1/4-Inch System-On-A-Chip (SOC) CMOS Digital Image Sensor

MT9D112

Features

- DigitalClarity™ CMOS imaging technology
- Superior low-light performance
- Ultra-low-power, low-cost
- Internal master clock generated by on-chip phase-locked loop oscillator (PLL)
- Electronic rolling shutter (ERS), progressive scan
- Integrated image flow processor (IFP) for single-die camera module
- Automatic image correction and enhancement, including 4-channel lens shading correction with independent corner correction
- Arbitrary image scaling with anti-aliasing
- Integrated microcontroller for flexibility
- Two-wire serial interface providing access to registers and microcontroller memory
- Selectable output data format: ITU-R BT.601 (YCbCr), 565RGB, 555RGB, 444RGB, processed bayer, RAW8, and RAW10-bit
- Output FIFO for data rate equalization
- Programmable I/O slew rate
- Parallel and serial mobile industry processor interface (MIPI) data output
- Xenon and LED flash support with fast exposure adaptation
- Flexible support for external auto focus, optical zoom, and mechanical shutter
- Independently configurable gamma correction

Applications

- Cellular phones
- PC cameras
- PDAs

Table 1: Key Performance Parameters

Parameter		Value
Optical format		1/4-inch (4:3)
Full resolution		1,600 x 1,200 pixels (UXGA)
Pixel size		2.2µm x 2.2µm
Chief ray angle		22.1 deg maximum at 75% image height
Color filter array		RGB Bayer pattern
Active pixel array area		3.56mm x 2.68mm
Shutter type		Electronic rolling shutter (ERS) with global reset
Input clock frequency		6–54 MHz
Maximum frame rate		15 fps at full resolution, 24 fps in preview mode, 30 fps in video mode
Maximum data rate/ master clock		80 MB/s 6 MHz to 80 MHz
Supply voltage	Analog	2.5V–3.1V
	Digital	1.7V–1.95V
	I/O	1.7V–3.1V
	PLL	2.5V–3.1V
	AF	1.7V–3.1V
ADC resolution		10-bit, on-die
Responsivity		0.53 V/lux-sec (preliminary)
Dynamic range		59.5dB (preliminary)
SNR_{MAX}		37.7dB (preliminary)
Power consumption		245mW at 15 fps, full resolution
		230mW at 30 fps, video mode 168mW at 24 fps, preview mode
		30µW, standby/shutdown
Operating temperature		–30°C to +70°C (at junction)
Package		Bare die

Table 2: Ordering Information

Part Number	Description
MT9D112D00STCK15AC1	Bare die

Table of Contents

List of Figures

List of Tables

General Description

The Micron®Imaging MT9D112 is a 1/4-inch 2-megapixel CMOS image sensor with an integrated advanced camera system. This camera system features a microcontroller (MCU), a sophisticated image flow processor (IFP), and both parallel and serial mobile industry processor interface (MIPI) interfaces. It also includes a programmable general purpose I/O module (GPIO), which can be used to control external auto focus (AF), optical zoom, or mechanical shutter. The microcontroller manages all components of the camera system and sets key operation parameters for the sensor core to optimize the quality of raw image data entering the IFP. The sensor core consists of an active pixel array of 1,616 x 1,216 pixels, programmable timing and control circuitry including a PLL and external flash support, analog signal chain with automatic offset correction and programmable gain, and two 10-bit analog-to-digital converters (ADC). The entire system-on-a-chip (SOC) has ultra-low power requirements and superior low light performance that is particularly suitable for mobile applications. The MT9D112 is based on DigitalClarity™ Technology—Micron's breakthrough 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.

MT9D112 Overview

The MT9D112 has a color image sensor with a Bayer color filter arrangement and a 2-megapixel active-pixel array with electronic rolling shutter and global reset. The sensor core readout is 10-bit and supports skipping, binning and can be flipped and/or mirrored. The sensor core also supports separate analog and digital gain for all four color channels (R, Gr, Gb, B).

The MT9D112 also has an embedded phase-locked loop oscillator (PLL) that can generate the internal sensor clock from the common wireless system clock. When in use, the PLL adjusts the incoming clock frequency up, allowing the MT9D112 to run at almost any desired resolution and frame rate within the sensor's capabilities. The PLL can be bypassed and powered down to reduce power consumption.

Low power consumption is a very important requirement for all components of wireless devices. The MT9D112 has numerous power-conserving features, including internal soft standby modes, hard, and an external SHUTDOWN that allows the internal power bus to be disabled.

Another important consideration for wireless devices is their electromagnetic interface (EMI). The MT9D112 can be used with either a serial MIPI interface or the parallel data output interface, which has a programmable I/O slew rate to minimize EMI and an output FIFO to eliminate output data bursts.

The advanced image flow processor and flexible programmability of the MT9D112 provide a variety of ways to enhance and optimize the image sensor performance. Built-in optimization algorithms enable the MT9D112 to operate at factory settings as a fully automatic, highly adaptable camera; however, most of its settings are user-programmable.

These algorithms include black level conditioning, lens shading correction (LC), defect correction, noise reduction, color interpolation, edge detection, color correction, aperture correction, and image formatting such as cropping and scaling.

The MT9D112 also includes a sequencer that coordinates all events triggered by the user. The sequencer manages auto focus, auto white balance, flicker detection, and auto exposure for the different operating modes, which include preview, still capture, video, and snapshot with flash.

A two-wire serial register interface bus enables read/write access to control registers, variables, and special function registers within the MT9D112. The hardware registers are grouped internally by pages and include sensor core controls, color pipeline controls, and output controls. Variables are located in the microcontroller's RAM memory and are used for drivers such as the auto exposure (AE), auto white balance (AWB), and auto focus (AF). Special function registers are registers connected to the local bus of the microcontroller and include GPIO and the waveform generator.

The general purpose I/O can be configured to allow the user to output a flash or shutter pulse or to achieve 10-bit parallel output, or they can be configured as inputs to enable the user to use features such as an external trigger.

Signal Description

Table 3 provides the signal descriptions for the MT9D112.

Table 3: Signal Description

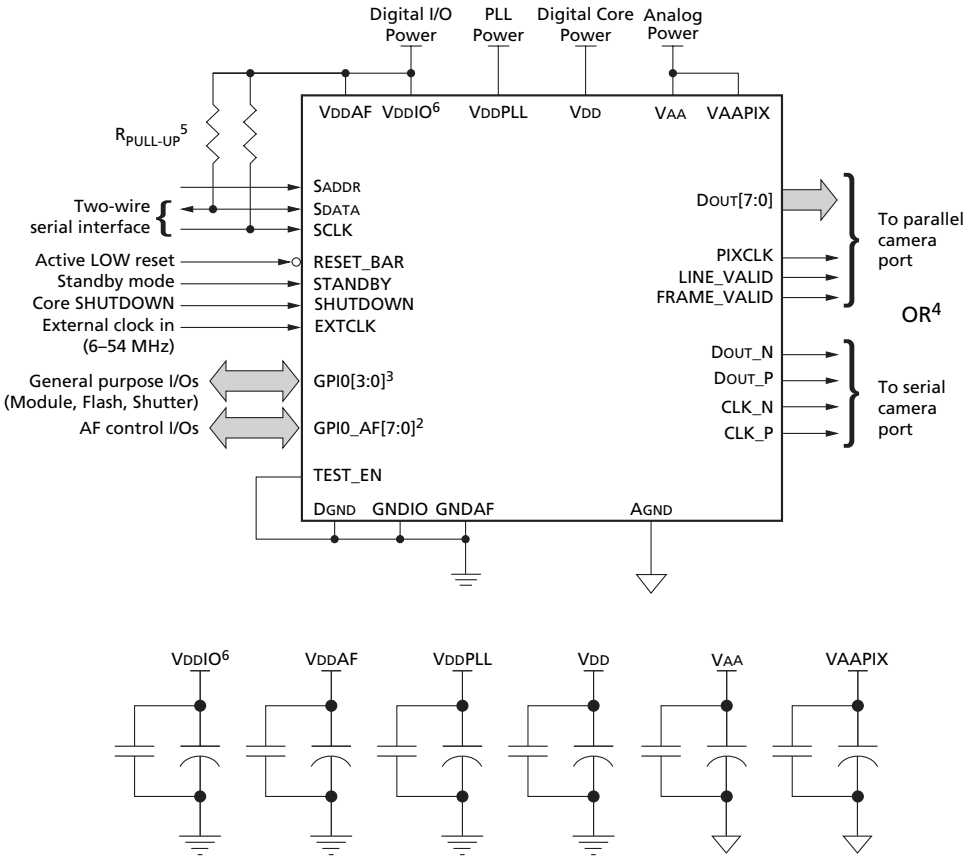
Name	Type	Description
SHUTDOWN	Input	Power down VDD, active HIGH.
TEST_EN	Input	Reserved for factory test. Tie to digital ground during normal operation (can be left floating if not used).
STANDBY	Input	Controls sensors standby mode, active HIGH.
SCLK	Input	Two-wire serial interface clock.
SADDR	Input	Selects device address for the two-wire serial interface. The address is 0x78 when SADDR is tied LOW, 0x7A if tied HIGH.
RESET_BAR	Input	Master reset signal, active LOW (can be left floating if not used).
EXTCLK	Input	Input clock signal 6–54 MHz.
SDATA	I/O	Two-wire serial interface data.
GPIO[3:0]	I/O	General purpose digital I/O, could be configured for Flash/Shutter/DOUT_LSB0/ DOUT_LSB1/Module_ID/OE_BAR/Trigger.
GPIO_AF[7:0]	I/O	General purpose digital I/O. Used for auto focus function (can be left floating if not used).
DOUT[7:0]	Output	Eight-bit image data output or most significant bits (MSB) of 10-bit sensor bypass mode.
DATA_OUT_N	Output	Differential MIPI data (sub-LVDS, negative) (must leave floating if not used).
DATA_OUT_P	Output	Differential MIPI data (sub-LVDS, positive) (must leave floating if not used).
CLK_OUT_N	Output	Differential MIPI clock (sub-LVDS, negative) (must leave floating if not used).
CLK_OUT_P	Output	Differential MIPI clock (sub-LVDS, positive) (must leave floating if not used).
VDD	Supply	Digital power (1.8V).
VAA_PIX	Supply	Pixel array power (2.8V).
VAA	Supply	Analog power (2.8V).
VDD_PLL	Supply	PLL power (2.8V).
VDD_IO	Supply	I/O power supply (1.8V to 2.8V).
GND_IO	Supply	I/O ground.
DGND	Supply	Digital, I/O, and PLL ground.
AGND	Supply	Analog ground.
VDD_AF	Supply	I/O power supply for GPIO_AF[7:0] pads (can be left floating if not used).
GND_AF	Supply	IO ground for GPIO_AF[7:0].

Typical Connections

Figure 1 shows typical MT9D112 device connections. For low-noise operation, the MT9D112 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 MT9D112 also supports different digital core (VDD/DGND) and I/O power (VDDIO/DGND) power domains that can be at different voltages. PLL requires a clean power source (VDDPLL).

Figure 1: Typical Configuration (connection)



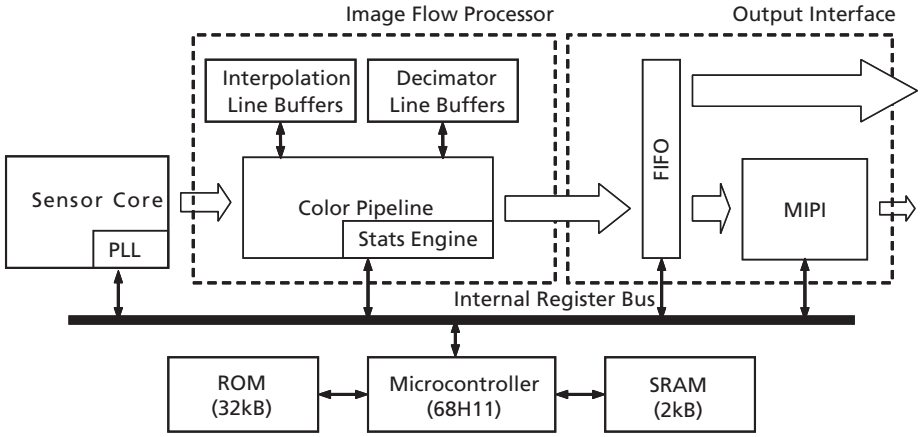
It is recommended that 0.1µF and 1µF decoupling capacitors for each power supply are mounted as close as possible to the pad. Actual values and results may vary depending on layout and design considerations.

Notes:1.Typical connection shows only one scenario out of multiple possible variations for this sensor.
2.If auto focus is not required the following pads can be left floating: VDDAF, GNDAF, and GPIO_AF.
3.The GPIO pads can serve multiple features that can be reconfigured. The function and direction will vary by applications
4.Only one of the output modes (serial or parallel) can be used at any time.
5.1.5kΩ resistor value is recommended for the two-wire serial interface RPULL-UP, however, greater value may be used for slower transmission speed.
6.All inputs must be configured with VDDIO.
7.VAA and VAAPIX must be tied together.

Architecture Overview

The MT9D112 combines a 2-Mp sensor core together with an image processor (image flow processor) to form a stand-alone solution that includes both image acquisition and processing. Both the sensor core and the image flow processor have internal registers that can be controlled by the user. In normal operation though, an integrated microcontroller controls most aspects of operation autonomously. The processed image data is transmitted to the host-system either via a parallel bus or a serial data interface through the output interface.

Figure 2: SOC Block Diagram



Sensor Core Description

The sensor core of the MT9D112 is a progressive-scan sensor that generates a stream of pixel data at a constant frame rate, qualified by LINE_VALID and FRAME_VALID. The maximum pixel rate is 80 megapixels/second, corresponding to a pixel clock rate of 80 MHz. Figure 3 on page 12 shows a block diagram of the sensor core. It includes a 2-megapixel 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 time interval between reset and readout. 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 10-bit value for each pixel in the array.

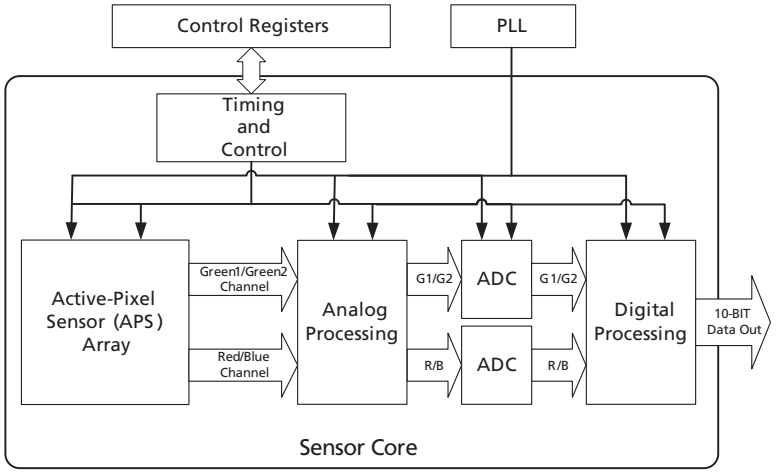
The pixel array contains optically active and light-shielded (dark) pixels. The dark pixels are used to provide data for the offset-correction algorithms (black level control).

The sensor core 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 are controlled by the SOC firmware and can be accessed through a two-wire serial interface. Register values written to the sensor core may be overwritten by firmware.

The output from the core 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.

A flash strobe output signal 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.

Figure 3: Sensor Core Block Diagram



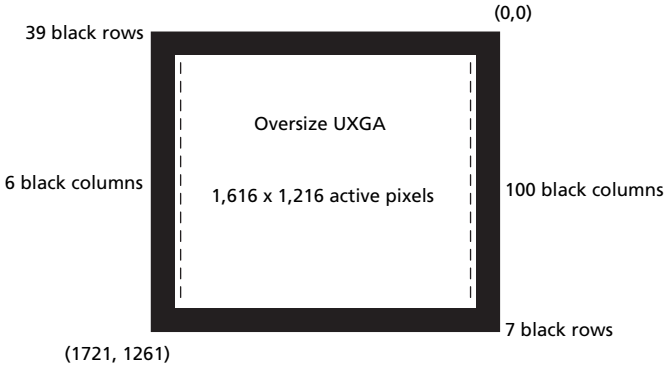
Pixel Array

Pixel Array Structure

The sensor core pixel array is configured as 1,722 columns by 1,262 rows (shown in Figure 4). The first 100 columns and the first 39 rows of pixels are optically black and are used for the automatic black level adjustment; the last 6 columns are also optically black.

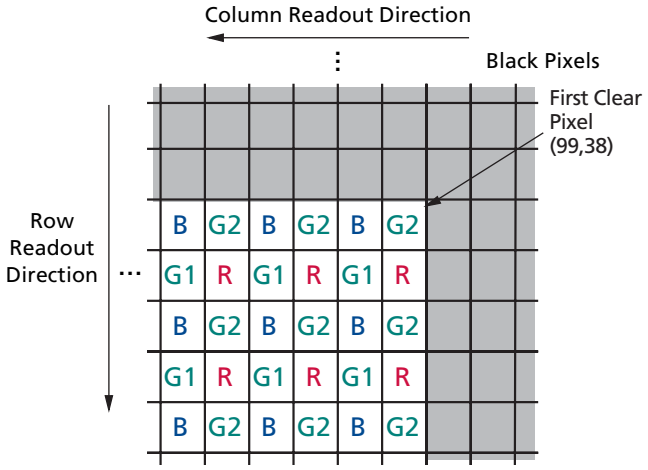
The optically active pixels are used as follows: In default mode a UXGA image (1,616 columns by 1,216 rows) is generated, starting at row 40, column 101. An 8-pixel boundary of active pixels is enabled around the image to avoid boundary effects during color interpolation and correction.

Figure 4: Pixel Array



The sensor core uses a Bayer color pattern, as shown in Figure 5. The even-numbered rows contain green and red color pixels; odd-numbered rows contain blue and green color pixels. Even-numbered columns contain green and blue color pixels; odd-numbered columns contain red and green color pixels.

Figure 5: Pixel Color Pattern Detail (Top Right Corner)

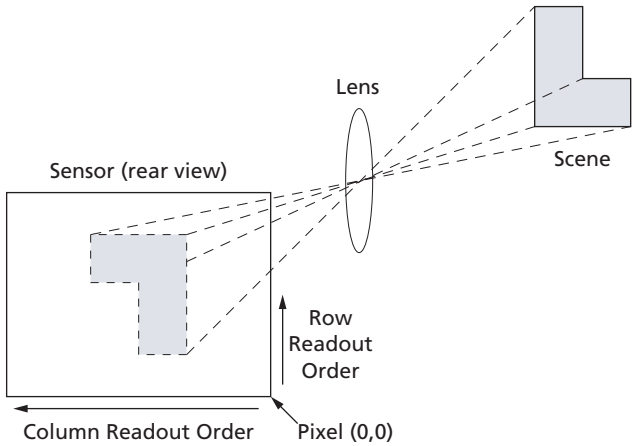


Default Readout Order

By convention, the sensor core pixel array is shown with pixel (0,0) in the top right-hand corner (see Figure 4 on page 12). This reflects the actual layout of the array on the die. When the sensor is imaging in a system, the active surface of the sensor faces the scene as shown in Figure 6.

When the image is read out of the sensor, it is read one row at a time, with the rows and columns sequenced. By convention, data from the sensor is shown with the first pixel read out-pixel (99,38) in the case of the sensor core-in the top lefthand corner.

Figure 6: Imaging a Scene



Analog Processing

Analog Readout Channel

The sensor core features two identical analog readout channels as shown in Figure 3 on page 12. The readout channel consists of two gain stages, a sample-and-hold stage with black level calibration capability, and a 10-bit ADC.

Timing and Control

Gain Options

The MT9D112 provides per-color gain control as well as 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.

Integer digital gains in the range 0–7 can be programmed. A digital gain of 0 sets all pixel values to 0 (the pixel data will simply represent the value applied by the pedestal block). Usually, when the gain settings are changed, the gain is updated on the next frame start.

Integration Time

The integration (exposure) time of the MT9D112 is controlled by the fine_integration_time and coarse_integration_time registers. While coarse integration time controls the integration duration steps of lines, the fine time allows for sub-line accuracy. To make manual adjustments to integration time, the MCU auto features must be disabled.

The limits for the fine integration time are defined by:

sensor_fine_IT_min_A/B ≥ fine_integration_time ≥ (sensor_lineLengthPck_A/B-sensor_fine_IT_max_margin_A/B)

Course integration time is < frame_length_lines_margin.

The actual integration time is given by:

$$integration\_time = \frac{((coarse\_integration\_time \times line\_length\_pck) + fine\_integration\_time)}{vt\_pix\_clk\_freq\_mhz/1x10^{6}} \quad \text{(EQ 1)}$$

line_length_pck = ((x_addr_end - x_addr_start +1)/xskip) +min_line_blanking_pck

coarse_integration_time is value of R0x3012

fine_integration_time is value of R0x3014

x_addr_end is a value of R0x3008

x_addr_start is a value of R0x3004

xskip is a value of R0x3016[7:5]

vt_pix_clk_freq_mhz is MCLK/2

It is fundamental to the operation of an electronic rolling shutter (ERS) that it is not possible to set an integration time that is greater than the frame time. Unlike earlier Micron Imaging parts, setting an integration time that is greater than the frame time does not affect the frame time; the behavior is undefined. On the MT9D112, it is necessary to reprogram the frame time (frame_length_lines) in order to make longer exposure times available. Long integration times increase the likelihood of image degradation due to increased accumulation of dark current.

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, it 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.

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.

PLL

PLL Generated Master Clock

The PLL can generate a master clock signal whose frequency is up to 80 MHz (input clock from 6 MHz through 54 MHz). R0x341C controls the frequency of the PLL-generated clock. It is possible to bypass the PLL and use CLKIN as master clock. In order to do so, one must set R0x341E[0] to “1.” If power consumption is a concern, R0x341E[1] should be also set to “1” a short time later, to put the bypassed PLL in power down mode. To enable the PLL again, the two bits must be set to “0” in the reverse order. By default, the PLL is bypassed and powered down.

PLL Setup

Because the input clock frequency is unknown, the part starts with the PLL disabled. The PLL takes time to power up. During this time, the behavior of its output clock signal is not guaranteed. The PLL output frequency is determined by two constants, M and N, and the input clock frequency.

PLL programming and power-up sequence is as follows:

1. Program PLL frequency settings, R0x341C (pll_m, pll_n) (master clock frequency is equal to fVCO_pll/8). With default settings master clock frequency of 80 MHz is obtained with f_{CLKIN} = 16 MHz.
2. Power up PLL, R0x341E[1] = 0.
3. Wait for PLL settling time > 1ms.
4. Turn off PLL bypass, R0x341E[0] = 0.

Allow one complete frame to effect the correct integration time after enabling PLL.

Note: Before enabling PLL, ensure the slew rate is optimized for pads.

The default M and N values are for 16 MHz.

Table 4: Frequency Parameters

Frequency	Equation	Min (MHz)	Max (MHz)
^{f}IN	–	6	54
^{f}PFD	^{f}clkin /(pll_n+1)	2	20
^{f}VCO	^{f}clkin * pll_m/(pll_n+1)	320	640

Readout Options

The sensor core supports different readout options to modify the image before it is sent to the IFP. 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 averaging in x and y direction.

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

Window Size

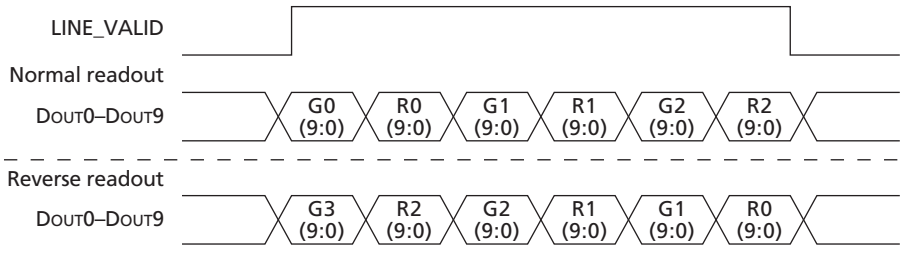
The image output size is set with registers x_addr_start, x_addr_end, y_addr_start, and y_addr_end. The edge pixels in the 1,616 x 1,216 array are present to avoid edge defects and should not be included in the visible window. Binning will change the image output size.

Readout Modes

Horizontal Mirror

When the horizontal_mirror bit (R0x3040[0]) is set in the read mode 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 7 shows a sequence of 6 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. This change in sensor core output is corrected for by the SOC.

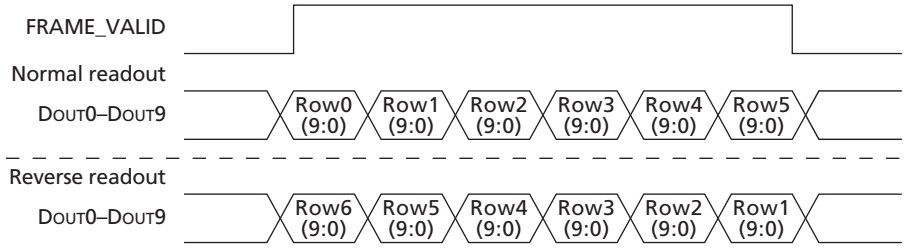
Figure 7: 6 Pixels in Normal and Column Mirror Readout Modes



Vertical Flip

When the vertical flip bit (R0x3040[1]) is set in the read mode 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 8 shows a sequence of 6 rows being read out with row mirror = 0 and row mirror = 1. Changing vertical_flip causes the bayer order of the output image to change; the new bayer order is reflected in the value of the pixel_order register. This change in sensor core output is corrected for by the SOC.

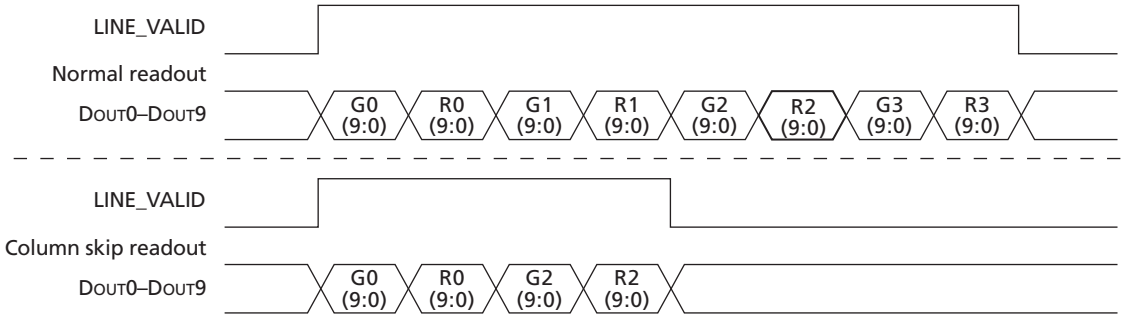
Figure 8: 6 Rows in Normal and Row Mirror Readout Modes



Column and Row Skip

The sensor core 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 setting x_odd_inc=3 and/or y_odd_inc=3. This reduces the amount of row and column data processed and is equivalent to the skip2 readout mode provided by earlier Micron Imaging sensors. When enabling skipping, the proper image output and crop sizes must be updated beforehand.

Figure 9: 8 Pixels in Normal and Column Skip 2x Readout Modes



The following waveform shows a sequence of data being read out with x_odd_inc=3 and y_odd_inc=1. The effect of the different subsampling settings on the pixel array readout is shown in Figures 10 through 13.

Figure 10: Pixel Readout (no skipping)

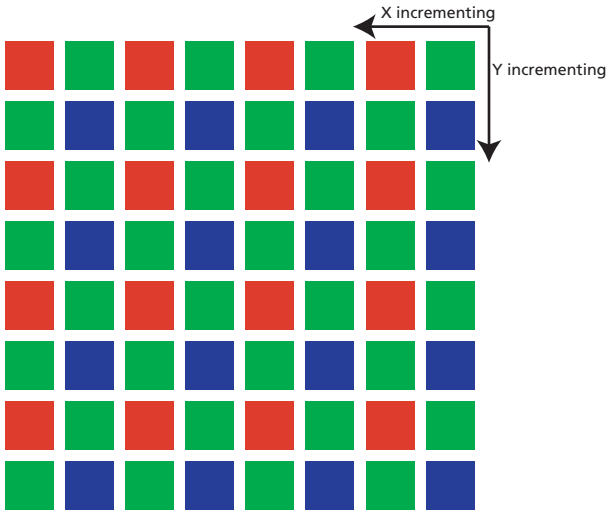


Figure 11: Pixel Readout (x_odd_inc=3, y_odd_inc=1)

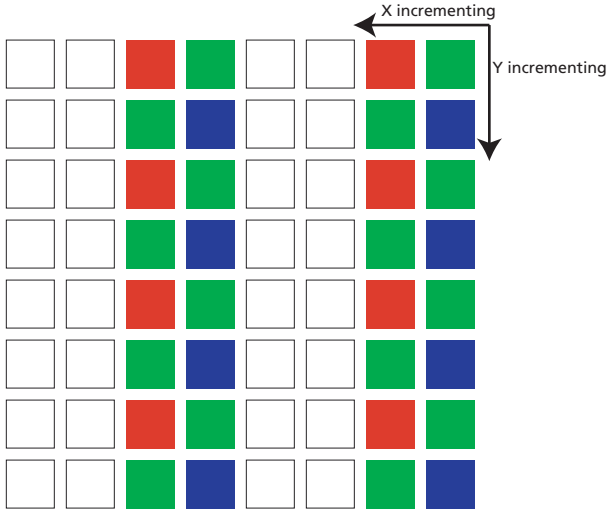


Figure 12: Pixel Readout (x_odd_inc=1, y_odd_inc=3)

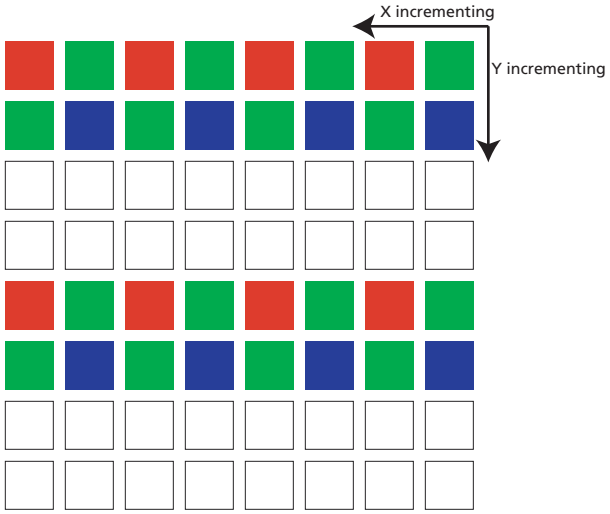
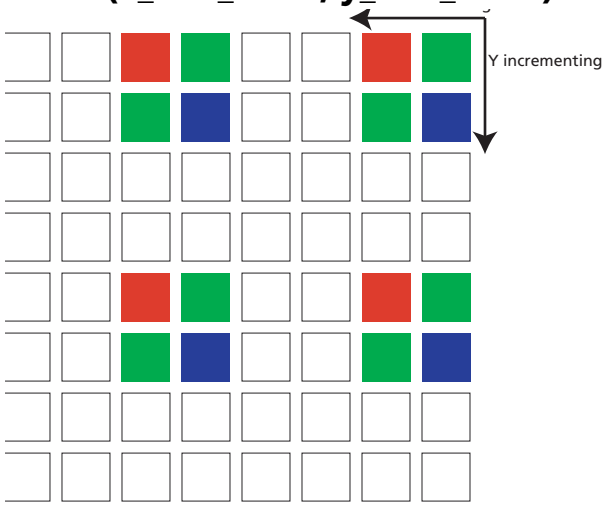


Figure 13: Pixel Readout (x_odd_inc=3, y_odd_inc=3)



Programming Restrictions when Skipping

When skipping is enabled as a viewfinder mode, and the sensor is switched back and forth between full resolution and skipping, 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.

When subsampling is enabled, it may be necessary to adjust the x_addr_end 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 rule:

remainder = (addr_end - addr_start + 1) AND 4;

if (remainder == 0) addr_end = addr_end - 2;

Table 5 on page 21 shows the row address sequencing for normal and subsampled (with y_odd_inc=3) readout. The same sequencing applies to column addresses for subsampled readout. There are two possible subsampling sequences (because the subsampling sequence only read half of the rows and columns) depending upon the alignment of the start address.

Binning

The MT9D112 sensor core supports 2x1 and 2x2 analogue binning (column binning, also called x-binning and row/column binning, also called xy-binning). Binning has many of the same characteristics as subsampling but because it gathers image data from all pixels in the active window (rather than a subset of them), it achieves superior image quality and 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 (Reg0x3040-1). As for subsampling, x_addr_end and y_addr_end may require adjustment when binning is enabled.

The effect of the different subsampling settings is shown in Figure 14 and Figure 15.

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

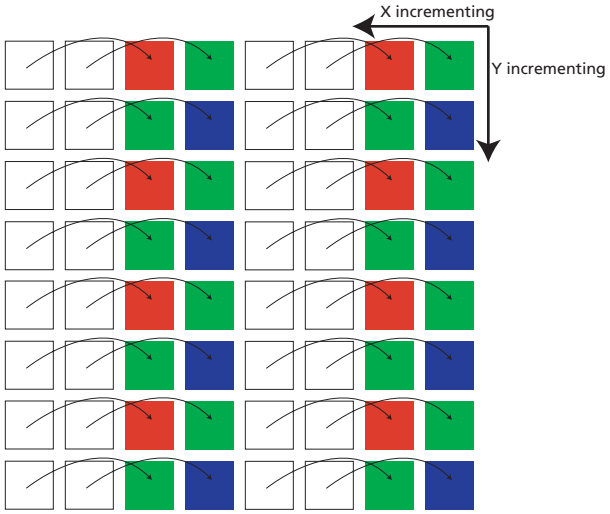
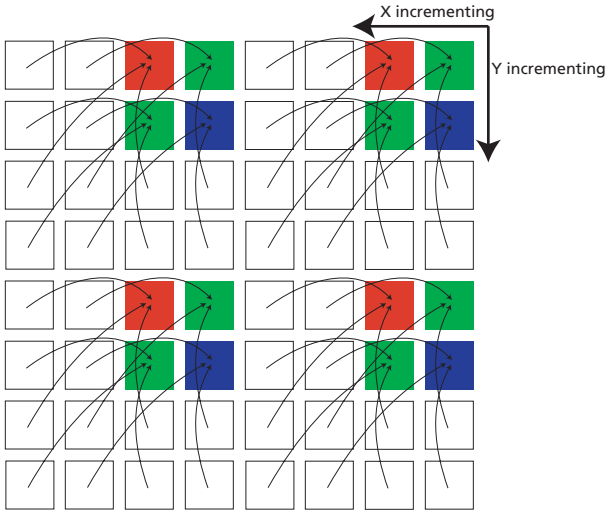


Figure 15: Pixel Readout (x_odd_inc=3, y_odd_inc=3, x_ybin=1)



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.

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. None of these adjustments are required for x-binning. The sensor must be taken out of streaming mode before switching between binned and nonbinned operation.

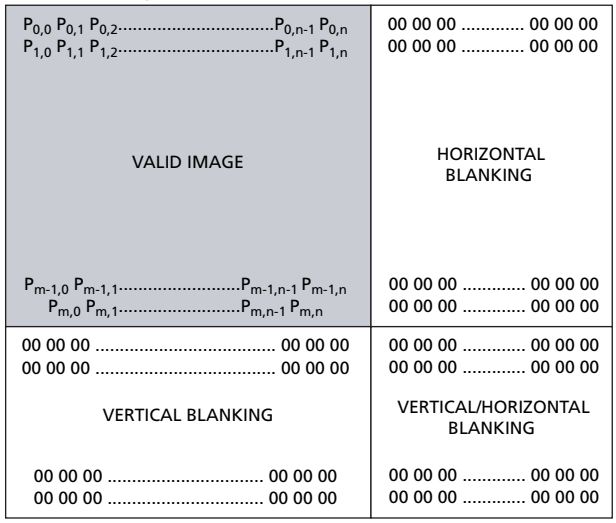
Table 5: Row Address Sequencing

Normal	Skipping	Skipping	Binned	Binned
0	0		0, 2	
1	1		1, 3	
2		2		2, 4
3		3		3, 5
4	4		4, 6	
5	5		5, 7	
6		6		6, 8
7		7		7, 9

Raw Data Format

The sensor core image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking as shown in Figure 16 on page 22. 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 16: Pixel Data Timing Example



Raw Data Timing

The sensor core output data is synchronized with the PIXCLK output. When LINE_VALID is HIGH, one pixel data is output on the 10-bit DOUT output every PIXCLK period. By default, the PIXCLK signal runs at the same frequency as the master clock, and its rising edges occur one-half of a master clock period after transitions on LINE_VALID, FRAME_VALID, and DOUT (Figure 17). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking period.

Figure 17: Pixel Data Timing Example

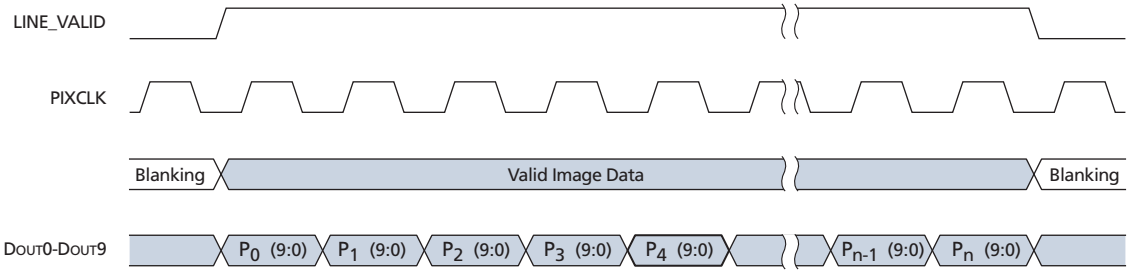
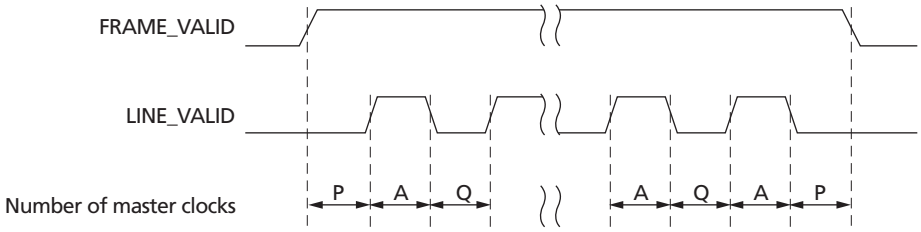


Figure 18: Row Timing and FRAME_VALID/LINE_VALID Signals



The sensor timing is shown in terms of pixel clock and master clock cycles (Figure 17 on page 22 and Figure 18 on page 22). Increasing the integration time to more than one frame causes the frame time to be extended.

Table 6: Row Timing Parameters

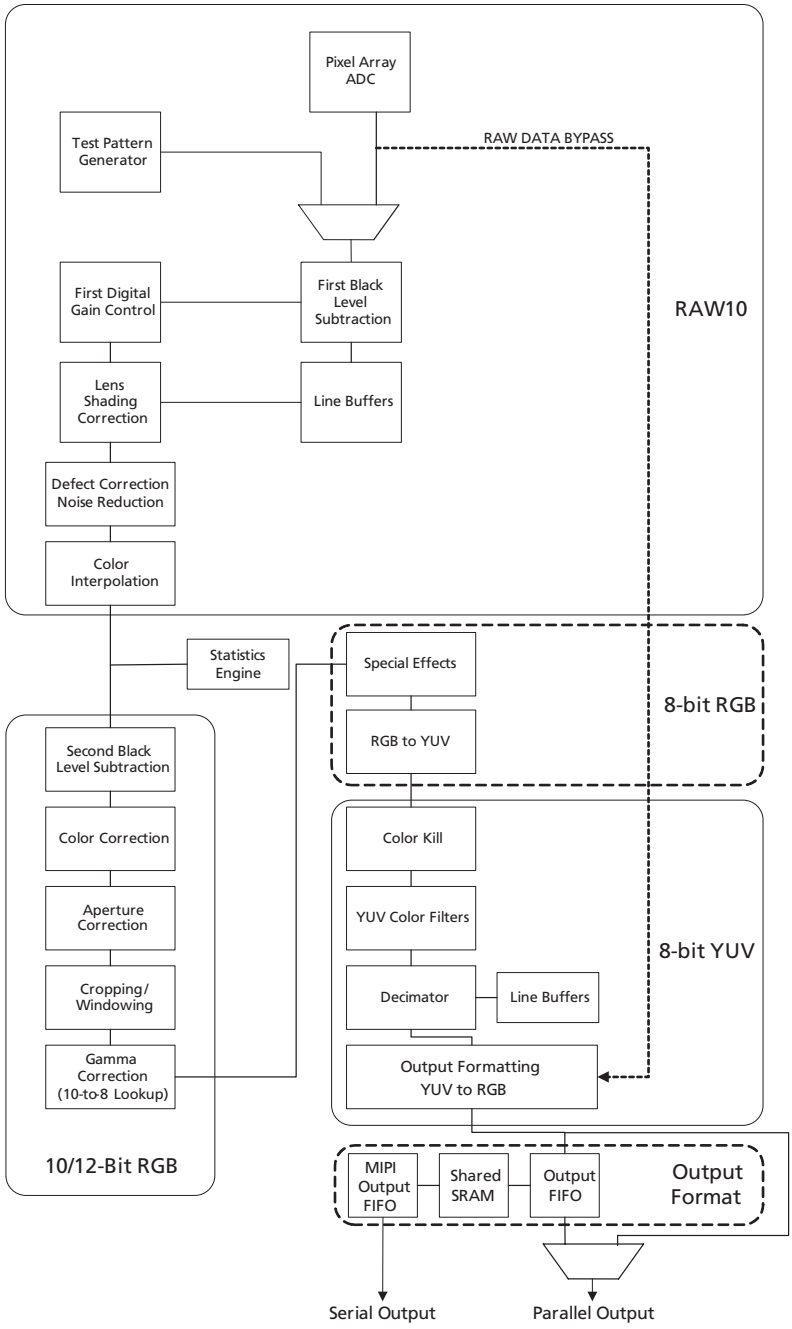
Parameter	Name	Default @ 54 MHz
PIXCLK_PERIOD	Pixel Clock Period	1 pixel clock = 18.52ns
S	Skip (Subsampling) Factor	1
A	Active Data Time	1600 pixel clocks = 29.63µs
P	Frame Start/End Blanking	6 pixel clocks = 111.11ns
Q	Horizontal Blanking	2284 - 1600 pixel clocks = 12.667µs
A + Q	Row Time	2284 pixel clocks = 42.296µs
N	Number of rows	1200 rows
V	Vertical Blanking	(1253 - 1200) * 2284 + 684 - 12 pixel clocks = 2253.481µs
Tfv	Frame Valid Time	(1200 * 2284) - 684 + 12 pixel clocks = 2740128 pixel clocks = 50.743ms
F	Total Frame Time	2284 pixel clocks/row * 1253 rows = 2861852 pixel clocks = 52.997ms

SOC Description

Image Flow Processor

Image and color processing in the MT9D112 is implemented as an image flow processor coded in hardware logic. The IFP can be controlled by registers from the outside but during normal operation the embedded microcontroller will automatically adjust the operation parameters. The IFP is broken down into different sections outlined in Figure 19.

Figure 19: Color Pipeline

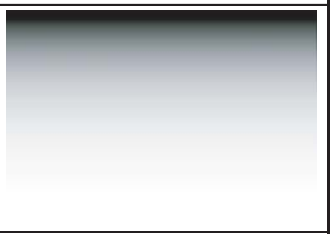
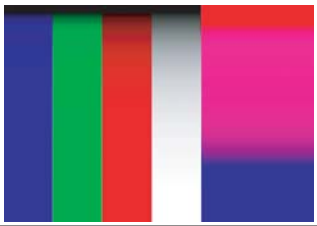


Test Patterns

During normal operation of the MT9D112, a stream of raw image data from the sensor core is continuously fed into the color pipeline. For test purposes, this stream 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 pipeline.

Test patterns are accessible using R0x3290 and are shown in Figure 20. Disabling MCU is recommended before enabling test patterns.

Figure 20: Test Patterns

Test Pattern	Register Value	Example
Flat Field	R0x3290 = 1	
Vertical Ramp	R0x3290 = 2	
Color Bar	R0x3290 = 3	
Vertical Stripes	R0x3290 = 4	
Pseudo-Random	R0x3290 = 5	

First Black Level Subtraction and Digital Gain

Image stream processing starts with black level subtraction (R0x3278) and multiplication of all pixel values by a programmable digital gain (R0x32DC). Both operations can be independently set to separate values for each color channel (R, Gr, Gb, B). Independent color channel digital gain is adjusted with registers R0x32D4 through R0x32DA. Independent color channel black level adjustments can be made with R0x327A through R0x3280. If the black level subtraction produces a negative result for a particular pixel, the value of this pixel is set to "1."

Lens Shading Correction

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 lens shading. The MT9D112 has an embedded lens shading correction module that can be programmed to counter the shading effect of a lens on each individual R, Gb, Gr, and B color signal. The LC module multiplies R, Gb, Gr, and B signals by a 2-dimensional correction function F (x,y), whose profile in both x and y direction is a piecewise quadratic polynomial with coefficients independently programmable for each direction and color. LC can be enabled and disabled with R0x3210[2].

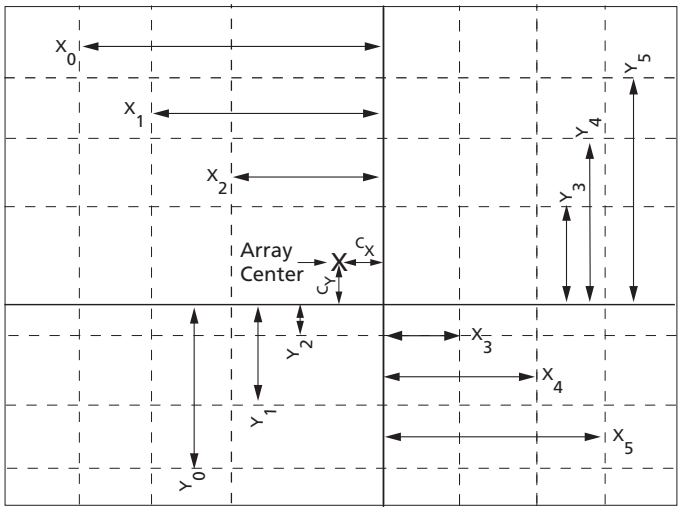
The MT9D112 also includes 16 independent corner parameters, K, for each of the color channels. The K factors are independently adjustable at each corner and are not dependent on the vertical and horizontal spatial dimension. They can be adjusted with R0x3546 through R0x3562.

LC can be enabled and disabled with R0x3210[2].

Lens Correction Zones

In order to increase the precision of the correction function, the image plane is divided into 8 zones in each dimension. The coordinates of zone boundaries are referenced with respect to the lens center, C. Each boundary as well as C (Cx, Cy) coordinate is stored as a byte, which represents the coordinate value divided by 4. There always three boundaries to the left (top) of the center and three to the right (bottom) of the center. These boundaries apply uniformly for all color channels. However, the correction functions are programmable independently for each color component. Boundary and lens center positions are also valid for the preview mode. Figure 21 illustrates the lens correction zones which are accessible through registers R0x34CE through R0x34DC.

Figure 21: Lens Correction Zones



Defect Correction and Noise Reduction

The IFP performs on-the-fly defect correction that can mask pixel array defects such as high dark current (hot) pixels and pixels that are darker or brighter than their neighbors due to photoresponse nonuniformity. Each pixel is compared with its 8 nearest neighbors having the same filter color. Suppose the value of the compared pixel is P and the values of the 8 neighbors range from PMIN to PMAX. If P > PMAX, P is replaced with PMAX. Likewise, if P < PMIN, P is replaced with PMIN. Otherwise, the pixel is not considered defective and its value P is not changed. Defect correction can be enabled and disabled with R0x3210[3].

