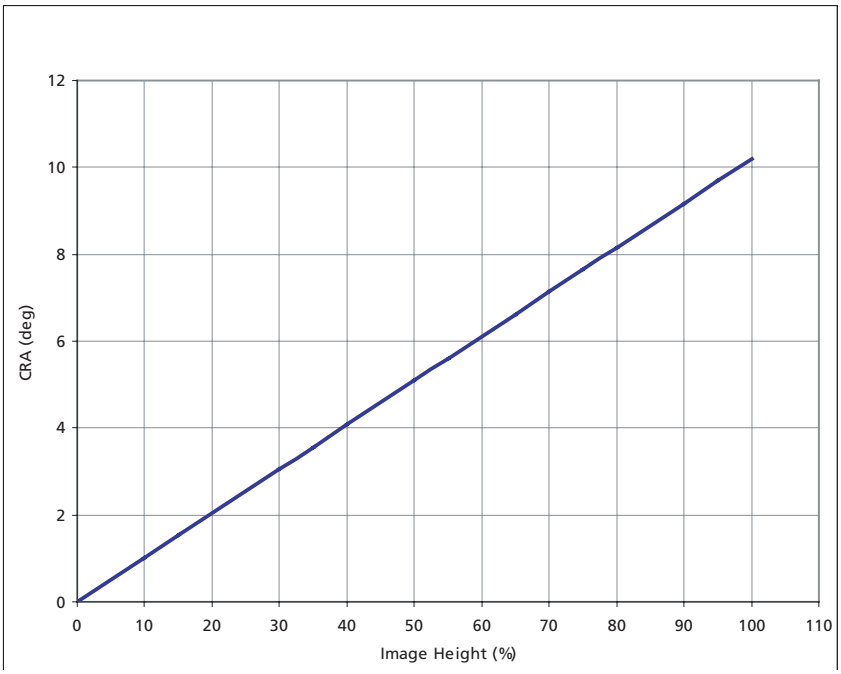
Quantum Efficiency

Figure 35: Quantum Efficiency



Spectral Characteristics

Figure 36: CRA vs. Image Height

CRA vs. Image Height Plot	Image Height (%)	Image Height (mm)	CRA (Deg)
	0	0	0
	5	0.179	0.51
	10	0.357	1.02
	15	0.536	1.53
	20	0.714	2.04
	25	0.893	2.55
	30	1.071	3.06
	35	1.250	3.57
	40	1.428	4.08
	45	1.607	4.59
	50	1.785	5.10
	55	1.964	5.60
	60	2.142	6.11
	65	2.321	6.62
	70	2.499	7.13
	75	2.678	7.64
	80	2.856	8.15
	85	3.035	8.66
	90	3.213	9.17
	95	3.392	9.68
	100	3.570	10.19

Timing Specifications

Power-up

It is recommended to simultaneously apply VDD, VDDIO, and VDDPLL first, followed by VAA and VAAPIX. The maximum time allowed between the first and last voltage applied is 500ms.