The image data for each color channel can be passed through an adaptive noise suppression module which averages over flat field areas while preserving edge information. The module is edge aware with exposure that is based on configurable thresholds. The thresholds are changed on-the-fly based on the brightness of the current scene. Noise reduction can be enabled and disabled with R0x33F4[3] and thresholds are set by R0x33F6 through R0x33FC.

Color Interpolation and Edge Detection

In the raw data stream fed by the sensor core to the IFP, each pixel is represented by a 10-bit integer number, which can be considered proportional to the pixel's response to a one-color light stimulus, red, green or blue, depending on the pixel's position under the color filter array. Initial data processing steps, up to and including the defect correction, preserve the 1-color-per-pixel nature of the data stream, but after the defect correction it must be converted to a 3-colors-per-pixel stream appropriate for standard color processing. The conversion is done by an edge-sensitive color interpolation module. The module pads the incomplete color information available for each pixel with information extracted from an appropriate set of neighboring pixels. The algorithm used to select this set and extract the information seeks the best compromise between preserving edges and filtering out high frequency noise in flat field areas. The edge threshold can be set with R0x328E.

Second Black Level Correction

After interpolation it might be necessary to recalibrate the black level. A second global black level substraction is possible at this stage in the pipeline, which is controlled via R0x3276. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to" 0."

Color Correction and Aperture Correction

In order to achieve good color fidelity of IFP output, interpolated RGB values of all pixels are subjected to color correction. The IFP multiplies each vector of three pixel colors by a 3 x 3 color correction matrix. The three components of the resulting color vector are all sums of three 10-bit numbers. Since such sums can have up to 12 significant bits, the bit width of the image data stream is widened to 12-bits per color (36-bits per pixel). The color correction matrix can be either programmed by the user or automatically selected by the auto white balance algorithm implemented in the IFP. Color correction should ideally produce output colors that are independent of the spectral sensitivity and color crosstalk characteristics of the image sensor. The optimal values of color correction matrix elements depend on those sensor characteristics and on the spectrum of light incident on the sensor. The color correction variables can be adjusted with R0x006 through R0x042.

To increase image sharpness, a programmable 1D or 2D aperture correction (sharpening filter) is applied to color corrected image data. Aperture correction is enabled using R0x3210. The gain and threshold for 1D and 2D correction can be defined via R0x326A and R0x326C.

Image Cropping

Image cropping takes place when the sensor core is programmed to output pixel values from a rectangular portion of its pixel array-a window-smaller than the default 1,600 x 1,200 window. Pixels outside the selected cropping window are not read out, which results in narrower field of view than at the default sensor settings. Irrespective of the size and position of the cropping window, the MT9D112 sensor core can also decimate outgoing images by skipping columns and/or rows of the pixel array, and/or by binning 2 x 2 groups of pixels of the same color. Since scaling by skipping (deletion) can cause aliasing (even if pixel binning is simultaneously enabled), it is generally better to change image size only by cropping and pixel binning.

In context A, the cropped window is defined by variables 0x051 through 0x057. In context B, it is defined by 0x05F and 0x065. In context A and B, the height and width definitions for the output window must be equal to or smaller than the cropped image.

Output width will always be as with no scaling; crop will still be in effect.

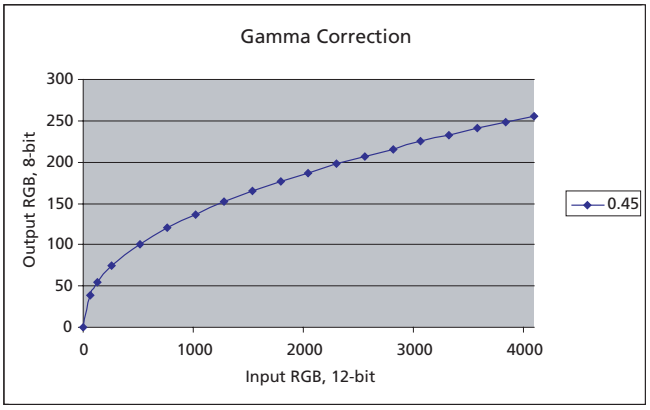
Contrast and Gamma Correction

Gamma correction is independently applied and independently configurable for each of the 12-bit R, G, and B color channels. The gamma correction curve is implemented as a piecewise linear function with 19 knee points, taking 12-bit arguments and mapping them to 8-bit output. The abscissas of the knee points are fixed at 0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4095. The 8-bit ordinates are programmable via IFP registers.

The MT9D112 IFP includes a block for gamma and contrast correction. A custom gamma/contrast correction table may be uploaded, or pre-set gamma and contrast settings may be selected.

The gamma and contrast correction block uses the following 12-bit input data points to form a piecewise linear transformation curve: 0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4095. These input points have been selected to provide more detail to the low end of the curve where gamma correction changes are typically the greatest. These points correspond to 8-bit output values that can be uploaded to the appropriate registers.

Figure 22: Gamma Correction Curve



For simplicity, predefined gamma and contrast tables may be selected, and the MT9D112 automatically combines these tables and upload them to the appropriate gamma correction registers.

The gamma and contrast tables may be selected at mode driver (ID = 7) offsets 67 and 68 (decimal) for mode A and mode B, respectively. The gamma settings are established at bits 0-2, and the contrast settings are established at bits 4-6.

The gamma setting values are shown in Table 7.

Table 7: Gamma Settings

Gamma Setting	Definition
0	Gamma = 1.0 (no gamma correction)
1	Gamma = 0.56
2	Gamma = 0.45
3	Use user-defined gamma table

S-Curve

The predefined contrast table values have been established by creating an "S" curve with highlight and shadow regions that blend smoothly with a linear midtone region. The slope and value of the highlight and shadow regions match the linear region at these transitions. In addition, the slope of the "S" curve is zero at the top (white) and bottom (black) points. The slope of the linear region determines how much contrast is applied; more contrast corresponds to a higher, midtone linear slope.

Figure 23: Contrast "S" Curve

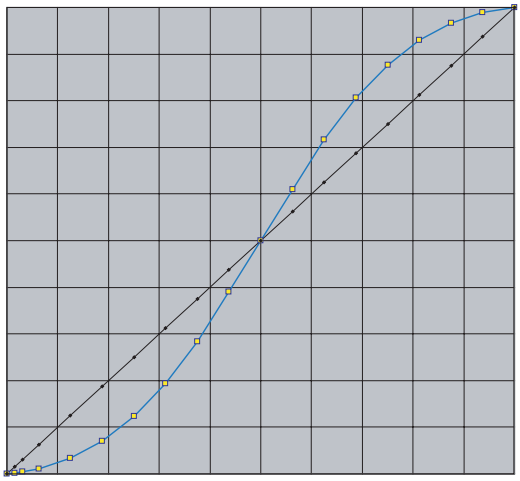


Table 8: Contrast Values

Contrast Setting	Definition
0	No contrast correction
1	Contrast slope = 1.25
2	Contrast slope = 1.50
3	Contrast slope = 1.75
4	Noise reduction contrast

The contrast curve function is applied to the gamma curve points used (whether the gamma curve points are predefined or user-uploaded).

S-curve is a function to correct image pixel values. When applied to pixel values, it typically compresses dark and bright tones, while stretching the midtones.

Special effects like negative image, sepia, or B/W can be applied to the data stream at this point. Special effects are enabled with R0x3348.

RGB to YUV Conversion

For further processing the data is converted from RGB color space to YUV color space.

Color Kill

To remove high light or low light color artifacts, a color kill circuit is included. It affects only pixels whose luminance exceeds a certain preprogrammed threshold. The U and V values of those pixels are attenuated proportionally to the difference between their luminance and the threshold.

YUV Color Filter

As an optional processing step noise suppression by 1-dimensional low-pass filtering of Y and/or UV signals is possible. A 3- or 5-tap filter can be selected for each signal.

Image Scaling

To ensure that the size of images output by the MT9D112 can be tailored to the needs of all users, the IFP includes a scaler module. When enabled, this module performs rescaling of incoming images-shrinks them to arbitrarily selected width and height without reducing the field of view and without discarding any pixel values.

The scaler performs pixel binning-divides each input image into rectangular bins corresponding to individual pixels of the desired output image, averages pixel values in these bins, and assembles the output image from the bin averages. Pixels lying on bin boundaries contribute to more than one bin average: their values are added to bin-wide sums of pixel values with fractional weights. The entire procedure preserves all image information that can be included in the downsized output image and filters out high frequency features that could cause aliasing.

The image cropping and scaler module can be used together to implement a digital zoom and pan. If the scaler is programmed to output images smaller than images coming from the sensor core, zoom effect can be produced by cropping the latter from their maximum size down to the size of the output images. The ratio of these two sizes determines the maximum attainable zoom factor. For example, a 1,600 x 1,200 image rendered on a 160 x 120 display can be zoomed up to 10 times, since 1,600/160 = 1,200/120 = 10. Panning effect can be achieved by fixing the size of the cropping window and moving it around the pixel array.

Due to the loss of sharpness due to pixel binning during image scaling, 1D aperture correction may be applied to increase image sharpness lost due to pixel binning during image scaling.

YUV-to-RGB/YUV Conversion and Output Formatting

The YUV data stream emerging from the scaler module can either exit the color pipeline as-is or be converted before exit to an alternative YUV or RGB data format as selected by R0x332E.

Color Conversion Formulas

Y'U'V'

This conversion is BT 601 scaled to make YUV range from 0 through 255. This setting is recommended for JPEG encoding and is the most popular, although it is not well defined and often misused in Windows.

Y' = 0.299 R' + 0.587 G' + 0.114 B'

U' = 0.564 (B' - Y') + 128

V' = 0.713 (R' - Y') + 128

There is an option where 128 is not added to U'V'.

Y'Cb'Cr' Using sRGB Formulas

The MT9D112 implements the sRGB standard. This option provides YCbCr coefficients for a correct 4:2:2 transmission. Note that 16 < Y60 1< 235, 16 < Cb, Cr < 240 and 0 < = RGB < = 255.

Y' = (0.2126*R' + 0.7152*G' + 0.0722*B')*219/256 + 16

Cb' = 0.5389*(B '- Y')*224/256 + 128

Cr' = 0.635*(R' - Y')*224/256 + 128

Y'U'V' Using sRGB Formulas

Similar to the previous set of formulas, but has YUV spanning a range of 0 through 255.

Y' = 0.2126*R' + 0.7152*G' + 0.0722*B'

U' = 0.5389*(B' - Y') + 128 = -0.1146*R - 0.3854*G + 0.5*B

V' = 0.635*(R' - Y') + 128 = 0.5*R - 0.4542*G - 0.0458*B

There is an option to disable adding 128 to U'V'. The reverse transform is as follows:

R' = Y + 1.5748*V

G' = Y - 0.1873*(U - 128) - 0.4681*(V - 128)

B' = Y + 1.8556*(U - 128)

Output Interface

Parallel and MIPI Output

The user can select to either use the serial MIPI output or the 8-bit parallel output to transmit the data. Only one of the output modes can be used at any time.

The parallel output, enabled by R0x301A[7], can be used with an output FIFO whose memory is shared with the MIPI output FIFO to retain a constant pixel output clock independent from the scaling factor.

When scaling the image or skipping lines, the data would be generated in bursts and the pixel clock would turn on and off in intervals, which might lead to EMI problems. The output FIFO will group all active pixel data together so the pixel clock can be run at a constant speed. The output FIFO is enabled by setting R0x3212[1:0] to 10. There are two registers to configure the output FIFO, watermark and line length.

The MIPI output transmitter implements a fully configurable serial differential subLVDS transmitter capable of up to 640 Mb/s. It supports multiple formats, error checking, and custom short packets.

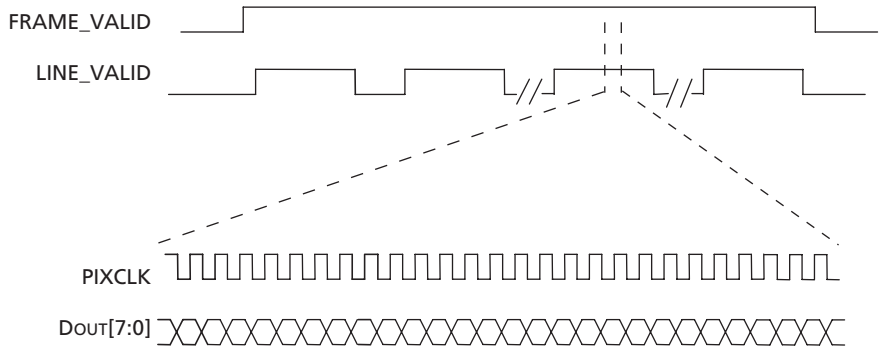
Output Format and Timing

YUV/RGB Uncompressed Output

Uncompressed YUV or RGB data can be output either directly from the output formatting block or via a FIFO buffer with a capacity of 800 bytes, enough to hold one-fourth uncompressed line at full resolution. Buffering of data is a way to equalize the data output rate when image scaling is used. Scaling produces an intermittent data stream consisting of short high-rate bursts separated by idle periods. High pixel clock frequency during bursts may be undesirable due to EMI concerns.

Figure 24 depicts the output timing of uncompressed YUV/RGB when a decimated data stream is equalized by buffering or when no scaling takes place. The pixel clock frequency remains constant during each LINE_VALID HIGH period. Decimated data are output at a lower frequency than full size frames, which helps to reduce EMI.

Figure 24: Timing of Uncompressed Full Frame Output or Decimated Output Passing Through the FIFO



Uncompressed YUV/RGB Data Ordering

The MT9D112 supports swapping YCrCb mode, as illustrated in Table 9.

Table 9: YCrCb Output Data Ordering

Mode				
Default (no swap)	Cb_i	Y_i	Cr_i	Y_{i+1}
Swapped CrCb	Cr_i	Y_i	Cb_i	Y_{i+1}
Swapped YC	Y_i	Cb_i	Y_{i+1}	Cr_i
Swapped CrCb, YC	Y_i	Cr_i	Y_{i+1}	Cb_i

The RGB output data ordering in default mode is shown in Table 10. The odd and even bytes are swapped when luma/chroma swap is enabled. R and B channels are bit-wise swapped when chroma swap is enabled.

Table 10: RGB Ordering in Default Mode

Mode (Swap Disabled)	Byte	$D_7D_6D_5D_4D_3D_2D_1D_0$
RGB 565	Odd	$R_7R_6R_5R_4R_3G_7G_6G_5$
	Even	$G_4G_3G_2B_7B_6B_5B_4B_3$
RGB 555	Odd	$0\ R_7R_6R_5R_4R_3G_7G_6$
	Even	$G_4G_3G_2B_7B_6B_5B_4B_3$
RGB 444x	Odd	$R_7R_6R_5R_4G_7G_6G_5G_4$
	Even	$B_7B_6B_5B_4\ 0\ 0\ 0\ 0$
RGB x444	Odd	$0\ 0\ 0\ 0\ R_7R6_6R_5R_4$
	Even	$G_7G_6G_5G_4B_7B_6B_5B_4$

Uncompressed 10-Bit Bypass Output

Raw 10-bit Bayer data from the sensor core can be output in bypass mode in two ways:

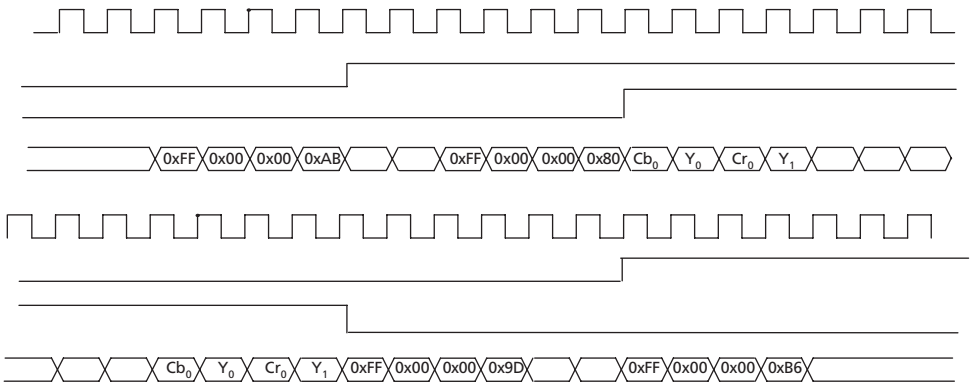
1. Using 8 data output pads (DOUT0-DOUT7), and GPIO[0:1].
2. Using only 8 pads (DOUT0-DOUT7) and a special 8 + 2 data format, shown in Table 11.

The timing of 10-bit or 8-bit data stream output in the bypass mode is qualitatively the same as that depicted in Figure 25 on page 34.

Table 11: 2-Byte RGB Format

Odd bytes	8 data bits	$D_9D_8D_7D_6D_5D_4D_3D_2$
Even bytes	2 data bits + 6 unused bits	0 0 0 0 0 0 D_1D_0

Figure 25: Example of Timing for Non-Decimated Uncompressed Output Bypassing Output FIFO



FIFO

During normal pipeline operation output data rate is determined by number of factors, for instance, input image size, degree of scaling, and sensor operation mode. As these parameters change during normal sensor operation, output frequency changes. This output frequency may generate RF noise, interfering with the mobile device. By using an output FIFO to maintain a constant output clock frequency, noise is easily filtered out.

The FIFO accumulates data and after certain number of bite is stored it will yield them in a single burst making sure that data rate within the burst remains constant. This approach may utilize a free running clock thus making possible RF interference to minimal.

In the default operation mode user must provide two values for proper FIFO operation:

1. the expected line length (R0x3220)
2. watermark (R0x321E)

Value for both has to be specified in bytes. Calculation of line byte count is straightforward; multiply the number of pixels by the number of bytes per pixel. Number of bytes per pixel is 2 for all output modes except processed Bayer in which it equals one byte per pixel.

Watermark

Calculation of the watermark should be done based on maximum number of bytes that pipeline could output in the given mode (with activated clip and no scaling) and number of bites it will yield with account of scaling. Also, account for the horizontal sensor binning mode if such is activated. In this mode, the number of pixels that pipeline can output should be doubled.

Camera Control

General Purpose I/Os

The four general purpose I/Os of the MT9D112 can be configured in multiple ways. Each of the I/Os can be used as a simple input/output that can be programmed from the host. The status of the GPIO is read at power up and can be used as a module ID to separate different module suppliers.

If 10-bit RAW output is required, GPIO[1:0] can be configured as bit 0 and bit 1 of a 10-bit data bus.

GPIO[3:2] can be configured to output a flash pulse to trigger an external Xenon or LED flash or a shutter pulse to control an external shutter.

To configure GPIO[3:0] pins:

- Set GPIO[1:0] to output DOUT_LSB[1:0]: R0x09:1[1:0]= “00”
- Set GPIO[2] to output Shutter: R0x51:1[0] = “1”
- Set GPIO[3] to output Flash: R0x09:1[4:3] = "01"

The general purpose inputs are enabled by setting reset_register[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[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
- Trigger
- Standby functions

The gpi_status register is used to associate a function with a general purpose input.

Output Enable Control

When the parallel pixel data interface is enabled, its signals can be switched asynchronously between the driven and High-Z under pin or register control, as shown in Table 12.

Table 12: Output Enable Control

OE_N Pin	Drive Signals Reg0x301A-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 13 on page 36.

Table 13: Trigger Control

TRIGGER	Global Trigger Reg0x3060-1[0]	Description
Disabled	0	Idle
Disabled	1	Trigger
0	0	Idle
X	1	Trigger
1	X	Trigger

Streaming/Standby Control

The MT9D112 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 14.

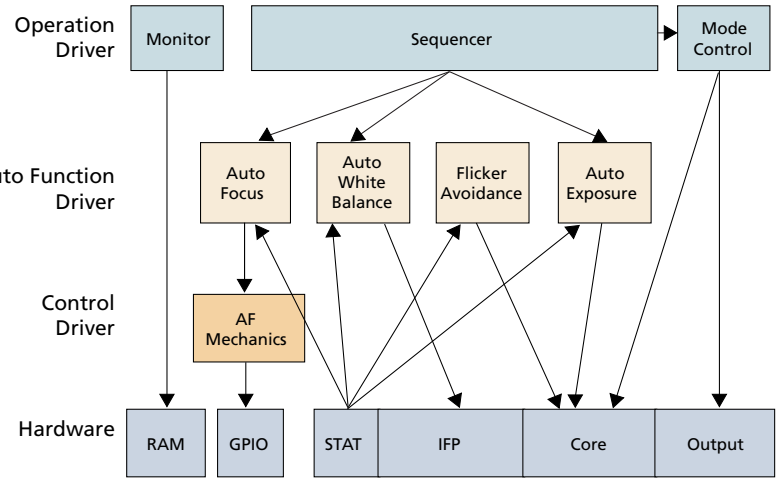
Table 14: Streaming/STANDBY

STANDBY	Streaming Reg0x301A-B[2]	Description
Disabled	0	Soft Standby
Disabled	1	Streaming
X	0	Soft Standby
0	1	Streaming
1	X	Soft Standby

Firmware Architecture

The firmware for the MT9D112 is implemented in multiple drivers that are responsible for different parts of operation. All drivers are executed on the 68H11 microcontroller from the 32kB ROM and use the 2kB RAM to store variables, firmware updates, and code. The firmware can access registers in the senor core and the IFP independently from the user through a ring-bus. The GPIO interface is accessible via special function registers (SFRs) in memory space.

Figure 26: Software Architecture



Sequencer

The sequencer is responsible for coordinating all events triggered by the user. It is implemented as a state machine. For example, sending a capture command to the sequencer will change the resolution from preview to full resolution, settle the AWB/AE, turn on or off an external LED and switch back to preview after capturing the frame. The setup of the sensor can be defined by the user for the following states:

- on entering preview
- in preview
- when leaving preview
- when entering capture

Context and Operational Modes

The MT9D112 can operate in several modes including preview, still capture (snapshot), and video. All modes of operation are individually configurable and are organized as two contexts-context A and context B. A context is defined by sensor image size, frame rate, resolution and other associated parameters. The user can switch between the two contexts by sending a command via the two-wire serial interface.

Preview Mode

Context A is primarily intended for use in the preview mode. During preview, the sensor usually outputs low resolution images at a relatively high frame rate, and its power consumption is kept to a minimum.

Still Capture and Video Modes

Context B can be configured for the still capture or video mode, as required by the user. For still capture configuration, the user typically specifies the desired output image size, if flash should be enabled, how many frames to capture, etc. For video, the user might select a different image size and a fixed frame rate.

Snapshot and Flash

To take a snapshot, the user must send a command that changes the context from A to B. Typical sequence of events after this command are:

- First, the camera may turn on its LED flash, if it has one and is required to use it. With the flash on, the camera exposure and white balance is automatically adjusted to the changed illumination of the scene.
- Next, the camera performs auto focusing. Once in focus, it captures one or more frames of desired size. A camera equipped with a Xenon flash strobes it during the capture. Completing the sequence, the camera automatically returns to context A and resume running preview.

Video

To start video capture, the user must change relevant context B settings, such as capture mode, image size and frame rate, and again send a context change command. Upon receiving it, the MT9D112 switches to the modified context B settings, while continuing to output YUV-encoded image data. Auto exposure and auto focus automatically switches smooth continuous operation. To exit the video capture mode, the user has to send another context change command causing the sensor to switch back to context A.

Auto Exposure

The auto exposure algorithm performs automatic adjustments of the image brightness by controlling exposure time and analog gains of the sensor core as well as digital gains applied to the image.

Auto exposure is implemented by a firmware driver that analyzes image statistics collected by the exposure measurement engine, makes a decision and programs the sensor core and color pipeline to achieve the desired exposure. The measurement engine subdivides the image into 16 windows organized as a 4 x 4 grid.

Two auto exposure algorithm modes are available:

1. preview (luminance-based)
2. evaluative (histogram-based)

Preview Mode

This exposure mode is activated during preview or video capture. It relies on the statistics engine that tracks speed and amplitude of the change of the overall luminance in the selected windows of the image.

The backlight compensation is achieved by weighting the luminance in the center of the image higher than the luminance on the periphery. Other algorithm features include the rejection of fast fluctuations in illumination (time averaging), control of speed of response, and control of the sensitivity to the small changes. While the default settings are adequate in most situations, the user can program target brightness, measurement window, and other parameters described above.

In preview AE, the driver calculates image brightness based on average luma values received from 16 programmable equal-size rectangular windows forming a 4 x 4 grid. In preview mode, 16 windows are combined in 2 segments: central and peripheral. Central segment includes four central windows. All remaining windows belong to peripheral segment. Scene brightness is calculated as average luma in each segment taken with certain weights. Variable ae.weights[3:0] specifies central zone weight, ae.weights[7:4] - peripheral zone weight.

The driver changes AE parameters (IT, Gains, and so on) to drive brightness (ae.CurrentY) to programmable target (ae.Target). Value of one step approach to target is defined by ae.JumpDivisor variable.

Expected brightness is:

Ynew = ae.CurrentY+(ae.Target-ae.CurrentY)/ ae.JumpDivisor

To avoid unwanted reaction of AE on small fluctuations of scene brightness or momentary scene changes, the AE driver uses temporal filter for luma and gate around AE luma target. The driver changes AE parameters only if buffered luma outsteps AE target gates.

Variable ae.lumaBufferSpeed defines buffering level.

32*Ybuf1=Ybuf0*(32-ae.lumaBufferSpeed)+Ycurr* ae.lumaBufferSpeed;

Values ae.lumaBufferSpeed=32 and ae.JumpDivisor=1 specify maximal AE speed.

Evaluative Algorithm

A scene evaluative AE algorithm is available for use in snapshot mode. The algorithm performs scene analysis and classification with respect to its brightness, contrast, and composure and then decides to increase, decrease, or keep original exposure target. It is the most effective for backlight and bright outdoor conditions.

Accelerated Settling During Overexposure

The AE speed is direction-dependent. Transitioning from over saturation to target can take more time than transitioning from under-saturation. The AE driver has a mode which speeds up AE for over exposed scenes (ae.status[7]).

The AE driver counts the number of AE windows (driver variable ae.numOE) which have average brightness equal to or greater than some value, 250 by default. For a scene having saturated regions, the average luma is underestimated due to signal clipping. The driver compensates underestimation by a factor defined by ae.numOE variable.

currentY = ae.CurrentY * coeffOE[ae.numOE] / 16;

where:

const BYTE coeffOE[17] = {16,17, 18, 19, 20, 21, 22, 24, 26, 28, 30, 32, 36, 40, 48, 54, 64};

Exposure Control

To achieve the required amount of exposure, the AE driver adjusts the sensor integration time R9:0, R12:0, gains, ADC reference, and IFP digital gains. To reject flicker, integration time is typically adjusted in increments of ae.R9_step. ae.R9_step specifies duration in row times equal to one flicker period. Thus, flicker is rejected if integration time is kept a natural factor of the flicker period.

Exposure is adjusted differently depending on illumination situation.

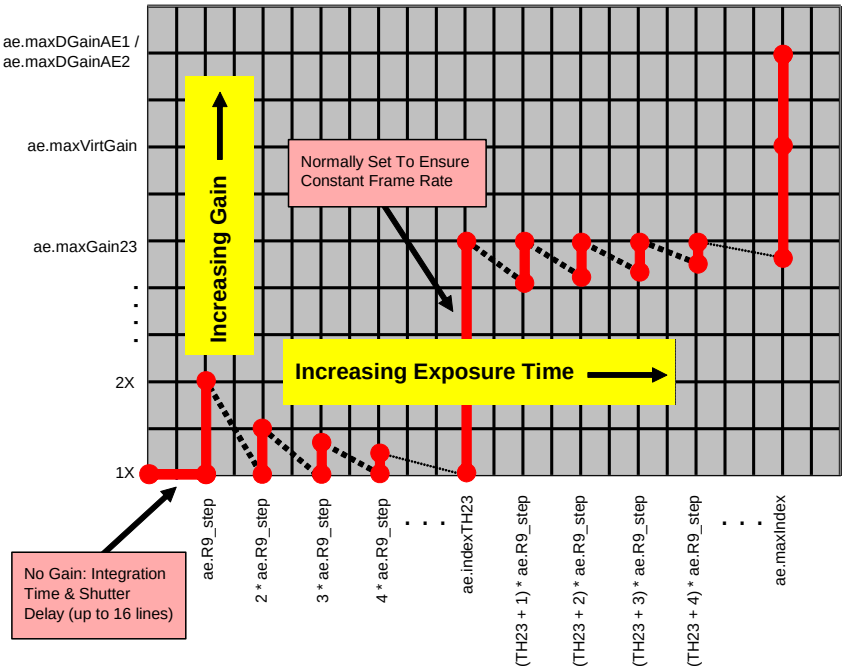
- In extremely bright conditions, the exposure is set using R12:0, R9:0 and analog gains. R12:0 is used to achieve very short integration times.

In this situation:

- R9:0<ae.R9_step and flicker are not rejected.
- In bright conditions where R9:0>=ae., R9_step R9:0 is set as a natural factor of ae.R9_step. Analog gains are also used, but the green gain, also called virtual gain, does not exceed 2x. ae.minVirtGain limits minimal integration time and is expressed in flicker periods. ae.Index indicates the current integration time expressed in the same form.
- Under medium-intensity illumination, the integration time can increase further. For any given exposure, the best signal-to-noise ratio can be typically obtained by using the longest exposure and the smallest gain setting. However, a long exposure time can slow down the output frame rate if the former exceeds the default frame rate, R9:0 >R3:0 + R6:0 + 1. Integration ae.IndexTH23 specifies the breakpoint where AE scheme, giving preference to increasing the shutter width, is replaced with another scheme giving preference to increase in gain. ae.maxGain23 specifies maximum allowed gain in this situation. ae.VirtGain indicates current green channel gain.
- In darker situations, the gain achieves ae.maxGain23 and the integration time is allowed to increase again up to ae.maxIndex.
- In yet darker situations, once the integration time achieves ae.maxIndex, the analog gain is allowed to increase up to ae.maxVirtGain.
- In very dark conditions, the digital IFP gains are allowed to increase up to ae.maxDGainAE1 and ae.maxDGainAE2.

ADC is used as an additional gain stage by adjusting reference levels. See Table 30, “2: AE Variables,” on page 140 for the ae.ADC* variables.

Figure 27: Gain vs. Exposure



AE MDR Mode

The MT9D112 also has an AE MDR mode to extend the dynamic range of a scene after AE has settled brightness. This mode can only be selected in PreviewEnter, PreviewLeave, and CaptureEnter states.

To set this mode, the following registers must be set:

seq.previewParEnter.ae = 4

seq.previewParLeave.ae = 4

seq.capParEnter.ae = 4

Auto White Balance

The MT9D112 has a built-in auto white balance algorithm designed to compensate for the effects of changing spectra of the scene illumination on the quality of the color rendition. The algorithm consists of two major parts: a measurement engine performing statistical analysis of the image and a driver performing the selection of the optimal color correction matrix, digital, and sensor core analog gains. While default settings of these algorithms are adequate in most situations, the user can reprogram base color correction matrices, place limits on color channel gains, and control the speed of both matrix and gain adjustments.

The MT9D112 AWB does not require the presence of gray or white elements in the image for good color rendition. The AWB does not attempt to locate brightest or grayest element of the image but instead performs statistical image analysis to differentiate between changes in predominant spectra of illumination and changes in predominant colors of the scene.

The MT9D112 also includes a module to detect edges based on hue variation which allows the high frequency patterns of the same color to be excluded from the AWB calculation.

Flicker Detection

Flicker occurs when the integration time is not an integer multiple of the period of the light intensity. The automatic flicker detection block does not compensate for the flicker, but rather avoids it by detecting the flicker frequency and adjusting the integration time. For integration times below the light intensity period (10ms for 50Hz environment), flicker cannot be avoided.

Flicker shows as horizontal bars rolling up or down. MCU looks for these rolling bars using a thin horizontal window, which outputs luma average and is applied to 48 points in the upper half of the image. MCU repeats same sampling on the next frame and subtracts 48 samples from previous frame from corresponding samples from the current frame. fd.skipFrame allows skipping more frames between subtraction. MCU then smooths the 48 sampling points, see fd.smooth_cnt, applies an amplitude threshold

fl.minAmplitude to avoid false detection and looks at the resulting waveform. If flicker is present, the waveform should have a frequency within the search range, see fd.search_f1/ 2_50/60. Assuming the flicker power is a sine wave, subtracting two frames results in

$\sin(wt)-\sin(wt+a) = 2\sin(a/2)\cos(wt+a/2)$

which is a cosine wave of the original frequency.

Auto Focus

Algorithm

The auto focus algorithm implemented in the MT9D112 firmware seeks to maximize sharpness of vertical lines in images output by the sensor, by guiding an external lens actuator to the position of best lens focus. The algorithm is actuator-independent, providing guidance by means of an abstract 1-dimensional position variable, leaving the translation of its changes into physical lens movements to a separate AF mechanics (AFM) driver.

For measuring line sharpness, the AF algorithm relies on focus measurement engine in the color pipeline, which is a programmable vertical-edge-filtering module. The module convolves two preprogrammed 1-dimensional digital filters with luminance (Y) data it receives row by row from the color interpolation module. In every interpolated image, the pixels whose Y values are used in the convolution form a rectangular block that can be arbitrarily positioned and sized, and in addition divided into up to 16 equal-size sub-blocks, referred to as AF windows or zones. The absolute values of convolution results are summed separately for each filter over each of the AF windows, yielding up to 32 sums per frame.

There are several motion sequences through which the MT9D112 AF algorithm can bring a lens to best focus position. All these sequences begin with a jump to a preselected start position, for example, the infinity focus position. This jump is referred to as

the first flyback. It is followed by a unidirectional series of steps that puts the lens at up to 19 preselected positions different from the start position. This series of steps is called the first scan.

During the scan, the AF algorithm stops the lens at each preselected position long enough to obtain valid sharpness scores. The normalized score for each AF window is stored along with information on how many zones had a high sharpness score and the position with the maximum sharpness score is determined taking into account the zone information. This way the algorithm can handle scenes with objects at different distances.

After the first scan, the AF algorithm provides a number of ways to proceed with final lens positioning. The user should select a way that best fits the magnitude of lens actuator hysteresis and desired lens proximity to the truly optimal position. Actuators with large, unknown or variable hysteresis should do a second flyback and either jump or retrace the steps of the first scan to the best scanned position. Actuators with constant hysteresis (like stepper motors) can be moved to that position directly from the end position of the scan-the AF algorithm offers an option to automatically increase the length of this move by a preprogrammed backlash-compensating step. Finally, if the first scan is coarse relative to the positioning precision of the lens actuator and depth of field of the lens, an optional second fine scan can be performed around the lens position voted best after the first scan. This second scan is done in the same way as the first, except that the positions it covers are not preselected. Instead, the AF algorithm user must set step size and number of steps for the second scan. The second scan must be followed by the same hysteresis-matching motion sequence as the first scan, for example, a third flyback and jump to the best position.

The AF driver is disabled on power-up. If AF is desired, it must be explicitly enabled after every power up or reset before it can be used. This can be accomplished by:

Establish AF motor type

- set afm.type (0x002) = 129 (for helimorph AF)
  afm.type (0x002) = 130 (for stepper motor AF)
  afm.type (0x002) = 131 (for AD5398)
- turn on AF in preview mode seq.previewPar.af=1
- call "refresh" sequencer command seq.cmd=5

Modes

There are four AF camera modes that the MT9D112 can fully support if it controls the position of the camera lens.

1. Snapshot mode
   In this mode, a camera performs auto focusing upon a user command to do so. When the auto focusing is finished, a snapshot is normally taken and there is no further AF activity until the next appropriate user command. The MT9D112 can do the auto focusing using its own AF algorithm described above or a substitute algorithm loaded into RAM. It can then wait or automatically proceed with other operations required to take a snapshot.
2. Locked mode
   The MT9D112 can be commanded to lock the lens in its current position. Between the command to lock the lens and another to release it, the lens does not respond to other commands or scene changes.
3. Focus-free mode
   In many situations, such as under low light or during video recording, it may be impossible or undesirable to focus the lens prior to every image capture. Instead, the

lens can be locked in a position most likely to produce satisfactory images (the hyperfocal position). This position can be programmed into the MT9D112, and it can move and hold the lens there on command.

4. Manual mode
   In this mode there is no AF activity— focusing the camera is left to the user. The user typically can move the camera lens in steps, by manually issuing commands to the lens actuator, and observe the effect of his actions on a preview display. The MT9D112 can provide 30 fps image input for the display and simultaneously translate user commands received via two-wire serial interface into digital waveforms driving the lens actuator.

Lens Actuator Interface

Actuators used to move lenses in AF cameras can be classified into several broad categories that differ significantly in their requirements for driving signals. These requirements also vary from one device to another within each category. To ensure its compatibility with many different actuators, the MT9D112 includes a general purpose input/output module (GPIO_AF).

The GPIO is a programmable rectangular waveform generator, with 8 individually controllable output pads (GPIO0 through GPIO7), a separate power supply pad (VDD_AF), and a separate clock domain that can be disconnected from the master clock to save power when the GPIO is not in use. The GPIO can toggle its output pads as fast as half the master clock frequency.

An external host processor or the embedded microcontroller of the MT9D112 have two ways to control the voltages on the GPIO output pads:

1. Setting or clearing bits in a control register
   The state of the GPIO pads is updated immediately after writing to the register. Since writing via the two-wire serial interface takes some time, this way does not give the host processor a very precise control over GPIO output timing.
2. Waveform programming
   The second way to obtain a desired output from the GPIO is to program a set of periodic waveforms to the control registers and initialize their generation. The GPIO then generates the programmed waveforms on its own, without waiting for any further input, and therefore with the best attainable timing precision. If necessary, the GPIO can notify the MCU and the host processor about reaching certain points in the waveforms generation, for example, the end of a particular waveform. Every GPIO notification has two components: the GPIO sends a wakeup signal to the MCU and sets a bit in its status register that can be polled by the MCU and/or the host processor. The wakeup signals have an effect only when the MCU is in sleep mode.

The MT9D112 can be set up not only to output digital signals to a lens actuator and/or other similar devices, but also to receive their digital feedback. All GPIO output pads are reconfigurable as high-impedance digital inputs. The logical state of each GPIO pad is mirrored by the state of a bit in a dedicated register, which allows the MCU and host processor to sample digital input signals at intervals equal to their respective register read times.

Two-Wire Serial Interface

The two-wire serial interface bus enables read/write access to control and status registers within the MT9D112.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 VDDIO 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.

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
- a 16-bit register address
- a(an) (no) acknowledge bit
- two data bytes
- 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, and 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 MT9D112 are 0x20 (write address) and 0x21 (read address). Alternate slave addresses of 0x30 (write address) and 0x31 (read address) can be selected by asserting the SADDR input signal.

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.

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.

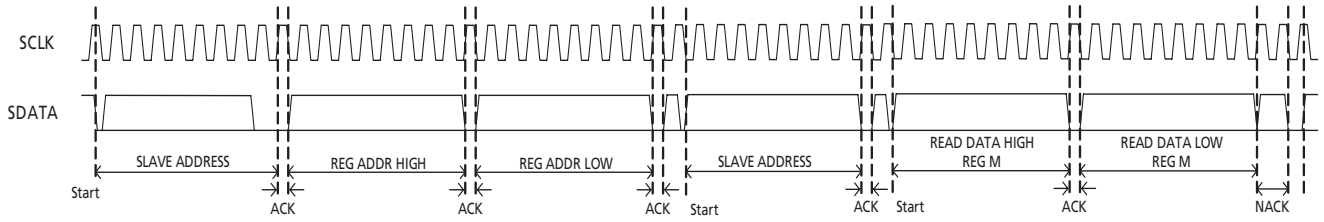
If the request was a write, the master then transfers the 16-bit register address to which a write should take place. This transfers takes place as two 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the data as an 8-bit sequence; the slave sends 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, just as in the 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, eight 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 (see Figure 28 on page 46) 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 28 shows how the internal register address maintained by the MT9D112 is loaded and incremented as the sequence proceeds.

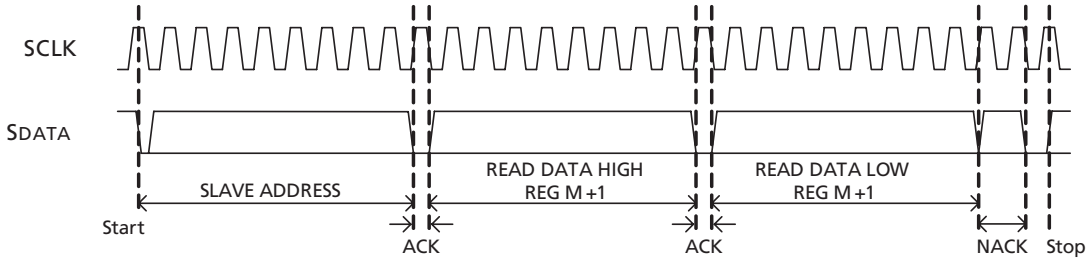
Figure 28: Single Read from Random Location



Single Read from Current Location

This sequence (Figure 29) performs a read using the current value of the MT9D112 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.

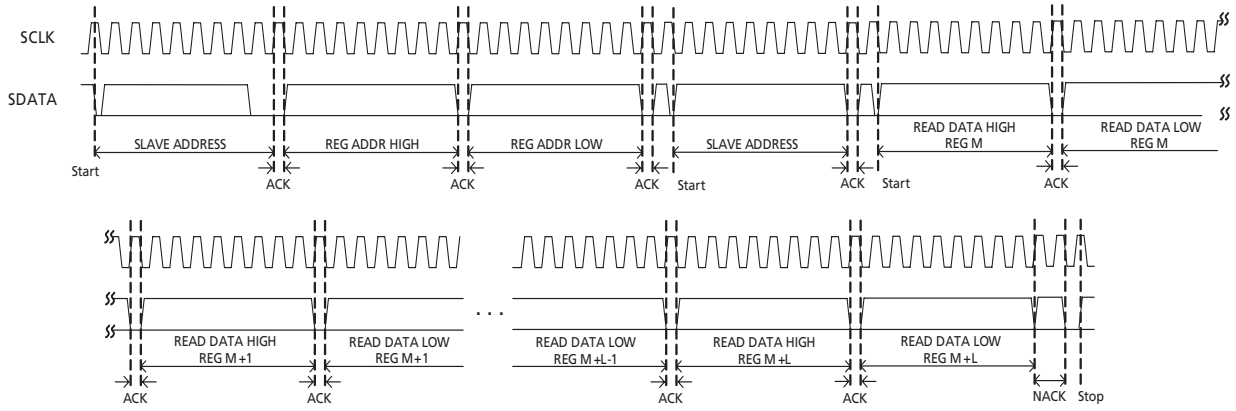
Figure 29: Single Read from Current Location



Sequential Read, Start from Random Location

This sequence (Figure 30) starts in the same way as the single read from random location (Figure 28). 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.

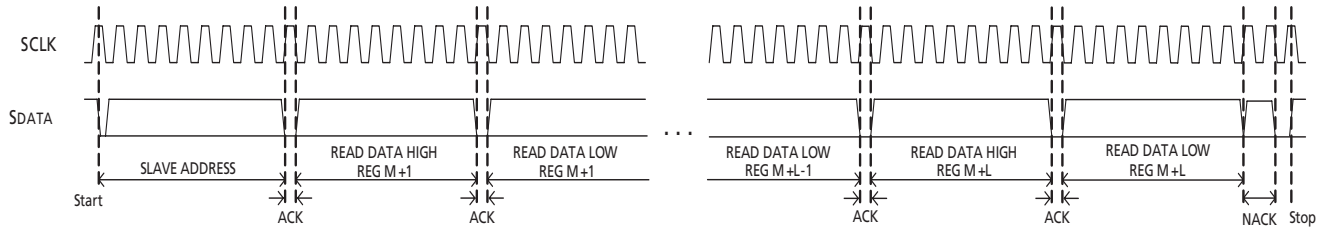
Figure 30: Sequential Read, Start from Random Location



Sequential Read, Start from Current Location

This sequence (Figure 31) starts in the same way as the single read from current 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 reads until L bytes have been read.

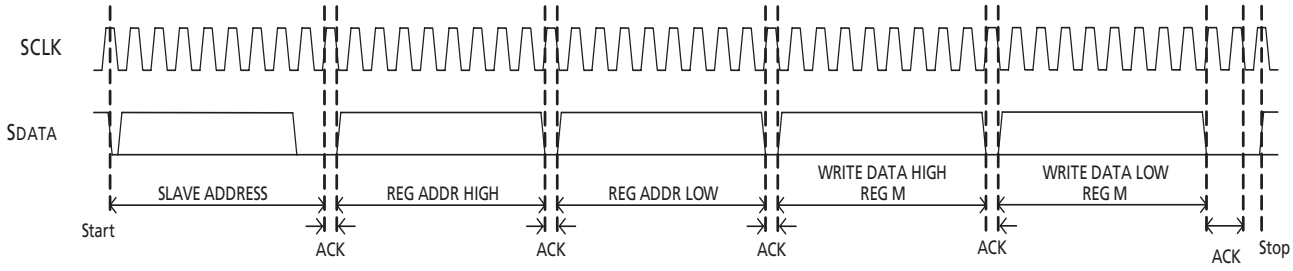
Figure 31: Sequential Read, Start from Current Location



Single Write to Random Location

This sequence (Figure 32) 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.

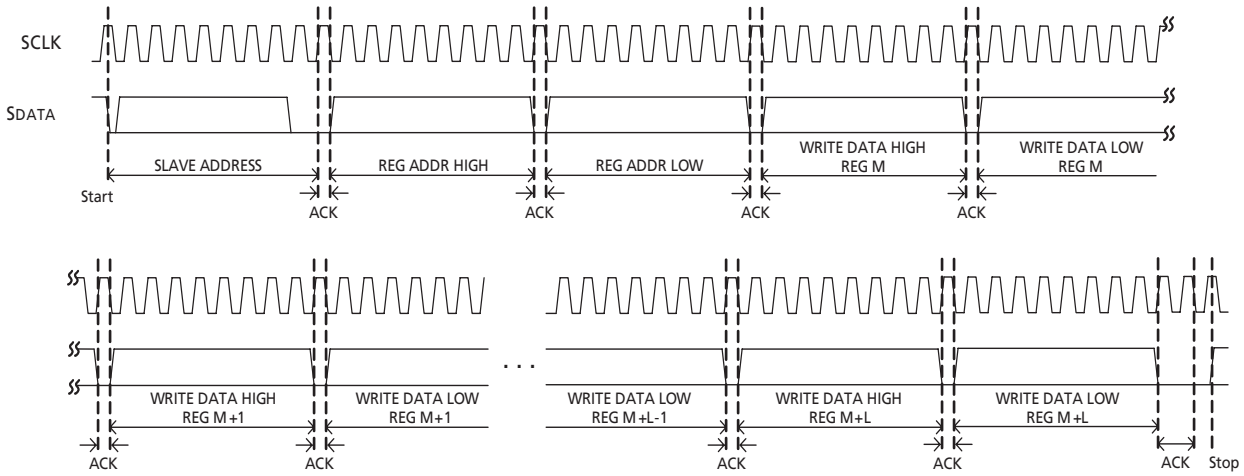
Figure 32: Single Write to Random Location



Sequential Write, Start at Random Location

This sequence (Figure 33) starts in the same way as the single write to random location (Figure 32). 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 33: Sequential Write, Start at Random Location



Registers and Variables

Four types of configuration controls are available:

1. Hardware registers
2. Driver variables
3. Special function registers
4. MCU SRAM

The following convention is used in the text below to designate registers and variables:

R0x3012, R0x3012[3:0] or R12306, R12306[3:0]

These refer to two-wire accessible register number 12306. [3:0] indicate bits. Registers numbers range 0..65535 and bits range 15..0. Not all register numbers are used.

- ae.Target
  This refers to variable 'Target" in the AE driver.
- SFR 0x1080 or SRAM 0x0400
  This refers to special function register or SRAM located at address 0x1080 in MCU memory space.

How to Access Registers and Variables

Registers, variables, and SFRs are accessed in different ways.

Registers

Hardware registers grouped internally by pages.

R0x3000 – R0x31FE = Senor Core Page 0

R0x3200 – R0x33FE = SoC Page 1

R0X3400 – R0x35BE = SoC Page 2

- Page 0 contains sensor controls.
- Page 1 contains color pipeline controls.
- Page 2 contains MIPI, output FIFO and more color pipeline controls.

Not all the registers in each page are used, not all the bits in each register might be used. Registers are accessed by serial interface READ and WRITE consisting of a 16-bit address and 16-bit data.

Variables

Variables are located in the microcontroller RAM memory. Each driver, such as auto exposure, white balance, auto focus, etc., has a unique driver ID (0..31) and a set of public variables organized as a structure. Each variable in this structure is uniquely identified by its offset from the top of the structure and its size. The size can be 1 or 2 bytes, while the offset is 1 byte.

All driver variables (public and private) can be accessed via R0x338C and R0x3390. While two access modes are available-access by physical address and by logical address, the public variables are typically accessed by the logical method. The logical address, which is set in R0x338C, consists of a 5-bit driver ID number and a variable offset. Examples are provided below.

To set the variable ae.Target=50:

- The variable has a driver ID of 2. Therefore, set R0x338C[12:8] = 2.

- The variable has an offset of 6. Therefore, set R0x338C[7:0] = 6.
- This is a logical access. Therefore, set R0x338C[14:13] = 01.
- The size of the variable is 8 bits. Therefore, set R0x338C[15] = 1.
- By combining these bits, R0x338C = 0xA206.
- Set R0x3390 = 50 for the value of the variable.

To read the variable ae.Target:
- Since this is the same variable as the above example, R0x338C = 0xA206.
- Read R0x3390 for the current variable value.

To set the variable mon.arg1=0x1234:
- The variable has a driver ID of 0. Therefore, set R338C[12:8] = 0.
- The variable has an offset of 3. Therefore, set R0x3390[7:0] = 3.
- This is a logical access. Therefore, set R0x338C[14:13] = 01.
- The size of the variable is 16 bits. Therefore, set R0x338C[15] = 0.
- By combining these bits, R0x338C = 0x2003.
- Set R0x3390 = 0x1234 for the value of the variable.

To read the variable mon.arg1:
- Since this is the same variable as the above example, R0x338C = 0x2003.
- Read R0x3390 for the current variable value.

Special Function Registers and MCU SRAM

Special function registers are registers connected to the local bus of the microcontroller. These registers include GPIO, waveform generator, and those important for firmware operation. SFR are accessed by physical address. MCU SRAM consists of 1K system memory and 1K user memory. Examples of access:

- Write into user SRAM. Use to upload code
  a. R0x338C = 0x400 // address
  b. R0x3390 = 0x1234 // write 16-bit value
- Read from user SRAM
  c. R0x338C = 0x400 // address
  d. Read R0x3390 // read 16-bit value
- Write to 8-bit GPIO register
  e. R0x338C = 0x9079 // GPIO_DIR_L at 0x1079
  f. R0x3390 = 0x00FE // Configure GPIO[0] as output
- Read from 8-bit GPIO register
  g. R0x3 38C = 0x9079 // GPIO_DIR_L at 0x1079
  h. Read R0x3390 //Check GPIO[7:0] pad state

Core Registers

Double-buffered Registers

Some sensor settings cannot be changed during frame readout. For example, changing x_addr_end R0x3008 part way through frame readout results in inconsistent LINE_VALID behavior. To avoid this, the sensor core 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. By default, this occurs 10 row times before FRAME_VALID goes HIGH.

R0x0248[15] can be used to inhibit transfers from the pending to the live registers. This control should be used when making many register changes that must take effect simultaneously.

Bad Frames

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

Many changes to the sensor register settings can cause a bad frame. For example, when x_addr_end R0x3008 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 has been integrated using the old row width. Consequently, reading it out using the new row width results in a frame with an incorrect integration time.

By default, most bad frames are masked: LINE_VALID and FRAME_VALID are inhibited for these frames so that the vertical blanking time between frames is extended by the frame time. This feature can be disabled with R0x0248[10].

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. 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 on the new value of the integration time.
3. When frame (*n* + 1) is read out, it is integrated using the new integration time. If the integration time is changed on successive frames, each value written is applied to a single frame; the latency between writing a value and it affecting the frame readout remains at two frames.

Changes to Gain Settings

When the gain settings are changed, the gain is usually updated on the next frame start. When the integration time and the gain are changed simultaneously, 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.

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

Register List and Default Value

Table 12: 0: Core Registers

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)
12288(0x3000)	model_id_	dddd dddd dddd dddd	5504 (0x1580)
12290(0x3002)	y_addr_start_	0000 0ddd dddd dddd	4 (0x0004)
12292(0x3004)	x_addr_start_	0000 dddd dddd dddd	4 (0x0004)
12294(0x3006)	y_addr_end_	0000 0ddd dddd dddd	1211 (0x04BB)
12296(0x3008)	x_addr_end_	0000 dddd dddd dddd	1611 (0x064B)
12298(0x300A)	frame_length_lines_	dddd dddd dddd dddd	1261 (0x04ED)
12300(0x300C)	line_length_pck_	0000 dddd dddd dddd	2084 (0x0824)
12302(0x300E)	sample_time_pck	0000 dddd dddd dddd	288 (0x0120)
12304(0x3010)	fine_correction	0ddd dddd dddd dddd	133 (0x0085)
12306(0x3012)	coarse_integration_time_	dddd dddd dddd dddd	16 (0x0010)
12308(0x3014)	fine_integration_time_	dddd dddd dddd dddd	330 (0x014A)
12310(0x3016)	row_speed	ddd0 00dd dddd 0ddd	8465 (0x2111)
12312(0x3018)	extra_delay	0000 dddd dddd dddd	0 (0x0000)
12314(0x301A)	reset_register	d00d dddd dddd dddd	584 (0x0248)
12316(0x301C)	image_orientation_mode_select_	0000 000d 0000 00dd	0 (0x0000)
12318(0x301E)	data_pedestal_	0000 00dd dddd dddd	42 (0x002A)
12320(0x3020)	software_reset_	0000 000d 0000 0000	0 (0x0000)
12322(0x3022)	mask_corrupted_frames_grouped_parameter_hold_	0000 000d 0000 000d	1 (0x0001)
12324(0x3024)	pixel_order_	0000 00?? 0000 00dd	0 (0x0000)
12326(0x3026)	gpi_status	dddd dddd d000 ????	65423 (0xFF8F)
12328(0x3028)	analogue_gain_code_global_	0000 0000 0ddd dddd	8 (0x0008)
12330(0x302A)	analogue_gain_code_greenR_	0000 0000 0ddd dddd	8 (0x0008)
12332(0x302C)	analogue_gain_code_red_	0000 0000 0ddd dddd	8 (0x0008)
12334(0x302E)	analogue_gain_code_blue_	0000 0000 0ddd dddd	8 (0x0008)
12336(0x3030)	analogue_gain_code_greenB_	0000 0000 0ddd dddd	8 (0x0008)
12338(0x3032)	digital_gain_greenR_	dddd dddd dddd ddd0	256 (0x0100)
12340(0x3034)	digital_gain_red_	dddd dddd dddd ddd0	256 (0x0100)
12342(0x3036)	digital_gain_blue_	dddd dddd dddd ddd0	256 (0x0100)
12344(0x3038)	digital_gain_greenB_	dddd dddd dddd ddd0	256 (0x0100)
12348(0x303C)	standby_status	0000 0000 0000 00dd	0 (0x0000)
12352(0x3040)	read_mode	dd00 dd0d dddd dddd	36 (0x0024)
12356(0x3044)	Reserved	–	1344 (0x0540)
12358(0x3046)	flash	??dd dddd 0000 0000	1536 (0x0600)
12360(0x3048)	flash_count	0000 00dd dddd dddd	8 (0x0008)
12374(0x3056)	green1_gain	0000 0ddd dddd dddd	272 (0x0110)
12376(0x3058)	blue_gain	0000 0ddd dddd dddd	272 (0x0110)
12378(0x305A)	red_gain	0000 0ddd dddd dddd	272 (0x0110)
12380(0x305C)	green2_gain	0000 0ddd dddd dddd	272 (0x0110)
12382(0x305E)	global_gain	0000 0ddd dddd dddd	272 (0x0110)
12400(0x3070)	test_pattern_mode_	0000 0000 0000 dddd	0 (0x0000)
12402(0x3072)	test_data_red_	0000 00dd dddd dddd	0 (0x0000)

Table 12: 0: Core Registers (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)
12404(0x3074)	test_data_greenR_	0000 00dd dddd dddd	0 (0x0000)
12406(0x3076)	test_data_blue_	0000 00dd dddd dddd	0 (0x0000)
12408(0x3078)	test_data_greenB_	0000 00dd dddd dddd	0 (0x0000)
12416(0x3080)	Reserved	–	32 (0x0020)
12418(0x3082)	Reserved	–	8224 (0x2020)
12420(0x3084)	Reserved	–	8224 (0x2020)
12422(0x3086)	Reserved	–	8224 (0x2020)
12424(0x3088)	Reserved	–	4128 (0x1020)
12426(0x308A)	Reserved	–	8199 (0x2007)
12428(0x308C)	Reserved	–	1284 (0x0504)
12430(0x308E)	Reserved	–	1805 (0x070D)
12432(0x3090)	Reserved	–	1543 (0x0607)
12434(0x3092)	Reserved	–	0 (0x0000)
12436(0x3094)	Reserved	–	0 (0x0000)
12438(0x3096)	Reserved	–	0 (0x0000)
12440(0x3098)	Reserved	–	24 (0x0018)
12442(0x309A)	Reserved	–	256 (0x0100)
12444(0x309C)	Reserved	–	51 (0x0033)
12446(0x309E)	Reserved	–	5137 (0x1411)
12448(0x30A0)	x_even_inc_	???? ???? ???? ????	1 (0x0001)
12450(0x30A2)	x_odd_inc_	0000 0000 0000 0ddd	1 (0x0001)
12452(0x30A4)	y_even_inc_	???? ???? ???? ????	1 (0x0001)
12454(0x30A6)	y_odd_inc_	0000 0000 0000 0ddd	1 (0x0001)
12464(0x30B0)	Reserved	–	0 (0x0000)
12470(0x30B6)	Green1 Frame Average	0000 0000 0??? ????	0 (0x0000)
12472(0x30B8)	Blue Frame Average	0000 0000 0??? ????	0 (0x0000)
12474(0x30BA)	Red Frame Average	0000 0000 0??? ????	0 (0x0000)
12476(0x30BC)	Green2 Frame Average	0000 0000 0??? ????	0 (0x0000)
12478(0x30BE)	calib_threshold	0ddd dddd 0ddd dddd	8989 (0x231D)
12480(0x30C0)	calib_control	dd0d 0000 dddd 0ddd	16512 (0x4080)
12482(0x30C2)	calib_green1	0000 000d dddd dddd	0 (0x0000)
12484(0x30C4)	calib_blue	0000 000d dddd dddd	0 (0x0000)
12486(0x30C6)	calib_red	0000 000d dddd dddd	0 (0x0000)
12488(0x30C8)	calib_green2	0000 000d dddd dddd	0 (0x0000)
12490(0x30CA)	Reserved	–	4 (0x0004)
12492(0x30CC)	Reserved	–	0 (0x0000)
12494(0x30CE)	Reserved	–	0 (0x0000)
12496(0x30D0)	Reserved	–	0 (0x0000)
12498(0x30D2)	Reserved	–	0 (0x0000)
12500(0x30D4)	Reserved	–	32800 (0x8020)
12502(0x30D6)	Reserved	–	512 (0x0200)
12504(0x30D8)	Reserved	–	0 (0x0000)
12506(0x30DA)	Reserved	–	1 (0x0001)
12508(0x30DC)	Reserved	–	0 (0x0000)

Table 12: 0: Core Registers (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)
12510(0x30DE)	Reserved	––	0 (0x0000)
12512(0x30E0)	Reserved	–	38408 (0x9608)
12514(0x30E2)	Reserved	–	31050 (0x794A)
12516(0x30E4)	Reserved	–	38521 (0x9679)
12518(0x30E6)	Reserved	–	30800 (0x7850)
12520(0x30E8)	Reserved	–	38523 (0x967B)
12522(0x30EA)	Reserved	–	11528 (0x2D08)
12524(0x30EC)	Reserved	–	18989 (0x4A2D)
12526(0x30EE)	Reserved	–	11540 (0x2D14)
12528(0x30F0)	Reserved	–	18995 (0x4A33)
12530(0x30F2)	Reserved	–	38405 (0x9605)
12532(0x30F4)	Reserved	–	38403 (0x9603)
12534(0x30F6)	Reserved	–	38421 (0x9615)
12536(0x30F8)	Reserved	–	38438 (0x9626)
12538(0x30FA)	Reserved	–	19208 (0x4B08)
12540(0x30FC)	Reserved	–	159 (0x009F)
12542(0x30FE)	Reserved	–	24394 (0x5F4A)
12544(0x3100)	Reserved	–	25422 (0x634E)
12546(0x3102)	Reserved	–	30819 (0x7863)
12548(0x3104)	Reserved	–	38272 (0x9580)
12550(0x3106)	Reserved	–	11272 (0x2C08)
12552(0x3108)	Reserved	–	18736 (0x4930)
12554(0x310A)	Reserved	–	38401 (0x9601)
12556(0x310C)	Reserved	–	38401 (0x9601)
12558(0x310E)	Reserved	–	38408 (0x9608)
12560(0x3110)	Reserved	–	38145 (0x9501)
12566(0x3116)	Reserved	–	38400 (0x9600)
12568(0x3118)	Reserved	–	4486 (0x1186)
12570(0x311A)	Reserved	–	38400 (0x9600)
12572(0x311C)	Reserved	–	16774 (0x4186)
12574(0x311E)	Reserved	–	38400 (0x9600)
12576(0x3120)	Reserved	–	16774 (0x4186)
12578(0x3122)	Reserved	–	11415 (0x2C97)
12592(0x3130)	Reserved	–	11264 (0x2C00)
12594(0x3132)	Reserved	–	10509 (0x290D)
12596(0x3134)	Reserved	–	9473 (0x2501)
12598(0x3136)	Reserved	–	10512 (0x2910)
12600(0x3138)	Reserved	–	11265 (0x2C01)
12602(0x313A)	Reserved	–	17094 (0x42C6)
12604(0x313C)	Reserved	–	11265 (0x2C01)
12606(0x313E)	Reserved	–	16902 (0x4206)
12608(0x3140)	Reserved	–	4459 (0x116B)
12610(0x3142)	Reserved	–	6252 (0x186C)
12612(0x3144)	Reserved	–	17489 (0x4451)

Table 12: 0: Core Registers (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)
12614(0x3146)	Reserved	–-	17489 (0x4451)
12640(0x3160)	global_seq_trigger	0000 0000 0000 0ddd	0 (0x0000)
12642(0x3162)	global_rst_end	dddd dddd dddd dddd	0 (0x0000)
12644(0x3164)	global_shutter_start	dddd dddd dddd dddd	0 (0x0000)
12646(0x3166)	global_read_start	dddd dddd dddd dddd	0 (0x0000)
12672(0x3180)	Reserved	–	32768 (0x8000)
12674(0x3182)	Reserved	–	0 (0x0000)
12676(0x3184)	Reserved	–	0 (0x0000)
12678(0x3186)	Reserved	–	0 (0x0000)
12680(0x3188)	Reserved	–	0 (0x0000)
12776(0x31E8)	horizontal_cursor_position_	0000 0ddd dddd dddd	0 (0x0000)
12778(0x31EA)	vertical_cursor_position_	0000 dddd dddd dddd	0 (0x0000)
12780(0x31EC)	horizontal_cursor_width_	0000 0ddd dddd dddd	0 (0x0000)
12782(0x31EE)	vertical_cursor_width_	0000 dddd dddd dddd	0 (0x0000)
12788(0x31F4)	Reserved	–	291 (0x0123)
12790(0x31F6)	Reserved	–	17767 (0x4567)
12792(0x31F8)	Reserved	–	35243 (0x89AB)
12794(0x31FA)	Reserved	–	52719 (0xCDEF)
12796(0x31FC)	I2C IDS	dddd dddd dddd dddd	31352 (0x7A78)
15872(0x3E00)	Reserved	–	0 (0x0000)
16128(0x3F00)	Reserved	–	0 (0x0000)

Table 13: 1: SOC

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)
12802(0x3202)	Standby Control	?000 0000 00dd dddd	9 (0x0009)
12804(0x3204)	Standby Done Status Bit	0000 0000 0000 0ddd	1 (0x0001)
12816(0x3210)	Color Pipeline Control	0000 0ddd dddd dddd	504 (0x01F8)
12818(0x3212)	Factory Bypass	dddd dddd dddd dddd	1 (0x0001)
12820(0x3214)	Pad Slew	0000 dddd dddd dddd	3200 (0x0C80)
12822(0x3216)	Internal clock control	0000 00dd dddd dddd	0 (0x0000)
12828(0x321C)	Output FIFO control and status	dddd d??? dddd dddd	2 (0x0002)
12830(0x321E)	Output FIFO watermark	0000 00dd dddd dddd	400 (0x0190)
12832(0x3220)	Output FIFO Line byte count	0000 dddd dddd dddd	1600 (0x0640)
12834(0x3222)	Lower X Boundary for Zoom Window	0000 0ddd dddd dddd	0 (0x0000)
12836(0x3224)	Upper X Boundary for Zoom Window	0000 0ddd dddd dddd	1600 (0x0640)
12838(0x3226)	Lower Y Boundary for Zoom Window	0000 0ddd dddd dddd	0 (0x0000)
12840(0x3228)	Upper Y Boundary for Zoom Window	0000 0ddd dddd dddd	1200 (0x04B0)
12842(0x322A)	Scaler Control	dddd dddd dddd dddd	0 (0x0000)
12844(0x322C)	Weight for Horizontal Scaling	0000 dddd dddd dddd	2048 (0x0800)
12846(0x322E)	Weight for Vertical Scaling	0000 dddd dddd dddd	2048 (0x0800)
12862(0x323E)	Edge Threshold for WB Statistic	dddd dddd dddd dddd	784 (0x0310)

Table 13: 1: SOC (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)
12864(0x3240)	Luminance Range of Pixels Considered in WB Statistics	dddd dddd dddd dddd	51220 (0xC814)
12888(0x3258)	Reserved	–	0 (0x0000)
12890(0x325A)	Right/Left Coordinates of AWB Measurement Window	dddd dddd dddd dddd	20480 (0x5000)
12892(0x325C)	Bottom/Top Coordinates of AWB Measurement Window	dddd dddd dddd dddd	15360 (0x3C00)
12896(0x3260)	Red Chrominance Measure Calculated by AWB	???? ???? ???? ????	0 (0x0000)
12898(0x3262)	Luminance Measure Calculated by AWB	???? ???? ???? ????	0 (0x0000)
12900(0x3264)	Blue Chrominance Measure Calculated by AWB	???? ???? ???? ????	0 (0x0000)
12902(0x3266)	Reserved	–	0 (0x0000)
12906(0x326A)	1D Aperture Parameters	00dd dddd dddd dddd	4616 (0x1208)
12908(0x326C)	Aperture Parameters	0ddd dddd dddd dddd	4616 (0x1208)
12910(0x326E)	Low pass YUV filters	0000 0000 dddd dddd	128 (0x0080)
12912(0x3270)	Threshold for Y Filter. R Channel	0000 dddd dddd dddd	1962 (0x07AA)
12914(0x3272)	Threshold for Y filter.G channel	0000 dddd dddd dddd	2020 (0x07E4)
12916(0x3274)	Threshold for Y filter. B channel	0000 0000 0ddd dddd	42 (0x002A)
12918(0x3276)	Second Black Level	0000 00dd dddd dddd	0 (0x0000)
12920(0x3278)	First Black Level	0000 00dd dddd dddd	42 (0x002A)
12922(0x327A)	First Black Level. Red	0000 00dd dddd dddd	42 (0x002A)
12924(0x327C)	First Black Level. Green_1	0000 00dd dddd dddd	42 (0x002A)
12926(0x327E)	First Black Level. Green_2	0000 00dd dddd dddd	42 (0x002A)
12928(0x3280)	First Black Level. Blue	0000 00dd dddd dddd	42 (0x002A)
12942(0x328E)	Edge Threshold for Interpolation	0000 0000 dddd dddd	8 (0x0008)
12944(0x3290)	Test Pattern	0000 0000 0ddd dddd	0 (0x0000)
12946(0x3292)	Test Pattern R Value	0000 00dd dddd dddd	256 (0x0100)
12948(0x3294)	Test Pattern G Value	0000 00dd dddd dddd	256 (0x0100)
12950(0x3296)	Test Pattern B Value	0000 00dd dddd dddd	256 (0x0100)
12956(0x329C)	Digital Gain 2	0000 0000 dddd dddd	32 (0x0020)
12958(0x329E)	Reserved	–	0 (0x0000)
12960(0x32A0)	Factory Test - CRC Data	???? ???? ???? ????	0 (0x0000)
12962(0x32A2)	Reserved	–	5696 (0x1640)
12964(0x32A4)	Reserved	–	32640 (0x7F80)
12992(0x32C0)	Color Correction Matrix Exponents for C11..C22	0ddd dddd dddd dddd	10531 (0x2923)
12994(0x32C2)	Color Correction Matrix Exponents for C22..C33	0000 dddd dddd dddd	1316 (0x0524)
12996(0x32C4)	Color Correction Matrix Elements 1 and 2	dddd dddd dddd dddd	48072 (0xBBC8)
12998(0x32C6)	Color Correction Matrix Elements 3 and 4	dddd dddd dddd dddd	51770 (0xCA3A)
13000(0x32C8)	Color Correction Matrix Elements 5 and 6	dddd dddd dddd dddd	15237 (0x3B85)
13002(0x32CA)	Color Correction Matrix Elements 7 and 8	dddd dddd dddd dddd	62063 (0xF26F)
13004(0x32CC)	Color Correction Matrix Element 9 and Signs	00dd dddd dddd dddd	15772 (0x3D9C)
13012(0x32D4)	Digital Gain 1 for Red Pixels	0000 00dd dddd dddd	128 (0x0080)
13014(0x32D6)	Digital Gain 1 for Green1 Pixels	0000 00dd dddd dddd	128 (0x0080)
13016(0x32D8)	Digital Gain 1 for Green2 Pixels	0000 00dd dddd dddd	128 (0x0080)
13018(0x32DA)	Digital Gain 1 for Blue Pixels	0000 00dd dddd dddd	128 (0x0080)

Table 13: 1: SOC (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)
13020(0x32DC)	Digital Gain 1 for All Colors	0000 00dd dddd dddd	128 (0x0080)
13044(0x32F4)	Boundaries of Flicker Measurement Window [Left/ Width]	dddd dddd dddd dddd	80 (0x0050)
13046(0x32F6)	Boundaries of Flicker Measurement Window [Top/ Height]	dddd dddd dddd dddd	136 (0x0088)
13048(0x32F8)	Flicker Measurement Window Size	dddd dddd dddd dddd	5120 (0x1400)
13050(0x32FA)	Measure of the Average Luminance in Flicker Measurement Window	0000 0000 ???? ????	0 (0x0000)
13100(0x332C)	Blank Frames	0000 0000 0000 000d	0 (0x0000)
13102(0x332E)	Output format configuration	0000 000d dddd dddd	0 (0x0000)
13104(0x3330)	Output Format Test	0000 dddd dddd dddd	0 (0x0000)
13106(0x3332)	Line Count	0000 ???? ???? ????	0 (0x0000)
13108(0x3334)	Frame Count	???? ???? ???? ????	0 (0x0000)
13110(0x3336)	Reserved	–	0 (0x0000)
13112(0x3338)	Reserved	–	0 (0x0000)
13114(0x333A)	Reserved	–	0 (0x0000)
13116(0x333C)	Reserved	–	0 (0x0000)
13118(0x333E)	Reserved	–	0 (0x0000)
13120(0x3340)	Reserved	–	0 (0x0000)
13122(0x3342)	Reserved	–	0 (0x0000)
13124(0x3344)	Reserved	–	0 (0x0000)
13126(0x3346)	Reserved	–	514 (0x0202)
13128(0x3348)	Special Effects	dddd dddd dddd dddd	25664 (0x6440)
13130(0x334A)	Sepia Constants	dddd dddd dddd dddd	45091 (0xB023)
13136(0x3350)	Reserved	–	0 (0x0000)
13138(0x3352)	Reserved	–	0 (0x0000)
13140(0x3354)	Reserved	–	0 (0x0000)
13142(0x3356)	Reserved	–	0 (0x0000)
13144(0x3358)	Reserved	–	0 (0x0000)
13146(0x335A)	Reserved	–	0 (0x0000)
13148(0x335C)	Reserved	–	0 (0x0000)
13150(0x335E)	Reserved	–	0 (0x0000)
13152(0x3360)	Reserved	–	0 (0x0000)
13154(0x3362)	Reserved	–	0 (0x0000)
13156(0x3364)	Gamma Curve Knees 0 and 1	dddd dddd dddd dddd	9984 (0x2700)
13158(0x3366)	Gamma Curve Knees 2 and 3	dddd dddd dddd dddd	18742 (0x4936)
13160(0x3368)	Gamma Curve Knees 4 and 5	dddd dddd dddd dddd	30820 (0x7864)
13162(0x336A)	Gamma Curve Knees 6 and 7	dddd dddd dddd dddd	38793 (0x9789)
13164(0x336C)	Gamma Curve Knees 8 and 9	dddd dddd dddd dddd	45220 (0xB0A4)
13166(0x336E)	Gamma Curve Knees 10 and 11	dddd dddd dddd dddd	50619 (0xC5BB)
13168(0x3370)	Gamma Curve Knees 12 and 13	dddd dddd dddd dddd	55246 (0xD7CE)
13170(0x3372)	Gamma Curve Knees 14 and 15	dddd dddd dddd dddd	59616 (0xE8E0)
13172(0x3374)	Gamma Curve Knees 16 and 17	dddd dddd dddd dddd	63728 (0xF8F0)
13174(0x3376)	Gamma Curve Knee 18	0000 0000 dddd dddd	255 (0x00FF)

Table 13: 1: SOC (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)
13176(0x3378)	Reserved	–	0 (0x0000)
13180(0x337C)	YUV/YCbCr Control	0000 0000 0000 dddd	6 (0x0006)
13182(0x337E)	Y/RGB Offset	dddd dddd dddd dddd	0 (0x0000)
13188(0x3384)	Reserved	–	0 (0x0000)
13190(0x3386)	Microcontroller Boot Mode	???? ???? dddd dddd	X
13196(0x338C)	Microcontroller variable/RAM address	dddd dddd dddd dddd	0 (0x0000)
13200(0x3390)	MCU variable/RAM data (burst 0)	dddd dddd dddd dddd	X
13202(0x3392)	Variable data burst SHIP access 1	dddd dddd dddd dddd	0 (0x0000)
13204(0x3394)	Variable data burst SHIP access 2	dddd dddd dddd dddd	0 (0x0000)
13206(0x3396)	Variable data burst SHIP access 3	dddd dddd dddd dddd	0 (0x0000)
13208(0x3398)	Variable data burst SHIP access 4	dddd dddd dddd dddd	0 (0x0000)
13210(0x339A)	Variable data burst SHIP access 5	dddd dddd dddd dddd	0 (0x0000)
13212(0x339C)	Variable data burst SHIP access 6	dddd dddd dddd dddd	0 (0x0000)
13214(0x339E)	Variable data burst SHIP access 7	dddd dddd dddd dddd	0 (0x0000)
13260(0x33CC)	Reserved	–	0 (0x0000)
13262(0x33CE)	Reserved	–	0 (0x0000)
13264(0x33D0)	Reserved	–	0 (0x0000)
13266(0x33D2)	Reserved	–	0 (0x0000)
13288(0x33E8)	Reserved	–	16385 (0x4001)
13290(0x33EA)	Reserved	–	0 (0x0000)
13292(0x33EC)	Reserved	–	0 (0x0000)
13294(0x33EE)	Reserved	–	0 (0x0000)
13300(0x33F4)	Reserved	–	788 (0x0314)
13302(0x33F6)	Threshold for noise reduction - Red	0000 0000 dddd dddd	32 (0x0020)
13304(0x33F8)	Threshold for noise reduction - Green	0000 0000 dddd dddd	32 (0x0020)
13306(0x33FA)	Threshold for noise reduction - Blue	0000 0000 dddd dddd	32 (0x0020)
13308(0x33FC)	Threshold for noise reduction - Min./ Max.	0000 0000 dddd dddd	255 (0x00FF)

Table 14: 2: SOC2

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)
13312(0x3400)	MIPI Control	0000 0000 000d dddd	12 (0x000C)
13314(0x3402)	MIPI Status	0000 000d dddd dddd	273 (0x0111)
13316(0x3404)	MIPI Data Type	000d dddd 00dd dddd	43 (0x002B)
13318(0x3406)	MIPI Frame Line Count	dddd dddd dddd dddd	1200 (0x04B0)
13320(0x3408)	Reserved	–	2000 (0x07D0)
13322(0x340A)	MIPI FIFO Watermark	00dd 00dd dddd dddd	750 (0x02EE)
13324(0x340C)	custom_short_pkt_wc	dddd dddd dddd dddd	0 (0x0000)
13326(0x340E)	MIPI Test	000d 0ddd 000d dddd	0 (0x0000)
13328(0x3410)	MIPI Pre HS Tx	0000 dddd 0000 dddd	2818 (0x0B02)
13330(0x3412)	MIPI Post HS Tx	00dd dddd 0000 dddd	2565 (0x0A05)
13332(0x3414)	MIPI Clock Overlap	00dd dddd 00dd dddd	2570 (0x0A0A)
13334(0x3416)	Clock Timing	0000 dddd 00dd dddd	1548 (0x060C)

Table 14: 2: SOC2 (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)
13336(0x3418)	Misc Timing	0000 0000 00dd dddd	8 (0x0008)
13338(0x341A)	Reserved	–	2128 (0x0850)
13340(0x341C)	PLL Dividers1	00dd dddd dddd dddd	336 (0x0150)
13342(0x341E)	PLL/ Clk_in control	dddd dddd 000d dddd	36619 (0x8F0B)
13440(0x3480)	Boundaries of First AF Measurement Window [Top/Left]	dddd dddd dddd dddd	3860 (0x0F14)
13442(0x3482)	Boundaries of First AF Measurement Window [Height/ Width]	dddd dddd dddd dddd	7720 (0x1E28)
13444(0x3484)	AF Measurement Window Size	dddd dddd dddd dddd	4800 (0x12C0)
13446(0x3486)	Average Luminance in AF Windows W12 and W11	???? ???? ???? ????	0 (0x0000)
13448(0x3488)	Average Luminance in AF Windows W14 and W13	???? ???? ???? ????	0 (0x0000)
13450(0x348A)	Average Luminance in AF Windows W22 and W21	???? ???? ???? ????	0 (0x0000)
13452(0x348C)	Average Luminance in AF Windows W24 and W23	???? ???? ???? ????	0 (0x0000)
13454(0x348E)	Average Luminance in AF Windows W32 and W31	???? ???? ???? ????	0 (0x0000)
13456(0x3490)	Average Luminance in AF Windows W34 and W33	???? ???? ???? ????	0 (0x0000)
13458(0x3492)	Average Luminance in AF Windows W42 and W41	???? ???? ???? ????	0 (0x0000)
13460(0x3494)	Average Luminance in AF Windows W44 and W43	???? ???? ???? ????	0 (0x0000)
13462(0x3496)	AF Filter 1	dddd dddd dddd dddd	40 (0x0028)
13464(0x3498)	AF Filter 1 Configuration	dddd dddd dddd dddd	49328 (0xC0B0)
13466(0x349A)	AF Filter 1 Average Sharpness Measure for AF Windows W12 and W11	???? ???? ???? ????	0 (0x0000)
13468(0x349C)	AF Filter 1 Average Sharpness Measure for AF Windows W14 and W13	???? ???? ???? ????	0 (0x0000)
13470(0x349E)	AF Filter 1 Average Sharpness Measure for AF Windows W22 and W21	???? ???? ???? ????	0 (0x0000)
13472(0x34A0)	AF Filter 1 Average Sharpness Measure for AF Windows W24 and W23	???? ???? ???? ????	0 (0x0000)
13474(0x34A2)	AF Filter 1 Average Sharpness Measure for AF Windows W32 and W31	???? ???? ???? ????	0 (0x0000)
13476(0x34A4)	AF Filter 1 Average Sharpness Measure for AF Windows W34 and W33	???? ???? ???? ????	0 (0x0000)
13478(0x34A6)	AF Filter 1 Average Sharpness Measure for AF Windows W42 and W41	???? ???? ???? ????	0 (0x0000)

Table 14: 2: SOC2 (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)
13480(0x34A8)	AF Filter 1 Average Sharpness Measure for AF Windows W44 and W43	???? ???? ???? ????	0 (0x0000)
13482(0x34AA)	AF Filter 2	dddd dddd dddd dddd	8320 (0x2080)
13484(0x34AC)	AF Filter 2 Configuration	dddd dddd dddd dddd	49456 (0xC130)
13486(0x34AE)	AF Filter 2 Average Sharpness Measure for AF Window W12 and W11	???? ???? ???? ????	0 (0x0000)
13488(0x34B0)	AF Filter 2 Average Sharpness Measure for AF Windows W14 and W13	???? ???? ???? ????	0 (0x0000)
13490(0x34B2)	AF Filter 2 Average Sharpness Measure for AF Windows W22 and W21	???? ???? ???? ????	0 (0x0000)
13492(0x34B4)	AF Filter 2 Average Sharpness Measure for AF Windows W24 and W23	???? ???? ???? ????	0 (0x0000)
13494(0x34B6)	AF Filter 2 Average Sharpness Measure for AF Windows W32 and W31	???? ???? ???? ????	0 (0x0000)
13496(0x34B8)	AF Filter 2 Average Sharpness Measure for AF Windows W34 and W33	???? ???? ???? ????	0 (0x0000)
13498(0x34BA)	AF Filter 2 Average Sharpness Measure for AF Windows W42 and W41	???? ???? ???? ????	0 (0x0000)
13500(0x34BC)	AF Filter 2 Average Sharpness Measure for AE Windows W44 and W43	???? ???? ???? ????	0 (0x0000)
13502(0x34BE)	Reserved	–	0 (0x0000)
13518(0x34CE)	Lens Correction Control	0000 0ddd dddd ddd0	352 (0x0160)
13520(0x34D0)	Zone Boundaries X1 and X2	dddd dddd dddd dddd	25650 (0x6432)
13522(0x34D2)	Zone Boundaries X0 and X3	dddd dddd dddd dddd	12950 (0x3296)
13524(0x34D4)	Zone Boundaries X4 and X5	dddd dddd dddd dddd	38500 (0x9664)
13526(0x34D6)	Zone Boundaries Y1 and Y2	dddd dddd dddd dddd	20520 (0x5028)
13528(0x34D8)	Zone Boundaries Y0 and Y3	dddd dddd dddd dddd	10360 (0x2878)
13530(0x34DA)	Zone Boundaries Y4 and Y5	dddd dddd dddd dddd	30800 (0x7850)
13532(0x34DC)	Center Offset	dddd dddd dddd dddd	0 (0x0000)
13534(0x34DE)	F[x] for Red Color at the First Pixel of the Array	0000 dddd dddd dddd	350 (0x015E)
13536(0x34E0)	F[x] for Green Color at the First Pixel of the Array	0000 dddd dddd dddd	323 (0x0143)
13538(0x34E2)	F[x] for Green2 Color at the First Pixel of the Array	0000 dddd dddd dddd	323 (0x0143)
13540(0x34E4)	F[x] for Blue Color at the First Pixel of the Array	0000 dddd dddd dddd	295 (0x0127)
13542(0x34E6)	F[y] for Red Color at the First Pixel of the Array	0000 dddd dddd dddd	300 (0x012C)

Table 14: 2: SOC2 (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)
13544(0x34E8)	F[y] for Green Color at the First Pixel of the Array	0000 dddd dddd dddd	259 (0x0103)
13546(0x34EA)	F[y] for Green2 Color at the First Pixel of the Array	0000 dddd dddd dddd	259 (0x0103)
13548(0x34EC)	F[y] for Blue Color at the First Pixel of the Array	0000 dddd dddd dddd	253 (0x00FD)
13550(0x34EE)	dF/dx for Red Color at the First Pixel of the Array	0000 dddd dddd dddd	3311 (0x0CEF)
13552(0x34F0)	dF/dx for Green Color at the First Pixel of the Array	0000 dddd dddd dddd	3384 (0x0D38)
13554(0x34F2)	dF/dx for Green2 Color at the First Pixel of the Array	0000 dddd dddd dddd	3384 (0x0D38)
13556(0x34F4)	dF/dx for Blue Color at the First Pixel of the Array	0000 dddd dddd dddd	3474 (0x0D92)
13558(0x34F6)	dF/dy for Red Color at the First Pixel of the Array	0000 dddd dddd dddd	3096 (0x0C18)
13560(0x34F8)	dF/dy for Green Color at the First Pixel of the Array	0000 dddd dddd dddd	3313 (0x0CF1)
13562(0x34FA)	dF/dy for Green2 Color at the First Pixel of the Array	0000 dddd dddd dddd	3313 (0x0CF1)
13564(0x34FC)	dF/dy for Blue Color at the First Pixel of the Array	0000 dddd dddd dddd	3333 (0x0D05)
13566(0x34FE)	Reserved	–	0 (0x0000)
13568(0x3500)	Second deriv. for Zone 0 Red Color	dddd dddd dddd dddd	2819 (0x0B03)
13570(0x3502)	Second deriv. for Zone 0 Green Color	dddd dddd dddd dddd	0 (0x0000)
13572(0x3504)	Second deriv. for Zone 0 Green2 Color	dddd dddd dddd dddd	0 (0x0000)
13574(0x3506)	Second deriv. for Zone 0 Blue Color	dddd dddd dddd dddd	256 (0x0100)
13576(0x3508)	Second deriv. for Zone 1 Red Color	dddd dddd dddd dddd	9524 (0x2534)
13578(0x350A)	Second deriv. for Zone 1 Green Color	dddd dddd dddd dddd	7219 (0x1C33)
13580(0x350C)	Second deriv. for Zone 1 Green2 Color	dddd dddd dddd dddd	7219 (0x1C33)
13582(0x350E)	Second deriv. for Zone 1 Blue Color	dddd dddd dddd dddd	6702 (0x1A2E)
13584(0x3510)	Second deriv. for Zone 2 Red color	dddd dddd dddd dddd	10810 (0x2A3A)
13586(0x3512)	Second deriv. for Zone 2 Green Color	dddd dddd dddd dddd	9517 (0x252D)
13588(0x3514)	Second deriv. for Zone 2 Green2 Color	dddd dddd dddd dddd	9517 (0x252D)
13590(0x3516)	Second deriv. for Zone 2 Blue Color	dddd dddd dddd dddd	10275 (0x2823)
13592(0x3518)	Second deriv. for Zone 3 Red Color	dddd dddd dddd dddd	3860 (0x0F14)
13594(0x351A)	Second deriv. for Zone 3 Green Color	dddd dddd dddd dddd	3360 (0x0D20)
13596(0x351C)	Second deriv. for Zone 3 Green2 Color	dddd dddd dddd dddd	3360 (0x0D20)
13598(0x351E)	Second deriv. for Zone 3 Blue Color	dddd dddd dddd dddd	1057 (0x0421)
13600(0x3520)	Second deriv. for Zone 4 Red Color	dddd dddd dddd dddd	3370 (0x0D2A)

Table 14: 2: SOC2 (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)
13602(0x3522)	Second deriv. for Zone 4 Green Color	dddd dddd dddd dddd	4119 (0x1017)
13604(0x3524)	Second deriv. for Zone 4 Green2 Color	dddd dddd dddd dddd	4119 (0x1017)
13606(0x3526)	Second deriv. for Zone 4 Blue Color	dddd dddd dddd dddd	5655 (0x1617)
13608(0x3528)	Second deriv. for Zone 5 Red Color	dddd dddd dddd dddd	5698 (0x1642)
13610(0x352A)	Second deriv. for Zone 5 Green Color	dddd dddd dddd dddd	5192 (0x1448)
13612(0x352C)	Second deriv. for Zone 5 Green2 Color	dddd dddd dddd dddd	5192 (0x1448)
13614(0x352E)	Second deriv. for Zone 5 Blue Color	dddd dddd dddd dddd	3918 (0x0F4E)
13616(0x3530)	Second deriv. for Zone 6 Red Color	dddd dddd dddd dddd	7975 (0x1F27)
13618(0x3532)	Second deriv. for Zone 6 Green Color	dddd dddd dddd dddd	6674 (0x1A12)
13620(0x3534)	Second deriv. for Zone 6 Green2 Color	dddd dddd dddd dddd	6674 (0x1A12)
13622(0x3536)	Second deriv. for Zone 6 Blue Color	dddd dddd dddd dddd	7702 (0x1E16)
13624(0x3538)	Second deriv. for Zone 7 Red Color	dddd dddd dddd dddd	5831 (0x16C7)
13626(0x353A)	Second deriv. for Zone 7 Green Color	dddd dddd dddd dddd	12741 (0x31C5)
13628(0x353C)	Second deriv. for Zone 7 Green2 Color	dddd dddd dddd dddd	12741 (0x31C5)
13630(0x353E)	Second deriv. for Zone 7 Blue Color	dddd dddd dddd dddd	10934 (0x2AB6)
13632(0x3540)	X2 Factors	dddd dddd dddd dddd	0 (0x0000)
13634(0x3542)	Global Offset of F[x y] Function	0000 0000 dddd dddd	2 (0x0002)
13636(0x3544)	K Factor in K F[x] F[y] red - Top right	0000 0ddd dddd dddd	398 (0x018E)
13638(0x3546)	K Factor in K F[x] F[y] green1 - Top right	0000 0ddd dddd dddd	398 (0x018E)
13640(0x3548)	K Factor in K F[x] F[y] green2 - Top right	0000 0ddd dddd dddd	398 (0x018E)
13642(0x354A)	K Factor in K F[x] F[y] blue - Top right	0000 0ddd dddd dddd	398 (0x018E)
13644(0x354C)	K Factor in K F[x] F[y] red - Top left	0000 0ddd dddd dddd	398 (0x018E)
13646(0x354E)	K Factor in K F[x] F[y] green1 - Top left	0000 0ddd dddd dddd	398 (0x018E)
13648(0x3550)	K Factor in K F[x] F[y] green2 - Top left	0000 0ddd dddd dddd	398 (0x018E)
13650(0x3552)	K Factor in K F[x] F[y] blue - Top left	0000 0ddd dddd dddd	398 (0x018E)
13652(0x3554)	K Factor in K F[x] F[y] red - Bottom right	0000 0ddd dddd dddd	398 (0x018E)
13654(0x3556)	K Factor in K F[x] F[y] green1 - Bottom right	0000 0ddd dddd dddd	398 (0x018E)
13656(0x3558)	K Factor in K F[x] F[y] green2 - Bottom right	0000 0ddd dddd dddd	398 (0x018E)
13658(0x355A)	K Factor in K F[x] F[y] blue - Bottom right	0000 0ddd dddd dddd	398 (0x018E)

Table 14: 2: SOC2 (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)
13660(0x355C)	K Factor in K F[x] F[y] red - Bottom left	0000 0ddd dddd dddd	398 (0x018E)
13662(0x355E)	K Factor in K F[x] F[y] green1 - Bottom left	0000 0ddd dddd dddd	398 (0x018E)
13664(0x3560)	K Factor in K F[x] F[y] green2 - Bottom left	0000 0ddd dddd dddd	398 (0x018E)
13666(0x3562)	K Factor in K F[x] F[y] blue - Bottom left	0000 0ddd dddd dddd	398 (0x018E)
13696(0x3580)	Boundaries of First AE Measurement Window [Top/Left]	dddd dddd dddd dddd	0 (0x0000)
13698(0x3582)	Boundaries of First AE Measurement Window [Height/ Width]	dddd dddd dddd dddd	15440 (0x3C50)
13700(0x3584)	AE Measurement Window Size[LSW]	dddd dddd dddd dddd	19200 (0x4B00)
13702(0x3586)	AE/AF Measurement Enable	0000 0000 0000 0ddd	6 (0x0006)
13704(0x3588)	Measure of the Average Luminance in AE windows W12 and W11	???? ???? ???? ????	0 (0x0000)
13706(0x358A)	Measure of the Average Luminance in AE windows W14 and W13	???? ???? ???? ????	0 (0x0000)
13708(0x358C)	Measure of the Average Luminance in AE windows W22 and W21	???? ???? ???? ????	0 (0x0000)
13710(0x358E)	Measure of the Average Luminance in AE windows W24 and W23	???? ???? ???? ????	0 (0x0000)
13712(0x3590)	Measure of the Average Luminance in AE windows W32 and W31	???? ???? ???? ????	0 (0x0000)
13714(0x3592)	Measure of the Average Luminance in AE windows W34 and W33	???? ???? ???? ????	0 (0x0000)
13716(0x3594)	Measure of the Average Luminance in AE windows W42 and W41	???? ???? ???? ????	0 (0x0000)
13718(0x3596)	Measure of the Average Luminance in AE windows W44 and W43	???? ???? ???? ????	0 (0x0000)
13730(0x35A2)	Dark Color Kill Controls	0000 0000 dddd dddd	20 (0x0014)
13732(0x35A4)	Bright Color Kill Controls	0000 0ddd dddd dddd	1428 (0x0594)
13734(0x35A6)	Histogram Window Lower Boundaries	dddd dddd dddd dddd	0 (0x0000)
13736(0x35A8)	Histogram Window Upper Boundaries	dddd dddd dddd dddd	38343 (0x95C7)
13738(0x35AA)	First Set of Bin Definitions	000d dddd dddd dddd	778 (0x030A)
13740(0x35AC)	Second Set of Bin Definitions	000d dddd dddd dddd	778 (0x030A)
13742(0x35AE)	Prescaler for the Historgram	0ddd dddd dddd dddd	10 (0x000A)
13744(0x35B0)	Pixel Counts for Bin 0 and Bin 1	???? ???? ???? ????	0 (0x0000)
13746(0x35B2)	Pixel Counts for Bin 2 and Bin 3	???? ???? ???? ????	0 (0x0000)
13748(0x35B4)	Pixel Counts for Bin 4 and Bin 5	???? ???? ???? ????	0 (0x0000)
13750(0x35B6)	Pixel Counts for Bin 6 and Bin 7	???? ???? ???? ????	0 (0x0000)
13752(0x35B8)	Definition of outlier counters	000d dddd dddd dddd	0 (0x0000)
13754(0x35BA)	Range for U and L counters	dddd dddd dddd dddd	64005 (0xFA05)

Table 14: 2: SOC2 (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)
13756(0x35BC)	Value for U_counter	???? ???? ???? ????	0 (0x0000)
13758(0x35BE)	Value for L_counter	???? ???? ???? ????	0 (0x0000)