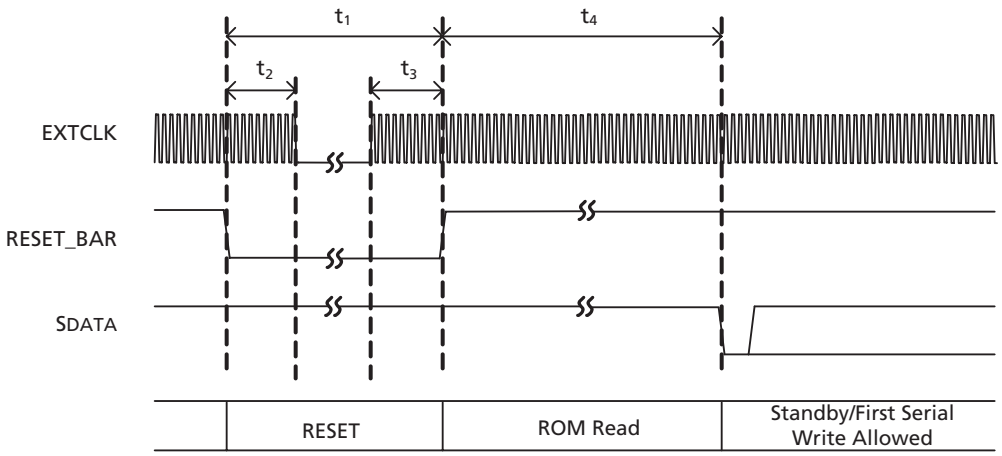
Reset

Two types of reset are available:

- A hard reset is issued by toggling RESET_BAR.
- A soft reset is issues by writing commands through the serial interface.

Hard Reset

Figure 37: Hard Reset



A hard reset sequence to the camera can be activated by the following steps:

1. Wait for all supplies to be stable.
2. Assert RESET_BAR for at least 30 EXTCLK cycles.
3. De-assert RESET_BAR (input clock must be running for at least 10 EXTCLK cycles).
4. Wait 6,000 clock cycles before using the two-wire serial interface.

Soft Reset

The sensor can be reset under software control by writing "1" to software_reset (R0x0103). A software reset asynchronously resets the sensor, truncating any frame that is in progress; the sensor then starts its internal initialization sequence. At this point, the behavior is exactly the same as for the power-on reset sequence.

Signal State During Reset

Table 24 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during software standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 24: Signal State During Reset

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Enabled. Must be driven to a valid logic level.	
RESET_BAR (XSHUTDOWN)	Input	Enabled. Must be driven to a valid logic level.	
LINE_VALID	Output	High-Z. Can be left disconnected/floating.	
FRAME_VALID	Output		
DOUT[11:0]	Output		
PIXCLK	Output		
SCLK	Input	Enabled. Must be pulled-up or driven to a valid logic level.	
SDATA	I/O	Enabled as an input. Must be pulled-up or driven to a valid logic level.	
FLASH	Output	High-Z.	Logic 0
SHUTTER	Output	High-Z.	Logic 0
GPI[3:0]	Input	Powered down. Can be left disconnected/floating.	
TEST_EN	Input	Enabled. Must be driven to a logic 0.	

Electrical Specifications

Table 25: Electrical Characteristics and Operating Conditions

[f]EXTCLK = 24 MHz; V_{DD} = 1.8V; V_{DDIO} = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Temperature (at junction) = 25°C; C_{LOAD} = 15pF