Table 15: 0: Monitor Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	Reserved	–	59597 (0xE8CD)
2(0x002)	cmd	0000 0000 dddd dddd	0 (0x0000)
3(0x003)	arg1	dddd dddd dddd dddd	0 (0x0000)
5(0x005)	arg2	dddd dddd dddd dddd	0 (0x0000)
7(0x007)	msgCount	0000 0000 dddd dddd	0 (0x0000)
8(0x008)	msg [HI]	dddd dddd dddd dddd	0 (0x0000)
10(0x00A)	msg [LO]	dddd dddd dddd dddd	0 (0x0000)
12(0x00C)	ver	0000 0000 dddd dddd	26 (0x001A)
13(0x00D)	modul ID	0000 0000 dddd dddd	0 (0x0000)

Table 16: 1: Sequencer Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59073 (0xE6C1)
2(0x002)	mode	0000 0000 dddd dddd	15 (0x000F)
3(0x003)	cmd	0000 0000 dddd dddd	0 (0x0000)
4(0x004)	state	0000 0000 dddd dddd	0 (0x0000)
5(0x005)	stepMode	0000 0000 dddd dddd	0 (0x0000)
6(0x006)	sharedParams.flashType	0000 0000 dddd dddd	0 (0x0000)
7(0x007)	sharedParams.aeContBuff	0000 0000 dddd dddd	8 (0x0008)
8(0x008)	sharedParams.aeContStep	0000 0000 dddd dddd	2 (0x0002)
9(0x009)	sharedParams.aeFastBuff	0000 0000 dddd dddd	32 (0x0020)
10(0x00A)	sharedParams.aeFastStep	0000 0000 dddd dddd	1 (0x0001)
11(0x00B)	sharedParams.awbContBuff	0000 0000 dddd dddd	8 (0x0008)
12(0x00C)	sharedParams.awbContStep	0000 0000 dddd dddd	2 (0x0002)
13(0x00D)	sharedParams.awbFastBuff	0000 0000 dddd dddd	32 (0x0020)
14(0x00E)	sharedParams.awbFastStep	0000 0000 dddd dddd	1 (0x0001)
15(0x00F)	Reserved	–	0 (0x0000)
16(0x010)	Reserved	–	0 (0x0000)
17(0x011)	sharedParams.options	0000 0000 dddd dddd	168 (0x00A8)
18(0x012)	sharedParams.totMaxFrames	0000 0000 dddd dddd	16 (0x0010)
19(0x013)	sharedParams.flashTH	0000 0000 dddd dddd	0 (0x0000)
20(0x014)	sharedParams_outdoorTH	0000 0000 dddd dddd	10 (0x000A)
21(0x015)	sharedParams.LLmode	0000 0000 dddd dddd	96 (0x0060)
22(0x016)	sharedParams.LLvirtGain1	0000 0000 dddd dddd	81 (0x0051)
23(0x017)	sharedParams.LLvirtGain2	0000 0000 dddd dddd	90 (0x005A)

Table 16: 1: Sequencer Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
24(0x018)	sharedParams.LLSat1	0000 0000 dddd dddd	128 (0x0080)
25(0x019)	sharedParams.LLSat2	0000 0000 dddd dddd	0 (0x0000)
26(0x01A)	sharedParams.LLInterpThresh1	0000 0000 dddd dddd	16 (0x0010)
27(0x01B)	sharedParams.LLInterpThresh2	0000 0000 dddd dddd	64 (0x0040)
28(0x01C)	sharedParams.LLApCorr1	0000 0000 dddd dddd	2 (0x0002)
29(0x01D)	sharedParams.LLApCorr2	0000 0000 dddd dddd	0 (0x0000)
30(0x01E)	sharedParams.LLApThresh1	0000 0000 dddd dddd	8 (0x0008)
31(0x01F)	sharedParams.LLApThresh2	0000 0000 dddd dddd	64 (0x0040)
32(0x020)	captureParams.mode	0000 0000 dddd dddd	0 (0x0000)
33(0x021)	captureParams.numFrames	0000 0000 dddd dddd	3 (0x0003)
34(0x022)	previewParEnter.ae	0000 0000 dddd dddd	0 (0x0000)
35(0x023)	previewParEnter.fd	0000 0000 dddd dddd	0 (0x0000)
36(0x024)	previewParEnter.awb	0000 0000 dddd dddd	0 (0x0000)
37(0x025)	previewParEnter.af	0000 0000 dddd dddd	0 (0x0000)
38(0x026)	previewParEnter.hg	0000 0000 dddd dddd	0 (0x0000)
39(0x027)	previewParEnter.flash	0000 0000 dddd dddd	0 (0x0000)
40(0x028)	previewParEnter.skipframe	0000 0000 dddd dddd	64 (0x0040)
41(0x029)	previewPar.ae	0000 0000 dddd dddd	3 (0x0003)
42(0x02A)	previewPar.fd	0000 0000 dddd dddd	2 (0x0002)
43(0x02B)	previewPar.awb	0000 0000 dddd dddd	3 (0x0003)
44(0x02C)	previewPar.af	0000 0000 dddd dddd	0 (0x0000)
45(0x02D)	previewPar.hg	0000 0000 dddd dddd	3 (0x0003)
46(0x02E)	previewPar.flash	0000 0000 dddd dddd	0 (0x0000)
47(0x02F)	previewPar.skipframe	0000 0000 dddd dddd	0 (0x0000)
48(0x030)	previewParLeave.ae	0000 0000 dddd dddd	1 (0x0001)
49(0x031)	previewParLeave.fd	0000 0000 dddd dddd	0 (0x0000)
50(0x032)	previewParLeave.awb	0000 0000 dddd dddd	1 (0x0001)
51(0x033)	previewParLeave.af	0000 0000 dddd dddd	0 (0x0000)
52(0x034)	previewParLeave.hg	0000 0000 dddd dddd	1 (0x0001)
53(0x035)	previewParLeave.flash	0000 0000 dddd dddd	0 (0x0000)
54(0x036)	previewParLeave.skipframe	0000 0000 dddd dddd	16 (0x0010)
55(0x037)	capParEnter.ae	0000 0000 dddd dddd	0 (0x0000)
56(0x038)	capParEnter.fd	0000 0000 dddd dddd	0 (0x0000)
57(0x039)	capParEnter.awb	0000 0000 dddd dddd	0 (0x0000)
58(0x03A)	capParEnter.af	0000 0000 dddd dddd	0 (0x0000)
59(0x03B)	capParEnter.hg	0000 0000 dddd dddd	0 (0x0000)
60(0x03C)	capParEnter.flash	0000 0000 dddd dddd	0 (0x0000)
61(0x03D)	capParEnter.skipframe	0000 0000 dddd dddd	64 (0x0040)
62(0x03E)	NR_minTH	0000 0000 dddd dddd	16 (0x0010)
63(0x03F)	NR_maxTH	0000 0000 dddd dddd	32 (0x0020)
64(0x040)	NR_GainTH	0000 0000 dddd dddd	0 (0x0000)
65(0x041)	NR_Slope	0000 0000 dddd dddd	16 (0x0010)
66(0x042)	standByStatus	0000 0000 dddd dddd	0 (0x0000)
67(0x043)	standByMode	dddd dddd dddd dddd	0 (0x0000)
69(0x045)	Reserved	–	68 (0x0044)

Table 16: 1: Sequencer Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
70(0x046)	Reserved	–	6 (0x0006)
71(0x047)	Reserved	–	38 (0x0026)
72(0x048)	Reserved	–	4 (0x0004)
73(0x049)	Reserved	–	0 (0x0000)
74(0x04A)	Reserved	–	7 (0x0007)
75(0x04B)	padClkFreq	dddd dddd dddd dddd	824 (0x0338)

Table 17: 2: AE Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	Vmt	dddd dddd dddd dddd	59133 (0xE6FD)
2(0x002)	windowPos	0000 0000 dddd dddd	0 (0x0000)
3(0x003)	windowSize	0000 0000 dddd dddd	239 (0x00EF)
4(0x004)	wakeUpLine	dddd dddd dddd dddd	570 (0x023A)
6(0x006)	Target	0000 0000 dddd dddd	60 (0x003C)
7(0x007)	Gate.	0000 0000 dddd dddd	10 (0x000A)
8(0x008)	SkipFrames	0000 0000 dddd dddd	0 (0x0000)
9(0x009)	JumpDivisor	0000 0000 dddd dddd	2 (0x0002)
10(0x00A)	lumaBufferSpeed	0000 0000 dddd dddd	8 (0x0008)
11(0x00B)	minIndex	0000 0000 dddd dddd	0 (0x0000)
12(0x00C)	maxIndex	0000 0000 dddd dddd	24 (0x0018)
13(0x00D)	minVirtGain	0000 0000 dddd dddd	16 (0x0010)
14(0x00E)	maxVirtGain	0000 0000 dddd dddd	128 (0x0080)
15(0x00F)	maxADChi	0000 0000 dddd dddd	11 (0x000B)
16(0x010)	minADChi	0000 0000 dddd dddd	10 (0x000A)
17(0x011)	minADClo	0000 0000 dddd dddd	9 (0x0009)
18(0x012)	maxDGainAE1	dddd dddd dddd dddd	128 (0x0080)
20(0x014)	maxDGainAE2	0000 0000 dddd dddd	32 (0x0020)
21(0x015)	IndexTH23	0000 0000 dddd dddd	8 (0x0008)
22(0x016)	maxGain23	0000 0000 dddd dddd	120 (0x0078)
23(0x017)	weights	0000 0000 dddd dddd	255 (0x00FF)
24(0x018)	status	0000 0000 dddd dddd	132 (0x0084)
25(0x019)	CurrentY	0000 0000 dddd dddd	0 (0x0000)
26(0x01A)	R12	dddd dddd dddd dddd	1032 (0x0408)
28(0x01C)	Index	0000 0000 dddd dddd	4 (0x0004)
29(0x01D)	VirtGain	0000 0000 dddd dddd	16 (0x0010)
30(0x01E)	ADC_hi	0000 0000 dddd dddd	11 (0x000B)
31(0x01F)	ADC_lo	0000 0000 dddd dddd	9 (0x0009)
32(0x020)	DGainAE1	dddd dddd dddd dddd	128 (0x0080)
34(0x022)	DGainAE2	0000 0000 dddd dddd	32 (0x0020)
35(0x023)	R9	dddd dddd dddd dddd	0 (0x0000)
37(0x025)	R65	dddd dddd dddd dddd	2315 (0x090B)
39(0x027)	Reserved	–	0 (0x0000)
41(0x029)	gainR12	0000 0000 dddd dddd	128 (0x0080)

Table 17: 2: AE Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
42(0x02A)	SkipFrames_cnt	0000 0000 dddd dddd	0 (0x0000)
43(0x02B)	BufferedLuma	dddd dddd dddd dddd	0 (0x0000)
45(0x02D)	dirAE_prev	0000 0000 dddd dddd	0 (0x0000)
46(0x02E)	R9_step	dddd dddd dddd dddd	157 (0x009D)
48(0x030)	Reserved	–	2 (0x0002)
49(0x031)	maxADClo	0000 0000 dddd dddd	10 (0x000A)
50(0x032)	physGainR	dddd dddd dddd dddd	16 (0x0010)
52(0x034)	physGainG	dddd dddd dddd dddd	16 (0x0010)
54(0x036)	physGainB	dddd dddd dddd dddd	16 (0x0010)
56(0x038)	Reserved	–	128 (0x0080)
58(0x03A)	Reserved	–	960 (0x03C0)
60(0x03C)	Reserved	–	3072 (0x0C00)
62(0x03E)	Reserved	–	8 (0x0008)
63(0x03F)	Reserved	–	10 (0x000A)
64(0x040)	Reserved	–	0 (0x0000)
65(0x041)	mmMeanEV	0000 0000 dddd dddd	0 (0x0000)
66(0x042)	mmShiftEV	0000 0000 dddd dddd	13 (0x000D)
67(0x043)	numOE	0000 0000 dddd dddd	0 (0x0000)
68(0x044)	TargetD	dddd dddd dddd dddd	61 (0x003D)
69(0x045)	maxNu	dddd dddd dddd dddd	100 (0x0064)
71(0x047)	Reserved	–	0 (0x0000)
73(0x049)	minTargetD	0000 0000 dddd dddd	40 (0x0028)
74(0x04A)	maxTargetD	0000 0000 dddd dddd	180 (0x00B4)
75(0x04B)	Reserved	–	40 (0x0028)
76(0x04C)	Reserved	–	180 (0x00B4)
77(0x04D)	Reserved	–	0 (0x0000)
78(0x04E)	Reserved	–	3 (0x0003)
79(0x04F)	coefG2	0000 0000 dddd dddd	128 (0x0080)
80(0x050)	EV_HiLo	0000 0000 dddd dddd	166 (0x00A6)
81(0x051)	targetDelta.	0000 0000 dddd dddd	168 (0x00A8)
82(0x052)	minTargetDelta	0000 0000 dddd dddd	134 (0x0086)
83(0x053)	maxNuDelta	0000 0000 dddd dddd	25 (0x0019)
84(0x054)	outliersSlope	0000 000d dddd dddd	16 (0x0010)

Table 18: 3: AWB Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59299 (0xE7A3)
2(0x002)	windowPos	0000 0000 dddd dddd	0 (0x0000)
3(0x003)	windowSize	0000 0000 dddd dddd	239 (0x00EF)
4(0x004)	wakeUpLine	dddd dddd dddd dddd	572 (0x023C)
6(0x006)	ccmL[0]	dddd dddd dddd dddd	645 (0x0285)
8(0x008)	ccmL[1]	dddd dddd dddd dddd	65176 (0xFE98)
10(0x00A)	ccmL[2]	dddd dddd dddd dddd	16 (0x0010)
12(0x00C)	ccmL[3]	dddd dddd dddd dddd	65365 (0xFF55)

Table 18: 3: AWB Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
14(0x00E)	ccmL[4]	dddd dddd dddd dddd	553 (0x0229)
16(0x010)	ccmL[5]	dddd dddd dddd dddd	65487 (0xFFCF)
18(0x012)	ccmL[6]	dddd dddd dddd dddd	65381 (0xFF65)
20(0x014)	ccmL[7]	dddd dddd dddd dddd	65017 (0xFDF9)
22(0x016)	ccmL[8]	dddd dddd dddd dddd	1035 (0x040B)
24(0x018)	ccmL[9]	dddd dddd dddd dddd	32 (0x0020)
26(0x01A)	ccmL[10]	dddd dddd dddd dddd	56 (0x0038)
28(0x01C)	ccmRL[0]	dddd dddd dddd dddd	65456 (0xFFB0)
30(0x01E)	ccmRL[1]	dddd dddd dddd dddd	101 (0x0065)
32(0x020)	ccmRL[2]	dddd dddd dddd dddd	65519 (0xFFEF)
34(0x022)	ccmRL[3]	dddd dddd dddd dddd	31 (0x001F)
36(0x024)	ccmRL[4]	dddd dddd dddd dddd	20 (0x0014)
38(0x026)	ccmRL[5]	dddd dddd dddd dddd	65494 (0xFFD6)
40(0x028)	ccmRL[6]	dddd dddd dddd dddd	95 (0x005F)
42(0x02A)	ccmRL[7]	dddd dddd dddd dddd	258 (0x0102)
44(0x02C)	ccmRL[8]	dddd dddd dddd dddd	65124 (0xFE64)
46(0x02E)	ccmRL[9]	dddd dddd dddd dddd	16 (0x0010)
48(0x030)	ccmRL[10]	dddd dddd dddd dddd	65517 (0xFFED)
50(0x032)	ccm[0]	dddd dddd dddd dddd	513 (0x0201)
52(0x034)	ccm[1]	dddd dddd dddd dddd	65274 (0xFEFA)
54(0x036)	ccm[2]	dddd dddd dddd dddd	6 (0x0006)
56(0x038)	ccm[3]	dddd dddd dddd dddd	65418 (0xFF8A)
58(0x03A)	ccm[4]	dddd dddd dddd dddd	429 (0x01AD)
60(0x03C)	ccm[5]	dddd dddd dddd dddd	65483 (0xFFCB)
62(0x03E)	ccm[6]	dddd dddd dddd dddd	65453 (0xFFAD)
64(0x040)	ccm[7]	dddd dddd dddd dddd	65234 (0xFED2)
66(0x042)	ccm[8]	dddd dddd dddd dddd	642 (0x0282)
68(0x044)	ccm[9]	dddd dddd dddd dddd	40 (0x0028)
70(0x046)	ccm[10]	dddd dddd dddd dddd	47 (0x002F)
72(0x048)	GainBufferSpeed	0000 0000 dddd dddd	8 (0x0008)
73(0x049)	JumpDivisor	0000 0000 dddd dddd	2 (0x0002)
74(0x04A)	GainMin	0000 0000 dddd dddd	102 (0x0066)
75(0x04B)	GainMax	0000 0000 dddd dddd	153 (0x0099)
76(0x04C)	GainR	0000 0000 dddd dddd	128 (0x0080)
77(0x04D)	GainG	0000 0000 dddd dddd	128 (0x0080)
78(0x04E)	GainB	0000 0000 dddd dddd	128 (0x0080)
79(0x04F)	CCMpositionMin	0000 0000 dddd dddd	0 (0x0000)
80(0x050)	CCMpositionMax	0000 0000 dddd dddd	127 (0x007F)
81(0x051)	CCMposition	0000 0000 dddd dddd	64 (0x0040)
82(0x052)	saturation	0000 0000 dddd dddd	128 (0x0080)
83(0x053)	mode	0000 0000 dddd dddd	0 (0x0000)
84(0x054)	GainR_buf	dddd dddd dddd dddd	0 (0x0000)
86(0x056)	GainB_buf	dddd dddd dddd dddd	0 (0x0000)
88(0x058)	sumR	0000 0000 dddd dddd	0 (0x0000)
89(0x059)	sumY	0000 0000 dddd dddd	0 (0x0000)

Table 18: 3: AWB Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
90(0x05A)	sumB	0000 0000 dddd dddd	0 (0x0000)
91(0x05B)	steadyBGainOutMin	0000 0000 dddd dddd	120 (0x0078)
92(0x05C)	steadyBGainOutMax	0000 0000 dddd dddd	134 (0x0086)
93(0x05D)	steadyBGainInMin	0000 0000 dddd dddd	126 (0x007E)
94(0x05E)	steadyBGainInMax	0000 0000 dddd dddd	130 (0x0082)
95(0x05F)	cntPxlTH	dddd dddd dddd dddd	100 (0x0064)
97(0x061)	TG_min0	0000 0000 dddd dddd	231 (0x00E7)
98(0x062)	TG_max0	0000 0000 dddd dddd	246 (0x00F6)
99(0x063)	X0	0000 0000 dddd dddd	16 (0x0010)
100(0x064)	kR_L	0000 0000 dddd dddd	170 (0x00AA)
101(0x065)	kG_L	0000 0000 dddd dddd	150 (0x0096)
102(0x066)	kB_L	0000 0000 dddd dddd	128 (0x0080)
103(0x067)	kR_R	0000 0000 dddd dddd	128 (0x0080)
104(0x068)	kG_R	0000 0000 dddd dddd	128 (0x0080)
105(0x069)	kB_R	0000 0000 dddd dddd	128 (0x0080)
106(0x06A)	EdgeTH	0000 0000 dddd dddd	128 (0x0080)
107(0x06B)	EdgeTH_min	0000 0000 dddd dddd	128 (0x0080)
108(0x06C)	EdgeTH_max	0000 0000 dddd dddd	128 (0x0080)

Table 19: 4: Flicker Varialbes

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59428 (0xE824)
2(0x002)	windowPosH	0000 0000 dddd dddd	29 (0x001D)
3(0x003)	windowHeight	0000 0000 dddd dddd	4 (0x0004)
4(0x004)	mode	0000 0000 dd?d ????	0 (0x0000)
5(0x005)	wakeUpLine	dddd dddd dddd dddd	64 (0x0040)
7(0x007)	smooth_cnt	0000 0000 dddd dddd	5 (0x0005)
8(0x008)	search_f1_50	0000 0000 dddd dddd	30 (0x001E)
9(0x009)	search_f2_50	0000 0000 dddd dddd	32 (0x0020)
10(0x00A)	search_f1_60	0000 0000 dddd dddd	37 (0x0025)
11(0x00B)	search_f2_60	0000 0000 dddd dddd	39 (0x0027)
12(0x00C)	skipFrame	0000 0000 dddd dddd	0 (0x0000)
13(0x00D)	stat_min	0000 0000 dddd dddd	3 (0x0003)
14(0x00E)	stat_max	0000 0000 dddd dddd	5 (0x0005)
15(0x00F)	stat	0000 0000 dddd dddd	0 (0x0000)
16(0x010)	minAmplitude	0000 0000 dddd dddd	5 (0x0005)
17(0x011)	R9_step_0_f60	dddd dddd dddd dddd	157 (0x009D)
19(0x013)	R9_step_0_f50	dddd dddd dddd dddd	188 (0x00BC)
21(0x015)	R9_step_1_f60	dddd dddd dddd dddd	0 (0x0000)
23(0x017)	R9_step_1_f50	dddd dddd dddd dddd	224 (0x00E0)
25(0x019)	Buffer[0]	0000 0000 dddd dddd	0 (0x0000)
26(0x01A)	Buffer[1]	0000 0000 dddd dddd	0 (0x0000)
27(0x01B)	Buffer[2]	0000 0000 dddd dddd	0 (0x0000)
28(0x01C)	Buffer[3]	0000 0000 dddd dddd	0 (0x0000)

Table 19: 4: Flicker Varialbes (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
29(0x01D)	Buffer[4]	0000 0000 dddd dddd	0 (0x0000)
30(0x01E)	Buffer[5]	0000 0000 dddd dddd	0 (0x0000)
31(0x01F)	Buffer[6]	0000 0000 dddd dddd	0 (0x0000)
32(0x020)	Buffer[7]	0000 0000 dddd dddd	0 (0x0000)
33(0x021)	Buffer[8]	0000 0000 dddd dddd	0 (0x0000)
34(0x022)	Buffer[9]	0000 0000 dddd dddd	0 (0x0000)
35(0x023)	Buffer[10]	0000 0000 dddd dddd	0 (0x0000)
36(0x024)	Buffer[11]	0000 0000 dddd dddd	0 (0x0000)
37(0x025)	Buffer[12]	0000 0000 dddd dddd	0 (0x0000)
38(0x026)	Buffer[13]	0000 0000 dddd dddd	0 (0x0000)
39(0x027)	Buffer[14]	0000 0000 dddd dddd	0 (0x0000)
40(0x028)	Buffer[15]	0000 0000 dddd dddd	0 (0x0000)
41(0x029)	Buffer[16]	0000 0000 dddd dddd	0 (0x0000)
42(0x02A)	Buffer[17]	0000 0000 dddd dddd	0 (0x0000)
43(0x02B)	Buffer[18]	0000 0000 dddd dddd	0 (0x0000)
44(0x02C)	Buffer[19]	0000 0000 dddd dddd	0 (0x0000)
45(0x02D)	Buffer[20]	0000 0000 dddd dddd	0 (0x0000)
46(0x02E)	Buffer[21]	0000 0000 dddd dddd	0 (0x0000)
47(0x02F)	Buffer[22]	0000 0000 dddd dddd	0 (0x0000)
48(0x030)	Buffer[23]	0000 0000 dddd dddd	0 (0x0000)
49(0x031)	Buffer[24]	0000 0000 dddd dddd	0 (0x0000)
50(0x032)	Buffer[25]	0000 0000 dddd dddd	0 (0x0000)
51(0x033)	Buffer[26]	0000 0000 dddd dddd	0 (0x0000)
52(0x034)	Buffer[27]	0000 0000 dddd dddd	0 (0x0000)
53(0x035)	Buffer[28]	0000 0000 dddd dddd	0 (0x0000)
54(0x036)	Buffer[29]	0000 0000 dddd dddd	0 (0x0000)
55(0x037)	Buffer[30]	0000 0000 dddd dddd	0 (0x0000)
56(0x038)	Buffer[31]	0000 0000 dddd dddd	0 (0x0000)
57(0x039)	Buffer[32]	0000 0000 dddd dddd	0 (0x0000)
58(0x03A)	Buffer[33]	0000 0000 dddd dddd	0 (0x0000)
59(0x03B)	Buffer[34]	0000 0000 dddd dddd	0 (0x0000)
60(0x03C)	Buffer[35]	0000 0000 dddd dddd	0 (0x0000)
61(0x03D)	Buffer[36]	0000 0000 dddd dddd	0 (0x0000)
62(0x03E)	Buffer[37]	0000 0000 dddd dddd	0 (0x0000)
63(0x03F)	Buffer[38]	0000 0000 dddd dddd	0 (0x0000)
64(0x040)	Buffer[39]	0000 0000 dddd dddd	0 (0x0000)
65(0x041)	Buffer[40]	0000 0000 dddd dddd	0 (0x0000)
66(0x042)	Buffer[41]	0000 0000 dddd dddd	0 (0x0000)
67(0x043)	Buffer[42]	0000 0000 dddd dddd	0 (0x0000)
68(0x044)	Buffer[43]	0000 0000 dddd dddd	0 (0x0000)
69(0x045)	Buffer[44]	0000 0000 dddd dddd	0 (0x0000)
70(0x046)	Buffer[45]	0000 0000 dddd dddd	0 (0x0000)
71(0x047)	Buffer[46]	0000 0000 dddd dddd	0 (0x0000)
72(0x048)	Buffer[47]	0000 0000 dddd dddd	0 (0x0000)

Table 20: 5: Auto Focus Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59438 (0xE82E)
2(0x002)	windowPos	0000 0000 dddd dddd	68 (0x0044)
3(0x003)	windowSize	0000 0000 dddd dddd	119 (0x0077)
4(0x004)	mode	0000 0000 dddd dddd	0 (0x0000)
5(0x005)	modeEx	0000 0000 dddd dddd	128 (0x0080)
6(0x006)	numSteps	0000 0000 dddd dddd	10 (0x000A)
7(0x007)	initPos	0000 0000 dddd dddd	0 (0x0000)
8(0x008)	numSteps2	0000 0000 dddd dddd	6 (0x0006)
9(0x009)	stepSize	0000 0000 dddd dddd	6 (0x0006)
10(0x00A)	wakeUpLine	dddd dddd dddd dddd	448 (0x01C0)
12(0x00C)	zoneWeights [HI]	dddd dddd dddd dddd	65535 (0xFFFF)
14(0x00E)	zoneWeights [LO]	dddd dddd dddd dddd	65535 (0xFFFF)
16(0x010)	distanceWeight	0000 0000 dddd dddd	255 (0x00FF)
17(0x011)	bestPosition	0000 0000 dddd dddd	0 (0x0000)
18(0x012)	shaTH	0000 0000 dddd dddd	10 (0x000A)
19(0x013)	modeEx2	0000 0000 dddd dddd	0 (0x0000)
20(0x014)	Position 0	0000 0000 dddd dddd	0 (0x0000)
21(0x015)	Position 1	0000 0000 dddd dddd	28 (0x001C)
22(0x016)	Position 2	0000 0000 dddd dddd	56 (0x0038)
23(0x017)	Position 3	0000 0000 dddd dddd	85 (0x0055)
24(0x018)	Position 4	0000 0000 dddd dddd	113 (0x0071)
25(0x019)	Position 5	0000 0000 dddd dddd	141 (0x008D)
26(0x01A)	Position 6	0000 0000 dddd dddd	170 (0x00AA)
27(0x01B)	Position 7	0000 0000 dddd dddd	198 (0x00C6)
28(0x01C)	Position 8	0000 0000 dddd dddd	226 (0x00E2)
29(0x01D)	Position 9	0000 0000 dddd dddd	255 (0x00FF)
30(0x01E)	Position 10	0000 0000 dddd dddd	27 (0x001B)
31(0x01F)	Position 11	0000 0000 dddd dddd	55 (0x0037)
32(0x020)	Position 12	0000 0000 dddd dddd	84 (0x0054)
33(0x021)	Position 13	0000 0000 dddd dddd	112 (0x0070)
34(0x022)	Position 14	0000 0000 dddd dddd	140 (0x008C)
35(0x023)	Position 15	0000 0000 dddd dddd	169 (0x00A9)
36(0x024)	Position 16	0000 0000 dddd dddd	197 (0x00C5)
37(0x025)	Position 17	0000 0000 dddd dddd	225 (0x00E1)
38(0x026)	Position 18	0000 0000 dddd dddd	254 (0x00FE)
39(0x027)	Position 19	0000 0000 dddd dddd	26 (0x001A)

Table 21: 6: AFM Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec(Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59454 (0xE83E)
2(0x002)	type	0000 0000 dddd dddd	0 (0x0000)
3(0x003)	curPos	0000 0000 dddd dddd	0 (0x0000)
4(0x004)	prePos	0000 0000 dddd dddd	0 (0x0000)
5(0x005)	status	0000 0000 dddd dddd	0 (0x0000)

Table 21: 6: AFM Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec(Hex)
6(0x006)	posMin	0000 0000 dddd dddd	0 (0x0000)
7(0x007)	posMax	0000 0000 dddd dddd	0 (0x0000)
8(0x008)	posMacro	0000 0000 dddd dddd	0 (0x0000)
9(0x009)	backlash	0000 0000 dddd dddd	0 (0x0000)
10(0x00A)	custCtrl	0000 0000 dddd dddd	0 (0x0000)
11(0x00B)	timer_vmt	dddd dddd dddd dddd	59627 (0xE8EB)
13(0x00D)	timer_startTime	dddd dddd dddd dddd	0 (0x0000)
15(0x00F)	timer_stopTime	dddd dddd dddd dddd	0 (0x0000)
17(0x011)	timer_hiWordMclkFreq	dddd dddd dddd dddd	0 (0x0000)
19(0x013)	timer_maxShortDelay	dddd dddd dddd dddd	0 (0x0000)
21(0x015)	timer_maxLongDelay	dddd dddd dddd dddd	0 (0x0000)
23(0x017)	timer_maxQuickMove	0000 0000 dddd dddd	0 (0x0000)
24(0x018)	timer_config	0000 0000 dddd dddd	0 (0x0000)
25(0x019)	si_vmt	dddd dddd dddd dddd	59635 (0xE8F3)
27(0x01B)	si_clkMask	dddd dddd dddd dddd	0 (0x0000)
29(0x01D)	si_dataMask	dddd dddd dddd dddd	0 (0x0000)
31(0x01F)	si_clkQtrPrd	dddd dddd dddd dddd	0 (0x0000)
33(0x021)	si_needsAck	0000 0000 dddd dddd	0 (0x0000)
34(0x022)	si_slaveAddr	0000 0000 dddd dddd	0 (0x0000)
35(0x023)	sm_enabMask	dddd dddd dddd dddd	0 (0x0000)
37(0x025)	sm_drv0Mask	0000 0000 dddd dddd	0 (0x0000)
38(0x026)	sm_drv1Mask	0000 0000 dddd dddd	0 (0x0000)
39(0x027)	sm_drv2Mask	0000 0000 dddd dddd	0 (0x0000)
40(0x028)	sm_drv3Mask	0000 0000 dddd dddd	0 (0x0000)
41(0x029)	sm_drvsQtrPrd	dddd dddd dddd dddd	0 (0x0000)
43(0x02B)	sm_drvsGenMode	0000 0000 dddd dddd	0 (0x0000)
44(0x02C)	sm_piEnabMask	dddd dddd dddd dddd	0 (0x0000)
46(0x02E)	sm_piOutMask	dddd dddd dddd dddd	0 (0x0000)
48(0x030)	sm_piEdgeOffset	0000 0000 dddd dddd	0 (0x0000)
49(0x031)	sm_piConfig	0000 0000 dddd dddd	0 (0x0000)

Table 22: 7: Mode Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59502 (0xE86E)
2(0x002)	context	0000 0000 dddd dddd	0 (0x0000)
3(0x003)	Output Width A	dddd dddd dddd dddd	800 (0x0320)
5(0x005)	Output Height A	dddd dddd dddd dddd	600 (0x0258)
7(0x007)	Output Width B	dddd dddd dddd dddd	1600 (0x0640)
9(0x009)	Output Height B	dddd dddd dddd dddd	1200 (0x04B0)
11(0x00B)	PLL_Lock_Delay	dddd dddd dddd dddd	200 (0x00C8)
13(0x00D)	sensor_row_start_A	dddd dddd dddd dddd	0 (0x0000)
15(0x00F)	sensor_col_start_A	dddd dddd dddd dddd	0 (0x0000)
17(0x011)	sensor_row_end_A	dddd dddd dddd dddd	1213 (0x04BD)
19(0x013)	sensor_col_end_A	dddd dddd dddd dddd	1613 (0x064D)

Table 22: 7: Mode Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
21(0x015)	sensor_x_delay_A	dddd dddd dddd dddd	0 (0x0000)
23(0x017)	sensor_row_speed_A	dddd dddd dddd dddd	8466 (0x2112)
25(0x019)	Read Mode A	dddd dddd dddd dddd	1132 (0x046C)
27(0x01B)	sensor_sample_time_pck_A	dddd dddd dddd dddd	984 (0x03D8)
29(0x01D)	sensor_fine_correction_A	dddd dddd dddd dddd	341 (0x0155)
31(0x01F)	sensor_fine_IT_min_A	dddd dddd dddd dddd	633 (0x0279)
33(0x021)	sensor_fine_IT_max_margin_A	dddd dddd dddd dddd	341 (0x0155)
35(0x023)	sensor_frameLengthLines_A	dddd dddd dddd dddd	659 (0x0293)
37(0x025)	sensor_lineLengthPck:_A	dddd dddd dddd dddd	2732 (0x0AAC)
39(0x027)	sensor_dac_id_4_5_A	dddd dddd dddd dddd	4112 (0x1010)
41(0x029)	sensor_dac_id_6_7_A	dddd dddd dddd dddd	8208 (0x2010)
43(0x02B)	sensor_dac_id_8_9_A	dddd dddd dddd dddd	4112 (0x1010)
45(0x02D)	sensor_dac_id_10_11_A	dddd dddd dddd dddd	4102 (0x1006)
47(0x02F)	sensor_row_start_B	dddd dddd dddd dddd	4 (0x0004)
49(0x031)	sensor_col_start_B	dddd dddd dddd dddd	4 (0x0004)
51(0x033)	sensor_row_end_B	dddd dddd dddd dddd	1211 (0x04BB)
53(0x035)	sensor_col_end_B	dddd dddd dddd dddd	1611 (0x064B)
55(0x037)	sensor_x_delay_B	dddd dddd dddd dddd	0 (0x0000)
57(0x039)	sensor_row_speed_B	dddd dddd dddd dddd	8465 (0x2111)
59(0x03B)	Read Mode B	dddd dddd dddd dddd	36 (0x0024)
61(0x03D)	sensor_sample_time_pck_B	dddd dddd dddd dddd	492 (0x01EC)
63(0x03F)	sensor_fine_correction_B	dddd dddd dddd dddd	164 (0x00A4)
65(0x041)	sensor_fine_IT_min_B	dddd dddd dddd dddd	330 (0x014A)
67(0x043)	sensor_fine_IT_max_margin_B	dddd dddd dddd dddd	164 (0x00A4)
69(0x045)	sensor_frameLengthLines_B	dddd dddd dddd dddd	1261 (0x04ED)
71(0x047)	sensor_lineLengthPck:_B	dddd dddd dddd dddd	2732 (0x0AAC)
73(0x049)	sensor_dac_id_4_5_B	dddd dddd dddd dddd	8224 (0x2020)
75(0x04B)	sensor_dac_id_6_7_B	dddd dddd dddd dddd	8224 (0x2020)
77(0x04D)	sensor_dac_id_8_9_B	dddd dddd dddd dddd	4128 (0x1020)
79(0x04F)	sensor_dac_id_10_11_B	dddd dddd dddd dddd	8198 (0x2006)
81(0x051)	crop_X0_A	dddd dddd dddd dddd	0 (0x0000)
83(0x053)	crop_X1_A	dddd dddd dddd dddd	800 (0x0320)
85(0x055)	crop_Y0_A	dddd dddd dddd dddd	0 (0x0000)
87(0x057)	crop_Y1_A	dddd dddd dddd dddd	600 (0x0258)
89(0x059)	dec_ctrl_A	dddd dddd dddd dddd	0 (0x0000)
91(0x05B)	width_ratio_A	dddd dddd dddd dddd	2048 (0x0800)
93(0x05D)	height_ratio_A	dddd dddd dddd dddd	2048 (0x0800)
95(0x05F)	crop_X0_B	dddd dddd dddd dddd	0 (0x0000)
97(0x061)	crop_X1_B	dddd dddd dddd dddd	1600 (0x0640)
99(0x063)	crop_Y0_B	dddd dddd dddd dddd	0 (0x0000)
101(0x065)	crop_Y1_B	dddd dddd dddd dddd	1200 (0x04B0)
103(0x067)	dec_ctrl_B	dddd dddd dddd dddd	0 (0x0000)
105(0x069)	width_ratio_B	dddd dddd dddd dddd	2048 (0x0800)
107(0x06B)	height_ratio_B	dddd dddd dddd dddd	2048 (0x0800)
109(0x06D)	gam_cont_A	0000 0000 dddd dddd	66 (0x0042)

Table 22: 7: Mode Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
110(0x06E)	gam_cont_B	0000 0000 dddd dddd	66 (0x0042)
111(0x06F)	gam_table_A_0	0000 0000 dddd dddd	0 (0x0000)
112(0x070)	gam_table_A_1	0000 0000 dddd dddd	39 (0x0027)
113(0x071)	gam_table_A_2	0000 0000 dddd dddd	53 (0x0035)
114(0x072)	gam_table_A_3	0000 0000 dddd dddd	72 (0x0048)
115(0x073)	gam_table_A_4	0000 0000 dddd dddd	99 (0x0063)
116(0x074)	gam_table_A_5	0000 0000 dddd dddd	119 (0x0077)
117(0x075)	gam_table_A_6	0000 0000 dddd dddd	136 (0x0088)
118(0x076)	gam_table_A_7	0000 0000 dddd dddd	150 (0x0096)
119(0x077)	gam_table_A_8	0000 0000 dddd dddd	163 (0x00A3)
120(0x078)	gam_table_A_9	0000 0000 dddd dddd	175 (0x00AF)
121(0x079)	gam_table_A_10	0000 0000 dddd dddd	186 (0x00BA)
122(0x07A)	gam_table_A_11	0000 0000 dddd dddd	196 (0x00C4)
123(0x07B)	gam_table_A_12	0000 0000 dddd dddd	206 (0x00CE)
124(0x07C)	gam_table_A_13	0000 0000 dddd dddd	215 (0x00D7)
125(0x07D)	gam_table_A_14	0000 0000 dddd dddd	224 (0x00E0)
126(0x07E)	gam_table_A_15	0000 0000 dddd dddd	232 (0x00E8)
127(0x07F)	gam_table_A_16	0000 0000 dddd dddd	240 (0x00F0)
128(0x080)	gam_table_A_17	0000 0000 dddd dddd	248 (0x00F8)
129(0x081)	gam_table_A_18	0000 0000 dddd dddd	255 (0x00FF)
130(0x082)	gam_table_B_0	0000 0000 dddd dddd	0 (0x0000)
131(0x083)	gam_table_B_1	0000 0000 dddd dddd	39 (0x0027)
132(0x084)	gam_table_B_2	0000 0000 dddd dddd	53 (0x0035)
133(0x085)	gam_table_B_3	0000 0000 dddd dddd	72 (0x0048)
134(0x086)	gam_table_B_4	0000 0000 dddd dddd	99 (0x0063)
135(0x087)	gam_table_B_5	0000 0000 dddd dddd	119 (0x0077)
136(0x088)	gam_table_B_6	0000 0000 dddd dddd	136 (0x0088)
137(0x089)	gam_table_B_7	0000 0000 dddd dddd	150 (0x0096)
138(0x08A)	gam_table_B_8	0000 0000 dddd dddd	163 (0x00A3)
139(0x08B)	gam_table_B_9	0000 0000 dddd dddd	175 (0x00AF)
140(0x08C)	gam_table_B_10	0000 0000 dddd dddd	186 (0x00BA)
141(0x08D)	gam_table_B_11	0000 0000 dddd dddd	196 (0x00C4)
142(0x08E)	gam_table_B_12	0000 0000 dddd dddd	206 (0x00CE)
143(0x08F)	gam_table_B_13	0000 0000 dddd dddd	215 (0x00D7)
144(0x090)	gam_table_B_14	0000 0000 dddd dddd	224 (0x00E0)
145(0x091)	gam_table_B_15	0000 0000 dddd dddd	232 (0x00E8)
146(0x092)	gam_table_B_16	0000 0000 dddd dddd	240 (0x00F0)
147(0x093)	gam_table_B_17	0000 0000 dddd dddd	248 (0x00F8)
148(0x094)	gam_table_B_18	0000 0000 dddd dddd	255 (0x00FF)
149(0x095)	output_format_A	dddd dddd dddd dddd	0 (0x0000)
151(0x097)	output_format_B	dddd dddd dddd dddd	0 (0x0000)
153(0x099)	spec_effects_A	dddd dddd dddd dddd	25664 (0x6440)
155(0x09B)	spec_effects_B	dddd dddd dddd dddd	25664 (0x6440)
157(0x09D)	y_rgb_offset_A	0000 0000 dddd dddd	0 (0x0000)
158(0x09E)	y_rgb_offset_B	0000 0000 dddd dddd	0 (0x0000)

Table 22: 7: Mode Variables (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
159(0x09F)	gam_shadow_0	0000 0000 dddd dddd	0 (0x0000)
160(0x0A0)	gam_shadow_1	0000 0000 dddd dddd	20 (0x0014)
161(0x0A1)	gam_shadow_2	0000 0000 dddd dddd	34 (0x0022)
162(0x0A2)	gam_shadow_3	0000 0000 dddd dddd	58 (0x003A)
163(0x0A3)	gam_shadow_4	0000 0000 dddd dddd	93 (0x005D)
164(0x0A4)	gam_shadow_5	0000 0000 dddd dddd	118 (0x0076)
165(0x0A5)	gam_shadow_6	0000 0000 dddd dddd	136 (0x0088)
166(0x0A6)	gam_shadow_7	0000 0000 dddd dddd	150 (0x0096)
167(0x0A7)	gam_shadow_8	0000 0000 dddd dddd	163 (0x00A3)
168(0x0A8)	gam_shadow_9	0000 0000 dddd dddd	175 (0x00AF)
169(0x0A9)	gam_shadow_10	0000 0000 dddd dddd	186 (0x00BA)
170(0x0AA)	gam_shadow_11	0000 0000 dddd dddd	196 (0x00C4)
171(0x0AB)	gam_shadow_12	0000 0000 dddd dddd	206 (0x00CE)
172(0x0AC)	gam_shadow_13	0000 0000 dddd dddd	215 (0x00D7)
173(0x0AD)	gam_shadow_14	0000 0000 dddd dddd	224 (0x00E0)
174(0x0AE)	gam_shadow_15	0000 0000 dddd dddd	232 (0x00E8)
175(0x0AF)	gam_shadow_16	0000 0000 dddd dddd	240 (0x00F0)
176(0x0B0)	gam_shadow_17	0000 0000 dddd dddd	248 (0x00F8)
177(0x0B1)	gam_shadow_18	0000 0000 dddd dddd	255 (0x00FF)

Table 23: 11: HG Variables

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
0(0x000)	vmt	dddd dddd dddd dddd	59667 (0xE913)
2(0x002)	DLevelBufferSpeed	0000 0000 dddd dddd	8 (0x0008)
3(0x003)	scaleGFactor	0000 0000 dddd dddd	1 (0x0001)
4(0x004)	maxDLevel	0000 0000 dddd dddd	64 (0x0040)
5(0x005)	percent	0000 0000 dddd dddd	0 (0x0000)
6(0x006)	lowerLimit1	0000 0000 dddd dddd	0 (0x0000)
7(0x007)	binSize1	0000 0000 dddd dddd	2 (0x0002)
8(0x008)	lowerLimit2	0000 0000 dddd dddd	192 (0x00C0)
9(0x009)	binSize2	0000 0000 dddd dddd	4 (0x0004)
10(0x00A)	DLevel	0000 0000 dddd dddd	0 (0x0000)
11(0x00B)	DLevel_buf	dddd dddd dddd dddd	0 (0x0000)
13(0x00D)	factorHI	0000 0000 dddd dddd	10 (0x000A)
14(0x00E)	percentHI	0000 0000 dddd dddd	0 (0x0000)
15(0x00F)	positionHI	???? ???? ???? ????	0 (0x0000)
17(0x011)	NI	dddd dddd dddd dddd	0 (0x0000)
19(0x013)	Nu	dddd dddd dddd dddd	0 (0x0000)

Table 24: 0: GPIO

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
4208(0x1070)	GPIO Data	0000 dddd dddd dddd	0 (0x0000)
4210(0x1072)	GPIO Output Toggle	0000 dddd dddd dddd	0 (0x0000)
4212(0x1074)	GPIO Output Set	0000 dddd dddd dddd	0 (0x0000)

Table 24: 0: GPIO (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
4214(0x1076)	GPIO Output Clear	0000 dddd dddd dddd	0 (0x0000)
4216(0x1078)	GPIO Direction	0000 dddd dddd dddd	4095 (0x0FFF)
4218(0x107A)	GPIO Dir. Reverse	0000 dddd dddd dddd	0 (0x0000)
4220(0x107C)	GPIO Dir. In	0000 dddd dddd dddd	0 (0x0000)
4222(0x107E)	GPIO Dir. Out	0000 dddd dddd dddd	0 (0x0000)
4224(0x1080)	1st Subperiod at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4225(0x1081)	1st Subperiod at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4226(0x1082)	2nd Subperiod at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4227(0x1083)	2nd Subperiod at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4228(0x1084)	3rd Subperiod at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4229(0x1085)	3rd Subperiod at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4230(0x1086)	4th Subperiod at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4231(0x1087)	4th Subperiod at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4232(0x1088)	5th Subperiod at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4233(0x1089)	5th Subperiod at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4234(0x108A)	Duration at AF_GPIO-1	0000 0000 dddd dddd	0 (0x0000)
4235(0x108B)	Duration at AF_GPIO-0	0000 0000 dddd dddd	0 (0x0000)
4236(0x108C)	1st Subperiod at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4237(0x108D)	1st Subperiod at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4238(0x108E)	2nd Subperiod at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4239(0x108F)	2nd Subperiod at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4240(0x1090)	3rd Subperiod at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4241(0x1091)	3rd Subperiod at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4242(0x1092)	4th Subperiod at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4243(0x1093)	4th Subperiod at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4244(0x1094)	5th Subperiod at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4245(0x1095)	5th Subperiod at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4246(0x1096)	Duration at AF_GPIO-3	0000 0000 dddd dddd	0 (0x0000)
4247(0x1097)	Duration at AF_GPIO-2	0000 0000 dddd dddd	0 (0x0000)
4248(0x1098)	1st Subperiod at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4249(0x1099)	1st Subperiod at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4250(0x109A)	2nd Subperiod at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4251(0x109B)	2nd Subperiod at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4252(0x109C)	3rd Subperiod at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4253(0x109D)	3rd Subperiod at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4254(0x109E)	4th Subperiod at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4255(0x109F)	4th Subperiod at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4256(0x10A0)	5th Subperiod at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4257(0x10A1)	5th Subperiod at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4258(0x10A2)	Duration at AF_GPIO-5	0000 0000 dddd dddd	0 (0x0000)
4259(0x10A3)	Duration at AF_GPIO-4	0000 0000 dddd dddd	0 (0x0000)
4260(0x10A4)	1st Subperiod at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4261(0x10A5)	1st Subperiod at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)
4262(0x10A6)	2nd Subperiod at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4263(0x10A7)	2nd Subperiod at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)

Table 24: 0: GPIO (continued)

Register #Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec (Hex)
4264(0x10A8)	3rd Subperiod at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4265(0x10A9)	3rd Subperiod at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)
4266(0x10AA)	4th Subperiod at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4267(0x10AB)	4th Subperiod at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)
4268(0x10AC)	5th Subperiod at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4269(0x10AD)	5th Subperiod at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)
4270(0x10AE)	Duration at AF_GPIO-7	0000 0000 dddd dddd	0 (0x0000)
4271(0x10AF)	Duration at AF_GPIO-6	0000 0000 dddd dddd	0 (0x0000)
4272(0x10B0)	Waveform Generator Config.	0000 0000 dddd dddd	0 (0x0000)
4273(0x10B1)	Chain Waveforms	0000 0000 0ddd dddd	0 (0x0000)
4274(0x10B2)	Waveform Clock Dividers	0000 0000 dddd dddd	0 (0x0000)
4275(0x10B3)	Clock Divider Selects	0000 0000 dddd dddd	0 (0x0000)
4276(0x10B4)	Sync Waveform to Frame	0000 0000 dddd dddd	0 (0x0000)
4277(0x10B5)	Waveform Resets	0000 0000 dddd dddd	0 (0x0000)
4278(0x10B6)	Waveform Suspends	0000 0000 dddd dddd	0 (0x0000)
4280(0x10B8)	Enable Notification Signals	0000 0000 dddd dddd	0 (0x0000)
4281(0x10B9)	Trigger Edge Selects	0000 dddd dddd dddd	0 (0x0000)
4283(0x10BB)	Trigger Mask	0000 dddd dddd dddd	4095 (0x0FFF)
4285(0x10BD)	Sync Waveform to STROBE	0000 0000 dddd dddd	0 (0x0000)
4286(0x10BE)	Signals Pending	0000 dddd dddd dddd	0 (0x0000)

Register Description

Table 25: 0: Core Registers

Reg. #	Bits	Default	Name
12288 0x3000	15:0	0x1580	model_id_ (RW)
	Chip version. Read-only. Can be made read/write by clearing Reg0x301A-B[3].		
12290 0x3002	15:0	0x0004	y_addr_start_ (RW)
	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. Must be an even value.		
12292 0x3004	15:0	0x0004	x_addr_start_ (RW)
	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. Must be an even value.		
12294 0x3006	15:0	0x04BB	y_addr_end_ (RW)
	The last row of visible pixels to be read out. Must be an odd value.		
12296 0x3008	15:0	0x064B	x_addr_end_ (RW)
	The last column of visible pixels to be read out. Must be an odd value.		
12298 0x300A	15:0	0x04ED	frame_length_lines_ (RW)
	The number of complete lines (rows) in the output frame. This includes visible lines and vertical blanking lines.		
12300 0x300C	15:0	0x0824	line_length_pck_ (RW)
	The number of pixel clock periods in one line (row) time. This includes visible pixels and horizontal blanking time.		
12302 0x300E	15:0	0x0120	sample_time_pck (RW)
	Number of pixel clocks from the start of the row time count (which goes from 0 to line_length_pck) to the start of the CB warm-up phase. The independent control provided by sample_time_pck allows the CB warm-up phase to overlap the row read sequence.		
12304 0x3010	15:0	0x0085	fine_correction (RW)
	Offset that is added/subtracted from the fine_integration_time when controlling the position of the shutter operations in the row sequence.		
12306 0x3012	15:0	0x0010	coarse_integration_time_ (RW)
	Integration time specified in multiples of line_length_pck_.		
12308 0x3014	15:0	0x014A	fine_integration_time_ (RW)
	Integration time specified as a number of pixel clocks.		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
	15:0	**0x2111**	**row_speed (RW)**
	15:13	0x0001	Reserved
	12:10	X	Reserved
	9:8	0x0001	Reserved
	7:4	0x0001	Pixel 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.
	3	X	Reserved
12310 0x3016	2:0	0x0001	Pixel Clock Speed A programmed value of N gives a pixel clock period of N system clocks. A value of 0 is illegal: it causes the clock to stop.
	15:0	**0x0000**	**extra_delay (RW)**
12312 0x3018	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.		
12314 0x301A	**15:0**	**0x0248**	**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.
	14:11	X	Reserved
	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.
	9	0x0001	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.
	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 Reg0x3026-7.
	7	0x0000	Parallel Enable 0 = the parallel data interface (DOUT[9:0], LINE_VALID, FRAME_VALID, and PIXCLK) is disabled and the outputs are placed in a high-impedance state. 1 = the parallel data interface is enabled. The output signals can be switched between a driven and a high-impedance state using output-enable control See “Streaming/Standby Control” on page 79.

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
	6	0x0001	Drive Pins 0 = the parallel data interface (DOUT[9:0], LINE_VALID, FRAME_VALID, and PIXCLK) may enter a high-impedance state (depending upon the configuration of Reg0x3026). See "Output-Enable Control" on page 76. 1 = the parallel data interface is driven.
	5	0x0000	Reserved
	4	0x0000	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.
	3	0x0001	Lock/Unlock Registers Many registers that are specified as read-only are actually implemented as read/write registers. Clearing this bit allows such registers to be written.
	2	0x0000	Start/Stop Streaming 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. .
	1	0x0000	Restart This bit always reads as 0. Setting this bit causes the sensor to truncate the current frame and start resetting the first row. The delay before the first valid frame is read out equals the integration time.
	0	0x0000	Reset This bit always reads as 0. Setting this bit initiates a reset sequence: the frame being generated will be truncated.
12316 0x301C	**15:0**	**0x0000**	**image_orientation_mode_select_ (RW)**
	15:9	X	Reserved
	8	0x0000	Alias of stream mode Reg 0x301A[2]
	7:2	X	Reserved
	1	0x0000	Vertical Flip This bit is an alias of Reg0x3040[1]
	0	0x0000	Horizontal Mirror This bit is an alias of Reg0x3040[0]
	This bit is an alias of Reg0x301A-B[2]		
12318 0x301E	**15:0**	**0x002A**	**data_pedestal_ (RW)**
	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 Reg0x301A-B[3].		
12320 0x3020	**15:0**	**0x0000**	**software_reset_ (RW)**
	This bit is an alias of Reg0x301A-B[0]		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12322 0x3022	**15:0**	**0x0001**	**mask_corrupted_frames_grouped_parameter_hold_ (RW)**
	15:9	X	Reserved
	8	0x0000	Alias of grouped_parameter_hold
	7:1	X	Reserved
	0	0x0001	Alias of mask_bad_frames
	This bit is an alias of Reg0x301A-B[15]		
12324 0x3024	**15:0**	**0x0000**	**pixel_order_ (RO)**
	15:2	X	Reserved
	1:0	0x0000	pixel_order 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 Reg0x3040[1:0] xor'ed with readout_order.

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12326 0x3026	**15:0**	**0xFF8F**	**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.
	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.
	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 Reg0x301A-B[8]=0.
	6:4	X	Reserved
	3	RO	GPI3 Read-only. Return the current state of the GPI3 input pin. Invalid if Reg0x301A-B[8]=0.
	2	RO	GPI2 Read-only. Return the current state of the GPI2 input pin. Invalid if Reg0x301A-B[8]=0.
	1	RO	GPI1 Read-only. Return the current state of the GPI1 input pin. Invalid if Reg0x301A-B[8]=0.
	0	RO	GPI0 Read-only. Return the current state of the GPI0 input pin. Invalid if Reg0x301A-B[8]=0. See "General-Purpose Inputs" on page 79
12328 0x3028	**15:0**	**0x0008**	**analogue_gain_code_global_ (RW)**
	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.		
12330 0x302A	**15:0**	**0x0008**	**analogue_gain_code_greenR_ (RW)**
	The gain code written to this register sets the gain for green pixels on red/green rows of the pixel array.		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12332 0x302C	15:0	0x0008	**analogue_gain_code_red_ (RW)**
	The gain code written to this register sets the gain for red pixels.		
12334 0x302E	15:0	0x0008	**analogue_gain_code_blue_ (RW)**
	The gain code written to this register sets the gain for blue pixels.		
12336 0x3030	15:0	0x0008	**analogue_gain_code_greenB_ (RW)**
	Writing a gain code to this register sets the gain for green pixels on blue/green rows of the pixel array.		
12338 0x3032	15:0	0x0100	**digital_gain_greenR_ (RW)**
	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:] are significant and are an alias of Reg0x3056[10:8].		
12340 0x3034	15:0	0x0100	**digital_gain_red_ (RW)**
	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 Reg0x305A[10:8].		
12342 0x3036	15:0	0x0100	**digital_gain_blue_ (RW)**
	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 Reg0x3058[10:8].		
12344 0x3038	15:0	0x0100	**digital_gain_greenB_ (RW)**
	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 Reg0x305C[10:8].		
12348 0x303C	15:0	0x0000	**standby_status (RW)**
	15:2	X	Reserved
	1	0x0000	standby_status_303c 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.
	0	0x0000	framesync_status This bit is reset each framesync and set every time there is a register write.Its intended function is to act as a debug flag to tell the SOC whether all writes for next frame "made it". If read at the start of frame, it will be 1 if there has been a write since framesync. This is then an error condition.
12352 0x3040	15:0	0x0024	**read_mode (RW)**

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
	15:14	0x0000	Special LINE_VALID 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
	13:12	X	Reserved
	11	0x0000	x bin enable Enable analogue binning in X (column) direction. When set, x_odd_inc must be set to 3 and y_odd_inc must be set to 1.
	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.
	9:8	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.
	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.
	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 Reg0x3024). The bit-order of bits [1:0] match the order in Reg0x301D but is reversed relative to earlier Micron Imaging sensors.
	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 Reg0x3024).

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12358 0x3046	**15:0**	**0x0600**	**flash (RW)**
	15	RO	Strobe Reflects the current state of the FLASH output signal.
	14	RO	Triggered Indicates that the FLASH output signal was asserted for the current frame.
	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.
	12:11	0x0000	Frame Delay Flash pulse delay measured in frames.
	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.
	9	0x0001	Every Frame 1 = Flash should be enabled every frame. 0 = Flash should be enabled for 1 frame only.
	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.
	7:0	X	Reserved
12360 0x3048	**15:0**	**0x0008**	**flash_count (RW)**
	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.		
12374 0x3056	**15:0**	**0x0110**	**green1_gain (RW)**
	15:11	X	Reserved
	10:8	0x0001	Digital Gain Digital Gain. Legal values 1-7.
	7	0x0000	Analog Gain Analog gain = (bit [7] + 1) * initial gain.
	6:0	0x0010	Initial Gain Initial gain = bits [6:0] * 1/16.
12376 0x3058	**15:0**	**0x0110**	**blue_gain (RW)**
	15:11	X	Reserved
	10:8	0x0001	Digital Gain Digital Gain. Legal values 1-7.
	7	0x0000	Analog Gain Analog gain = (bit [7] + 1) * initial gain.
	6:0	0x0010	Initial Gain Initial gain = bits [6:0] * 1/16.

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12378 0x305A	**15:0**	**0x0110**	**red_gain (RW)**
	15:11	X	Reserved
	10:8	0x0001	Digital Gain Digital Gain. Legal values 1-7.
	7	0x0000	Analog Gain Analog gain = (bit [7] + 1) * initial gain.
	6:0	0x0010	Initial Gain Initial gain = bits [6:0] * 1/16.
12380 0x305C	**15:0**	**0x0110**	**green2_gain (RW)**
	15:11	X	Reserved
	10:8	0x0001	Digital Gain Digital Gain. Legal values 1-7.
	7	0x0000	Analog Gain Analog gain = (bit [7] + 1) * initial gain.
	6:0	0x0010	Initial Gain Initial gain = bits [6:0] * 1/16.
12382 0x305E	**15:0**	**0x0110**	**global_gain (RW)**
	15:11	X	Reserved
	10:8	0x0001	Digital Gain
	7	0x0000	Analog Gain
	6:0	0x0010	Initial Gain
	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 green2_gain register.		
12400 0x3070	**15:0**	**0x0000**	**test_pattern_mode_ (RW)**
	0 = Normal operation: generate output data from pixel array 1 = Solid colour test pattern. 2 = 100% colour bar test pattern 3 = Fade to grey colour bar test pattern 4 = PN9 Link integrity test pattern 5-7 = Reserved.		
12402 0x3072	**15:0**	**0x0000**	**test_data_red_ (RW)**
	The value for red pixels in the bayer data used for the solid colour test pattern and the test cursors.		
12404 0x3074	**15:0**	**0x0000**	**test_data_greenR_ (RW)**
	The value for green pixels in read/green rows of the bayer data used for the solid colour test pattern and the test cursors.		
12406 0x3076	**15:0**	**0x0000**	**test_data_blue_ (RW)**
	The value for blue pixels in the bayer data used for the solid colour test pattern and the test cursors.		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12408 0x3078	15:0	0x0000	**test_data_greenB_ (RW)**
	The value for green pixels in blue/green rows of the bayer data used for the solid colour test pattern and the test cursors.		
12448 0x30A0	15:0	0x0001	**x_even_inc_ (RO)**
	Read-only.		
12450 0x30A2	15:0	0x0001	**x_odd_inc_ (RW)**
	This register field is an alias of Reg0x3040[7:5]		
12452 0x30A4	15:0	0x0001	**y_even_inc_ (RO)**
	Read-only.		
12454 0x30A6	15:0	0x0001	**y_odd_inc_ (RW)**
	This register field is an alias of Reg0x3040[4:2]		
12470 0x30B6	15:0	0x0000	**Green1 Frame Average (RO)**
	The frame averaged Green1 black level that is used in the black level calibration algorithm.		
12472 0x30B8	15:0	0x0000	**Blue Frame Average (RO)**
	The frame averaged blue black level that is used in the black level calibration algorithm.		
12474 0x30BA	15:0	0x0000	**Red Frame Average (RO)**
	The frame averaged red black level that is used in the black level calibration algorithm.		
12476 0x30BC	15:0	0x0000	**Green2 Frame Average (RO)**
	The frame averaged green2 black level that is used in the black level calibration algorithm.		
12478 0x30BE	15:0	0x231D	**calib_threshold (RW)**
	15	X	Reserved
	14:8	0x0023	Upper Threshold Upper threshold for targeted black level in ADC LSBs.
	7	X	Reserved
	6:0	0x001D	Lower Threshold Lower threshold for targeted black level in ADC LSBs.

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12480 0x30C0	**15:0**	**0x4080**	**calib_control (RW)**
	15	0x0000	Disable Rapid Sweep Mode Disables the rapid sweep mode in the black level algorithm. The averaging mode will still be enabled.
	14	0x0001	Rapid_sweep_only When set Calib slow algorithm is off, i.e. only rapid sweep occurs.
	13	X	Reserved
	12	0x0000	Recalculate When set, the rapid-sweep mode will be triggered if enabled, and the running frame average will be reset to the current frame average. This bit is write - 1 but always reads back as 0.
	11:8	X	Reserved
	7:5	0x0004	Frames to Average Over Two to the power of this value decide how many frames to average over when the black level algorithm is in the averaging mode. In this mode, the running frame average will be calculated from the following formula: Running frame average = Old running frame average - (old running frame average)/2n + (new frame average)/2n.
	4	0x0000	Step Size Forced to One When set, the step size will be forced to "1" for the rapidsweep algorithm. Default operation (0) is to start at a higher step size when in rapid-sweep mode, to converge to the correct value faster.
	3	X	Reserved
	2	0x0000	Same Red/Blue When this bit is set, the same calibration value will be used for red and blue pixels: Calib blue = calib red.
	1	0x0000	Same Green When this bit is set, the same calibration value will be used for all green pixels: Calib green2 = calib green1.
	0	0x0000	Manual Override 0 = Normal operation 1 = Override automatic black level values with values programmed into Reg0x30C2, Reg0x30C4, Reg0x30C6, Reg0x30C8
12482 0x30C2	**15:0**	**0x0000**	**calib_green1 (RW)**
	Analog calibration offset for green1 pixels, represented as a two's complement signed 8-bit value (if [8] is clear, the offset is positive and the magnitude is given by [7:0]. If [8] is set, the offset is negative and the magnitude is given by not([7:0]) + 1). If Reg0x30C0[0] = 0, this register is read-only and returns the current value computed by the black level calibration algorithm. If Reg0x30C0[0] = 1, this register is read/write, and can be used to set the calibration offset manually. Green1 pixels share rows with red pixels.		
12484 0x30C4	**15:0**	**0x0000**	**calib_blue (RW)**
	Analog calibration offset for blue pixels, represented as a two's complement signed 8-bit value (if [8] is clear, the offset is positive and the magnitude is given by [7:0]. If [8] is set, the offset is negative and the magnitude is given by not([7:0]) + 1). If Reg0x30C0[0] = 0, this register is read-only and returns the current value computed by the black level calibration algorithm. If Reg0x30C0[0] = 1, this register is read/write and can be used to set the calibration offset manually.		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12486 0x30C6	15:0	0x0000	**calib_red (RW)**
	Analog calibration offset for red pixels, represented as a two's complement signed 8-bit value (if [8] is clear, the offset is positive and the magnitude is given by [7:0]. If [8] is set, the offset is negative and the magnitude is given by not([7:0]) + 1). If Reg0x30C0[0] = 0, this register is read-only and returns the current value computed by the black level calibration algorithm. If Reg0x30C0[0] = 1, this register is read/write and can be used to set the calibration offset manually.		
12488 0x30C8	15:0	0x0000	**calib_green2 (RW)**
	Analog calibration offset for green2 pixels, represented as a two's complement signed 8-bit value (if [8] is clear, the offset is positive and the magnitude is given by [7:0]. If [8] is set, the offset is negative and the magnitude is given by not([7:0]) + 1). If Reg0x30C0[0] = 0, this register is read-only and returns the current value computed by the black level calibration algorithm. If Reg0x30C0[0] = 1, this register is read/write and can be used to set the calibration offset manually. Green2 pixels share rows with blue pixels.		
12640 0x3160	15:0	0x0000	**global_seq_trigger (RW)**
	15:3	X	Reserved
	2	0x0000	Global Flash TBD
	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 synchronisation 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).
	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.
12642 0x3162	15:0	0x0000	**global_rst_end (RW)**
12644 0x3164	15:0	0x0000	**global_shutter_start (RW)**
12646 0x3166	15:0	0x0000	**global_read_start (RW)**
12776 0x31E8	15:0	0x0000	**horizontal_cursor_position_ (RW)**
	Specify the start column for the test cursor.		
12778 0x31EA	15:0	0x0000	**vertical_cursor_position_ (RW)**
	Specify the start column for the test cursor. See "Test Cursors" on page 102		

Table 25: 0: Core Registers (continued)

Reg. #	Bits	Default	Name
12780 0x31EC	**15:0**	**0x0000**	**horizontal_cursor_width_ (RW)**
	Specify the width, in rows, of the horizontal test cursor. A width of 0 disables the cursor. See "Test Cursors" on page 102		
12782 0x31EE	**15:0**	**0x0000**	**vertical_cursor_width_ (RW)**
	Specify the width, in columns, of the vertical test cursor. A width of 0 disables the cursor. See "Test Cursors" on page 102		
12796 0x31FC	**15:0**	**0x7A78**	**I2C IDS (RW)**
	I2C Slave address. SADDR = '1', I2C base address = reg[15:8] SADDR = '0', I2C base address = reg[7:0]		

Table 26: 1: SOC1

Reg. #	Bits	Default	Name
12802 0x3202	**15:0**	**0x0009**	**Standby Control (RW)**
	15	RO	SOC standby pin is mapped to this bit
	14:6	X	Reserved
	5	0x0000	Reserved
	4	0x0000	Synchronize standby entry with the end of the frame If bit is set than standby entry is postponed until the end of current frame. If standby request comes between frames standby will be entery will have 1 frame latency.
	3	0x0001	Enable IRQ. When this bit is set IRQ is used to enter standby procedure 2020SOC enters standby mode by means of IRQ. If this bit is set standby functionality is enabled. If this bit is cleared both pin and I2C means of standby entry are disabled. 2020SOC can only enter standby through MCU command in this case.
	2	0x0000	Stop MCU on power up This bit stops MCU and allows host to upload configuration settings
	1	0x0000	Do not initialize variables after standby. If this bit is set most of the memory content will be preserved on power up.
	0	0x0001	Standby SHiP Standby request through SHiP interface. Set High to request Standby Mode.
12804 0x3204	**15:0**	**0x0001**	**Standby Done Status Bit (RW)**
	15:3	X	Reserved
	2	0x0000	Reserved
	1	0x0000	This bit is set to '1' when initial initializateon is done. This bit is set to '1' when initial initializateon is done. Only works if 0x3202[2] is set.
	0	0x0001	This bit will be set upon standby entry Set to '1' one when chip is in a standby. Should be used for host to indicate that it is safe to gate clock.
	This bit indicates that standby procedure has been completed.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12816 0x3210	**15:0**	**0x01F8**	**Color Pipeline Control (RW)**
	15:11	X	Reserved
	10	0x0000	Enable 1D aperture correction 1=1D aperture correction 0=Bypass 1D aperture correction
	9	0x0000	Reserved
	8	0x0001	Scaler 1=Enable scaler 0=Bypass scaler
	7	0x0001	Enable gamma correction 1=Enable gamma correction 0=Bypass gamma correction
	6	0x0001	Invert output pixel clock 1=invert output pixel clock (in all modes , except 10- bit bypass) 0=Do not invert output pixel clock
	5	0x0001	Enable color correction 1=enable color correction 0=bypass color correction (unity color matrix)
	4	0x0001	Enable 2D aperture correction 1=Enable 2D aperture correction 0=Bypass 2D aperture correction
	3	0x0001	Enable on-the-fly defect correction 1=Enable on-the-fly defect correction 0=Bypass on-the-fly defect correction
	2	0x0000	Enable lens shading 1=Enable lens shading 0=Bypass lens shading
	1	0x0000	Toggles the assumption about Bayer CFA (horizontal shift) Toggles the assumption about Bayer CFA (column shift). 0—column with Blue comes first. 1—column with Red comes first.
	0	0x0000	Toggles the assumption about Bayer CFA (vertical shift) Toggles the assumption about Bayer CFA (vertical shift) ‘0’ – column with Blue comes first '1’ – column with Red comes first
12818 0x3212	**15:0**	**0x0001**	**Factory Bypass (RW)**
	15:5	X	Reserved
	4:3	0x0000	GPIO output bypass. (Confidential)
	2	0x0000	Set DOUT[7:0] FRAME_VALID LINE_VALID and PIXCLK to a known state 0 = set DOUT[7:0], FRAME_VALID, LINE_VALID and PIXCLK to zero. 1 = set those outputs to 1.
	1:0	0x0001	Data output bypass. 00=10-bit sensor 01=SOC no FIFO 10=SOC with constant rate FIFO 11=output a constant value

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0C80**	**Pad Slew (RW)**
	15:12	X	Reserved
	11	0x0001	Enables power down of VDD GPIO
	10:8	0x0004	Slew rate for PIXCLK 7=fastest slew; 0=slowest. Actual slew depends on load, temperature and I/O voltage. See pad datasheet for details.
	7	0x0001	Input pad power down enable Reserved
	6:4	0x0000	Slew rate for GPIO 7=fastest slew; 0=slowest. Actual slew depends on load, temperature and I/O voltage. See pad datasheet for details.
	3	X	Reserved
	2:0	0x0000	Slew rate for DOUT[7:0] PIXCLK FRAME_VALID and LINE_VALID. 7=fastest slew rate; 0=slowest slew rate. Actual slew rate depends on load, temperature and I/O voltage. See pad datasheet for details.
12820 0x3214	7=fastest slew; 0=slowest. ёActual slew depends on load, temperature and I/O voltage. See pad datasheet for details.		
	15:0	**0x0000**	**Internal clock control (RW)**
	15:10	X	Reserved
	9	0x0000	This signal OR'ed with external CLK_EN provides master clock for the chip This signal OR'ed with external CLK_EN provides master clock for the chip 1=enable clock. WARNING: If Standby enabled and this bit=1 then clock to chip is gated and can only be re-enabled by releasing STANDBY!
	8	0x0000	Reserved
	7	0x0000	Reserved
	6	0x0000	Reserved
	5	0x0000	Reserved
	4	0x0000	Reserved
	3	0x0000	Reserved
	2	0x0000	Reserved
	1	0x0000	Reserved
12822 0x3216	0	0x0000	Reserved

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12828 0x321C	**15:0**	**0x0002**	**Output FIFO control and status (RW)**
	15:11	X	Reserved
	10	RO	FIFO empty FIFO empty
	9	RO	FIFO underflow
	8	RO	FIFO overflow Indicates overflow for output FIFO
	7	X	Reserved
	6:1	0x0001	Reserved
	0:3	X	Reserved
	2	0x0000	Half rate mode Enables half speed mode for constant rate fifo
	1	X	Reserved
	0	0x0000	Keep flags. If this bit is set bits 9 and 8 will keep high value if set, otherwise reset will occur at the beginning of each line
	This register contain control (7:0) and status (15:8) bits for FIFO operation		
12830 0x321E	**15:0**	**0x0190**	**Output FIFO watermark (RW)**
	FIFO watermark is a number of bytes that has to be stored in the FIFO prior to initiating read operation.		
12832 0x3220	**15:0**	**0x0640**	**Output FIFO Line byte count (RW)**
	Value for byte count in the line. Should be calculated as number of bits per pixel x number of pixels. Used for generation of line valid signal.		
12834 0x3222	**15:0**	**0x0000**	**Lower X Boundary for Zoom Window (RW)**
	Lower X Boundary for Zoom Window. In preview mode the value loaded to the register is divided by X skip factor		
12836 0x3224	**15:0**	**0x0640**	**Upper X Boundary for Zoom Window (RW)**
	Upper X Boundary for Zoom Window. In preview mode the value loaded to the register is divided by X skip factor		
12838 0x3226	**15:0**	**0x0000**	**Lower Y Boundary for Zoom Window (RW)**
	Lower Y Boundary for Zoom Window. In preview mode the value loaded to the register is divided by Y skip factor		
12840 0x3228	**15:0**	**0x04B0**	**Upper Y Boundary for Zoom Window (RW)**
	Upper Y Boundary for Zoom Window.In preview mode the value loaded to the register is divided by Y skip factor		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12842 0x322A	**15:0**	**0x0000**	**Scaler Control (RW)**
	15:7	X	Reserved
	6	0x0000	Reserved
	5	0x0000	Reserved
	4	0x0000	Reserved
	3	0x0000	Reserved
	2	0x0000	High precision mode Additional bits for scaling result are stored. Only for scaling >2
	1	0x0000	Reserved
	0	0x0000	Reserved
	This register controls operation of the scaler		
12844 0x322C	**15:0**	**0x0800**	**Weight for Horizontal Scaling (RW)**
	Weight for Horizontal Scaling Xsize =int(XInputSize/2048*Value)		
12846 0x322E	**15:0**	**0x0800**	**Weight for Vertical Scaling (RW)**
	Weight for Vertical Scaling Ysize=int(YInputSize/2048*Value) InputSize is defined by R0x3222-0x3228		
12862 0x323E	**15:0**	**0x0310**	**Edge Threshold for WB Statistic (RW)**
	15	X	Reserved
	14	0x0000	This bit enables mode when AWB considers pixels by hue edges only.
	13	0x0000	this bit enables mode when AWB considers pixels by luma and hue edges
	12:8	0x0003	Hue edge threshold value for WB
	7:0	0x0010	Luma edge threshold
12864 0x3240	**15:0**	**0xC814**	**Luminance Range of Pixels Considered in WB Statistics (RW)**
	15:8	0x00C8	Upper limit of luminance for WB statistics
	7:0	0x0014	Lower limit of luminance for WB statistics
	In order to avoid skewing WB statistics by very dark or very bright values this register allows programming the luminance range of pixels to be used for WB computation.		
12890 0x325A	**15:0**	**0x5000**	**Right/Left Coordinates of AWB Measurement Window (RW)**
	15:8	0x0050	Right window boundary
	7:0	0x0000	Left window boundary
	This register specifies the Right/Left coordinates of the window used by AWB measurement engine. The values programmed in the registers are desired boundaries divided by 8.		
12892 0x325C	**15:0**	**0x3C00**	**Bottom/Top Coordinates of AWB Measurement Window (RW)**
	15:8	0x003C	Bottom window boundary
	7:0	0x0000	Top window boundary
	This register specifies the Bottom/Top coordinates of the window used by AWB measurement engine. The values programmed in the registers are desired boundaries divided by 8.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12896 0x3260	15:0	0x0000	**Red Chrominance Measure Calculated by AWB (RO)**
	This register contains a measure of red chrominance obtained using AWB measurement algorithm. The measure is normalized to an arbitrary max. value, the same for R0x3260, R0x3262 and R0x3264. Because of this normalization, only the ratios of values of registers R0x3260, R0x3262 and R0x3264 are meaningful.		
12898 0x3262	15:0	0x0000	**Luminance Measure Calculated by AWB (RO)**
	This register contains a measure of image luminance obtained using AWB measurement algorithm. The measure is normalized to an arbitrary max. value, the same for R0x3260, R0x3262 and R0x3264. Because of this normalization, only the ratios of values of registers R0x3260, R0x3262 and R0x3264 are meaningful.		
12900 0x3264	15:0	0x0000	**Blue Chrominance Measure Calculated by AWB (RO)**
	This register contains a measure of blue chrominance obtained using AWB measurement algorithm. The measure is normalized to an arbitrary max. value, the same for R0x3260, R0x3262 and R0x3264. Because of this normalization, only the ratios of values of registers R0x3260, R0x3262 and R0x3264 are meaningful.		
12906 0x326A	15:0	0x1208	**1D Aperture Parameters (RW)**
	15:14	X	Reserved
	13:11	0x0002	Ap_exp Exponent for gain for aperture signal
	10:8	0x0002	Ap_gain Gain for aperture signal
	7:0	0x0008	Ap_knee Threshold for aperture signal to lower noise
	This apperture correction is applied after scaler and only can improve sharpness of the vertical lines (1D). Work similar to 2D aperture correcion (see description in 0x326C)		
12908 0x326C	15:0	0x1208	**Aperture Parameters (RW)**
	15	X	Reserved
	14	0x0000	Ap_abs
	13:11	0x0002	Ap_exp Exponent for gain for aperture signal
	10:8	0x0002	Ap_gain Gain for aperture signal
	7:0	0x0008	Ap_knee threshold for aperture signal to lower noise
	Defines 2D aperture gain and threshold. If ap_in is aperture signal from interpolation and ap_out is resulting aperture, when (abs(ap_in)-threshold)>0, ap_out = sgn(ap_in) * gain * (abs(ap_in) - threshold) * 2^(ap_exp-3).		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0080**	**Low pass YUV filters (RW)**
	15:8	X	Reserved
	7	0x0001	B/W filter enable switch When "1" Y filter can be switched off for B/W areas as controlled by U/V thresholds specified in registers 0x3270(11:7) and 0x3272(11:7).
	6	0x0000	Switch for adaptive Y filter threshold Switch for adaptive Y filter threshold. If it is "0," control signals are combined by AND function if "1" they are combined by OR (see description 0x3270, 0x3272, and 3274)
	5	0x0000	Permanently enable Y filter
	4:3	0x0000	Y filter mode 00 - no filter 01 - median 3 10 - median 5
	2:0	0x0000	UV filter 000 - no filter 001 - 11110 010 - 01100 011 - 01210 100 - 12221 101 - median 3 110 - median 5
12910 **0x326E**	YUV signals could be processed by low pass averaging or midian filters. Average and median option is avalable for UV and madian option is avalable for Y signals. Should be used with coution since can reduce resolution.		
	15:0	**0x07AA**	**Threshold for Y Filter. R Channel (RW)**
	15:12	X	Reserved
	11:7	0x000F	U Threshold
	6	0x0000	Invert control signal
	5	0x0001	Enable control signal for R Channel
	4:0	0x000A	Threshold value Works in 8 wide bit space.
12912 **0x3270**	Registers 0x3270–0x3274 control operation of adaptive Y filter. Control signal for each channel is created by comparison of pixel value (in 256 bit scale) with threshold (bits 4:0 x 8). If pixel value is larger than threshold, control value is "1." Bit 6 inverts control value if set to "1" and such control signal would indicate that the pixel is below threshold. If bit 5 is "0," the control bit is always "0." Three control bits for R,G, and B channels can be [(R or B) and G] (0) or [R or B or G] depending on value of bit 6 in Register 0x326e. The result of this operation enables Y filter for analyzed pixel. Enable signal can be inhibited for gray areas by specifying U and V threshold in registers 0x3270– 0x3272.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12914 0x3272	**15:0**	**0x07E4**	**Threshold for Y filter.G channel (RW)**
	15:12	X	Reserved
	11:7	0x000F	V Threshold
	6	0x0001	Invert control signal
	5	0x0001	Enable control signal for G channel
	4:0	0x0004	Threshold value Works in 8 wide bit space.
	Registers 0x3270–0x3274 control operation of adaptive Y filter. Control signal for each channel is created by comparison of pixel value (in 256 bit scale) with threshold (bits 4:0 x 8). If pixel value is larger than threshold, control value is "1." Bit 6 inverts control value if set to "1" and such control signal would indicate that the pixel is below threshold. If bit 5 is "0," the control bit is always "0." Three control bits for R,G, and B channels can be [(R or B) and G] (0) or [R or B or G] depending on value of bit 6 in Register 0x326e. The result of this operation enables Y filter for analyzed pixel. Enable signal can be inhibited for gray areas by specifying U and V threshold in registers 0x3270– 0x3272.		
12916 0x3274	**15:0**	**0x002A**	**Threshold for Y filter. B channel (RW)**
	15:7	X	Reserved
	6	0x0000	Invert control signal
	5	0x0001	Enable control signal for B channel
	4:0	0x000A	Threshold value Works in 8 wide bit space.
	Registers 0x3270–0x3274 control operation of adaptive Y filter. Control signal for each channel is created by comparison of pixel value (in 256 bit scale) with threshold (bits 4:0 x 8). If pixel value is larger than threshold, control value is "1." Bit 6 inverts control value if set to "1" and such control signal would indicate that the pixel is below threshold. If bit 5 is "0," the control bit is always "0." Three control bits for R,G, and B channels can be [(R or B) and G] (0) or [R or B or G] depending on value of bit 6 in Register 0x326e. The result of this operation enables Y filter for analyzed pixel. Enable signal can be inhibited for gray areas by specifying U and V threshold in registers 0x3270– 0x3272.		
12918 0x3276	**15:0**	**0x0000**	**Second Black Level (RW)**
	This register contains the value subtracted by IFP from pixel values prior to CCM. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0.		
12920 0x3278	**15:0**	**0x002A**	**First Black Level (RW)**
	This register contains the value subtracted by IFP from raw pixel values before applying lens shading correction and digital gains. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0. Typically, the subtracted value should be equal to the black level targeted by the sensor. This value is subtracted from all test patterns as well.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12922 0x327A	15:0	0x002A	First Black Level. Red (RW)
	This register contains the value (for red pixels) subtracted by IFP from raw pixel values before applying lens shading correction and digital gains. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0. Typically, the subtracted value should be equal to the black level targeted by the sensor. This value is subtracted from all test patterns as well.		
12924 0x327C	15:0	0x002A	First Black Level. Green_1 (RW)
	This register contains the value (for green_1 pixels) subtracted by IFP from raw pixel values before applying lens shading correction and digital gains. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0. Typically, the subtracted value should be equal to the black level targeted by the sensor. This value is subtracted from all test patterns as well.		
12926 0x327E	15:0	0x002A	First Black Level. Green_2 (RW)
	This register contains the value (for green_2 pixels) subtracted by IFP from raw pixel values before applying lens shading correction and digital gains. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0. Typically, the subtracted value should be equal to the black level targeted by the sensor. This value is subtracted from all test patterns as well.		
12928 0x3280	15:0	0x002A	First Black Level. Blue (RW)
	This register contains the value (for blue pixels) subtracted by IFP from raw pixel values before applying lens shading correction and digital gains. If the subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0. Typically, the subtracted value should be equal to the black level targeted by the sensor. This value is subtracted from all test patterns as well.		
12942 0x328E	15:0	0x0008	Edge Threshold for Interpolation (RW)
	Threshold for identifying pixel neighborhood as having an edge. A neighborhood without edge receives more averaging during interpolation for noise reduction purposes.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0000**	**Test Pattern (RW)**
	15:7	X	Reserved
	6:5	0x0000	Running '1' test pattern "01" enables 8 bit running '1' test pattern. "11" enables 10 bit running '1' test pattern. To use this test pattern chip has to be configured in 10 bit bypass mode. Sequence for 8 bit running one pattern is following: 00-01-02-04...-80-FF. For 10 bit sequence is similar.
	4	0x0000	Switch row Switches row interpretation for the test pattern.
	3	0x0000	Switch column Switches column interpretation for the test pattern.
	2:0	0x0000	Test mode 001- flat field values for RGB from R0x3292-0x3296 010-vertical ramp 011-regular color test pattern 100-vertical strips. Intensities are in R0x3292 and R0x3294 101-Linear shift feedback register test mode (Noise)
12944 **0x3290**	Test modes: 001- flat field values for RGB from R0x3292-0x3296 010-vertical ramp 011-regular color test pattern 100-vertical strips. Intensities are in R0x3292 and R0x3294 101-Linear shift feedback register test mode (psudo random pattern)		
12946 **0x3292**	**15:0**	**0x0100**	**Test Pattern R Value (RW)**
	Value for the red component on the flat field test pattern		
12948 **0x3294**	**15:0**	**0x0100**	**Test Pattern G Value (RW)**
	Value for the green component on the flat field test pattern		
12950 **0x3296**	**15:0**	**0x0100**	**Test Pattern B Value (RW)**
	Value for the blue component on the flat field test pattern		
12956 **0x329C**	**15:0**	**0x0020**	**Digital Gain 2 (RW)**
	Default setting 32 corresponds to gain value of 1. Gain scales linearly with value		
12960 **0x32A0**	**15:0**	**0x0000**	**Factory Test – CRC Data (RO)**
12992 **0x32C0**	**15:0**	**0x2923**	**Color Correction Matrix Exponents for C11..C22 (RW)**
	2:0—matrix element 1 (C11) exponent 5:3—matrix element 2 (C12) exponent 8:6—matrix element 3 (C13) exponent 11:9—matrix element 4 (C21) exponent 14:12—matrix element 5 (C22) exponent		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
12994 0x32C2	**15:0**	**0x0524**	**Color Correction Matrix Exponents for C22..C33 (RW)**
	Color Correction Matrix Exponents for C22..C33 2:0—matrix element 6 (C23) exponent 5:3—matrix element 7(C31) exponent 8:6—matrix element 8(C32) exponent 11:9— matrix element 9(C33) exponent The value of a matrix coefficient is calculated Cij=(1 - 2*Sij)*Mij*2^-(Eij + 4), 0< = Eij < = 4 Here Sij is coefficient's sign, Mij is the mantissa and Eij is the exponent.		
12996 0x32C4	**15:0**	**0xBBC8**	**Color Correction Matrix Elements 1 and 2 (RW)**
	15:8	0x00BB	Color correction matrix element 2(C12)
	7:0	0x00C8	Color correction matrix element 1(C11)
12998 0x32C6	**15:0**	**0xCA3A**	**Color Correction Matrix Elements 3 and 4 (RW)**
	15:8	0x00CA	Color correction matrix element 4(C21)
	7:0	0x003A	Color correction matrix element 3(C13)
13000 0x32C8	**15:0**	**0x3B85**	**Color Correction Matrix Elements 5 and 6 (RW)**
	15:8	0x003B	Color correction matrix element 6(C23)
	7:0	0x0085	Color correction matrix element 5(C22)
13002 0x32CA	**15:0**	**0xF26F**	**Color Correction Matrix Elements 7 and 8 (RW)**
	15:8	0x00F2	Color correction matrix element 8(C32)
	7:0	0x006F	Color correction matrix element 7(C31)
13004 0x32CC	**15:0**	**0x3D9C**	**Color Correction Matrix Element 9 and Signs (RW)**
	15:14	X	Reserved
	13:8	0x003D	Signs for off-diagonal CCM elements Signs for off-diagonal CCM elements. Bit 8—sign for C12 Bit 9—sign for C13 Bit 10—sign for C21 Bit 11—sign for C23 Bit 12—sign for C31 Bit 13—sign for C32 1— indicates negative 0—indicates positive Signs for C11, C22, and C33 are assumed always positive.
	7:0	0x009C	Color Correction Matrix Element 9 (C33)
13012 0x32D4	**15:0**	**0x0080**	**Digital Gain 1 for Red Pixels (RW)**
	Default setting 128 corresponds to gain value of 1. Gain scales linearly with value		
13014 0x32D6	**15:0**	**0x0080**	**Digital Gain 1 for Green1 Pixels (RW)**
	Default setting 128 corresponds to gain value of 1. Gain scales linearly with value		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
13016 0x32D8	15:0	0x0080	**Digital Gain 1 for Green2 Pixels (RW)**
	Default setting 128 corresponds to gain value of 1. Gain scales linearly with value		
13018 0x32DA	15:0	0x0080	**Digital Gain 1 for Blue Pixels (RW)**
	Default setting 128 corresponds to gain value of 1. Gain scales linearly with value		
13020 0x32DC	15:0	0x0080	**Digital Gain 1 for All Colors (RW)**
	Write 128 to set all gains [R0x32D4-R0x32DA] to 1. When read, this register returns the value of R0x32D6.		
13044 0x32F4	15:0	0x0050	**Boundaries of Flicker Measurement Window [Left/Width] (RW)**
	15:8	0x0000	Left window boundary
	7:0	0x0050	Window width
	This register specifies the boundaries of the window used by the flicker measurement engine. Values programmed in the registers are the desired boundaries divided by 8.		
13046 0x32F6	15:0	0x0088	**Boundaries of Flicker Measurement Window [Top/Height] (RW)**
	15:6	0x0002	Top window boundary
	5:0	0x0008	Window height
	This register specifies the boundaries of the window used by the flicker measurement engine.		
13048 0x32F8	15:0	0x1400	**Flicker Measurement Window Size (RW)**
	This register specifies the number of pixels in the window used by the flicker measurement engine.		
13050 0x32FA	15:0	0x0000	**Measure of the Average Luminance in Flicker Measurement Window (RO)**
	Average Luminance value in Window		
13100 0x332C	15:0	0x0000	**Blank Frames (RW)**
	1- blank outgoing frames. The bit is synchronized with frame enable. See freeze bit R0x3330 [7]		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0000**	**Output format configuration (RW)**
	15:9	X	Reserved
	8	0x0000	Turn on processed Bayer mode Chip output data in Bayer fomat. As a result datarate decreses in two times.
	7:6	0x0000	RGB output format RGB output format: 00 = 16-bit RGB565 01 = 15-bit RGB555 10 = 12-bit RGB444x 11 = 12-bit RGBx444
	5	0x0000	RGB/YUV output RGB/YUV output 1=output RGB (see R0x332E [7:6]) 0=output YUV
	4	0x0000	Use CCIR656 codes when bypassing FIFO 1=use CCIR656 codes when bypassing FIFO 0xAB = frame start 0x80 = line start 0x9D = line end 0xB6 = frame end
	3	0x0000	Monochrome output Monochrome output.
	2	0x0000	Reserved
	1	0x0000	Swaps chrominance byte with luminance byte in YUV output. In RGB mode, swaps odd and even bytes. This bit is subject to synchronous update.
	0	0x0000	Swap Channels In YUV output mode, swaps Cb and Cr channels. In RGB mode, swaps R and B. This bit is subject to synchronous update.
13102 **0x332E**	Output format configuration		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0000**	**Output Format Test (RW)**
	15:12	X	Reserved
	11	0x0000	Enables output test ramp Enables output test ramp
	10	0x0000	Reserved
	9	0x0000	Reserved
	8	0x0000	Enable Lens Correction Bypass Bypass 8+2, after lens correction. Used for calibrating lens correction. 8+2 bypass needs to be enabled to use this mode.
	7	0x0000	Freeze update of R0x332E and SOC size registers 1=freeze update of R0x332E and SOC size registers
	6	0x0000	Enable 8+2 bypass 1=enable 8+2 bypass
	5:3	0x0000	Test ramp output 00 – off 01 – by column 10 – by row 11 – by frame
	2	0x0000	Disable Cb channel 1=disable Cb channel (B in RGB mode)
	1	0x0000	Disable Y channel 1=disable Y channel (G in RGB mode)
13104 0x3330	0	0x0000	Disable Cr channel 1=disable Cr channel (R in RGB mode)
13106 0x3332	**15:0**	**0x0000**	**Line Count (RO)**
	Line number inside frame		
13108 0x3334	**15:0**	**0x0000**	**Frame Count (RO)**
	Number of frames sent since reset. Counter wraps around at 16-bit boundary.		
	15:0	**0x6440**	**Special Effects (RW)**
	15:8	0x0064	Solarization threshhold
	7	X	Reserved
	6	0x0001	Dither Luma only 1= dither only in luma channel 0 = dither in all color channels
	5:3	0x0000	Bit width of dither Bit width of dither (valid values 1, 2, 3, and 4; no dither if value = 0, 5, 6, or 7)
	2:0	0x0000	Special effect selection bits 0 - disabled 1 - monochrome 2 - sepia 3 - negative 4 - solarization with unmodified UV 5 - solarization with -UV
13128 0x3348			

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
13130 0x334A	**15:0**	**0xB023**	**Sepia Constants (RW)**
	15:8	0x00B0	Sepia constant for Cb
	7:0	0x0023	Sepia constant for Cr
13156 0x3364	**15:0**	**0x2700**	**Gamma Curve Knees 0 and 1 (RW)**
	15:8	0x0027	Gamma curve knee point 1
	7:0	0x0000	Gamma curve knee point 0
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13158 0x3366	**15:0**	**0x4936**	**Gamma Curve Knees 2 and 3 (RW)**
	15:8	0x0049	Gamma curve knee point 3
	7:0	0x0036	Gamma curve knee point 2
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13160 0x3368	**15:0**	**0x7864**	**Gamma Curve Knees 4 and 5 (RW)**
	15:8	0x0078	Gamma curve knee point 5
	7:0	0x0064	Gamma curve knee point 4
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13162 0x336A	**15:0**	**0x9789**	**Gamma Curve Knees 6 and 7 (RW)**
	15:8	0x0097	Gamma curve knee point 7
	7:0	0x0089	Gamma curve knee point 6
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13164 0x336C	**15:0**	**0xB0A4**	**Gamma Curve Knees 8 and 9 (RW)**
	15:8	0x00B0	Gamma curve knee point 9
	7:0	0x00A4	Gamma curve knee point 8
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13166 0x336E	**15:0**	**0xC5BB**	**Gamma Curve Knees 10 and 11 (RW)**
	15:8	0x00C5	Gamma curve knee point 11
	7:0	0x00BB	Gamma curve knee point 10
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
13168 0x3370	15:0	0xD7CE	**Gamma Curve Knees 12 and 13 (RW)**
	15:8	0x00D7	Gamma curve knee point 13
	7:0	0x00CE	Gamma curve knee point 12
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13170 0x3372	15:0	0xE8E0	**Gamma Curve Knees 14 and 15 (RW)**
	15:8	0x00E8	Gamma curve knee point 15
	7:0	0x00E0	Gamma curve knee point 14
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13172 0x3374	15:0	0xF8F0	**Gamma Curve Knees 16 and 17 (RW)**
	15:8	0x00F8	Gamma curve knee point 17
	7:0	0x00F0	Gamma curve knee point 16
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13174 0x3376	15:0	0x00FF	**Gamma Curve Knee 18 (RW)**
	There are 19 points on the Gamma curve corresponding to values of the values of the RGB data inputs to gamma correction. The knee points are at input data values (0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096).		
13180 0x337C	15:0	0x0006	**YUV/YCbCr Control (RW)**
	15:4	X	Reserved
	3	0x0000	Clip Y and UV Clips Y values to 16-235 Clips UV values to 16-240
	2	0x0001	Add 128 to UV values Adds 128 to U and V values
	1	0x0001	Coefficient control 1=YCbCr coefficients 0=YUV coefficients
	0	0x0000	Clips Y values to 16–235; clips UV values to 16–240. Clips Y values to 16–235; clips UV values to 16–240.
13182 0x337E	15:0	0x0000	**Y/RGB Offset (RW)**
	15:8	0x0000	Y Offset
	7:0	0x0000	RGB Offset

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
13190 0x3386	**15:0**	**X**	**Microcontroller Boot Mode (RW)**
	15	RO	Illegal opcode restart occurred 1=Illegal opcode restart occurred
	14	RO	Watchdog restart occurred 1=watchdog restart occurred
	13	RO	Warm restart occurred 1=warm restart occurred
	12	RO	Reserved Reserved
	11:8	RO	Step number in MCU initialization sequence Step number in MCU initialization sequence
	7	0x0000	MCU Debug Indicator
	6:5	0x0000	Reserved Reserved
	4	0x0000	Do not initialize driver tables Do not initialize driver tables
	3	0x0000	Boot in safe mode Boot in safe mode (do not run drivers)
	2	0x0000	Factory memory test select Factory memory test select 1=test MCU SRAM 0=test MCU R
	1	0x0000	Perform factory memory test
	0	0x0000	Reset MCU 1=Put MCU into reset 0=Release reset to MCU
13196 0x338C	**15:0**	**0x0000**	**Microcontroller variable/RAM address (RW)**
	15	0x0000	8-bit access 1=8-bit access; 0=16-bit
	14:13	0x0000	Select Logical Access Bits 14:13 of address for physical access; R0x338C [14:13]=01 select logical access
	12:8	0x0000	Driver ID Bits 12:8 of address for physical access; driver ID for logical access
	7:0	0x0000	Driver variable Bits 7:0 of address for physical access; driver variable offset for logical access
	Microcontroller variables are similar to SHIP registers, except that they are located in MCU memory. Variables are accessed by specifying their address in and reading/writing the value to . Variables can be accessed as 8-bit (byte) and 16-bit (word) at a time. Variable address can be specified as physical or logical. Physical address is the actual address of the variable in the micro-controller's 64K address space. Use physical address to upload custom binary code to a known memory location. Use logical address to configure driver variables. A logical address consists of a driver ID (0=monitor, 1=sequencer, etc) and an offset into the driver data structure.		