Symbol	Definition	Conditions	Min	Typ	Max	Units
V_{DD}	Core digital voltage		1.7	1.8	1.9	V
V_{DDIO}	I/O digital voltage		2.4	1.8	1.9	V
			2.4	2.8	3.1	V
V_{AA}	Analog voltage		2.4	2.8	3.1	V
VAAPIX	Pixel supply voltage		2.4	2.8	3.1	V
VDDPLL	PLL supply voltage		2.5	2.8	3.1	V
$I_{DD}1$	Digital operating current	Snapshot, fullres 11 fps	30	40	50	mA
$I_{DDIO}1$	I/O digital operating current	Snapshot V_{DDIO} = 1.8V	10	20	30	mA
$I_{DDIO}1$	I/O digital operating current	Snapshot V_{DDIO} = 2.8V	20	35	50	mA
$I_{AA}1$	Analog operating current	Snapshot	120	165	190	mA
IAAPIX1	Pixel supply current	Snapshot	0.1	1.5	7.0	mA
IDDPLL1	PLL supply current	Snapshot	4.0	5.0	6.5	mA
	Total Power Consumption	Snapshot	457	650	880	mW
$I_{DD}2$	Digital operating current	Preview, binning 30 fps	20	30	40	mA
$I_{DDIO}2$	I/O digital operating current	Preview V_{DDIO} = 1.8	5	15	25	mA
$I_{DDIO}2$	I/O digital operating current	Preview V_{DDIO} = 2.8	10	20	30	mA
$I_{AA}2$	Analog operating current	Preview	120	165	190	mA
IAAPIX2	Pixel supply current	Preview	0.1	3.0	7.0	mA
IDDPLL2	PLL supply current	Preview	4.0	5.0	6.5	mA
	Total Power Consumption	Preview	411	594	726	mW
$I_{DD}2$	Digital operating current	Low power preview, binning 30 fps	10	20	30	mA
$I_{DDIO}2$	I/O digital operating current	Low power preview V_{DDIO} = 1.8	5	15	25	mA
$I_{DDIO}2$	I/O digital operating current	Low power preview V_{DDIO} = 2.8	10	20	30	mA
$I_{AA}2$	Analog operating current	Preview	60	80	95	mA
IAAPIX2	Pixel supply current	Low power preview	0.1	1.5	7.0	mA
IDDPLL2	PLL supply current	Low power preview	4.0	5.0	6.5	mA
	Total Power Consumption	Low power preview	225	334	442	mW
IDDSTDBY1	Digital standby current	Hard standby/EXTCLK En	650	800	950	µA
IDDIOSTDBY1	I/O digital standby current	Standby/EXTCLK En V_{DDIO} = 1.8	5	20	30	µA
IDDIOSTDBY1	I/O digital standby current	Standby/EXTCLK En V_{DDIO} = 2.8	5	35	50	µA
IAASTDBY1	Analog standby current	Standby/EXTCLK En	0.0	0.15	0.5	µA
IAAPIXSTDBY1	Pixel supply standby current	Standby/EXTCLK En	0	0.3	0.5	µA
IDDPLLSTDBY1	PLL standby current	Standby/EXTCLK En	5	16	25	µA
IDDSTDBY2	Digital standby current	Hard standby/EXTCLK Dis	5	20	40	µA
IDDIOSTDBY2	I/O digital standby current	Standby/EXTCLK Dis V_{DDIO} = 1.8	1.0	4	8	µA
IDDIOSTDBY2	I/O digital standby current	Standby/EXTCLK Dis V_{DDIO} = 2.8	1.0	8	15	µA
IAASTDBY2	Analog standby current	Standby/EXTCLK Dis	0.0	0.15	0.5	µA
IAAPIXSTDBY2	Pixel supply standby current	Standby/EXTCLK Dis	0	0.3	0.5	µA
IDDPLLSTDBY2	PLL standby current	Standby/EXTCLK Dis	0	0.2	0.4	µA
IDDSTDBY3	Digital standby current	Soft standby	650	800	950	µA
IDDIOSTDBY3	I/O digital standby current	Soft standby V_{DDIO} = 1.8	5	20	30	µA

Table 25: Electrical Characteristics and Operating Conditions (continued)

[f]EXTCLK = 24 MHz; VDD = 1.8V; VDDIO = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Temperature (at junction) = 25°C; CLOAD = 15pF

Symbol	Definition	Conditions	Min	Typ	Max	Units
IDDIOSTDBY3	I/O digital standby current	Soft standby VDDIO = 2.8	5	35	50	µA
IAASTDBY3	Analog standby current	Soft standby	0.0	0.15	0.5	µA
IAAPIXSTDBY3	Pixel supply standby current	Soft standby	0	.3	.5	µA
IDDPLLSTDBY3	PLL standby current	Soft standby	5	35	60	µA

Table 26: I/O Parameters

[f]EXTCLK = 24 MHz; VDD = 1.8V; VDDIO = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Lighting conditions = 0 lux