Table 26: 1: SOC1 (continued)

Reg. #	Bits	Default	Name
13200 0x3390	15:0	0x0000	**MCU variable/RAM data (burst 0) (RW)**
	To read a variable from MCU memory, set address in R0x338C and read data from Rox3390. To write to a variable, set address in R0x338C and write to R0x3390. 16-bit and 8-bit variables can be accessed, see R0x3390. When reading an 8-bit variable (R0x338C [15]=0) R0x3390 [15:8] are set to 0. When writing to an 8-bit variable, R0x3390 [15:8] are ignored		
13202 0x3392	15:0	0x0000	**Variable data burst SHIP access 1 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst SHIP access mode. The variables must have consecutive addresses		
13204 0x3394	15:0	0x0000	**Variable data burst SHIP access 2 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst SHIP access mode. The variables must have consecutive addresses		
13206 0x3396	15:0	0x0000	**Variable data burst SHIP access 3 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst SHIP access mode. The variables must have consecutive addresses		
13208 0x3398	15:0	0x0000	**Variable data burst SHIP access 4 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst SHIP access mode. The variables must have consecutive addresses		
13210 0x339A	15:0	0x0000	**Variable data burst SHIP access 5 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst two-wire serial interface access mode. The variables must have consecutive addresses.		
13212 0x339C	15:0	0x0000	**Variable data burst SHIP access 6 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst two-wire serial interface access mode. The variables must have consecutive addresses.		
13214 0x339E	15:0	0x0000	**Variable data burst SHIP access 7 (RW)**
	Use these registers to read or write up to 16 bytes of variable data using the burst two-wire serial interface access mode. The variables must have consecutive addresses.		
13302 0x33F6	15:0	0x0020	**Threshold for noise reduction - Red (RW)**
13304 0x33F8	15:0	0x0020	**Threshold for noise reduction - Green (RW)**
13306 0x33FA	15:0	0x0020	**Threshold for noise reduction - Blue (RW)**
13308 0x33FC	15:0	0x00FF	**Threshold for noise reduction - Min./ Max. (RW)**

Table 27: 2: SOC2

Reg. #	Bits	Default	Name
	15:0	**0x000C**	**MIPI Control (RW)**
	15:5	X	Reserved
	4	0x0000	Update Sync'd Registers Forces the update of registers that are normally updated at the start of a frame
	3	0x0001	Continuous MIPI Clock When asserted the high speed clock continues to be output on the clock lane between packets NOTE: this must only be toggled whilst Enable MIPI is de-asserted
	2	0x0001	Standby Enable Asserted when chip is preparing to enter low power standby mode. Reset to 1 and must be de-asserted to exit ultra low power mode
	1	0x0000	Reset MIPI Immediately reset MIPI interface
13312 **0x3400**	0	0x0000	Enable MIPI Asserted by host or MCU to enable the MIPI interface
	15:0	**0x0111**	**MIPI Status (RO)**
	15:9	X	Reserved
	8	0x0001	FIFO_empty Asserted when the FIFO memory is empty. Read only
	7	0x0000	FIFO_underflow Asserted if an attempt is made to read from an empty FIFO memory. Write 1 to clear
	6	0x0000	FIFO_overflow Asserted if an attempt is made to write to a full FIFO memory. Write 1 to clear
	5	0x0000	MIPI_rdy_for_data Asserted once the wake-up and initialization periods (see D-PHY spec) are complete. Any frame presented to the MIPI interface prior to this bit being asserted will be 'ignored'. Read only
	4	0x0001	MIPI_idle Asserted if MIPI interface is not cu0rrently transmitting packet data. Read only
	3	0x0000	line_size_mismatch Asserted if the size of a line from the SOC colour-pipeline differs from the size specified by the line_byte_count register. Write 1 to clear
	2	0x0000	frame_size_mismatch Asserted if the number of lines in a frame from the SOC colour-pipeline differs from the number specified by the frame_line_count register. Write 1 to clear
	1	0x0000	frame_truncated Asserted if a frame is terminated by standby before the programmed number of lines have been transmitted. Write 1 to clear
13314 **0x3402**	0	0x0001	MIPI_stby Asserted once the MIPI interface has completed its transition to ultra low power mode in response to a standby request and it is safe to stop clocks, disconnect power etc. Read only.

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13316 0x3404	**15:0**	**0x002B**	**MIPI Data Type (RW)**
	15:13	X	Reserved
	12	0x0000	custom_sp_frame_sync Insert the requested custom short packet between frames (otherwise will insert after the current packet)
	11	0x0000	custom_sp_req Asserted to request the transmission of a short packet with data type defined by 'custom_sp_data_type'. Automatically reset to 0 once the packet has been transmitted. No further request should be issued until the transition to 0 has been detected.
	10:8	0x0000	custom_sp_data_type The least significant 3 bits (the ms 3 bits are fixed to 'b001) of the data type code to insert in the requested short packet.
	7:6	X	Reserved
	5:0	0x002B	lp_data type Data type code as used in the long packet header - 0x1e - yuv422 - 0x20 - rgb444 - 0x21 - rgb555 - 0x22 - rgb565 - 0x2a - raw8 (processed bayer) - 0x2b - raw10
13318 0x3406	**15:0**	**0x04B0**	**MIPI Frame Line Count (RW)**
	15:0	0x04B0	frame_line_count Number of lines in the frame (and hence the expected number of long packets)
13322 0x340A	**15:0**	**0x02EE**	**MIPI FIFO Watermark (RW)**
	15:14	X	Reserved
	13	0x0000	frame_count_reset Resets the 8-bit frame counter to 1. This bit is reset once the counter reset is complete
	12	0x0000	frame_count_insert_disable When asserted the automatic insertion of the frame count value into the WC field of the frame start/frame end short packets is disabled, and a value of 0 is inserted instead
	11:10	X	Reserved
	9:0	0x02EE	fifo_watermark Number of bytes that must be stored in the FIFO before the packet transmission can begin to ensure a contiguous output data stream. The watermark is capped to line_byte_count ie. if a full line gets buffered then the FIFO will start to be emptied, regardless of the programmed watermark value
13324 0x340C	**15:0**	**0x0000**	**custom_short_pkt_wc (RW)**
	Value to insert in the WC field of a custom short packet		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13326 0x340E	**15:0**	**0x0000**	**MIPI Test (RW)**
	15:13	X	Reserved
	12	0x0000	io_tst Forces the MIPI IO lanes to go tristate for test purposes
	11	X	Reserved
	10	0x0000	bist_pass Asserted if the completed BIST sequence has detected no errors. Read only
	9	0x0000	bist_end Asserted on completion of the BIST sequence. Write 1 to clear
	8	0x0000	bist_en Trigger to start the BIST sequence for the FIFO memory
	7:5	X	Reserved
	4	0x0000	test_en When asserted the output is driven with test data as dtermined by the test_mode field
	3:0	0x0000	test_mode Test mode (valid for as long as test_en is asserted): 0x0 - force data and clock lane outputs to LP-00 0x1 - force data and clock lane outputs to LP-11 0x2 - force data lane output to HS-0 0x3 - force data lane output to HS-1 0x4 - enables a constant HS value to be output on the clock lane (to force a constant HS state on the clock lane the pll output should be bypassed allowing CP/CN to be driven directly from the CLOCK_IN pad)) 0x5- drive a fixed frequency HS square wave out on the data lane 0x6- replace pixel data from the SOC with data generated by an 8-bit fast parallel PRBS, sequence starting from a known value at the start of a frame. Data transmitted as long packets with data type code 0x30 0x7- replace pixel data from the SOC with data generated by an 8-bit incrementer starting from zero at the start of a frame. Data transmitted as long packets with data type code 0x30 (The frame/line based data sequences (modes 6 and 7) are inserted before the FIFO - all other test values are inserted after the FIFO)
13328 0x3410	**15:0**	**0x0B02**	**MIPI Pre HS Tx (RW)**
	15:12	X	Reserved
	11:8	0x000B	t_hs_zero Time to drive HS-0 before the sync sequence (NOTE: programmed value must = 4n-1 where n is an integer)
	7:4	X	Reserved
	3:0	0x0002	t_hs_prepare Time to drive LP-00 to prepare for HS transmission (NOTE: programmed value must = 4n-2 where n is an integer)

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13330 0x3412	15:0	0x0A05	**MIPI Post HS Tx (RW)**
	15:14	X	Reserved
	13:8	0x000A	t_hs_exit Time to drive LP-11 after HS burst
	7:4	X	Reserved
	3:0	0x0005	t_hs_trail Time to drive flipped differential state after last payload data bit of a HS transmission burst
13332 0x3414	15:0	0x0A0A	**MIPI Clock Overlap (RW)**
	15:14	X	Reserved
	13:8	0x000A	t_clk_post Time to drive the HS clock after the data lane has gone into low power mode
	7:6	X	Reserved
	5:0	0x000A	t_clk_pre Time to drive the HS clock signal before any data lane might start up
13334 0x3416	15:0	0x060C	**Clock Timing (RW)**
	15:12	X	Reserved
	11:8	0x0006	t_clk_trail Time to drive HS differential state after last payload clock bit of a HS transmission burst
	7:6	X	Reserved
	5:0	0x000C	t_clk_zero Minimum lead HS-0 clock lane drive period before starting clock
13336 0x3418	15:0	0x0008	**Misc Timing (RW)**
	15:6	X	Reserved
	5:0	0x0008	t_lpx Length of any low power state period
13340 0x341C	15:0	0x0150	**PLL Dividers1 (RW)**
	15:14	X	Reserved
	13:8	0x0001	pll_n_div PLL n-divider value NOTE: pll vco frequency = clkin frequency * pll_m/(pll_n+1)
	7:0	0x0050	pll_m_div PLL m-divider value

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13342 0x341E	**15:0**	**0x8F0B**	**PLL/ Clk_in control (RW)**
	15:12	0x0008	pll_pfd PLL phase detector gain
	11:8	0x000F	pll_div8_en Independent enables for each of the four identical clock feeds
	7:5	X	Reserved
	4	0x0000	ip_pd Asserted to disable the clock input receiver
	3	0x0001	hyst_en Asserted to enable hysteresis on the clock pin
	2	0x0000	phy_test Asserted to power down PLL and assume normal full chip operation using the external clock as source
	1	0x0001	pll_pd PLL power down
	0	0x0001	pll_bypass Asserted to bypass the PLL (for test and low power modes)
13440 0x3480	**15:0**	**0x0F14**	**Boundaries of First AF Measurement Window [Top/Left] (RW)**
	15:8	0x000F	Top window boundary Top window boundary
	7:0	0x0014	Left window boundary
	This register specifies top and left boundaries of the first from 16 (left-top) window used by AF measurement engine. The values programmed in the registers are desired boundaries divided by 8.		
13442 0x3482	**15:0**	**0x1E28**	**Boundaries of First AF Measurement Window [Height/Width] (RW)**
	15:8	0x001E	Window height
	7:0	0x0028	Window width
	This register specifies height and width of the first from 16 window used by AF measurement engine. The values programmed in the registers are desired boundaries divided by 2.		
13444 0x3484	**15:0**	**0x12C0**	**AF Measurement Window Size (RW)**
	This register specifies number of pixels in the window used by AF measurement engine.		
13446 0x3486	**15:0**	**0x0000**	**Average Luminance in AF Windows W12 and W11 (RO)**
	15:8	RO	Average Y in W12
	7:0	RO	Average Y in W11
13448 0x3488	**15:0**	**0x0000**	**Average Luminance in AF Windows W14 and W13 (RO)**
	15:8	RO	Average Y in W14
	7:0	RO	Average Y in W13
13450 0x348A	**15:0**	**0x0000**	**Average Luminance in AF Windows W22 and W21 (RO)**
	15:8	RO	Average Y in W22
	7:0	RO	Average Y in W21

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13452 0x348C	**15:0**	**0x0000**	**Average Luminance in AF Windows W24 and W23 (RO)**
	15:8	RO	Average Y in W24
	7:0	RO	Average Y in W23
13454 0x348E	**15:0**	**0x0000**	**Average Luminance in AF Windows W32 and W31 (RO)**
	15:8	RO	Average Y in W32
	7:0	RO	Average Y in W31
13456 0x3490	**15:0**	**0x0000**	**Average Luminance in AF Windows W34 and W33 (RO)**
	15:8	RO	Average Y in W34
	7:0	RO	Average Y in W33
13458 0x3492	**15:0**	**0x0000**	**Average Luminance in AF Windows W42 and W41 (RO)**
	15:8	RO	Average Y in W42
	7:0	RO	Average Y in W41
13460 0x3494	**15:0**	**0x0000**	**Average Luminance in AF Windows W44 and W43 (RO)**
	15:8	RO	Average Y in W44
	7:0	RO	Average Y in W43
13462 0x3496	**15:0**	**0x0028**	**AF Filter 1 (RW)**
	15:12	0x0000	C4
	11:8	0x0000	C3
	7:4	0x0002	C2
	3:0	0x0008	C1
13464 0x3498	**15:0**	**0xC0B0**	**AF Filter 1 Configuration (RW)**
	15:12	0x000C	C0 coefficient For the center tap in odd-sized filter. If filter coefficients larger then 1 the user has to use appropriate scale factor to avoid overflow. For example, use scale factor /2 for filter 0 0 0 2 -2 0 0 0.
	11	X	Reserved
	10	0x0000	Sign for C4
	9	0x0000	Sign for C3
	8	0x0000	Sign for C2
	7	0x0001	Sign for C1
	6	0x0000	Sign for C0
	5	0x0001	Symmetric filter If '1' filter is symmetric else anti-symmetric
	4	0x0001	Odd/even [9/8] filter size 1=odd length (9 taps); 0=even length (8 taps, C0 ignored)
	3:0	0x0000	Scale factor 0- 1; 1- /2; 2- /4; 3- /8; 4- /16; 5- /32; 6- /64; 7- /128; 8- /256; 9- /512; 11- *2; 12- *4; 13- *8; 14- *16; 15- *32

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13466 0x349A	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W12 and W11 (RO)**
	15:8	RO	Average Sharpness in W12
	7:0	RO	Average Sharpness in W11
13468 0x349C	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W14 and W13 (RO)**
	15:8	RO	Average Sharpness in W14
	7:0	RO	Average Sharpness in W13
13470 0x349E	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W22 and W21 (RO)**
	15:8	RO	Average Sharpness in W22
	7:0	RO	Average Sharpness in W21
13472 0x34A0	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W24 and W23 (RO)**
	15:8	RO	Average Sharpness in W24
	7:0	RO	Average Sharpness in W23
13474 0x34A2	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W32 and W31 (RO)**
	15:8	RO	Average Sharpness in W32
	7:0	RO	Average Sharpness in W31
13476 0x34A4	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W34 and W33 (RO)**
	15:8	RO	Average Sharpness in W34
	7:0	RO	Average Sharpness in W33
13478 0x34A6	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W42 and W41 (RO)**
	15:8	RO	Average Sharpness in W42
	7:0	RO	Average Sharpness in W41
13480 0x34A8	15:0	0x0000	**AF Filter 1 Average Sharpness Measure for AF Windows W44 and W43 (RO)**
	15:8	RO	Average Sharpness in W44
	7:0	RO	Average Sharpness in W43
13482 0x34AA	15:0	0x2080	**AF Filter 2 (RW)**
	15:12	0x0002	C4
	11:8	0x0000	C3
	7:4	0x0008	C2
	3:0	0x0000	C1

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13484 0x34AC	**15:0**	**0xC130**	**AF Filter 2 Configuration (RW)**
	15:12	0x000C	C0 coefficient For the center tap in odd-sized filter. If filter coefficients larger then 1 then the user has to use appropriate scale factor to avoid overflow. For example, use scale factor /2 for filter 0 0 0 2 -2 0 0 0.
	11	X	Reserved
	10	0x0000	Sign for C4
	9	0x0000	Sign for C3
	8	0x0001	Sign for C2
	7	0x0000	Sign for C1
	6	0x0000	Sign for C0
	5	0x0001	Symmetric filter If '1' filter is symmetric else anti-symmetric
	4	0x0001	Odd/even [9/8] filter size 1=odd length (9 taps); 0=even length (8 taps, C0 ignored)
	3:0	0x0000	Scale factor 0- 1; 1- /2; 2- /4; 3- /8; 4- /16; 5- /32; 6- /64; 7- /128; 8- /256; 9- /512; 11- *2; 12- *4; 13- *8; 14- *16; 15- *32
13486 0x34AE	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Window W12 and W11 (RO)**
	15:8	RO	Average Sharpness in W12
	7:0	RO	Average Sharpness in W11
13488 0x34B0	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W14 and W13 (RO)**
	15:8	RO	Average Sharpness in W14
	7:0	RO	Average Sharpness in W13
13490 0x34B2	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W22 and W21 (RO)**
	15:8	RO	Average Sharpness in W22
	7:0	RO	Average Sharpness in W21
13492 0x34B4	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W24 and W23 (RO)**
	15:8	RO	Average Sharpness in W24
	7:0	RO	Average Sharpness in W23
13494 0x34B6	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W32 and W31 (RO)**
	15:8	RO	Average Sharpness in W32
	7:0	RO	Average Sharpness in W31
13496 0x34B8	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W34 and W33 (RO)**
	15:8	RO	Average Sharpness in W34
	7:0	RO	Average Sharpness in W33

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13498 0x34BA	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AF Windows W42 and W41 (RO)**
	15:8	RO	Average Sharpness in W42
	7:0	RO	Average Sharpness in W41
13500 0x34BC	**15:0**	**0x0000**	**AF Filter 2 Average Sharpness Measure for AE Windows W44 and W43 (RO)**
	15:8	RO	Average Sharpness in W44
	7:0	RO	Average Sharpness in W43
13518 0x34CE	**15:0**	**0x0160**	**Lens Correction Control (RW)**
	15:11	X	Reserved
	10	0x0000	Shift row LC specific
	9	0x0000	Shift column LC specific
	8:6	0x0005	Divisor for the first Derivative Y Direction 000-/1, 001-/2, 010-/4, ... 111-/128.
	5:3	0x0004	Divisor for the first derivative X direction 000-/1, 001-/2, 010-/4, ... 111-/128.
	2	0x0000	If 1 all the second Y derivatives are doubled
	1	0x0000	If 1 all the second X derivatives are doubled
	0	X	Reserved
	This register controls behavior of lens correction.		
13520 0x34D0	**15:0**	**0x6432**	**Zone Boundaries X1 and X2 (RW)**
	15:8	0x0064	Zone 1 boundary [/4]. X direction
	7:0	0x0032	Zone 2 boundary [/4]. X direction
	Boundaries for zone 1 and 2.		
13522 0x34D2	**15:0**	**0x3296**	**Zone Boundaries X0 and X3 (RW)**
	15:8	0x0032	Zone 3 boundary [/4]. X direction
	7:0	0x0096	Zone 0 boundary [/4]. X direction Zone 0 boundary [/4]. X direction
	Boundaries for zone 3 and 0.		
13524 0x34D4	**15:0**	**0x9664**	**Zone Boundaries X4 and X5 (RW)**
	15:8	0x0096	Zone 5 boundary [/4]. X direction
	7:0	0x0064	Zone 4 boundary [/4]. X direction
	Boundaries for zone 4 and 5.		
13526 0x34D6	**15:0**	**0x5028**	**Zone Boundaries Y1 and Y2 (RW)**
	15:8	0x0050	Zone 1 boundary [/4]. Y direction
	7:0	0x0028	Zone 2 boundary [/4]. Y direction
	Boundaries for zone 1 and 2.		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13528 0x34D8	**15:0**	**0x2878**	**Zone Boundaries Y0 and Y3 (RW)**
	15:8	0x0028	Zone 3 boundary [/4]. Y direction
	7:0	0x0078	Zone 0 boundary [/4]. Y direction
	Boundaries for zone 0 and 3.		
13530 0x34DA	**15:0**	**0x7850**	**Zone Boundaries Y4 and Y5 (RW)**
	15:8	0x0078	Zone 5 boundary [/4]. Y direction
	7:0	0x0050	Zone 4 boundary [/4]. Y direction
	Boundaries for zone 4 and 5.		
13532 0x34DC	**15:0**	**0x0000**	**Center Offset (RW)**
	15:8	0x0000	Center offset Y [/4]
	7:0	0x0000	Center offset X [/4]
	Offset is the central point for the lens correction (relative to the center of imaging array)		
13534 0x34DE	**15:0**	**0x015E**	**F[x] for Red Color at the First Pixel of the Array (RW)**
13536 0x34E0	**15:0**	**0x0143**	**F[x] for Green Color at the First Pixel of the Array (RW)**
13538 0x34E2	**15:0**	**0x0143**	**F[x] for Green2 Color at the First Pixel of the Array (RW)**
13540 0x34E4	**15:0**	**0x0127**	**F[x] for Blue Color at the First Pixel of the Array (RW)**
13542 0x34E6	**15:0**	**0x012C**	**F[y] for Red Color at the First Pixel of the Array (RW)**
13544 0x34E8	**15:0**	**0x0103**	**F[y] for Green Color at the First Pixel of the Array (RW)**
13546 0x34EA	**15:0**	**0x0103**	**F[y] for Green2 Color at the First Pixel of the Array (RW)**
13548 0x34EC	**15:0**	**0x00FD**	**F[y] for Blue Color at the First Pixel of the Array (RW)**
13550 0x34EE	**15:0**	**0x0CEF**	**dF/dx for Red Color at the First Pixel of the Array (RW)**
13552 0x34F0	**15:0**	**0x0D38**	**dF/dx for Green Color at the First Pixel of the Array (RW)**
13554 0x34F2	**15:0**	**0x0D38**	**dF/dx for Green2 Color at the First Pixel of the Array (RW)**
13556 0x34F4	**15:0**	**0x0D92**	**dF/dx for Blue Color at the First Pixel of the Array (RW)**

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13558 0x34F6	**15:0**	**0x0C18**	**dF/dy for Red Color at the First Pixel of the Array (RW)**
13560 0x34F8	**15:0**	**0x0CF1**	**dF/dy for Green Color at the First Pixel of the Array (RW)**
13562 0x34FA	**15:0**	**0x0CF1**	**dF/dy for Green2 Color at the First Pixel of the Array (RW)**
13564 0x34FC	**15:0**	**0x0D05**	**dF/dy for Blue Color at the First Pixel of the Array (RW)**
13568 0x3500	**15:0**	**0x0B03**	**Second deriv. for Zone 0 Red Color (RW)**
	15:8	0x000B	d2F/dy2 for zone 0 red color
	7:0	0x0003	d2F/dx2 for zone 0 red color
	Second derivative for red color in zone 0.		
13570 0x3502	**15:0**	**0x0000**	**Second deriv. for Zone 0 Green Color (RW)**
	15:8	0x0000	d2F/dy2 for zone 0 green color
	7:0	0x0000	d2F/dx2 for zone 0 green color
	Second derivative for green color in zone 0.		
13572 0x3504	**15:0**	**0x0000**	**Second deriv. for Zone 0 Green2 Color (RW)**
	15:8	0x0000	d2F/dy2 for zone 0 green2 color
	7:0	0x0000	d2F/dx2 for zone 0 green2 color
	Second derivative for green2 color in zone 0.		
13574 0x3506	**15:0**	**0x0100**	**Second deriv. for Zone 0 Blue Color (RW)**
	15:8	0x0001	d2F/dy2 for zone 0 blue color
	7:0	0x0000	d2F/dx2 for zone 0 blue color
	Second derivative for blue color in zone 0.		
13576 0x3508	**15:0**	**0x2534**	**Second deriv. for Zone 1 Red Color (RW)**
	15:8	0x0025	d2F/dy2 for zone 1 red color
	7:0	0x0034	d2F/dx2 for zone 1 red color
	Second derivative for red color in zone 1.		
13578 0x350A	**15:0**	**0x1C33**	**Second deriv. for Zone 1 Green Color (RW)**
	15:8	0x001C	d2F/dy2 for zone 1 green color
	7:0	0x0033	d2F/dx2 for zone 1 green color
	Second derivative for green color in zone 1.		
13580 0x350C	**15:0**	**0x1C33**	**Second deriv. for Zone 1 Green2 Color (RW)**
	15:8	0x001C	d2F/dy2 for zone 1 green2 color
	7:0	0x0033	d2F/dx2 for zone 1 green2 color
	Second derivative for green2 color in zone1.		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13582 0x350E	**15:0**	**0x1A2E**	**Second deriv. for Zone 1 Blue Color (RW)**
	15:8	0x001A	d2F/dy2 for zone 1 blue color
	7:0	0x002E	d2F/dx2 for zone 1 blue color
	Second derivative for blue color in zone 1.		
13584 0x3510	**15:0**	**0x2A3A**	**Second deriv. for Zone 2 Red color (RW)**
	15:8	0x002A	d2F/dy2 for zone 2 red color
	7:0	0x003A	d2F/dx2 for zone 2 red color
	Second derivative for red color in zone 2.		
13586 0x3512	**15:0**	**0x252D**	**Second deriv. for Zone 2 Green Color (RW)**
	15:8	0x0025	d2F/dy2 for zone 2 green color
	7:0	0x002D	d2F/dx2 for zone 2 green color
	Second derivative for green color in zone 2.		
13588 0x3514	**15:0**	**0x252D**	**Second deriv. for Zone 2 Green2 Color (RW)**
	15:8	0x0025	d2F/dy2 for zone 2 green2 color
	7:0	0x002D	d2F/dx2 for zone 2 green2 color
	Second derivative for green2 color in zone 2.		
13590 0x3516	**15:0**	**0x2823**	**Second deriv. for Zone 2 Blue Color (RW)**
	15:8	0x0028	d2F/dy2 for zone 2 blue color
	7:0	0x0023	d2F/dx2 for zone 2 blue color
	Second derivative for blue color in zone 2.		
13592 0x3518	**15:0**	**0x0F14**	**Second deriv. for Zone 3 Red Color (RW)**
	15:8	0x000F	d2F/dy2 for zone 3 red color
	7:0	0x0014	d2F/dx2 for zone 3 red color
	Second derivative for red color in zone 3.		
13594 0x351A	**15:0**	**0x0D20**	**Second deriv. for Zone 3 Green Color (RW)**
	15:8	0x000D	d2F/dy2 for zone 3 green color
	7:0	0x0020	d2F/dx2 for zone 3 green color
	Second derivative for green color in zone 3.		
13596 0x351C	**15:0**	**0x0D20**	**Second deriv. for Zone 3 Green2 Color (RW)**
	15:8	0x000D	d2F/dy2 for zone 3 green2 color
	7:0	0x0020	d2F/dx2 for zone 3 green2 color
	Second derivative for green2 color in zone 3.		
13598 0x351E	**15:0**	**0x0421**	**Second deriv. for Zone 3 Blue Color (RW)**
	15:8	0x0004	d2F/dy2 for zone 3 blue color
	7:0	0x0021	d2F/dx2 for zone 3 blue color
	Second derivative for blue color in zone 3.		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13600 0x3520	**15:0**	**0x0D2A**	**Second deriv. for Zone 4 Red Color (RW)**
	15:8	0x000D	d2F/dy2 for zone 4 red color
	7:0	0x002A	d2F/dx2 for zone 4 red color
	Second derivative for red color in zone 4.		
13602 0x3522	**15:0**	**0x1017**	**Second deriv. for Zone 4 Green Color (RW)**
	15:8	0x0010	d2F/dy2 for zone 4 green color
	7:0	0x0017	d2F/dx2 for zone 4 green color
	Second derivative for green color in zone 4.		
13604 0x3524	**15:0**	**0x1017**	**Second deriv. for Zone 4 Green2 Color (RW)**
	15:8	0x0010	d2F/dy2 for zone 4 green2 color
	7:0	0x0017	d2F/dx2 for zone 4 green2 color
	Second derivative for green2 color in zone 4		
13606 0x3526	**15:0**	**0x1617**	**Second deriv. for Zone 4 Blue Color (RW)**
	15:8	0x0016	d2F/dy2 for zone 4 blue color
	7:0	0x0017	d2F/dx2 for zone 4 blue color
	Second derivative for blue color in zone 4.		
13608 0x3528	**15:0**	**0x1642**	**Second deriv. for Zone 5 Red Color (RW)**
	15:8	0x0016	d2F/dy2 for zone 5 red color
	7:0	0x0042	d2F/dx2 for zone 5 red color
	Second derivative for red color in zone 5.		
13610 0x352A	**15:0**	**0x1448**	**Second deriv. for Zone 5 Green Color (RW)**
	15:8	0x0014	d2F/dy2 for zone 5 green color
	7:0	0x0048	d2F/dx2 for zone 5 green color
	Second derivative for green color in zone 5.		
13612 0x352C	**15:0**	**0x1448**	**Second deriv. for Zone 5 Green2 Color (RW)**
	15:8	0x0014	d2F/dy2 for zone 5 green2 color
	7:0	0x0048	d2F/dx2 for zone 5 green2 color
	Second derivative for green2 color in zone 5.		
13614 0x352E	**15:0**	**0x0F4E**	**Second deriv. for Zone 5 Blue Color (RW)**
	15:8	0x000F	d2F/dy2 for zone 5 blue color
	7:0	0x004E	d2F/dx2 for zone 5 blue color
	Second derivative for blue color in zone 5.		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13616 0x3530	15:0	0x1F27	**Second deriv. for Zone 6 Red Color (RW)**
	15:8	0x001F	d2F/dy2 for zone 6 red color
	7:0	0x0027	d2F/dx2 for zone 6 red color
	Second derivative for red color in zone 6.		
13618 0x3532	15:0	0x1A12	**Second deriv. for Zone 6 Green Color (RW)**
	15:8	0x001A	d2F/dy2 for zone 6 green color
	7:0	0x0012	d2F/dx2 for zone 6 green color
	Second derivative for green color in zone 6.		
13620 0x3534	15:0	0x1A12	**Second deriv. for Zone 6 Green2 Color (RW)**
	15:8	0x001A	d2F/dy2 for zone 6 green2 color
	7:0	0x0012	d2F/dx2 for zone 6 green2 color d2F/dx2 for zone 6 green2 color
	Second derivative for green2 color in zone 6.		
13622 0x3536	15:0	0x1E16	**Second deriv. for Zone 6 Blue Color (RW)**
	15:8	0x001E	d2F/dy2 for zone 6 blue color
	7:0	0x0016	d2F/dx2 for zone 6 blue color
	Second derivative for blue color in zone 6.		
13624 0x3538	15:0	0x16C7	**Second deriv. for Zone 7 Red Color (RW)**
	15:8	0x0016	d2F/dy2 for zone 7 red color
	7:0	0x00C7	d2F/dx2 for zone 7 red color
	Second derivative for red color in zone 7.		
13626 0x353A	15:0	0x31C5	**Second deriv. for Zone 7 Green Color (RW)**
	15:8	0x0031	d2F/dy2 for zone 7 green color
	7:0	0x00C5	d2F/dx2 for zone 7 green color
	Second derivative for green color in zone 7.		
13628 0x353C	15:0	0x31C5	**Second deriv. for Zone 7 Green2 Color (RW)**
	15:8	0x0031	d2F/dy2 for zone 7 green2 color
	7:0	0x00C5	d2F/dx2 for zone 7 green2 color
	Second derivative for green2 color in zone 7.		
13630 0x353E	15:0	0x2AB6	**Second deriv. for Zone 7 Blue Color (RW)**
	15:8	0x002A	d2F/dy2 for zone 7 blue color
	7:0	0x00B6	d2F/dx2 for zone 7 blue color
	Second derivative for blue color in zone 7.		

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13632 0x3540	**15:0**	**0x0000**	**X2 Factors (RW)**
	15:8	0x0000	X2 factors for Y zones If 1 value of the value of d2F/dy2 is multiplied by 2
	7:0	0x0000	X2 factors for X zones If 1 value of the value of d2F/dx2 is multiplied by 2
13634 0x3542	**15:0**	**0x0002**	**Global Offset of F[x y] Function (RW)**
	Offset for F(y) and F(x) LC functions. Effect is analogous to gain. Only effective if LC is enabled.		
13636 0x3544	**15:0**	**0x018E**	**K Factor in K F[x] F[y] red - Top right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13638 0x3546	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green1 - Top right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13640 0x3548	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green2 - Top right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13642 0x354A	**15:0**	**0x018E**	**K Factor in K F[x] F[y] blue - Top right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13644 0x354C	**15:0**	**0x018E**	**K Factor in K F[x] F[y] red - Top left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13646 0x354E	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green1 - Top left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13648 0x3550	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green2 - Top left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13650 0x3552	**15:0**	**0x018E**	**K Factor in K F[x] F[y] blue - Top left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13652 0x3554	**15:0**	**0x018E**	**K Factor in K F[x] F[y] red - Bottom right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13654 0x3556	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green1 - Bottom right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13656 0x3558	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green2 - Bottom right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13658 0x355A	**15:0**	**0x018E**	**K Factor in K F[x] F[y] blue - Bottom right (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13660 0x355C	**15:0**	**0x018E**	**K Factor in K F[x] F[y] red - Bottom left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13662 0x355E	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green1 - Bottom left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13664 0x3560	**15:0**	**0x018E**	**K Factor in K F[x] F[y] green2 - Bottom left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value
13666 0x3562	**15:0**	**0x018E**	**K Factor in K F[x] F[y] blue - Bottom left (RW)**
	15:11	X	Reserved
	10	0x0000	Sign for K factor
	9:0	0x018E	Absolute Value

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13696 0x3580	**15:0**	**0x0000**	**Boundaries of First AE Measurement Window [Top/Left] (RW)**
	15:8	0x0000	Top window boundary
	7:0	0x0000	Left window boundary
	This register specifies top and left boundaries of the first from 16 (left-top) window used by AE measurement engine. The values programmed in the registers are desired boundaries divided by 8.		
13698 0x3582	**15:0**	**0x3C50**	**Boundaries of First AE Measurement Window [Height/Width] (RW)**
	15:8	0x003C	Window Height
	7:0	0x0050	Window Width
	This register specifies height and width of the first from 16 window used by AE measurment engine. The values programmed in the registers are desired boundaries divided by 2.		
13700 0x3584	**15:0**	**0x4B00**	**AE Measurement Window Size[LSW] (RW)**
	This register number of pixels in the window used by AE measurement engine.		
13702 0x3586	**15:0**	**0x0006**	**AE/AF Measurement Enable (RW)**
	15:3	X	Reserved
	2	0x0001	AF Statistic Enable
	1	0x0001	AE Statistic Enable
	0	0x0000	MS bit of the AE Measurement Window Size
	This register specifies number of pixels in the window used by AE measurement engine.		
13704 0x3588	**15:0**	**0x0000**	**Measure of the Average Luminance in AE windows W12 and W11 (RO)**
	15:8	RO	Average Y in W12
	7:0	RO	Average Y in W11
13706 0x358A	**15:0**	**0x0000**	**Measure of the Average Luminance in AE windows W14 and W13 (RO)**
	15:8	RO	Average Y in W14
	7:0	RO	Average Y in W13
13708 0x358C	**15:0**	**0x0000**	**Measure of the Average Luminance in AE windows W22 and W21 (RO)**
	15:8	RO	Average Y in W22
	7:0	RO	Average Y in W21
13710 0x358E	**15:0**	**0x0000**	**Measure of the Average Luminance in AE windows W24 and W23 (RO)**
	15:8	RO	Average Y in W24
	7:0	RO	Average Y in W23
13712 0x3590	**15:0**	**0x0000**	**Measure of the Average Luminance in AE windows W32 and W31 (RO)**
	15:8	RO	Average Y in W32
	7:0	RO	Average Y in W31

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13714 0x3592	15:0	0x0000	**Measure of the Average Luminance in AE windows W34 and W33 (RO)**
	15:8	RO	Average Y in W34
	7:0	RO	Average Y in W33
13716 0x3594	15:0	0x0000	**Measure of the Average Luminance in AE windows W42 and W41 (RO)**
	15:8	RO	Average Y in W42
	7:0	RO	Average Y in W41
13718 0x3596	15:0	0x0000	**Measure of the Average Luminance in AE windows W44 and W43 (RO)**
	15:8	RO	Average Y in W44
	7:0	RO	Average Y in W43
13730 0x35A2	15:0	0x0014	**Dark Color Kill Controls (RW)**
	15:8	X	Reserved
	7	0x0000	Enables dark color kill operation Enables dark color kill operation
	6	0x0000	Reserved
	5:3	0x0002	Reserved
	2:0	0x0004	Reserved
13732 0x35A4	15:0	0x0594	**Bright Color Kill Controls (RW)**
	15:11	X	Reserved
	10	0x0001	Enables bright color kill Enables bright color kill
	9	0x0000	Reserved
	8:6	0x0006	Color kill threshold Color kill effects only pixels with value larger then 128*value.
	5:3	0x0002	Color kill gain Determines rate of gain decrease above threshold. If pixel value is above threshold the difference is multiplied by 2^ value-1 (for 0 factor is “0”) and subtracted from the base gain.
	2:0	0x0004	Color kill saturation point Saturation; 100 = 100 percent. Scales linearly with value.
13734 0x35A6	15:0	0x0000	**Histogram Window Lower Boundaries (RW)**
	15:8	0x0000	Y0/8
	7:0	0x0000	X0/8
13736 0x35A8	15:0	0x95C7	**Histogram Window Upper Boundaries (RW)**
	15:8	0x0095	Y1/8
	7:0	0x00C7	X1/8

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13738 0x35AA	**15:0**	**0x030A**	**First Set of Bin Definitions (RW)**
	15:13	X	Reserved
	12:11	0x0000	Histogram statistics criteria
	10:8	0x0003	Bin width 0-4LSB,1-8LSB,2-16LSB,…7-512LSB on a 10-bit scale
	7:0	0x000A	Offset for bin 0 Divided by 4 on 10-bit scale
13740 0x35AC	**15:0**	**0x030A**	**Second Set of Bin Definitions (RW)**
	15:13	X	Reserved
	12:11	0x0000	Histogram statistics criteria
	10:8	0x0003	Bin width 0-4LSB,1-8LSB,2-16LSB,…7-512LSB on a 10-bit scale
	7:0	0x000A	Offset for bin 0 Divided by 4 on 10-bit scale
13742 0x35AE	**15:0**	**0x000A**	**Prescaler for the Historgram (RW)**
	15	X	Reserved
	14	0x0000	Selects where statistics for histogram and outlier modules is taken. Two options are before (0) and after (1) color correction matrix. Selects where statistics for histogram and outlier modules is taken. Two options are before (0) and after (1) color correction matrix. In case statistics is taken from after color correction it is influenced by the second black level.
	13	0x0000	Select bin set to read '0'- bin set 1, '1'-bin set 2
	12:0	0x000A	Reserved
13744 0x35B0	**15:0**	**0x0000**	**Pixel Counts for Bin 0 and Bin 1 (RO)**
	15:8	RO	Reserved
	7:0	RO	Reserved
	through 0x35B6, based on value of bit 13 in reg 0x35AE[2].		
13746 0x35B2	**15:0**	**0x0000**	**Pixel Counts for Bin 2 and Bin 3 (RO)**
	15:8	RO	Bin 3; Pixel count divided by G prescaler
	7:0	RO	Bin 2; Pixel count divided by G prescaler
	See 0x0x35B0		
13748 0x35B4	**15:0**	**0x0000**	**Pixel Counts for Bin 4 and Bin 5 (RO)**
	15:8	RO	Bin 5; Pixel count divided by G prescaler
	7:0	RO	Bin 4; Pixel count divided by G prescaler
	See 0x0x35B0		
13750 0x35B6	**15:0**	**0x0000**	**Pixel Counts for Bin 6 and Bin 7 (RO)**
	15:8	RO	Bin 7; Pixel count divided by G prescaler
	7:0	RO	Bin 6; Pixel count divided by G prescaler

Table 27: 2: SOC2 (continued)

Reg. #	Bits	Default	Name
13752 0x35B8	**15:0**	**0x0000**	**Definition of outlier counters (RW)**
	15:13	X	Reserved
	12:9	0x0000	Specifies Zone number from which outline counters have to be read Zone definitions for the outlier counters match zone definition for the autoexposure zones.
	8:7	0x0000	Select criteria for U statistics. 00-Y 01-RGB_min 10-RGB_max
	6:5	0x0000	Select criteria for L statistics.
	4:0	0x0000	Divisor of outliers counts (GO prescaler) The number of pixels that satisfy the selection criteria are divided by the number specified in this register +1 (0-1,1-2..). For example, if the window size is 100x100 the maximum count for the window will be 1e4 if the divisor is 0. If the devisor is 1, the maximum count will be 5E3
	Outlier counters can count the number of pixels for 16 zones. Zone definitions coincide with that for auto exposure zones. For each zone two counters exist. The first counts pixels above specified value(H_counter) and the second counts pixels below specified value(L_counter). Bits 6:5 and 8:7 select criteria for statistics selection. Could be the value of Y, RGB_min, and RGB_max		
13754 0x35BA	**15:0**	**0xFA05**	**Range for U and L counters (RW)**
	15:8	0x00FA	U range Statistics collected if (Pixel_val>U_range)
	7:0	0x0005	L range Statistics collected if (Pixel_val<L_range)
	Pixels are counted in the lower outlier counter when their value is less than L_range. Pixels are counted in the upper outlier counter when their value is more than U_range. Both U_range and L_range are specified in 8-bit scale.		
13756 0x35BC	**15:0**	**0x0000**	**Value for U_counter (RO)**
	Highlight counter. This register contains the number of outlier pixels for the selected zone (divided by GO prescaler). Counters are not double buffered, thus readout has to occur between frames.		
13758 0x35BE	**15:0**	**0x0000**	**Value for L_counter (RO)**
	Lowlight counter. This register contain number of outlier pixels for the selected zone (divided by GO prescaler). Counters are not double buffered, thus readout has to occur between frames.		

Table 28: 0: Monitor Variables

Reg. #	Bits	Default	Name
2 0x002	15:0	0x0000	**cmd (RW)**
	Monitor command Setting this variable to 1 causes the monitor to call a firmware function whose address equals mon.arg1. If the function needs an argument, it is given the value of mon.arg2. If the function returns a 1- or 2-byte value, it is put in mon.arg2. Setting mon.cmd to 2 causes the monitor to calculate CRC for a memory block defined by a starting address in mon.arg1 and length in mon.arg2. The CRC is returned in mon.arg2. The monitor starts executing each received command at the earliest opportunity. Before starting, it sets bit 7 of mon.cmd to 1. The bit remains set until the command execution is finished. Usage: Set mon.arg1 and mon.arg2 and write command number to mon.cmd. Wait until mon.cmd=0. Retrieve numerical result, if any, from mon.arg2.		
3 0x003	15:0	0x0000	**arg1 (RW)**
	First argument for monitor command		
5 0x005	15:0	0x0000	**arg2 (RW)**
	Second argument for monitor command		
7 0x007	15:0	0x0000	**msgCount (RW)**
	Number of posted messages Usage: Poll this variable to detect messages. When mon.msgCount > 0, read mon.msg. TBD—clear counter.		
8 0x008	15:0	0x0000	**msg [HI] (RW)**
	First message (Hi)		
10 0x00A	15:0	0x0000	**msg [LO] (RW)**
	First message (Lo)		
12 0x00C	15:0	0x001A	**ver (RW)**
	Bits [6:0] = firmware version Bit 7 = reserved		
13 0x00D	15:0	0x0000	**modul ID (RW)**

1: Sequencer Variables

Table 29: 1: Sequencer Variables

Reg. #	Bits	Default	Name
0 0x000	**15:0**	**0xE6C1**	**vmt (RW)**
	Pointer to virtual method table		
	15:0	**0x000F**	**mode (RW)**
	15:8	X	Reserved
	7:6	0x0000	TBD TBD
	5	0x0000	6500K outdoor WB If this bit is set then sequencer automatically forces outdoor white balance settings if average for given scene exposure value (ae.mmMeanEV) exceeds seq.sharedParams.outdoorTH threshold.
	4	0x0000	Auto Focus Auto focus driver. 1 - enable, 0 - disable
	3	0x0001	Histogram Histogram driver. 1 - enable, 0 - disable
	2	0x0001	Auto White Balance Auto white balance driver. 1 - enable, 0 - disable
	1	0x0001	Flicker Detection Flicker detection driver. 1 - enable, 0 - disable
	0	0x0001	AutoExposure Auto exposure driver. 1 - enable, 0 - disable
2 0x002	Set of 1-bit switches enabling various drivers and outdoor white balance option. Writing 1 to a particular switch enables the corresponding driver or option. Changes take effect immediately.		
	15:0	**0x0000**	**cmd (RW)**
3 0x003	Command or program to execute ; 0—Run 1—Go to Preview mode ; 2—Go to Capture mode ; 3—Go to Standby mode ; 4—Do lock ; 5—Refresh ; 6—Refresh mode ; Writing a positive value to this variable commands the sequencer to execute the corresponding program. The sequencer resets cmd to 0, executes the program, and then resumes running in its current state.Changes take effect immediately.		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
4 0x004	15:0	0x0000	**state (RW)**
	Current state of sequencer 0—Initialize 1—Mode Change to Preview 2—Enter Preview 3—Preview 4—Leave Preview 5—Mode Change to Capture 6—Enter Capture 7—Capture 8—Leave Capture 9—Standby		
5 0x005	15:0	0x0000	**stepMode (RW)**
	15:2	X	Reserved
	1	0x0000	Force seq next step 1 forces the sequencer to do next step
	0	0x0000	step mode On/Off Step mode On/Off (1 = On)
	Step-by-step mode for sequencer Bit 0—Step mode On/Off (1 = On) Bit 1—1 forces the sequencer to do next step. Changes take effect immediately.		
6 0x006	15:0	0x0000	**sharedParams.flashType (RW)**
	15:8	X	Reserved
	7	0x0000	flash On in Lock mode Bit 7—1 means the flash was on in LOCK mode
	6:0	0x0000	FlashType 0—None 1—LED 2—Xenon 3—Xenon burst
	Type of flash to be used		
7 0x007	15:0	0x0008	**sharedParams.aeContBuff (RW)**
	Weighting factor determining to what extent AE driver averages luma over time in continuous AE mode. Setting of 32 disables luma averaging—only current luma values are used. Lower settings enable luma averaging.		
8 0x008	15:0	0x0002	**sharedParams.aeContStep (RW)**
	Number of steps in which AE driver is expected to reach target brightness in continuous AE mode		
9 0x009	15:0	0x0020	**sharedParams.aeFastBuff (RW)**
	Weighting factor determining to what extent AE driver averages luma over time in fast mode. Setting of 32 disables luma averaging—only current luma values are used. Lower settings enable luma averaging.		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
10 0x00A	15:0	0x0001	**sharedParams.aeFastStep (RW)**
	Number of steps in which AE driver is expected to reach target brightness in fast AE mode		
11 0x00B	15:0	0x0008	**sharedParams.awbContBuff (RW)**
	Weighting factor determining to what extent AWB driver averages luma over time in continuous AWB mode. Setting of 32 disables luma averaging—only current luma values are used. Lower settings enable luma averaging.		
12 0x00C	15:0	0x0002	**sharedParams.awbContStep (RW)**
	Number of steps in which AWB driver is expected to reach target color balance in continuous AWB mode		
13 0x00D	15:0	0x0020	**sharedParams.awbFastBuff (RW)**
	Weighting factor determining to what extent AWB driver averages luma over time in fast mode. Setting of 32 disables luma averaging—only current luma values are used. Lower settings enable luma averaging.		
14 0x00E	15:0	0x0001	**sharedParams.awbFastStep (RW)**
	Number of steps in which AWB driver is expected to reach target color balance in fast AWB mode		
17 0x011	15:0	0x00A8	**sharedParams.options (RW)**
	15:8	X	Reserved
	7	0x0001	Powerup done This bit indicates that powerup sequence was done
	6	X	Reserved
	5	0x0001	use crop window for AF statistic
	4	0x0000	use crop window for AWB statistic
	3	0x0001	use crop window for AE statistic
	2	0x0000	Skip_ModeChenge_states Skip ModeChanange states
	1	0x0000	Sync_EOF Synchronize sensor StandBy with EOF
	0	0x0000	Highlight_On Highlight number of frames taken in PreviewEnter, Preview, PreviewLeave and CaptureEnter states using debug boxes (see regs 0x019B - 0x01A2)
18 0x012	15:0	0x0010	**sharedParams.totMaxFrames (RW)**
	Maximal number of frames allowed in states 2,4,6. Changes take effect immediately.		
19 0x013	15:0	0x0000	**sharedParams.flashTH (RW)**
	Exposure value (EV) threshold. If current EV is below this threshold, flash will be used in autoevaluate flash mode. Changes take effect immediately.		
20 0x014	15:0	0x000A	**sharedParams_outdoorTH (RW)**
	Exposure value (EV) threshold. If current EV is above this threshold and bit 5 of seq.mode equals 1, AWB driver is forced to choose color correction settings suitable for daylight color temperature of 6500 K. Changes take effect immediately.		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
21 0x015	15:0	0x0060	**sharedParams.LLmode (RW)**
	15:8	X	Reserved
	7	0x0000	ClusterDC vs. Gain On/Off
	6	0x0001	Reset Clip Level
	5	0x0001	Tx_boost vs. Gains On/Off
	4	0x0000	Y Filter On
	3	0x0000	Reduce ap threshold
	2	0x0000	Reduce ap correction
	1	0x0000	Reduce color saturation
	0	0x0000	Change Interp threshold
	Low Light mode parameters. Changes take effect immediately.		
22 0x016	15:0	0x0051	**sharedParams.LLvirtGain1 (RW)**
	Minimum virtual gain in Low Light mode. Defined as (ae.VirtGain/2 + ae.DGainAE1/16 + ae.DGainAE2/4). Changes take effect immediately.		
23 0x017	15:0	0x005A	**sharedParams.LLvirtGain2 (RW)**
	Maximum virtual gain in Low Light mode. Defined as (ae.VirtGain/2 + ae.DGainAE1/16 + ae.DGainAE2/4). Min. and max. LL virtual gains define when low-light parameters start and stop changing. Changes take effect immediately.		
24 0x018	15:0	0x0080	**sharedParams.LLSat1 (RW)**
	Maximum color saturation for color correction matrix (128 means 100%) applied if LL virtual gain less then SEQ_LLVIRTGAIN1. Changes take effect immediately.		
25 0x019	15:0	0x0000	**sharedParams.LLSat2 (RW)**
	Minimum color saturation for color correction matrix (128 means 100%) applied if LL virtual gain larger then SEQ_LLVIRTGAIN2. Changes take effect immediately.		
26 0x01A	15:0	0x0010	**sharedParams.LLInterpThresh1 (RW)**
	Minimum threshold for interpolation (applied if LL virtual gain less then SEQ_LLVIRTGAIN1). Changes take effect immediately.		
27 0x01B	15:0	0x0040	**sharedParams.LLInterpThresh2 (RW)**
	Maximum threshold for interpolation (applied if LL virtual gain larger then SEQ_LLVIRTGAIN2). Changes take effect immediately.		
28 0x01C	15:0	0x0002	**sharedParams.LLApCorr1 (RW)**
	Maximum aperture correction value (applied if LL virtual gain less then SEQ_LLVIRTGAIN1). Changes take effect immediately.		
29 0x01D	15:0	0x0000	**sharedParams.LLApCorr2 (RW)**
	Minimum aperture correction value (applied if LL virtual gain larger then SEQ_LLVIRTGAIN2). Changes take effect immediately.		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
30 0x01E	15:0	0x0008	**sharedParams.LLApThresh1 (RW)**
	Minimum aperture correction threshold (applied if LL virtual gain less then SEQ_LLVIRTGAIN1). Changes take effect immediately.		
31 0x01F	15:0	0x0040	**sharedParams.LLApThresh2 (RW)**
	Maximum aperture correction threshold (applied if LL virtual gain larger then SEQ_LLVIRTGAIN2). Changes take effect immediately.		
32 0x020	15:0	0x0000	**captureParams.mode (RW)**
	15:7	X	Reserved
	6	0x0000	HG On in Video mode
	5	0x0000	AWB On in Video mode
	4	0x0000	AE On in Video mode
	3	0x0000	AF On in Video mode
	2	0x0000	turn flash off before last frame in capture state
	1	0x0000	Capture Video
	0	0x0000	Capture w/ Xenon Flash
	Capture mode parameters. Take effect in Capture state only.		
33 0x021	15:0	0x0003	**captureParams.numFrames (RW)**
	Number of frames captured in still capture mode. Changes take effect in Capture state only.		
34 0x022	15:0	0x0000	**previewParEnter.ae (RW)**
	Auto exposure configuration in PreviewEnter state 0 - Off 1 - Fast settle 2 - Manual 3 - Continuous 4 - Fast settle + MDR		
35 0x023	15:0	0x0000	**previewParEnter.fd (RW)**
	Flicker detection configuration in PreviewEnter state 0—Off 1—Continuous 2—Manual		
36 0x024	15:0	0x0000	**previewParEnter.awb (RW)**
	Auto white balance configuration in PreviewEnter state 0—Off 1—Fast settling 2—Manual 3—Continuous		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
37 0x025	15:0	0x0000	**previewParEnter.af (RW)**
	Auto focus configuration in PreviewEnter state 0—Off 1—Fast 2—Manual 5—Creep compensation		
38 0x026	15:0	0x0000	**previewParEnter.hg (RW)**
	Histogram configuration in PreviewEnter state 0—Off 1—Fast 2—Manual 3—Continuous		
39 0x027	15:0	0x0000	**previewParEnter.flash (RW)**
	15:8	X	Reserved
	7	0x0000	Load user defined settings Load user defined settings 0 —Off 1—On 2—Locked 3—AutoEvaluate Bit 7—Load user defined settings
	6:0	X	Reserved
	Flash configuration in PreviewEnter state		
40 0x028	15:0	0x0040	**previewParEnter.skipframe (RW)**
	15:8	X	Reserved
	7	0x0000	Turn off fen 1 - turn off Frame Enable output
	6	0x0001	Skip State 1 - skip state
	5	0x0000	Skip 1st when LED On 1 - skip first frame when LED On
	4:0	X	Reserved
	Bit 7—1 - turn off fen ёBit 6—1 means skip state ёBit 5—1 means skip first frame when LED On		
41 0x029	15:0	0x0003	**previewPar.ae (RW)**
	Auto exposure configuration in Preview state ё0—Off ё1—Fast settling ё2—Manual ё3—Continuous		
42 0x02A	15:0	0x0002	**previewPar.fd (RW)**
	Flicker detection configuration in Preview state 0—Off 1—Continuous 2—Manual		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
43 0x02B	15:0	0x0003	**previewPar.awb (RW)**
	Auto white balance configuration in Preview state 0—Off 1—Fast settling 2—Manual 3—Continuous		
44 0x02C	15:0	0x0000	**previewPar.af (RW)**
	Auto focus configuration in Preview state 0—Off 1—Fast 2—Manual 5—Creep compensation		
45 0x02D	15:0	0x0003	**previewPar.hg (RW)**
	Histogram configuration in Preview state 0—Off 1—Fast 2—Manual 3—Continuous		
46 0x02E	15:0	0x0000	**previewPar.flash (RW)**
	15:8	X	Reserved
	7	0x0000	Load user defined settings Load user defined settings
	6:0	X	Reserved
	Flash configuration in Preview state 0—Off 1—On 2—Locked 3—AutoEvaluate Bit 7—Load user defined settings		
47 0x02F	15:0	0x0000	**previewPar.skipframe (RW)**
	15:8	X	Reserved
	7	0x0000	Turn off fen 1 means turn off fen
	6	0x0000	Skip State 1 means skip state
	5	0x0000	Skip 1st when LED On 1 means skip first frame when LED On
	4:0	X	Reserved
	Bit 7—1means turn off fen Bit 6—1means skip state Bit 5—1means skip first frame when LED On		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
48 0x030	15:0	0x0001	**previewParLeave.ae (RW)**
	Auto exposure configuration in PreviewLeave state ё0 - Off ё1 - Fast settle ё2 - Manual ё3 - Continuous4 - Fast settle + MDR		
49 0x031	15:0	0x0000	**previewParLeave.fd (RW)**
	Flicker detection configuration in PreviewLeave state 0—Off 1—Continuous 2—Manual		
50 0x032	15:0	0x0001	**previewParLeave.awb (RW)**
	Auto white balance configuration in PreviewLeave state 0—Off 1—Fast settling 2—Manual 3—Continuous		
51 0x033	15:0	0x0000	**previewParLeave.af (RW)**
	Auto focus configuration in PreviewLeave state 0—Off 1—Fast 2—Manual 5—Creep compensation		
52 0x034	15:0	0x0001	**previewParLeave.hg (RW)**
	Histogram configuration in PreviewLeave state 0—Off 1—Fast 2 —Manual 3—Continuous		
53 0x035	15:0	0x0000	**previewParLeave.flash (RW)**
	15:8	X	Reserved
	7	0x0000	Load user defined settings Load user defined settings
	6:0	X	Reserved
	Flash configuration in PreviewLeave state 0—Off 1—On 2—Locked 3—AutoEvaluate Bit 7—Load user defined settings.		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
	15:0	**0x0010**	**previewParLeave.skipframe (RW)**
	15:8	X	Reserved
	7	0x0000	Turn off fen 1 means turn off fen
	6	0x0000	Skip State 1 means skip state
	5	0x0000	Skip 1st when LED On 1 means skip first frame when LED On
	4	0x0001	skip PreviewLeave state if all auto functions were settled in preview state
	3:0	X	Reserved
54 **0x036**	Bit 7—1 means turn off fen ёBit 6—1 means skip state ёBit 5—1 means skip first frame when LED On, Bit 4 - skip state if all auto functions were settled.		
	15:0	**0x0000**	**capParEnter.ae (RW)**
55 **0x037**	Auto exposure configuration in CaptureEnter state ё0 - Off ё1 - Fast settle ё2 - Manual ё3 - Continuous4 - Fast settle + MDR		
	15:0	**0x0000**	**capParEnter.fd (RW)**
56 **0x038**	Flicker detection configuration in CaptureEnter state 0—Off 1—Continuous 2—Manual		
	15:0	**0x0000**	**capParEnter.awb (RW)**
57 **0x039**	Auto white balance configuration in CaptureEnter state 0—Off 1—Fast settling 2—Manual 3—Continuous		
	15:0	**0x0000**	**capParEnter.af (RW)**
58 **0x03A**	Auto focus configuration in CaptureEnter state 0—Off 1—Fast 2—Manual 5—Creep compensation		
	15:0	**0x0000**	**capParEnter.hg (RW)**
59 **0x03B**	Histogram configuration in CaptureEnter state 0—Off 1—Fast 2—Manual 3—Continuous		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
60 0x03C	**15:0**	**0x0000**	**capParEnter.flash (RW)**
	15:8	X	Reserved
	7	0x0000	Load user defined settings Load user defined settings
	6:0	X	Reserved
	Flash configuration in CaptureEnter state 0—Off 1—On 2—Locked 3—AutoEvaluate Bit 7—Load user defined settings		
61 0x03D	**15:0**	**0x0040**	**capParEnter.skipframe (RW)**
	15:8	X	Reserved
	7	0x0000	Turn off fen 1 means turn off fen
	6	0x0001	Skip State 1 means skip state
	5	0x0000	Skip 1st when LED On 1 means skip first frame when LED On
	4:0	X	Reserved
	Bit 7—1 means turn off fen ёBit 6—1 means skip state ёBit 5—1 means skip first frame when LED On		
62 0x03E	**15:0**	**0x0010**	**NR_minTH (RW)**
	Minimal noise reduction threshold (See description of register 251 :1). Changes take effect immediately.		
63 0x03F	**15:0**	**0x0020**	**NR_maxTH (RW)**
	Maximal noise reduction threshold (See description of register 251 :1)		
64 0x040	**15:0**	**0x0000**	**NR_GainTH (RW)**
	Virtual gain threshold above that mcu starts changing noise reduction threshold. Changes take effect immediately.		
65 0x041	**15:0**	**0x0010**	**NR_Slope (RW)**
	Slope of the noise reduction threshold		

Table 29: 1: Sequencer Variables (continued)

Reg. #	Bits	Default	Name
66 0x042	**15:0**	**0x0000**	**standByStatus (RW)**
	15:8	X	Reserved
	7	0x0000	This bit indicates PLL bypass state
	6	0x0000	This bit indicates PLL down state.
	5	0x0000	reserved
	4	0x0000	If this bit is set f/w go to standby right from IRQ function otherwise chip go to soft standby by DO_STANDBY command
	3	0x0000	allow interrupt during MCU slip only
	2	0x0000	This bit indicates MIPI was used
	1	0x0000	This bit indicates standby interrupt occurred
	0	0x0000	This bit indicates modul ID was set
	Standby status		
67 0x043	**15:0**	**0x0000**	**standByMode (RW)**
	Default to all "0". Has the same definition as R3216. This variable will be copy over to register R3216 during standby procedure. Host could over write this to control what clock get gated off during standby.		
75 0x04B	**15:0**	**0x0338**	**padClkFreq (RW)**
	HIWORD of pad clock frequency (0x0338 – 54MHz). Changes take effect immediately.		

Table 30: 2: AE Variables

Reg. #	Bits	Default	Name
0 0x000	**15:0**	**0xE6FD**	**Vmt (RW)**
	Pointer to virtual method table		
2 0x002	**15:0**	**0x0000**	**windowPos (RW)**
	15:8	X	Reserved
	7:4	0x0000	Y0 Y0 in units 1/16th of frame height
	3:0	0x0000	X0 X0 in units of 1/16th of frame width
	Position of first (upper left) AE window New position written to this variable takes effect only after REFRESH_MODE or REFRESH command is given to the sequencer (seq.cmd = 5 or 6).		

Table 30: 2: AE Variables (continued)

Reg. #	Bits	Default	Name
3 0x003	15:0	0x00EF	**windowSize (RW)**
	15:8	X	Reserved
	7:4	0x000E	height height (in units of 1/16th of frame height) decremented by 1
	3:0	0x000F	width width (in units of 1/16th of frame width) decremented by 1
	Size of 4 x 4 array of AE windows New data written to this variable take effect only after REFRESH_MODE or REFRESH command is given to the sequencer (seq.cmd = 5 or 6).		
4 0x004	15:0	0x023A	**wakeUpLine (RW)**
	Row number at which MCU wakes up to do AE calculations. This number is automatically set after REFRESH_MODE or REFRESH command (seq.cmd = 5 or 6)		
6 0x006	15:0	0x003C	**Target (RW)**
	Target brightness. Changes take effect immediately.		
7 0x007	15:0	0x000A	**Gate. (RW)**
	AE sensitivity. Changes take effort immediately		
8 0x008	15:0	0x0000	**SkipFrames (RW)**
	Frequency of AE updates. Changes take effect immediately.		
9 0x009	15:0	0x0002	**JumpDivisor (RW)**
	Factor reducing exposure jumps (1 = fastest jumps). Changes take effect immediately.		
10 0x00A	15:0	0x0008	**lumaBufferSpeed (RW)**
	Speed of buffering (32 = fastest, 1= slowest)		
11 0x00B	15:0	0x0000	**minIndex (RW)**
	intend to use to limit the min integration time. but it is not working, and there is not intention to fix it. (20060124)		
12 0x00C	15:0	0x0018	**maxIndex (RW)**
	Maximum allowed zone number (i.e. maximum integration time). Changes take effect only after REFRESH command.		
13 0x00D	15:0	0x0010	**minVirtGain (RW)**
	Minimum allowed virtual gain		
14 0x00E	15:0	0x0080	**maxVirtGain (RW)**
	Maximum allowed virtual gain. Changes take effect only after REFRESH command.		
15 0x00F	15:0	0x000B	**maxADChi (RW)**
	Maximum ADC Vref_hi setting		

Table 30: 2: AE Variables (continued)

Reg. #	Bits	Default	Name
16 0x010	15:0	0x000A	**minADChi (RW)**
	Minimum ADC Vref_hi setting. Changes take effect only after REFRESH command.		
17 0x011	15:0	0x0009	**minADClo (RW)**
	Minimum ADC Vref_lo setting		
18 0x012	15:0	0x0080	**maxDGainAE1 (RW)**
	Maximum digital gain pre-LC. Changes take effect only after REFRESH command.		
20 0x014	15:0	0x0020	**maxDGainAE2 (RW)**
	Maximum digital gain post-LC. Changes take effect only after REFRESH command.		
21 0x015	15:0	0x0008	**IndexTH23 (RW)**
	Zone number to start gain increase in low-light. Sets frame rate at normal illumination.		
22 0x016	15:0	0x0078	**maxGain23 (RW)**
	Maximum gain to increase in low-light before dropping frame rate. Sets frame rate at low-light. Changes take effect only after REFRESH command.		
23 0x017	15:0	0x00FF	**weights (RW)**
	15:8	X	Reserved
	7:4	0x000F	background weight background weight
	3:0	0x000F	center zone weight center zone weight
	AE windows weights. Changes take effect immediately.		
24 0x018	15:0	0x0084	**status (RW)**
	15:8	X	Reserved
	7	0x0001	On/Off OverExpose compensation
	6	0x0000	Exclude outliers exceeding 16EV from statistic
	5	0x0000	do one step MDR mode in ModeChange state
	4	0x0000	MDR Mode is Done
	3	0x0000	do MDR mode
	2	0x0001	Ready
	1	0x0000	1-R9 is changed (need to skip frame)
	0	0x0000	1-AE reached the limit
	AE Status		
25 0x019	15:0	0x0000	**CurrentY (RW)**
	Last measured luminance		
26 0x01A	15:0	0x0408	**R12 (RW)**
	Current shutter delay value		

Table 30: 2: AE Variables (continued)

Reg. #	Bits	Default	Name
28 0x01C	15:0	0x0004	**Index (RW)**
	Current zone (integration time)		
29 0x01D	15:0	0x0010	**VirtGain (RW)**
	Current virtual gain		
30 0x01E	15:0	0x000B	**ADC_hi (RW)**
	Current ADC VREF_HI value		
31 0x01F	15:0	0x0009	**ADC_lo (RW)**
	Current ADC VREF_LO value		
32 0x020	15:0	0x0080	**DGainAE1 (RW)**
	Current digital gain pre-LC		
34 0x022	15:0	0x0020	**DGainAE2 (RW)**
	Current digital gain post-LC		
35 0x023	15:0	0x0000	**R9 (RW)**
	Current R9:0 value (integration time)		
37 0x025	15:0	0x090B	**R65 (RW)**
	Current ADC VREF values		
41 0x029	15:0	0x0080	**gainR12 (RW)**
	Gain compensating too low maxR12 (128 = unit gain)		
42 0x02A	15:0	0x0000	**SkipFrames_cnt (RW)**
	Counter of skipped frames		
43 0x02B	15:0	0x0000	**BufferedLuma (RW)**
	Current time-buffered measured luma		
45 0x02D	15:0	0x0000	**dirAE_prev (RW)**
	Previous state of AE		
46 0x02E	15:0	0x009D	**R9_step (RW)**
	Integration time of one zone		
49 0x031	15:0	0x000A	**maxADClo (RW)**
	Maximum ADC Vref_lo setting		
50 0x032	15:0	0x0010	**physGainR (RW)**
	Physical (analog) R gain		
52 0x034	15:0	0x0010	**physGainG (RW)**
	Physical (analog) G gain		

Table 30: 2: AE Variables (continued)

Reg. #	Bits	Default	Name
54 0x036	15:0	0x0010	**physGainB (RW)**
	Physical (analog) B gain		
65 0x041	15:0	0x0000	**mmMeanEV (RW)**
	Mean EV (Exposure Value)		
66 0x042	15:0	0x000D	**mmShiftEV (RW)**
	Exposure Value shift. Adjust this value for given lens. Changes take effect immediately.		
67 0x043	15:0	0x0000	**numOE (RW)**
	Number of overexposed zones		
68 0x044	15:0	0x003D	**TargetD (RW)**
	Current AE target in MDR mode. Changes take effect immediately.		
69 0x045	15:0	0x0064	**maxNu (RW)**
	Maximal allowed number of upper outliers		
73 0x049	15:0	0x0028	**minTargetD (RW)**
	Minimal AE target in MDR mode. Changes take effect immediately.		
74 0x04A	15:0	0x00B4	**maxTargetD (RW)**
	Maximal AE target in MDR mode. Changes take effect immediately.		
79 0x04F	15:0	0x0080	**coefG2 (RW)**
	Green2 / Green1* 128 ratio		
80 0x050	15:0	0x00A6	**EV_HiLo (RW)**
	15:8	X	Reserved
	7:4	0x000A	EV_Hi Hi EV threshold for AE parameters. For bright scenes when average scene brightness (ae.mmMeanEV) above then threshold defined by EV_Hi, AE driver (in MDR mode only) increace AE target (ae.Target) and minimal AE target (ae.minTargetD) by values defined in ae,targetDelta[7:4] and ae.minTargetDelta[7:4]
	3:0	0x0006	EV_Low Low EV threshold for AE parameters. For dark scenes when average scene brightness (ae.mmMeanEV) below then threshold defined by EV_Low, AE driver (in MDR mode only) reduces AE target (ae.Target) and minimal AE target (ae.minTargetD) by values defined in ae,targetDelta[3:0] and ae.minTargetDelta[3:0]
	Low and Hi EV threshold for AE parameters defining dark and bright scenes. For MDR mode only. Changes take effect immediately.		

Table 30: 2: AE Variables (continued)

Reg. #	Bits	Default	Name
81 0x051	**15:0**	**0x00A8**	**targetDelta. (RW)**
	15:8	X	Reserved
	7:4	0x000A	HiTargetDelta AE tareget shift divided by 2 for bright scenes if ae.meanEV >= ae.EV_HiLo[7:4] then AE target increaced by HI_TARGET_DELTA/2 value. (MDR mode only)
	3:0	0x0008	LoTargetDelta AE target shift (divided by 2) for dark scenes if ae.meanEV < ae.EV_HiLo[3:0] then AE target reduced by LO_TARGET_DELTA/2 value. (MDR mode only)
	Changes take effect immediately.		
82 0x052	**15:0**	**0x0086**	**minTargetDelta (RW)**
	15:8	X	Reserved
	7:4	0x0008	Hi_minTargetDelta this value (divided by 2) added to minTarget for bright scenes.if ae.meanEV >= ae.EV_HiLo[7:4] ёthen AE minTarget increaced by LO_MINTARGET_DELTA/2 value
	3:0	0x0006	Lo_minTargetDelta this value (divided by 2) subtracted from minTarget for dark scenes. if ae.meanEV < ae.EV_HiLo[3:0] ёthen AE minTarget reduced by LO_MINTARGET_DELTA/2 value
	this variable specifies shift of minimal value of the AE target for dark and bright scenes. (MDR mode only). Changes take effect immediately.		
83 0x053	**15:0**	**0x0019**	**maxNuDelta (RW)**
	Addition (divided by 4) to maximal allowed top outliers. if ae.meanEV >= ae.EV_HiLo[7:4] ёthen value of ae.maxNu is increaced by maxNuDelta/4 value. (for MDR mode only). Changes take effect immediately.		
84 0x054	**15:0**	**0x0010**	**outliersSlope (RW)**
	This variable specifies scale factor for outliers. It is used in ONE STEP MDR mode only. Shift of ae.Target is defined as hg.Nu or hg.NI value divided by ae.outliersSlope. Changes take effect immediately.		

Table 31: 3: AWB Variables

Reg. #	Bits	Default	Name
0 0x000	**15:0**	**0xE7A3**	**vmt (RW)**
	Pointer to virtual method table		

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
2 0x002	**15:0**	**0x0000**	**windowPos (RW)**
	15:8	X	Reserved
	7:4	0x0000	Y0 Y0 in units 1/16th of frame height
	3:0	0x0000	X0 X0 in units of 1/16th of frame width
	Position of upper left corner of AWB window. New position written to this variable takes effect only after REFRESH_MODE or REFRESH command is given to the sequencer (seq.cmd = 5 or 6)		
3 0x003	**15:0**	**0x00EF**	**windowSize (RW)**
	15:8	X	Reserved
	7:4	0x000E	Height Height (in units of 1/16th of frame height) decremented by 1
	3:0	0x000F	Width Width (in units of 1/16th of frame width) decremented by 1
	Size of AWB window. New size written to this variable takes effect only after REFRESH_MODE or REFRESH command is given to the sequencer (seq.cmd = 5 or 6)		
4 0x004	**15:0**	**0x023C**	**wakeUpLine (RW)**
	Number of image row at which MCU wakes up to perform AWB calculations. This number is automatically adjusted after each change of the position and dimensions of the AWB window. Changes take effect immediately.		
6 0x006	**15:0**	**0x0285**	**ccmL[0] (RW)**
	Left color correction matrix element K11. Value is encoded in signed two's complement form; 256 means 1.0. Changes take effect immediately.		
8 0x008	**15:0**	**0xFE98**	**ccmL[1] (RW)**
	Left color correction matrix element K12. Changes take effect immediately.		
10 0x00A	**15:0**	**0x0010**	**ccmL[2] (RW)**
	Left color correction matrix element K13. Changes take effect immediately.		
12 0x00C	**15:0**	**0xFF55**	**ccmL[3] (RW)**
	Left color correction matrix element K21. Changes take effect immediately.		
14 0x00E	**15:0**	**0x0229**	**ccmL[4] (RW)**
	Left color correction matrix element K22. Changes take effect immediately.		
16 0x010	**15:0**	**0xFFCF**	**ccmL[5] (RW)**
	Left color correction matrix element K23. Changes take effect immediately.		
18 0x012	**15:0**	**0xFF65**	**ccmL[6] (RW)**
	Left color correction matrix element K31. Changes take effect immediately.		

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
20 0x014	15:0	0xFDF9	ccmL[7] (RW)
	Left color correction matrix element K32. Changes take effect immediately.		
22 0x016	15:0	0x040B	ccmL[8] (RW)
	Left color correction matrix element K33. Changes take effect immediately.		
24 0x018	15:0	0x0020	ccmL[9] (RW)
	Left color correction matrix red/green gain ratio. Value is interpreted as unsigned; 32 means 1.0. Changes take effect immediately.		
26 0x01A	15:0	0x0038	ccmL[10] (RW)
	Left color correction matrix blue/green gain ratio. Value is interpreted as unsigned; 32 means 1.0. Changes take effect immediately.		
28 0x01C	15:0	0xFFB0	ccmRL[0] (RW)
	Delta color correction matrix element D11. Value is encoded in signed two's complement form; 256 means 1.0. Changes take effect immediately.		
30 0x01E	15:0	0x0065	ccmRL[1] (RW)
	Delta color correction matrix element D12. Changes take effect immediately.		
32 0x020	15:0	0xFFEF	ccmRL[2] (RW)
	Delta color correction matrix element D13. Changes take effect immediately.		
34 0x022	15:0	0x001F	ccmRL[3] (RW)
	Delta color correction matrix element D21. Changes take effect immediately.		
36 0x024	15:0	0x0014	ccmRL[4] (RW)
	Delta color correction matrix element D22. Changes take effect immediately.		
38 0x026	15:0	0xFFD6	ccmRL[5] (RW)
	Delta color correction matrix element D23. Changes take effect immediately.		
40 0x028	15:0	0x005F	ccmRL[6] (RW)
	Delta color correction matrix element D31. Changes take effect immediately.		
42 0x02A	15:0	0x0102	ccmRL[7] (RW)
	Delta color correction matrix element D32. Changes take effect immediately.		
44 0x02C	15:0	0xFE64	ccmRL[8] (RW)
	Delta color correction matrix element D33. Changes take effect immediately.		
46 0x02E	15:0	0x0010	ccmRL[9] (RW)
	Delta color correction matrix red/green gain ratio. Value is interpreted as unsigned; 32 means 1.0. Changes take effect immediately.		

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
48 0x030	15:0	0xFFED	**ccmRL[10] (RW)**
	Delta color correction matrix blue/green gain ratio. Value is interpreted as unsigned; 32 means 1.0. Changes take effect immediately.		
50 0x032	15:0	0x0201	**ccm[0] (RW)**
	Current color correction matrix element C11. Value is stored in signed two's complement form; 256 means 1.0.		
52 0x034	15:0	0xFEFA	**ccm[1] (RW)**
	Current color correction matrix element C12		
54 0x036	15:0	0x0006	**ccm[2] (RW)**
	Current color correction matrix element C13		
56 0x038	15:0	0xFF8A	**ccm[3] (RW)**
	Current color correction matrix element C21		
58 0x03A	15:0	0x01AD	**ccm[4] (RW)**
	Current color correction matrix element C22		
60 0x03C	15:0	0xFFCB	**ccm[5] (RW)**
	Current color correction matrix element C23		
62 0x03E	15:0	0xFFAD	**ccm[6] (RW)**
	Current color correction matrix element C31		
64 0x040	15:0	0xFED2	**ccm[7] (RW)**
	Current color correction matrix element C32		
66 0x042	15:0	0x0282	**ccm[8] (RW)**
	Current color correction matrix element C33		
68 0x044	15:0	0x0028	**ccm[9] (RW)**
	Current color correction matrix red/green gain ratio. Value is interpreted as unsigned; 32 means 1.0.		
70 0x046	15:0	0x002F	**ccm[10] (RW)**
	Current color correction matrix blue/green gain ratio. Value is interpreted as unsigned; 32 means 1.0.		
72 0x048	15:0	0x0008	**GainBufferSpeed (RW)**
	Speed of time-buffering (32 = fastest, 1= slowest). Changes take effect immediately.		
73 0x049	15:0	0x0002	**JumpDivisor (RW)**
	Derating factor for WB change (1= fastest). Changes take effect immediately.		
74 0x04A	15:0	0x0066	**GainMin (RW)**
	Minimum allowed AWB digital gain (128 means 1.0) Changes take effect immediately.		

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
75 0x04B	15:0	0x0099	**GainMax (RW)**
	Maximum allowed AWB digital gain (128 means 1.0). Changes take effect immediately.		
76 0x04C	15:0	0x0080	**GainR (RW)**
	Current R digital gain (128 means1.0)		
77 0x04D	15:0	0x0080	**GainG (RW)**
	Current G digital gain (128 means1.0)		
78 0x04E	15:0	0x0080	**GainB (RW)**
	Current B digital gain (128 means1.0)		
79 0x04F	15:0	0x0000	**CCMpositionMin (RW)**
	Leftmost position of color correction matrix (corresponding to incandescent light). Changes effect immediately.		
80 0x050	15:0	0x007F	**CCMpositionMax (RW)**
	Rightmost position of color correction matrix (corresponding to daylight). Changes take effect immediately.		
81 0x051	15:0	0x0040	**CCMposition (RW)**
	Current position of color correction matrix (0 corresponds to incandescent light, 127 to daylight). Changes take effect immediately.		
82 0x052	15:0	0x0080	**saturation (RW)**
	Saturation of color correction matrix (128 means 100%). Changes take effect immediately.		
83 0x053	15:0	0x0000	**mode (RW)**
	15:8	X	Reserved
	7	0x0000	AWB_OUTDOOR 1 = force outdoor settings, set by the sequencer
	6	0x0000	AWB_CCMNORMOFF 1 = disable CCM normalization
	5	0x0000	AWB_UNITYGAINS 1 = force AWB Digital Gains to unit; program value into awb.CCMPosition to manually set matrix position
	4	0x0000	AWB_NO_STAT 1 = no awb statistic
	3	0x0000	RESERVED 1 = debug
	2	0x0000	AWB_TG 1 = reserved
	1	0x0000	AWB_LIMITS 1 = AWB limit is reached, 0 - limit not reach
	0	0x0000	AWB_DONE 1 = AWB is done.

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
84 0x054	15:0	0x0000	GainR_buf (RW)
	Time-buffered R gain (0x1000 means 1.0)		
86 0x056	15:0	0x0000	GainB_buf (RW)
	Time-buffered B gain (0x1000 means 1.0)		
88 0x058	15:0	0x0000	sumR (RW)
	AWB statistics. Scale normalized to an arbitrary number.		
89 0x059	15:0	0x0000	sumY (RW)
	AWB statistics. Scale normalized to an arbitrary number.		
90 0x05A	15:0	0x0000	sumB (RW)
	AWB statistics. Scale normalized to an arbitrary number.		
91 0x05B	15:0	0x0078	steadyBGainOutMin (RW)
	Blue gain min. threshold to start search. When AWB blue digital gain falls below this threshold, the AWB driver starts searching for new color correction matrix position. Threshold setting of 128 corresponds to gain of 1.0; higher setting means higher gain. Changes take effect immediately.		
92 0x05C	15:0	0x0086	steadyBGainOutMax (RW)
	Blue gain max. threshold to start search. When AWB blue digital gain goes above this threshold, the AWB driver starts searching for new color correction matrix position. Threshold setting of 128 corresponds to gain of 1.0; higher setting means higher gain. Changes take effect immediately.		
93 0x05D	15:0	0x007E	steadyBGainInMin (RW)
	Blue gain min. threshold to end search. When AWB blue digital gain is between awb.steadyBGainInMin and awb.steadyBGainInMax, the AWB driver maintains current color correction matrix position. Threshold setting of 128 corresponds to gain of 1.0; higher setting means higher gain. Changes take effect immediately.		
94 0x05E	15:0	0x0082	steadyBGainInMax (RW)
	Blue gain max. threshold to end search. See awb.steadyBGainInMin above. Changes take effect immediately.		
95 0x05F	15:0	0x0064	cntPxlTH (RW)
	Reserved		
97 0x061	15:0	0x00E7	TG_min0 (RW)
	Reserved		
98 0x062	15:0	0x00F6	TG_max0 (RW)
	Reserved		
99 0x063	15:0	0x0010	X0 (RW)
	CCM position threshold between Day and Incandescent normalization coefficients. Changes take effect immediately.		

Table 31: 3: AWB Variables (continued)

Reg. #	Bits	Default	Name
100 0x064	15:0	0x00AA	kR_L (RW)
	CCM red row normalization coefficient for left matrix position (128 - minimal number). Changes take effect immediately.		
101 0x065	15:0	0x0096	kG_L (RW)
	CCM green row normalization coefficient for left matrix position (128 - minimal number). Changes take effect immediately.		
102 0x066	15:0	0x0080	kB_L (RW)
	CCM blue row normalization coefficient for left matrix position (128 - minimal number). Changes take effect immediately.		
103 0x067	15:0	0x0080	kR_R (RW)
	CCM red row normalization coefficient for right matrix position (128 - minimal number). Changes take effect immediately.		
104 0x068	15:0	0x0080	kG_R (RW)
	CCM green row normalization coefficient for right matrix position (128 - minimal number). Changes take effect immediately.		
105 0x069	15:0	0x0080	kB_R (RW)
	CCM blue row normalization coefficient for right matrix position (128 - minimal number). Changes take effect immediately.		
106 0x06A	15:0	0x0080	EdgeTH (RW)
	current edge threshold for WB statistic		
107 0x06B	15:0	0x0080	EdgeTH_min (RW)
	minimum edge threshold for WB statistic. Changes take effect only after REFRESH command.		
108 0x06C	15:0	0x0080	EdgeTH_max (RW)
	maximum edge threshold for WB statistic.Changes take effect only after REFRESH command.		

Table 32: 4: Flicker Variables

Reg. #	Bits	Default	Name
0 0x000	15:0	0xE824	vmt (RW)
	Pointer to virtual method table		

Table 32: 4: Flicker Variables (continued)

Reg. #	Bits	Default	Name
2 0x002	**15:0**	**0x001D**	**windowPosH (RW)**
	15:8	X	Reserved
	7:4	0x0001	x0 Left boundary of the flicker mesurement window (in units of 1/16th of frame width)
	3:0	0x000D	width width (in units of 1/16th of frame width) decremented by 1
	Width of flicker measurement window and position of its left boundary X0 ;Bits [3:0]—width (in units of 1/16th of frame width) decremented by 1; Bits [7:4]—X0 (in units of 1/16th of frame width); At the beginning of each flicker measurement, the top boundary (Y0) of the flicker measurement window is set at row 64. Average luminance in the window is measured by the IFP measurement engine and stored in fd.Buffer[0]. Then Y0 is incremented by fd.windowHeight and the buffer index by 1. By repeating these steps 47 times, the entire fd.Buffer is filled with average luminance values from a vertical array of 48 equal-size windows.		
3 0x003	**15:0**	**0x0004**	**windowHeight (RW)**
	15:6	X	Reserved
	5:0	0x0004	flicker measurement window height in rows flicker measurement window height in rows
	Bits [5:0]—flicker measurement window height in rows		
4 0x004	**15:0**	**0x0000**	**mode (RW)**
	15:8	X	Reserved
	7	0x0000	Manual Mode 1 enables manual mode, 0 disables it
	6	0x0000	Manual Flicker Mode Setting 0 selects 60Hz settings for manual mode, 1 selects 50Hz settings
	5	RO	Current Flicker State read-only, indicates current settings (0–60Hz, 1– 50Hz)
	4	0x0000	Debug Mode 1 enables debug mode, 0 disables it
	3:0	RO	Reserved read-only, reserved for auto FD
	Flicked detection switches and indicators.		
5 0x005	**15:0**	**0x0040**	**wakeUpLine (RW)**
	Number of image row at which MCU wakes up to perform flicker detection. Changing the value of fd.wakeUpLine has no effect on the functioning of the FD driver, because it does not use this variable. It simply sets it to 64 at initialization, to indicate the top of the flicker measurement area. See fd.windowPosH above.		
7 0x007	**15:0**	**0x0005**	**smooth_cnt (RW)**
	Reserved		
8 0x008	**15:0**	**0x001E**	**search_f1_50 (RW)**
	Lower limit of period range of interest in 50 Hz flicker detection. If the FD driver is searching for 50 Hz flicker and detects in a frame a flicker-like signal whose period in rows is between fl.search_f1_50 and fl.search_f2_50, it counts the frame as one containing flicker.		

Table 32: 4: Flicker Variables (continued)

Reg. #	Bits	Default	Name
9 0x009	15:0	0x0020	**search_f2_50 (RW)**
	Upper limit of period range of interest in 50 Hz flicker detection. See fd.search_f1_50 above.		
10 0x00A	15:0	0x0025	**search_f1_60 (RW)**
	Lower limit of period range of interest in 60 Hz flicker detection. If the FD driver is searching for 60 Hz flicker and detects in a frame a flicker-like signal whose period in rows is between fl.search_f1_60 and fl.search_f2_60, it counts the frame as one containing flicker.		
11 0x00B	15:0	0x0027	**search_f2_60 (RW)**
	Upper limit of period range of interest in 60 Hz flicker detection. See fd.search_f1_60 above.		
12 0x00C	15:0	0x0000	**skipFrame (RW)**
	Skip frame before subtracting two frames.		
13 0x00D	15:0	0x0003	**stat_min (RW)**
	Flicker is considered detected if fd.stat_min out of fd.stat_max consecutive frames contain flicker (periodic signal of sought frequency).		
14 0x00E	15:0	0x0005	**stat_max (RW)**
	See fd.stat_min.		
15 0x00F	15:0	0x0000	**stat (RW)**
	Shift register, whose every bit corresponds to a frame. Bits equal to 1 indicate frames containing flicker.		
16 0x010	15:0	0x0005	**minAmplitude (RW)**
	Reserved		
17 0x011	15:0	0x009D	**R9_step_0_f60 (RW)**
	Minimal shutter width step for 60Hz AC in context A.		
19 0x013	15:0	0x00BC	**R9_step_0_f50 (RW)**
	Minimal shutter width step for 50Hz AC in context A		
21 0x015	15:0	0x0000	**R9_step_1_f60 (RW)**
	Minimal shutter width step for 60Hz AC in context B.		
23 0x017	15:0	0x00E0	**R9_step_1_f50 (RW)**
	Minimal shutter width step for 50Hz AC in context B		
25 0x019	15:0	0x0000	**Buffer[0] (RW)**
	Temporary buffer for flicker.		
26 0x01A	15:0	0x0000	**Buffer[1] (RW)**
	Temporary buffer for flicker.		
27 0x01B	15:0	0x0000	**Buffer[2] (RW)**
	Temporary buffer for flicker.		

Table 32: 4: Flicker Variables (continued)

Reg. #	Bits	Default	Name
28 0x01C	15:0	0x0000	**Buffer[3] (RW)**
	Temporary buffer for flicker.		
29 0x01D	15:0	0x0000	**Buffer[4] (RW)**
	Temporary buffer for flicker.		
30 0x01E	15:0	0x0000	**Buffer[5] (RW)**
	Temporary buffer for flicker.		
31 0x01F	15:0	0x0000	**Buffer[6] (RW)**
	Temporary buffer for flicker.		
32 0x020	15:0	0x0000	**Buffer[7] (RW)**
	Temporary buffer for flicker.		
33 0x021	15:0	0x0000	**Buffer[8] (RW)**
	Temporary buffer for flicker.		
34 0x022	15:0	0x0000	**Buffer[9] (RW)**
	Temporary buffer for flicker.		
35 0x023	15:0	0x0000	**Buffer[10] (RW)**
	Temporary buffer for flicker.		
36 0x024	15:0	0x0000	**Buffer[11] (RW)**
	Temporary buffer for flicker.		
37 0x025	15:0	0x0000	**Buffer[12] (RW)**
	Temporary buffer for flicker.		
38 0x026	15:0	0x0000	**Buffer[13] (RW)**
	Temporary buffer for flicker.		
39 0x027	15:0	0x0000	**Buffer[14] (RW)**
	Temporary buffer for flicker.		
40 0x028	15:0	0x0000	**Buffer[15] (RW)**
	Temporary buffer for flicker.		
41 0x029	15:0	0x0000	**Buffer[16] (RW)**
	Temporary buffer for flicker.		
42 0x02A	15:0	0x0000	**Buffer[17] (RW)**
	Temporary buffer for flicker.		

Table 32: 4: Flicker Variables (continued)

Reg. #	Bits	Default	Name
43 0x02B	15:0	0x0000	**Buffer[18] (RW)**
	Temporary buffer for flicker.		
44 0x02C	15:0	0x0000	**Buffer[19] (RW)**
	Temporary buffer for flicker.		
45 0x02D	15:0	0x0000	**Buffer[20] (RW)**
	Temporary buffer for flicker.		
46 0x02E	15:0	0x0000	**Buffer[21] (RW)**
	Temporary buffer for flicker.		
47 0x02F	15:0	0x0000	**Buffer[22] (RW)**
	Temporary buffer for flicker.		
48 0x030	15:0	0x0000	**Buffer[23] (RW)**
	Temporary buffer for flicker.		
49 0x031	15:0	0x0000	**Buffer[24] (RW)**
	Temporary buffer for flicker.		
50 0x032	15:0	0x0000	**Buffer[25] (RW)**
	Temporary buffer for flicker.		
51 0x033	15:0	0x0000	**Buffer[26] (RW)**
	Temporary buffer for flicker.		
52 0x034	15:0	0x0000	**Buffer[27] (RW)**
	Temporary buffer for flicker.		
53 0x035	15:0	0x0000	**Buffer[28] (RW)**
	Temporary buffer for flicker.		
54 0x036	15:0	0x0000	**Buffer[29] (RW)**
	Temporary buffer for flicker.		
55 0x037	15:0	0x0000	**Buffer[30] (RW)**
	Temporary buffer for flicker.		
56 0x038	15:0	0x0000	**Buffer[31] (RW)**
	Temporary buffer for flicker.		
57 0x039	15:0	0x0000	**Buffer[32] (RW)**
	Temporary buffer for flicker.		
58 0x03A	15:0	0x0000	**Buffer[33] (RW)**
	Temporary buffer for flicker.		

Table 32: 4: Flicker Variables (continued)

Reg. #	Bits	Default	Name
59 0x03B	15:0	0x0000	Buffer[34] (RW)
	Temporary buffer for flicker.		
60 0x03C	15:0	0x0000	Buffer[35] (RW)
	Temporary buffer for flicker.		
61 0x03D	15:0	0x0000	Buffer[36] (RW)
	Temporary buffer for flicker.		
62 0x03E	15:0	0x0000	Buffer[37] (RW)
	Temporary buffer for flicker.		
63 0x03F	15:0	0x0000	Buffer[38] (RW)
	Temporary buffer for flicker.		
64 0x040	15:0	0x0000	Buffer[39] (RW)
	Temporary buffer for flicker.		
65 0x041	15:0	0x0000	Buffer[40] (RW)
	Temporary buffer for flicker.		
66 0x042	15:0	0x0000	Buffer[41] (RW)
	Temporary buffer for flicker.		
67 0x043	15:0	0x0000	Buffer[42] (RW)
	Temporary buffer for flicker.		
68 0x044	15:0	0x0000	Buffer[43] (RW)
	Temporary buffer for flicker.		
69 0x045	15:0	0x0000	Buffer[44] (RW)
	Temporary buffer for flicker.		
70 0x046	15:0	0x0000	Buffer[45] (RW)
	Temporary buffer for flicker.		
71 0x047	15:0	0x0000	Buffer[46] (RW)
	Temporary buffer for flicker.		
72 0x048	15:0	0x0000	Buffer[47] (RW)
	Temporary buffer for flicker.		