Symbols	Definition	Conditions	Min	Max	Units
VIH	Input HIGH voltage	VDDIO = 1.8V	1.4	VDDIO + 0.3	V
VIH	Input HIGH voltage	VDDIO = 2.8V	2.4	VDDIO + 0.3	V
VIL	Input LOW voltage	VDDIO = 1.8V	GND - 0.3	0.4	V
VIL	Input LOW voltage	VDDIO = 2.8V	GND - 0.3	0.8	V
IIN	Input leakage current	No pull-up resistor; Vin = VDD or DGND	-20	20	µA
VOH	Output HIGH voltage	At specified IOH	VDDIO - 0.4V	–	V
VOL	Output LOW voltage	At specified IOL	–	0.4	V
IOH	Output HIGH current	At specified VOH	–	-12	mA
IOL	Output LOW current	At specified VOL	–	9	mA
IOZ	Tri-state output leakage current		–	10	µA

Table 27: Typical Power

Frame Rate	CIF	QVGA	VGA	UXGA	QXGA	QSXGA	Units
Preview	500	500	500	500	505	505	mW
Snapshot	525	525	525	530	535	540	mW

Table 28: Optimized Video Power Consumption

[f]EXTCLK = 24 MHz; VDD = 1.8V; VDDIO = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Lighting conditions = 0 lux.
Reserved registers 0x3174, 0x3176, and 0x3178; set to "0x1212."

VAA	VDDPLL	VDD	VAAPIX	Total
224.72	6.94	32.62	10.95	281.66

I/O Timing

Figure 38: I/O TIming

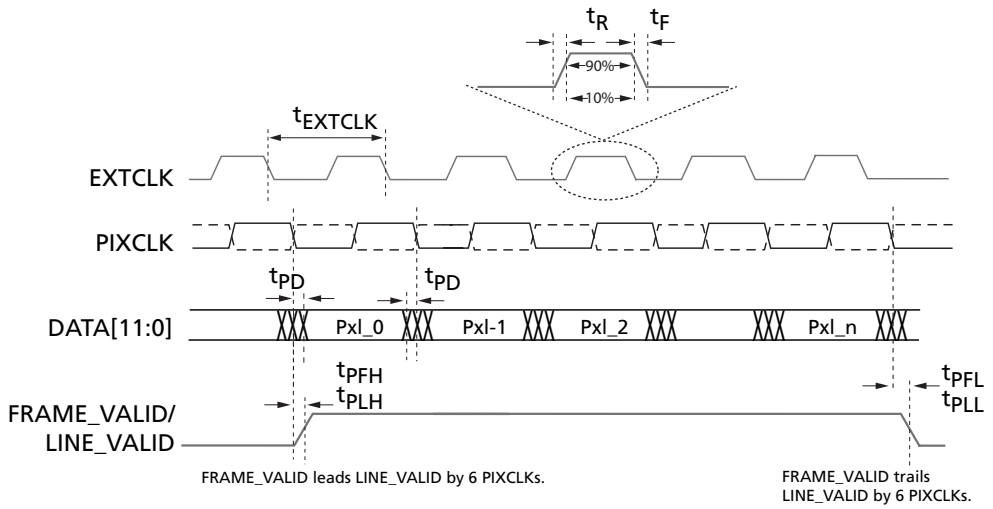


Table 29: I/O Timing

Symbol	Definition	Conditions	Min	Typ	Max	Units
^{f}EXTCLK	Input clock frequency	PLL enabled	6	24	48	MHz
^{t}EXTCLK	Input clock period	PLL enabled	166	41	20	ns
^{t}R	Input clock rise time		0.1	–	1	V/ns
^{t}F	Input clock fall time		0.1	–	1	V/ns
	Clock duty cycle		45	50	55	%
^{t}JITTER	Input clock jitter		–	–	0.3	ns
Output pin slew	Fastest	CLOAD = 15pF	–	0.7	–	V/ns
^{f}PIXCLK	PIXCLK frequency	Default	–	96	–	MHz
^{t}PD	PIXCLK to data valid	Default	–	–	3	ns
^{t}PFH	PIXCLK to FRAME_VALID HIGH	Default	–	–	3	ns
^{t}PLH	PIXCLK to LINE_VALID HIGH	Default	–	–	3	ns
^{t}PFL	PIXCLK to FRAME_VALID LOW	Default	–	–	3	ns
^{t}PLL	PIXCLK to LINE_VALID LOW	Default	–	–	3	ns

Power on Reset (POR)

Figure 39 shows the power on reset.

Figure 39: Power On Reset

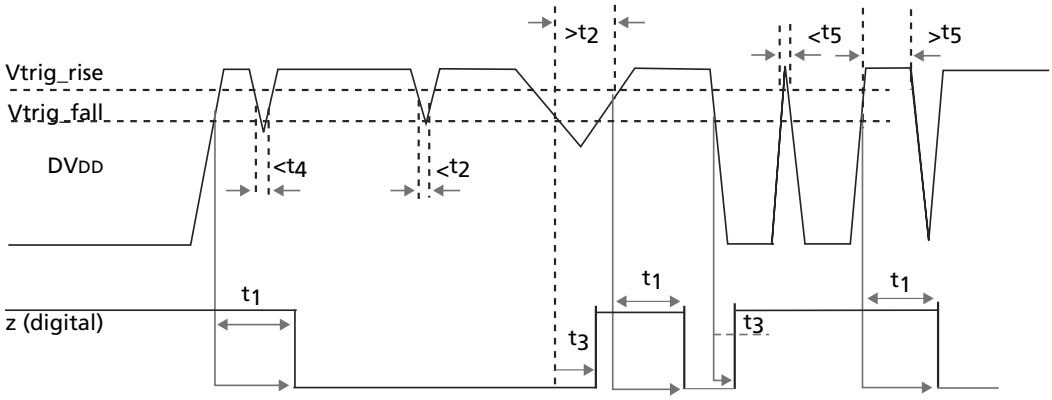
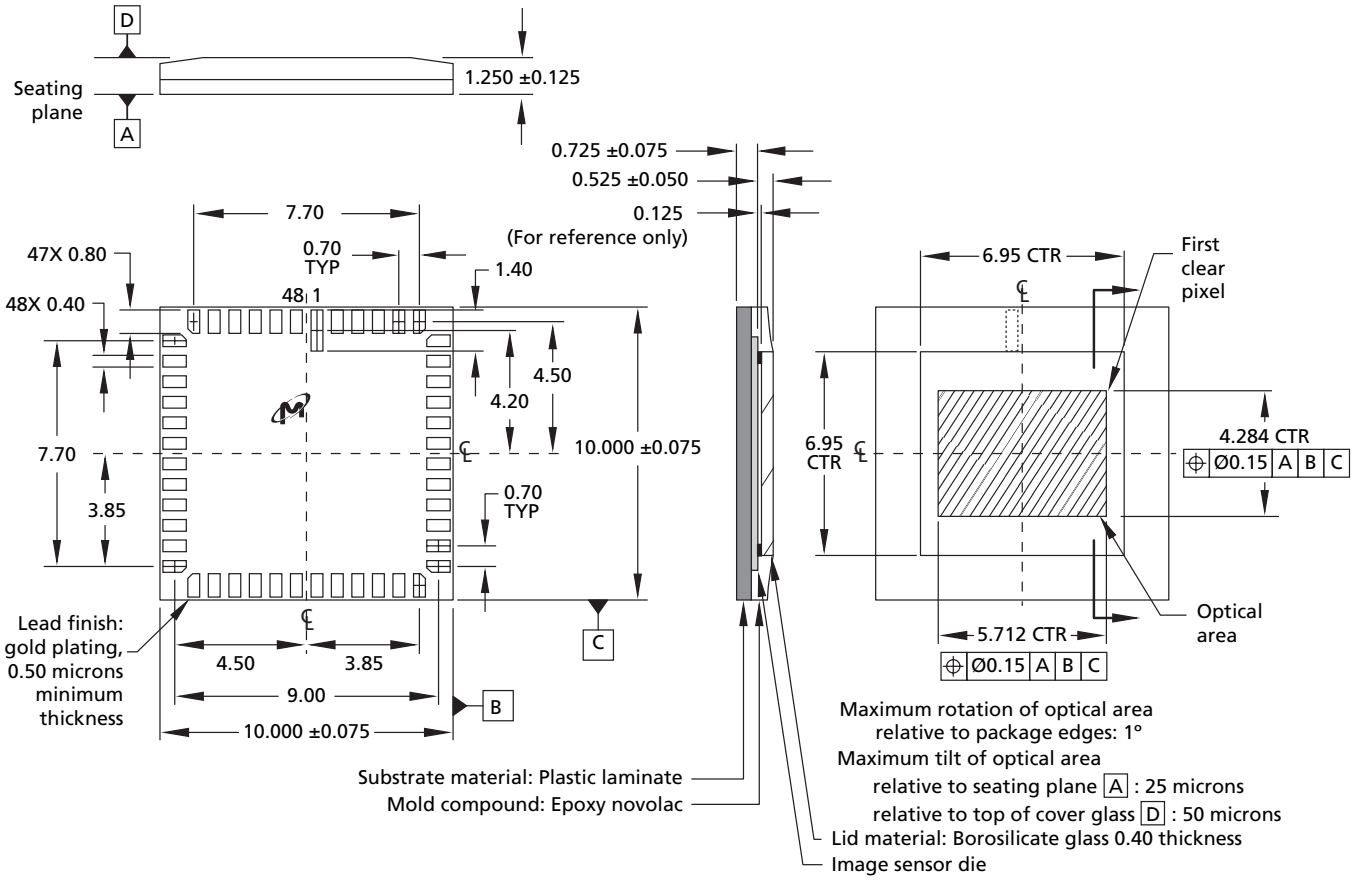


Table 30: POR Characterization

Symbol	Typical
t_1	15.5µs
t_2	0.4µs
t_3	0.9µs
t_4	2.0µs
Vtrig_rising	0.83Vµs
Vtrig_falling	1.07V

Package Dimensions

Figure 40: 48-Pin iLCC Package Outline Drawing



Note: All dimensions are in millimeters.

8000 S. Federal Way, P.O. Box 6, Boise, ID 83707-0006, Tel: 208-368-3900
prodmktg@micron.com www.micron.com Customer Comment Line: 800-932-4992

Revision History

Rev D 1/07

- Update Table 2, “Key Performance Parameters,” on page 1
- Update Figure 2: “Typical Configuration (connection),” on page 9
- Update "Pedestal" on page 14
- Add "SMIA Windowing" on page 18
- Update Equation on page 19
- Update "Programming Restrictions when Subsampling" on page 22
- Add “Scaler” on page 26
- Add Table 9, “Row Timing Constants,” on page 30
- Add Figure 35: “Quantum Efficiency,” on page 89
- Add Table 28, “Optimized Video Power Consumption,” on page 94

Rev C, Preliminary 10/06

- Update Table 2, “Key Performance Parameters,” on page 1
- Update Table 2, “Available Part Numbers,” on page 1
- Update "General Description" on page 6
- Update Table 3, “Signal Description,” on page 7
- Update Figure 1: “48-Pin ILCC 10 x 10 Package Pinout Diagram (Top View),” on page 8
- Update Figure 2: “Typical Configuration (connection),” on page 9
- Update Figure 3: “Block Diagram,” on page 10
- Update "Pixel Array" on page 11
- Update "Default Readout Order" on page 12
- Update "Using Per-color or Global Gain Control" on page 13
- Update "Timing and Control" on page 12
- Update "SMIA Gain Model" on page 13
- Update "Micron Imaging Gain Model" on page 13
- Update Table 4, “Recommended Gain Settings,” on page 14
- Update "Digital Gain" on page 14
- Update "PLL" on page 15
- Update Equation 6 on page 15
- Update "PLL Setup" on page 17
- Update "Readout Options" on page 17
- Update "Pixel Border" on page 17
- Update Figure 8: “8 Pixels in Normal and Column Mirror Readout Modes,” on page 19
- Update "Programming Restrictions when Subsampling" on page 22
- Update "Shading Correction (SC)" on page 27
- Update "Output Data Timing (Parallel Pixel Data Interface)" on page 28
- Update Table 8, “Row Timing Parameters,” on page 29
- Update "General Purpose Inputs" on page 31
- Update "Output Enable Control" on page 31
- Update Table 11, “Output Enable Control,” on page 31
- Update Table 12, “Trigger Control,” on page 32
- Update "Streaming/Standby Control" on page 32
- Update "Low Power Mode" on page 33
- Update "Slave Address/Data Direction Byte" on page 35

- Update "Two-Wire Serial Interface" on page 35
- Update "Registers" on page 39
- Update "Byte Ordering" on page 40
- Update "Bad Frames" on page 41
- Update Table 17, “SMIA Configuration,” on page 44
- Update Table 18, “1: SMIA Parameter Limits,” on page 46
- Update Table 19, “3: Manufacturer Specific,” on page 48
- Update Table 20, “0: SMIA Configuration,” on page 57
- Update Table 21, “1: SMIA Parameter Limits,” on page 62
- Update Table 22, “3: Manufacturer Specific,” on page 66
- Add "Electrical Specifications" on page 93
- Add Table 25, “Electrical Characteristics and Operating Conditions,” on page 93
- Add Table 26, “I/O Parameters,” on page 94
- Add Table 27, “Typical Power,” on page 94
- Add Table 29, “I/O Timing,” on page 95
- Add Figure 38: “I/O TIming,” on page 95
- Add Figure 39: “Power On Reset,” on page 96
- Add Table 30, “POR Characterization,” on page 96

Rev B, Advance 7/06

- Update "Features" on page 1
- Update "General Description" on page 6
- Update Table 3, “Signal Description,” on page 7
- Update "Typical Connections" on page 9
- Update Figure 3: “Block Diagram,” on page 10
- Update "Sensor Core Description" on page 11
- Update "Analog Gain Options" on page 12
- Update "Gain Code Mapping" on page 14
- Update "Pedestal" on page 14
- Update "Integration Time" on page 14
- Update "PLL" on page 15
- Update Figure 8: “8 Pixels in Normal and Column Mirror Readout Modes,” on page 19
- Update Figure 12: “Pixel Readout (no skipping, x_odd_inc = 1, y_odd_inc = 1),” on page 20
- Update "Programming Restrictions when Subsampling" on page 22
- Update Figure 20: “Pixel Data Timing Example,” on page 28
- Update Table 8, “Row Timing Parameters,” on page 29
- Update "General Purpose Inputs" on page 31
- Update "Output Enable Control" on page 31
- Update "Snapshot and Flash" on page 32
- Update Table 15, “Address Space Regions,” on page 39
- Update "Register Notation" on page 39
- Update "Register Aliases" on page 39
- Update Table 17, “SMIA Configuration,” on page 44
- Update Table 18, “1: SMIA Parameter Limits,” on page 46
- Update Table 19, “3: Manufacturer Specific,” on page 48
- Update Table 20, “0: SMIA Configuration,” on page 57
- Update Table 21, “1: SMIA Parameter Limits,” on page 62

- Update Table 22, “3: Manufacturer Specific,” on page 66
- Add Figure 34: “CRA vs. Image Height,” on page 87
- Add Figure 40: “48-Pin iLCC Package Outline Drawing,” on page 97

Rev A, Advance **2/06**

- Initial release