Table 33: 5: Auto Focus Variables

Reg. #	Bits	Default	Name
0 0x000	**15:0**	**0xE82E**	**vmt (RW)**
	Pointer to virtual method table (VMT)		
2 0x002	**15:0**	**0x0044**	**windowPos (RW)**
	15:8	X	Reserved
	7:4	0x0004	y coordinate y coordinate (vertical offset from the upper left corner of the frame) in units of 1/16th of frame height.
	3:0	0x0004	x coordinate x coordinate (horizontal offset from the upper left corner of the frame) in units of 1/16th of frame width,
	Encoded position of upper left corner of first AF window (W11): Bits [3:0] - x coordinate (horizontal offset from the upper left corner of the frame) in units of 1/16th of frame width, Bits [7:4] - y coordinate (vertical offset from the upper left corner of the frame) in units of 1/16th of frame height. New position written to this variable takes effect after REFRESH_MODE command is given to Sequencer driver (sq.cmd is set to 6).		
3 0x003	**15:0**	**0x0077**	**windowSize (RW)**
	15:8	X	Reserved
	7:4	0x0007	height height (in units of 1/16th of frame height) decremented by 1.
	3:0	0x0007	width width (in units of 1/16th of frame width) decremented by 1,
	Encoded dimensions of the 4 x 4 array of AF windows: Bits [3:0] - width (in units of 1/16th of frame width) decremented by 1, Bits [7:4]-height (in units of 1/16th of frame height) decremented by 1. New dimensions written to this variable take effect only after REFRESH_MODE command is given to the Sequencer (sq.cmd is set to 6).		

Table 33: 5: Auto Focus Variables (continued)

Reg. #	Bits	Default	Name
4 0x004	**15:0**	**0x0000**	**mode (RW)**
	15:8	X	Reserved
	7	0x0000	Manual mode Manual mode switch (0 - manual mode disabled, 1 - enabled)
	6	0x0000	Creep compensation mode Creep compensation mode switch (0 - creep compensation mode disabled, 1 - enabled).
	5:1	X	Reserved
	0	0x0000	Auto focus trigger If AF is enabled in the Sequencer (sq.mode bit 4 = 1) and manual mode is disabled (af.mode bit 7 = 0), a snapshot AF sequence can be triggered at any time by setting af.mode bit 0 to 1.
	Two mode switches and 5 bits reserved for use in default snapshot AF mode: Bit 7 - manual mode switch (0 - manual mode disabled, 1 - enabled), Bit 6 - creep compensation mode switch (0 - creep compensation mode disabled, 1 - enabled), Bits [4:0] - reserved for use in snapshot AF mode. If AF is enabled in the Sequencer (sq.mode bit 4 = 1) and manual mode is disabled (af.mode bit 7 = 0), a snapshot AF sequence can be triggered at any time by setting af.mode bit 0 to 1. Bits [4:0] are used in the sequence and automatically cleared at its end.		
5 0x005	**15:0**	**0x0080**	**modeEx (RW)**
	15:8	X	Reserved
	7	0x0001	Second flyback Switch enabling the second flyback (jump to the start position of the first scan) and then jump to best focus position.
	6	0x0000	Stepping to best position Switch enabling the second flyback and retracing of scan steps to best focus position (0 - option disabled, 1 - enabled).
	5	0x0000	Fine scan Switch enabling the second scan (0 - disabled, 1 - enabled).
	4	0x0000	AF error AF algorithm status indicator, set to 1if an algorithm failure has occurred or has been anticipated and avoided by ignoring a request to refocus. The AF algorithm is likely to fail if sensor exposure settings are not stable during the search for best focus position. Hence, all attempts to use the AF algorithm simultaneously with auto exposure algorithm are thwarted by Sequencer driver. It signals preventing an anticipated failure of the AF algorithm by setting bit 4 of af.modeEx to 1. If the Sequencer allows auto focusing to proceed, the AF driver clears bit 4 of af.modeEx at the start of the first flyback. At the end of the first scan, the AF driver checks how much normalized sharpness scores from all relevant AF windows varied during the scan. If the variation for all the windows was lower than the minimum specified by af.shaTH, the AF driver sets bit 4 of af.modeEx to 1 to signal that it has no data to select best lens position. This is the only situation recognized by the AF driver as a failure of the AF algorithm.
	3	0x0000	Extra wait after AFM stops moving If bit 7 of af.mode equals 0, this bit enables skipping 1 extra frame after detecting that bit 1 of afm.status has been cleared (i.e. after the end of every lens movement).
	2	0x0000	Extra wait is in progress Status indicator, set to 1 when an extra frame is being skipped.

Table 33: 5: Auto Focus Variables (continued)

Reg. #	Bits	Default	Name
5 0x005	1	0x0000	Fine scan is in progress Status indicator, set to 1 when the second scan is in progress.
	0	0x0000	Sharpness scores are ready Status indicator, set to 1 when sharpness scores are ready.
	Four option switches and 4 status indicators: Bit 7 - switch enabling the second flyback (jump to the start position of the first scan) and then jump to best focus position, Bit 6 - switch enabling the second flyback and retracing of scan steps to best focus position (0 - option disabled, 1 - enabled), Bit 5 - switch enabling the second scan (0 - disabled, 1 - enabled), Bit 4 - AF algorithm status indicator, set to 1if an algorithm failure has occurred or has been anticipated and avoided by ignoring a request to refocus (see more below), Bit 3 - if bit 7 of af.mode equals 0, this bit enables skipping 1 extra frame after detecting that bit 1 of afm.status has been cleared (i.e. after the end of every lens movement), Bit 2 - status indicator, set to 1 when an extra frame is being skipped, Bit 1 - status indicator, set to 1 when the second scan is in progress, Bit 0- status indicator, set to 1 when sharpness scores are ready. The AF algorithm is likely to fail if sensor exposure settings are not stable during the search for best focus position. Hence, all attempts to use the AF algorithm simultaneously with auto exposure algorithm are thwarted by Sequencer driver. It signals preventing an anticipated failure of the AF algorithm by setting bit 4 of af.modeEx to 1. If the Sequencer allows auto focusing to proceed, the AF driver clears bit 4 of af.modeEx at the start of the first flyback. At the end of the first scan, the AF driver checks how much normalized sharpness scores from all relevant AF windows varied during the scan. If the variation for all the windows was lower than the minimum specified by af.shaTH, the AF driver sets bit 4 of af.modeEx to 1 to signal that it has no data to select best lens position. This is the only situation recognized by the AF driver as a failure of the AF algorithm.		
6 **0x006**	**15:0**	**0x000A**	**numSteps (RW)**
	Number of steps (lens positions tried) in the first scan.		
7 **0x007**	**15:0**	**0x0000**	**initPos (RW)**
	Number (index) of start position, af.positions[af.initPos], used in the first scan and optional second scan. Must be 0 at the beginning of the first scan if the second is enabled. Otherwise, can be set before the first scan to any value below 20- af.numSteps. The AF driver makes af.initPos equal to af.numSteps at the beginning of the second scan and equal to 0 at the end of last scan (first or second, whichever is last).		
8 **0x008**	**15:0**	**0x0006**	**numSteps2 (RW)**
	15:8	X	Reserved
	7:4	0x0000	Actual num. of steps in 2nd scan Actual number of steps in the second scan (calculated by the AF driver at the beginning of the scan).
	3:0	0x0006	Des. num. of steps in 2nd scan Desired number of steps in second scan (max. allowed number is 14).
	Bits [3:0] - desired number of steps in second scan (max. allowed number is 14) Bits [7:4] - actual number of steps in the second scan (calculated by the AF driver at the beginning of the scan).		

Table 33: 5: Auto Focus Variables (continued)

Reg. #	Bits	Default	Name
9 0x009	15:0	0x0006	**stepSize (RW)**
	Logical step size for the second scan. Because the logical range of motion is from 0 to 255, af.stepSize=6 means that during the second scan the AF driver will try to move then lens in increments equal to 6/255 of the length of its entire motion range. However, the lens actuator may or may not be able to move the lens in steps of precisely that size. It is important to ascertain that lens movements requested by the AF driver during the second scan can be at least reasonably approximated by the lens actuator. The quality of the approximation may depend on how well the physical limitations of the actuator are accounted for in the source code and /or configuration of the AFM driver.		
10 0x00A	15:0	0x01C0	**wakeUpLine (RW)**
	Number of image row at which the MCU wakes up to execute AF driver code. When the function AF_SetSize resizes the 4 x 4 array of AF windows according to new values of af.windowPos and af.windowSize, it automatically makes af.wakeUpLine equal to the number of the second row below the bottom of the array. If af.windowPos and af.windowSize are such that the bottom of the array is outside the frame, the value given to af.wakeUpLine by AF_SetSize is greater than frame height, i.e. invalid. It must be changed to something less than the frame height, otherwise AF will not work.		
12 0x00C	15:0	0xFFFF	**zoneWeights [HI] (RW)**
	Upper word of double-word (64-bit) variable af.zoneWeights holding the weights of all 16 AF windows or zones. Bits [1:0] of this variable represent the weight of window W11, bits [3:2] the weight of W12, and so on to bits [31:30] that represent the weight of W44. Since each weight is represented by just 2 bits, it is allowed to have only 4 values, 0, 1/3, 2/3 or 1. Value stored in each 2 bits equals window weight times 3, so the value of 1 signifies the weight of 1/3, 2 stands for 2/3, and 3 for 1.		
14 0x00E	15:0	0xFFFF	**zoneWeights [LO] (RW)**
	Lower word of double-word (64-bit) variable af.zoneWeights holding the weights of all 16 AF windows or zones. Bits [1:0] of this variable represent the weight of window W11, bits [3:2] the weight of W12, and so on to bits [31:30] that represent the weight of W44. Since each weight is represented by just 2 bits, it is allowed to have only 4 values, 0, 1/3, 2/3 or 1. Value stored in each 2 bits equals window weight times 3, so the value of 1 signifies the weight of 1/3, 2 stands for 2/3, and 3 for 1.		
16 0x010	15:0	0x00FF	**distanceWeight (RW)**
	Reserved.		

Table 33: 5: Auto Focus Variables (continued)

Reg. #	Bits	Default	Name
17 0x011	15:0	0x0000	**bestPosition (RW)**
	This variable is used in 3 different ways depending on values of bits 6 and 7 of af.mode. When bit 7 equals 1 (in manual lens control mode), the position of AF lens can be changed by changing the value of af.bestPosition, which is interpreted by the AF driver as logical lens position desired by its user. The AF driver reads af.bestPosition once every frame, and if it differs from current logical lens position (afm.curPos), the AF driver gives the AFM driver a command to make these variables equal by moving the lens. Physical movement of the lens corre-sponding to the change of afm.curPos to af.bestPosition always takes some time, during which it is best not to change the value of af.bestPosition to avoid possible errors. When af.mode bits [7:6] are both 0, af.bestPosition serves to store AF algo-rithm output rather than user input. After both first and second scan, the AF algorithm outputs to this variable the offset of programmable logical lens po-sition found to be best relative to the start position of the scan. In other words, after each scan, the best lens position found is af.positions[af.initPos+af.bestPosition]. When af.mode bit 7 is 0 and bit 6 is 1 (creep compensation mode is enabled) af.bestPosition is used during lens re-positioning triggered by setting bit 1 of af.mode to 1. It is used to store the desired final lens position, which is assumed to be the same as afm.curPos before the re-positioning. As a result, after every succesfull re-positioning, af.bestPosition equals afm.curPos.		
18 0x012	15:0	0x000A	**shaTH (RW)**
	Sharpness score variability threshold. Only AF windows whose min. and max. normalized sharpness scores satisfy the condition 1-(min.score/max.score) ? af.shaTH/256 are used to select best focus position.		
19 0x013	15:0	0x0000	**modeEx2 (RW)**
	15:1	X	Reserved
	0	0x0000	automatic zone weighting mode
20 0x014	15:0	0x0000	**Position 0 (RW)**
	Programmable logical lens position 0		
21 0x015	15:0	0x001C	**Position 1 (RW)**
	Programmable logical lens position 1		
22 0x016	15:0	0x0038	**Position 2 (RW)**
	Programmable logical lens position 2		
23 0x017	15:0	0x0055	**Position 3 (RW)**
	Programmable logical lens position 3		
24 0x018	15:0	0x0071	**Position 4 (RW)**
	Programmable logical lens position 4		
25 0x019	15:0	0x008D	**Position 5 (RW)**
	Programmable logical lens position 5		
26 0x01A	15:0	0x00AA	**Position 6 (RW)**
	Programmable logical lens position 6		

Table 33: 5: Auto Focus Variables (continued)

Reg. #	Bits	Default	Name
27 0x01B	15:0	0x00C6	**Position 7 (RW)**
	Programmable logical lens position 7		
28 0x01C	15:0	0x00E2	**Position 8 (RW)**
	Programmable logical lens position 8		
29 0x01D	15:0	0x00FF	**Position 9 (RW)**
	Programmable logical lens position 9		
30 0x01E	15:0	0x001B	**Position 10 (RW)**
	Programmable logical lens position 10		
31 0x01F	15:0	0x0037	**Position 11 (RW)**
	Programmable logical lens position 11		
32 0x020	15:0	0x0054	**Position 12 (RW)**
	Programmable logical lens position 12		
33 0x021	15:0	0x0070	**Position 13 (RW)**
	Programmable logical lens position 13		
34 0x022	15:0	0x008C	**Position 14 (RW)**
	Programmable logical lens position 14		
35 0x023	15:0	0x00A9	**Position 15 (RW)**
	Programmable logical lens position 15		
36 0x024	15:0	0x00C5	**Position 16 (RW)**
	Programmable logical lens position 16		
37 0x025	15:0	0x00E1	**Position 17 (RW)**
	Programmable logical lens position 17		
38 0x026	15:0	0x00FE	**Position 18 (RW)**
	Programmable logical lens position 18		
39 0x027	15:0	0x001A	**Position 19 (RW)**
	Programmable logical lens position 19		

Table 34: 6: AFM Variables

Reg. #	Bits	Default	Name
0 0x000	15:0	0xE83E	**vmt (RW)**
	Pointer to the driver's virtual method table (VMT). The AFM driver includes a separate set of control methods for each supported type of auto focus mechanism (see afm.type below). Each set of methods is acessible via a separate VMT. Pointing afm.vmt to one of those VMTs either enables the AF driver to control the corresponding type of AF mechanism via the GPIO or takes away the control of the GPIO from the AF driver.		
2 0x002	15:0	0x0000	**type (RW)**
	Type of AF mechanism (lens actuator) used: 0–none, 1–helimorph, 2–stepper motor, 3–any actuator controlled by an AD5398 DAC IC. At sequencer initialization, this variable is set to 0 and the afm.vmt is pointed to the default VMT of the AFM driver, which makes the GPIO inaccessible to the AF driver. Enabling the AF driver to control a lens actuator via the GPIO involves 2 steps. First, afm.type must be set to t+128, where t is 1, 2 or 3. Second, the sequencer must be given REFRESH command by setting seq.cmd to 5. The nonzero 7th bit in afm.type forces the sequencer to call AFM_Init function upon that command. The function makes afm.type equal to t and points afm.vmt to the VMT through which the AFM driver methods for controlling actuator type t can be called.		
3 0x003	15:0	0x0000	**curPos (RW)**
	Current logical position of AF lens.		
4 0x004	15:0	0x0000	**prePos (RW)**
	Previous logical position of AF lens		
5 0x005	15:0	0x0000	**status (RW)**
	15:5	X	Reserved
	4:3	0x0000	Physical position modulo 4 Physical position modulo 4. Used only when type = 2
	2	0x0000	Direction of last physical move 0: forward (+); 1: backward (-)
	1	0x0000	Motion indicator 0: AF lens actuator is stationary; 1: AF lens actuator is moving
	0	0x0000	Error indicator
	Lens actuator status: Bit 0 — 0 if all is OK, 1 if the actuator reported an error, Bit 1 — 0 if the lens is stationary, 1 if it is moving, Bit 2 — 0 if direction of last lens movement was forward (+), 1 if the direction was backward (-), Bits [4:3] — number of current stepper motor position if afm.type=2; otherwise unused, Bits [7:5] — unused.After sending a command to change lens position to the lens actuator, the AFM driver sets bit 1 of afm.status to 1. The value of the bit remains 1 until the AFM driver gets information that the lens is not moving (either has stopped at the desired new position or has failed to reach it). The AF driver waits through all times when the lens is moving by having afm.status updated once every frame, reading its bit 1 and immediately going back to sleep if it equals 1.		
6 0x006	15:0	0x0000	**posMin (RW)**
	Lower limit of physical motion range of AF lens		

Table 34: 6: AFM Variables (continued)

Reg. #	Bits	Default	Name
7 0x007	15:0	0x0000	posMax (RW)
	Upper limit of physical motion range of AF lens. Can be set below afm.posMin to swap forward (+) and backward (-) directions of lens motion.		
8 0x008	15:0	0x0000	posMacro (RW)
	Logical macro position of AF lens		
9 0x009	15:0	0x0000	backlash (RW)
	Logical size of backlash-compensating step that the AF driver can optionally use in lens positioning after the first scan. If bits [7:6] of af.mode are set to 0, this positioning is done by moving the lens directly from the end position of the scan, af.positions[af.initPos+af.numSteps-1] to the logical position found best, af.positions[af.initPos+af.bestPosition]. The direction of this move is opposite to the direction of the scan, and therefore the move may not bring the lens to its intended physical destination, unless its logical length is adjusted upward to compensate for lens actuator backlash. To make this adjustment, the AF driver subtracts afm.backlash from the value of af.positions[af.initPos+af.bestPosition] and gives the result to the AFM driver as the logical position to move the lens to. Negative results of the subtraction are replaced with 0. Note that subtracting afm.backlash makes sense only when af.positions[af.initPos+af.numSteps-1] > af.positions[af.initPos+af.bestPosition]; otherwise addition is required. Since the AF driver always subtracts afm.backlash, backlash compensation using this variable is not recommended after scans done in the negative direction (e.g. from logical position 255 to logical position 0).		
10 0x00A	15:0	0x0000	custCtrl (RW)
	Custom controls: 1-bit option switches and fine-tuning parameters for actuator control methods. The function of different bits of this variable depends on the current value of afm.type. If afm.type = 1, then: Bit 0 — selects the length of commands sent to HD80 helimorph driver by function AFM_SetPosHelimorph (0 – 2 bytes, 1 – 3 bytes including enable byte), Bit 1 — selects one of 2 possible relations between the argument of the function AFM_SetPosHelimorph, bPos, and position byte sent to HD80 helimorph driver (0 means send bPos, 1 – send 255-bPos, to reverse the direction of lens movement), Bit 2 — selects one of 2 positions that helimorph can assume upon command to exit standby (0 – afm.posMin, 1 – afm.posMax), Bits [7:3] — unused. If afm.type = 2, then: Bit 0 — selects direction of lens motion if bit 1 is set to 1, Bit 1 — determines how function AFM_SetPosStMotor interprets its 1-byte argument (0 – as desired logical lens position, 1 – as number of physical steps to make in the direction indicated by bit 0), Bit 2 — enables periodic forcing of stepper-motor-driving outputs into calculated logical states (0 – forcing disabled, outputs are only toggled as needed, 1 – forcing enabled), Bit 3 — when set to 1, enables powering stepper motor down after every movement (the motor is always powered up before movements, but powering down is optional), Bit 4 — enables repositioning of stepper motor by function AFM_ResetStMotor upon command to enter standby (1 – enable, 0 – disable), Bit 5 — enables repositioning of stepper motor by function AFM_ResetStMotor upon command to exit standby (1 – enable, 0 – disable), Bits [7:6] — allow one to slow down initial portions of stepper-motor-driving waveforms that cannot be entirely generated by MT9D111 waveform generator (higher value = slower waveforms). If afm.type = 0, afm.custCtrl is unused.		

Table 34: 6: AFM Variables (continued)

Reg. #	Bits	Default	Name
11 0x00B	15:0	0xE8EB	**timer_vmt (RW)**
	Pointer to timer VMT. Default timer VMT located in ROM contains pointers to the following public functions: AFM_Wait, AFM_TimerSetDelay, AFM_TimerSetTimeToMove, AFM_TimerIsStopped. The pointers are all of type void* and have the following names: pWait, pSetDelay, pSetTimeToMove, pTimerIsStopped.		
13 0x00D	15:0	0x0000	**timer_startTime (RW)**
	Timer start time.		
15 0x00F	15:0	0x0000	**timer_stopTime (RW)**
	Timer stop time		
17 0x011	15:0	0x0000	**timer_hiWordMclkFreq (RW)**
	Master clock frequency in Hz divided by 65536. Used to convert delay times in milliseconds (for example, the values of afm.timer.maxShortDelay and afm.timer.maxLongDelay) to corresponding counts of master clock cycles that can be programmed into the timer.		
19 0x013	15:0	0x0000	**timer_maxShortDelay (RW)**
	Maximum expected duration of a short lens move. Should be given in milliseconds. Used by the AFM driver function AFM_TimerSetTimeToMove to compute lens travel time estimates.		
21 0x015	15:0	0x0000	**timer_maxLongDelay (RW)**
	Maximum expected duration of a long lens move. Should be given in milliseconds. Used by the AFM driver function AFM_TimerSetTimeToMove to compute lens travel time estimates.		
23 0x017	15:0	0x0000	**timer_maxQuickMove (RW)**
	Maximum length of short lens move (or threshold between short and long moves). Used by the AFM driver function AFM_TimerSetTimeToMove to compute lens travel time estimates.		

Table 34: 6: AFM Variables (continued)

Reg. #	Bits	Default	Name
24 0x018	**15:0**	**0x0000**	**timer_config (RW)**
	15:2	X	Reserved
	1	0x0000	Delay used when curPos = prePos
	0	0x0000	Motion time estimation method
	Bits [1:0] of this variable determine how afm.timer.maxShortDelay, afm.timer.maxLongDelay, and afm.timer.maxQuickMove are used to estimate duration of lens movements. Bits [7:2] are unused.If a command-driven lens actuator does not provide any feedback about its status after receiving a command to move an AF lens, the AFM driver must somehow predict how long the lens will be moving, to prevent the AF driver from collecting sharpness scores and issuing new commands during its movement. The need for predictions of lens travel time is satisfied rather inexpensively by the AFM driver function AFM_TimerSetTimeToMove, which takes as arguments 2 logical lens positions and estimates the time required to move the lens between them. The function can use 2 different estimation methods, both of which rely on 3 user-set parameters, afm.timer.maxShortDelay, afm.timer.maxLongDelay, and afm.timer.maxQuickMove, as a sole source of information about how fast the lens actuator moves the lens. The default method of piecewise linear estimation is used when bit 0 of afm.timer.config is cleared. Setting this bit to 1 enables the alternative bipolar method. The bipolar method is very simple: if the distance between the 2 logical positions given to AFM_TimerSetTimeToMove as arguments exceeds afm.timer.maxQuickMove, then afm.timer.maxLongDelay is selected as the proper lens travel time estimate. Otherwise, unless the 2 logical positions are the same, the estimate equals afm.timer.maxShortDelay. If the 2 positions are the same, the estimate should be 0, and indeed is 0 if bit 1 of afm.timer.config is cleared. However, if this bit is set to 1 and the positions are the same, the function AFM_TimerSetTimeToMove outputs afm.timer.maxShortDelay instead of 0.Please see page 37 for a description of the piecewise linear estimation method.		
25 0x019	**15:0**	**0xE8F3**	**si_vmt (RW)**
	Pointer to serial interface VMT. Default serial interface VMT located in ROM contains pointers to the following public functions: AFM_SiSendCmd, AFM_SiSetActvFlag, AFM_SiSendByte, AFM_SiRecvByte.The pointers are all of type void* and have the following names: pSendCmd, pSetActvFlag, pSendByte, pRecvByte.		
27 0x01B	**15:0**	**0x0000**	**si_clkMask (RW)**
	Mask selecting one of GPIO pads as the clock line of dedicated two-wire serial interface between the MT9D111 and a lens actuator (for example, helimorph).		
29 0x01D	**15:0**	**0x0000**	**si_dataMask (RW)**
	Mask selecting one of GPIO pads as the data line of the dedicated two-wire serial interface to a lens actuator.		
31 0x01F	**15:0**	**0x0000**	**si_clkQtrPrd (RW)**
	Delay for slowing down serial interface transmissions. The period of serial interface clock is asymptotically proportional to si.clkQtrPrd.		
33 0x021	**15:0**	**0x0000**	**si_needsAck (RW)**
	Switch enabling detection of ACK bits from the lens actuator (0-detectiondisabled, 1-enabled).		

Table 34: 6: AFM Variables (continued)

Reg. #	Bits	Default	Name
34 0x022	15:0	0x0000	**si_slaveAddr (RW)**
	Lens actuator address used in serial interface transmissions.		
35 0x023	15:0	0x0000	**sm_enabMask (RW)**
	Mask selecting one of the GPIO pads as stepper-motor-enabling output.		
37 0x025	15:0	0x0000	**sm_drv0Mask (RW)**
	Mask selecting one of the GPIO pads as first stepper-motor-driving output.		
38 0x026	15:0	0x0000	**sm_drv1Mask (RW)**
	Mask selecting one of the GPIO pads as second stepper-motor-driving output.		
39 0x027	15:0	0x0000	**sm_drv2Mask (RW)**
	Mask selecting one of the GPIO pads as third stepper-motor-driving output.		
40 0x028	15:0	0x0000	**sm_drv3Mask (RW)**
	Mask selecting one of the GPIO pads as fourth stepper-motor-driving output.		
41 0x029	15:0	0x0000	**sm_drvsQtrPrd (RW)**
	Delay for lengthening the period of stepper motor driving waveforms. The number of master/GPIO clock cycles in this period asymptotically approaches 8 times the sm.drvsQtrPrd.		
43 0x02B	15:0	0x0000	**sm_drvsGenMode (RW)**
	15:8	X	Reserved
	7:4	0x0000	Minimum GPIO clock divider setting
	3	0x0000	GPIO clock divider
	2	0x0000	GPIO counter mode
	1	0x0000	Fine stepper motor positioning
	0	0x0000	Finest stepper motor positioning
	This variable tells the AFM driver how to program the GPIO to generate stepper motor driving waveforms. Bits [1:0] — size of smallest stepper motor move (0 – 4 steps, 1 – 1 step, 2 – 2 steps), Bit 2 — if 0, use the waveform generator in 8-bit counter mode, if 1, use it in 16-bit counter mode, Bit 3 — if 0, use clock divider 1, if 1, use clock divider 2,Bits [7:4] — clock divider setting (used by the AFM driver only if it is higher than the setting the driver has automatically calculated).		
44 0x02C	15:0	0x0000	**sm_piEnabMask (RW)**
	Mask selecting one of GPIO pads as photointerrupter-enabling output.		
46 0x02E	15:0	0x0000	**sm_piOutMask (RW)**
	Mask selecting one of GPIO pads as photointerrupter-sensing input.		
48 0x030	15:0	0x0000	**sm_piEdgeOffset (RW)**
	Distance (in units of smallest stepper motor move) between the position of photointerrupter signal edge and desired initial position of stepper motor.		

Table 34: 6: AFM Variables (continued)

Reg. #	Bits	Default	Name
49 0x031	**15:0**	**0x0000**	**sm_piConfig (RW)**
	15:8	X	Reserved
	7:6	0x0000	Expected PI settling time Expected PI settling time
	5	0x0000	Max. step size forcing Max. step size forcing: 0: disabled - PI edge is approached in 1-, 2-, or 4-step jumps, depending on bits 0 and 1 of afm.sm.drvsGenMode. 1: enabled - PI edge is approached in 4-step jumps, irrespective of the value of bits 0 and 1 of afm.sm.drvsGenMode.
	4	0x0000	Logical value assigned to initial pos. Logical value assigned to initial position (0: 0; 1: 255)
	3	0x0000	PI state at initial stepper position PI state at initial stepper position (0: PI inactive state; 1: PI active state)
	2	0x0000	PI active state PI active state (0: logical low; 1: logical high)
	1	0x0000	Location of PI signal edge Location of PI signal edge (0: near logical position 0; 1: near logical position 255)
	0	0x0000	PI use in stepper motor positioning PI use in stepper motor positioning(1: use PI; 0: do not use PI move motor to logical position indicated by bit 1)
	Photointerrupter (PI) configuration: Bit 0 — if 0, do not use PI, if 1, use it for initial stepper positioning, Bit 1 — if 0, PI signal edge is near logical position 0, if 1 – near 255, Bit2 — PI active state (0–low, 1–high), Bit 3 — if 1, PI is in the active state at initial stepper position, if 0 – it is not, Bit 4 — if 0, initial logical stepper position is 0, if 1 – 255, Bit 5 — if 0, PI edge is approached in 1-, 2-, or 4-step jumps, depending on bits 0 and 1 of afm.sm.drvsGenMode, if 1, PI edge is approached in 4-step jumps, irrespective of the value of afm.sm.drvsGenMode, Bits [7:6] — delay between powering up the PI and sensing its output the first time.		

7: Mode Variables

Table 35: 7: Mode Variables

Reg. #	Bits	Default	Name
0 0x000	**15:0**	**0xE86E**	**vmt (RW)**
	Pointer to virtual method table.		
2 0x002	**15:0**	**0x0000**	**context (RW)**
	Current context (0 = A, 1 = B).		
3 0x003	**15:0**	**0x0320**	**Output Width A (RW)**
	Output size of final image for context A. Must be equal to or smaller than crop dimension. Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
5 0x005	15:0	0x0258	**Output Height A (RW)**
	Output size of final image for context A. Must be equal to or smaller than crop dimension. Changes take effect only after REFRESH command.		
7 0x007	15:0	0x0640	**Output Width B (RW)**
	Output size of final image for context B. Must be equal to or smaller than crop dimension. Changes take effect only after REFRESH command.		
9 0x009	15:0	0x04B0	**Output Height B (RW)**
	Output size of final image for context B. Must be equal to or smaller than crop dimension. Changes take effect only after REFRESH command.		
11 0x00B	15:0	0x00C8	**PLL_Lock_Delay (RW)**
13 0x00D	15:0	0x0000	**sensor_row_start_A (RW)**
	context A shadow register R0x3002. Changes take effect only after REFRESH_MODE command.		
15 0x00F	15:0	0x0000	**sensor_col_start_A (RW)**
	context A shadow register R0x3004. Changes take effect only after REFRESH_MODE command.		
17 0x011	15:0	0x04BD	**sensor_row_end_A (RW)**
	context A shadow register R0x3006. Changes take effect only after REFRESH_MODE command.		
19 0x013	15:0	0x064D	**sensor_col_end_A (RW)**
	context A shadow register R0x3008. Changes take effect only after REFRESH_MODE command.		
21 0x015	15:0	0x0000	**sensor_x_delay_A (RW)**
	context A shadow register R0x3018. Changes take effect only after REFRESH_MODE command.		
23 0x017	15:0	0x2112	**sensor_row_speed_A (RW)**
	15:13	0x0001	CB Delay Reserved. Do not change from default value.
	12:10	X	Reserved
	9:8	0x0001	Row Seq Scale Reserved. Do not change from default value.
	7:4	0x0001	Pixel 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.
	3	0x0000	Reserved Reserved
	2:0	0x0002	Pixel clock speed in mclk periods A programmed value of N gives a pixel clock period of N system clocks. A value of 0 is illegal: it causes the clock to stop.
	context A shadow register R0x3016. Changes take effect only after REFRESH_MODE command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
	15:0	**0x046C**	**Read Mode A (RW)**
	15:14	0x0000	Special LINE_VALID 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
	13:12	X	Reserved
	11	0x0000	x bin enable Enable analogue binning in X (column) direction. When set, x_odd_inc must be set to 3 and y_odd_inc must be set to 1.
	10	0x0001	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.
	9:8	X	Reserved
	7:5	0x0003	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.
	4:2	0x0003	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.
	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 Reg0x3024). The bit-order of bits [1:0] match the order in Reg0x301C but is reversed relative to earlier Micron Imaging sensors.
25 **0x019**	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 Reg0x3024).
context A shadow register R0x20:0. Changes take effect only after REFRESH_MODE command.			
27 **0x01B**	**15:0**	**0x03D8**	**sensor_sample_time_pck_A (RW)**
context A shadow register R07:0. Changes take effect only after REFRESH_MODE command.			
29 **0x01D**	**15:0**	**0x0155**	**sensor_fine_correction_A (RW)**
context A shadow register R08:0. Changes take effect only after REFRESH_MODE command.			
31 **0x01F**	**15:0**	**0x0279**	**sensor_fine_IT_min_A (RW)**
context A minimal fine integration time. Changes take effect only after REFRESH_MODE command.			
33 **0x021**	**15:0**	**0x0155**	**sensor_fine_IT_max_margin_A (RW)**
context A maximal fine integration time margin. Changes take effect only after REFRESH_MODE command.			

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
35 0x023	15:0	0x0293	**sensor_frameLengthLines_A (RW)**
	context A shadow register R05:0. Changes take effect only after REFRESH_MODE command.		
37 0x025	15:0	0x0AAC	**sensor_lineLengthPck:_A (RW)**
	context A shadow register R06:0. Changes take effect only after REFRESH_MODE command.		
39 0x027	15:0	0x1010	**sensor_dac_id_4_5_A (RW)**
	context A shadow register R0x42:0. Changes take effect only after REFRESH_MODE command.		
41 0x029	15:0	0x2010	**sensor_dac_id_6_7_A (RW)**
	context A shadow register R0x43:0. Changes take effect only after REFRESH_MODE command.		
43 0x02B	15:0	0x1010	**sensor_dac_id_8_9_A (RW)**
	context A shadow register R0x44:0. Changes take effect only after REFRESH_MODE command.		
45 0x02D	15:0	0x1006	**sensor_dac_id_10_11_A (RW)**
	context A shadow register R0x45:0. Changes take effect only after REFRESH_MODE command.		
47 0x02F	15:0	0x0004	**sensor_row_start_B (RW)**
	context B shadow register R01:0. Changes take effect only after REFRESH_MODE command.		
49 0x031	15:0	0x0004	**sensor_col_start_B (RW)**
	context B shadow register R02:0. Changes take effect only after REFRESH_MODE command.		
51 0x033	15:0	0x04BB	**sensor_row_end_B (RW)**
	context B shadow register R03:0. Changes take effect only after REFRESH_MODE command.		
53 0x035	15:0	0x064B	**sensor_col_end_B (RW)**
	context B shadow register R04:0. Changes take effect only after REFRESH_MODE command.		
55 0x037	15:0	0x0000	**sensor_x_delay_B (RW)**
	context B shadow register R0x0C:0. Changes take effect only after REFRESH_MODE command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
57 0x039	**15:0**	**0x2111**	**sensor_row_speed_B (RW)**
	15:13	0x0001	CB Delay Reserved. Do not change from default value.
	12:10	X	Reserved
	9:8	0x0001	Row Seq Scale Reserved. Do not change from default value.
	7:4	0x0001	Pixel 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.
	3	0x0000	Reserved
	2:0	0x0001	Pixel clock speed in mclk periods A programmed value of N gives a pixel clock period of N system clocks. A value of 0 is illegal: it causes the clock to stop.
	context B shadow register R0x0B:0. Changes take effect only after REFRESH_MODE command.		
	15:0	**0x0024**	**Read Mode B (RW)**
	15:14	0x0000	Special LINE_VALID 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
	13:12	X	Reserved
	11	0x0000	x bin enable Enable analogue binning in X (column) direction. When set, x_odd_inc must be set to 3 and y_odd_inc must be set to 1.
	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.
	9:8	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.
	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.
59 0x03B	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 Reg0x3024). The bit-order of bits [1:0] match the order in Reg0x301C but is reversed relative to earlier Micron Imaging sensors.

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
	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 Reg0x3024).
	context B shadow register R0x20:0. Changes take effect only after REFRESH_MODE command.		
61 0x03D	**15:0**	**0x01EC**	**sensor_sample_time_pck_B (RW)**
	context B shadow register R07:0. Changes take effect only after REFRESH_MODE command.		
63 0x03F	**15:0**	**0x00A4**	**sensor_fine_correction_B (RW)**
	context B shadow register R08:0. Changes take effect only after REFRESH_MODE command.		
65 0x041	**15:0**	**0x014A**	**sensor_fine_IT_min_B (RW)**
	context B minimal fine integration time. Changes take effect only after REFRESH_MODE command.		
67 0x043	**15:0**	**0x00A4**	**sensor_fine_IT_max_margin_B (RW)**
	context B maximal fine integration time margin. Changes take effect only after REFRESH_MODE command.		
69 0x045	**15:0**	**0x04ED**	**sensor_frameLengthLines_B (RW)**
	context B shadow register R05:0. Changes take effect only after REFRESH_MODE command.		
71 0x047	**15:0**	**0x0AAC**	**sensor_lineLengthPck:_B (RW)**
	context B shadow register R06:0. Changes take effect only after REFRESH_MODE command.		
73 0x049	**15:0**	**0x2020**	**sensor_dac_id_4_5_B (RW)**
	context B shadow register R0x42:0. Changes take effect only after REFRESH_MODE command.		
75 0x04B	**15:0**	**0x2020**	**sensor_dac_id_6_7_B (RW)**
	context B shadow register R0x43:0. Changes take effect only after REFRESH_MODE command.		
77 0x04D	**15:0**	**0x1020**	**sensor_dac_id_8_9_B (RW)**
	context B shadow register R0x44:0. Changes take effect only after REFRESH_MODE command.		
79 0x04F	**15:0**	**0x2006**	**sensor_dac_id_10_11_B (RW)**
	context B shadow register R0x45:0. Changes take effect only after REFRESH_MODE command.		
81 0x051	**15:0**	**0x0000**	**crop_X0_A (RW)**
	Lower-x scaler zoom window (context A shadow register). Changes take effect only after REFRESH command.		
83 0x053	**15:0**	**0x0320**	**crop_X1_A (RW)**
	Upper-x scaler zoom window (context A shadow register). Changes take effect only after REFRESH command.		
85 0x055	**15:0**	**0x0000**	**crop_Y0_A (RW)**
	Lower-y scaler zoom window (context A shadow register). Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
87 0x057	**15:0**	**0x0258**	**crop_Y1_A (RW)**
	Upper-y scaler zoom window (context A shadow register). Changes take effect only after REFRESH command.		
89 0x059	**15:0**	**0x0000**	**dec_ctrl_A (RW)**
	15:7	X	Reserved
	6	0x0000	Completion for row Completion for row
	5	0x0000	Completion for column Completion for column
	4	0x0000	Reserved(not used)
	3	0x0000	Output Width has no scaling Pixel enable is always "1". Output width will always be as with no scaling. Crop will still be in effect.
	2	0x0000	High precision mode Additional bits for scaling result are stored. Only for scaling >2
	1	0x0000	U_first U_first. Color signal stored in memory first [U or V] [for 4:2:2 internal format]
	0	0x0000	Reserved (not used) Reserved (not used)
	Scaler control register (context A shadow register). Changes take effect only after REFRESH command.		
91 0x05B	**15:0**	**0x0800**	**width_ratio_A (RW)**
	Scaler width ratio (context A shadow register). Temporary storage for calculated value.		
93 0x05D	**15:0**	**0x0800**	**height_ratio_A (RW)**
	Scaler height ratio (context A shadow register). Temporary storage for calculated value.		
95 0x05F	**15:0**	**0x0000**	**crop_X0_B (RW)**
	Lower-x scaler zoom window (context B shadow register). Changes take effect only after REFRESH command.		
97 0x061	**15:0**	**0x0640**	**crop_X1_B (RW)**
	Upper-x scaler zoom window (context B shadow register). Changes take effect only after REFRESH command.		
99 0x063	**15:0**	**0x0000**	**crop_Y0_B (RW)**
	Lower-y scaler zoom window (context B shadow register). Changes take effect only after REFRESH command.		
101 0x065	**15:0**	**0x04B0**	**crop_Y1_B (RW)**
	Upper-y scaler zoom window (context B shadow register). Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
103 0x067	**15:0**	**0x0000**	**dec_ctrl_B (RW)**
	15:7	X	Reserved
	6	0x0000	Completion for row Completion for row
	5	0x0000	Completion for column Completion for column
	4	0x0000	Reserved(not used)
	3	0x0000	Output Width has no scaling Pixel enable is always "1". Output width will always be as with no scaling. Crop will still be in effect.
	2	0x0000	High precision mode Additional bits for scaling result are stored. Only for scaling >2
	1	0x0000	U_first U_first. Color signal stored in memory first [U or V] [for 4:2:2 internal format]
	0	0x0000	Reserved (not used) Reserved (not used)
	Scaler control register (context B shadow register). Changes take effect only after REFRESH command.		
105 0x069	**15:0**	**0x0800**	**width_ratio_B (RW)**
	Scaler width ratio (context B shadow register). Temporary storage for calculated value.		
107 0x06B	**15:0**	**0x0800**	**height_ratio_B (RW)**
	Scaler height ratio (context B shadow register). Temporary storage for calculated value.		
109 0x06D	**15:0**	**0x0042**	**gam_cont_A (RW)**
	15:7	X	Reserved
	6:4	0x0004	Contrast Setting Contrast Setting: 0—no contrast increase (slope = 100%) 1—some contrast increase (slope = 125%) 2—more contrast increase (slope = 150%) 3—most contrast increase (slope = 175%) 4—noise-reduction contrast
	3	X	Reserved
	2:0	0x0002	Gamma Setting Gamma Setting: 0—gamma = 1.0 1—gamma = 0.56 2—gamma = 0.45 3—use user-defined gamma table
	Gamma and contrast settings (context A). Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
110 0x06E	**15:0**	**0x0042**	**gam_cont_B (RW)**
	15:7	X	Reserved
	6:4	0x0004	Contrast Setting Contrast Setting: 0—no contrast increase (slope = 100%) 1—some contrast increase (slope = 125%) 2—more contrast increase (slope = 150%) 3—most contrast increase (slope = 175%) 4—noise-reduction contrast
	3	X	Reserved
	2:0	0x0002	Gamma Setting Gamma Setting: 0—gamma = 1.0 1—gamma = 0.56 2—gamma = 0.45 3—use user-defined gamma table
	Gamma and contrast settings (context B). Changes take effect only after REFRESH command.		
111 0x06F	**15:0**	**0x0000**	**gam_table_A_0 (RW)**
	User-defined gamma table values (context A shadow register).		
112 0x070	**15:0**	**0x0027**	**gam_table_A_1 (RW)**
	User-defined gamma table values (context A shadow register).		
113 0x071	**15:0**	**0x0035**	**gam_table_A_2 (RW)**
	User-defined gamma table values (context A shadow register).		
114 0x072	**15:0**	**0x0048**	**gam_table_A_3 (RW)**
	User-defined gamma table values (context A shadow register).		
115 0x073	**15:0**	**0x0063**	**gam_table_A_4 (RW)**
	User-defined gamma table values (context A shadow register).		
116 0x074	**15:0**	**0x0077**	**gam_table_A_5 (RW)**
	User-defined gamma table values (context A shadow register).		
117 0x075	**15:0**	**0x0088**	**gam_table_A_6 (RW)**
	User-defined gamma table values (context A shadow register).		
118 0x076	**15:0**	**0x0096**	**gam_table_A_7 (RW)**
	User-defined gamma table values (context A shadow register).		
119 0x077	**15:0**	**0x00A3**	**gam_table_A_8 (RW)**
	User-defined gamma table values (context A shadow register).		
120 0x078	**15:0**	**0x00AF**	**gam_table_A_9 (RW)**
	User-defined gamma table values (context A shadow register).		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
121 0x079	15:0	0x00BA	gam_table_A_10 (RW)
	User-defined gamma table values (context A shadow register).		
122 0x07A	15:0	0x00C4	gam_table_A_11 (RW)
	User-defined gamma table values (context A shadow register).		
123 0x07B	15:0	0x00CE	gam_table_A_12 (RW)
	User-defined gamma table values (context A shadow register).		
124 0x07C	15:0	0x00D7	gam_table_A_13 (RW)
	User-defined gamma table values (context A shadow register).		
125 0x07D	15:0	0x00E0	gam_table_A_14 (RW)
	User-defined gamma table values (context A shadow register).		
126 0x07E	15:0	0x00E8	gam_table_A_15 (RW)
	User-defined gamma table values (context A shadow register).		
127 0x07F	15:0	0x00F0	gam_table_A_16 (RW)
	User-defined gamma table values (context A shadow register).		
128 0x080	15:0	0x00F8	gam_table_A_17 (RW)
	User-defined gamma table values (context A shadow register).		
129 0x081	15:0	0x00FF	gam_table_A_18 (RW)
	User-defined gamma table values (context A shadow register).		
130 0x082	15:0	0x0000	gam_table_B_0 (RW)
	User-defined gamma table values (context B shadow register).		
131 0x083	15:0	0x0027	gam_table_B_1 (RW)
	User-defined gamma table values (context B shadow register).		
132 0x084	15:0	0x0035	gam_table_B_2 (RW)
	User-defined gamma table values (context B shadow register).		
133 0x085	15:0	0x0048	gam_table_B_3 (RW)
	User-defined gamma table values (context B shadow register).		
134 0x086	15:0	0x0063	gam_table_B_4 (RW)
	User-defined gamma table values (context B shadow register).		
135 0x087	15:0	0x0077	gam_table_B_5 (RW)
	User-defined gamma table values (context B shadow register).		
136 0x088	15:0	0x0088	gam_table_B_6 (RW)
	User-defined gamma table values (context B shadow register).		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
137 0x089	15:0	0x0096	gam_table_B_7 (RW)
	User-defined gamma table values (context B shadow register).		
138 0x08A	15:0	0x00A3	gam_table_B_8 (RW)
	User-defined gamma table values (context B shadow register).		
139 0x08B	15:0	0x00AF	gam_table_B_9 (RW)
	User-defined gamma table values (context B shadow register).		
140 0x08C	15:0	0x00BA	gam_table_B_10 (RW)
	User-defined gamma table values (context B shadow register).		
141 0x08D	15:0	0x00C4	gam_table_B_11 (RW)
	User-defined gamma table values (context B shadow register).		
142 0x08E	15:0	0x00CE	gam_table_B_12 (RW)
	User-defined gamma table values (context B shadow register).		
143 0x08F	15:0	0x00D7	gam_table_B_13 (RW)
	User-defined gamma table values (context B shadow register).		
144 0x090	15:0	0x00E0	gam_table_B_14 (RW)
	User-defined gamma table values (context B shadow register).		
145 0x091	15:0	0x00E8	gam_table_B_15 (RW)
	User-defined gamma table values (context B shadow register).		
146 0x092	15:0	0x00F0	gam_table_B_16 (RW)
	User-defined gamma table values (context B shadow register).		
147 0x093	15:0	0x00F8	gam_table_B_17 (RW)
	User-defined gamma table values (context B shadow register).		
148 0x094	15:0	0x00FF	gam_table_B_18 (RW)
	User-defined gamma table values (context B shadow register).		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
149 0x095	**15:0**	**0x0000**	**output_format_A (RW)**
	15:9	X	Reserved
	8	0x0000	Turn on processed Bayer mode Chip output data in Bayer fomat. As a result datarate decreses in two times.
	7:6	0x0000	RGB output format RGB output format: 00 = 16-bit RGB565 01 = 15-bit RGB555 10 = 12-bit RGB444x 11 = 12-bit RGBx444
	5	0x0000	RGB/YUV output RGB/YUV output 1=output RGB (see R0x332E [7:6]) 0=output YUV
	4	0x0000	Use CCIR656 codes when bypassing FIFO 1=use CCIR656 codes when bypassing FIFO 0xAB = frame start 0x80 = line start 0x9D = line end 0xB6 = frame end
	3	0x0000	Monochrome output Monochrome output.
	2	0x0000	Progressive Bayer Progressive Bayer
	1	0x0000	Swaps chrominance byte with luminance byte in YUV output. In RGB mode, swaps odd and even bytes. This bit is subject to synchronous update.
	0	0x0000	Swap Channels In YUV output mode, swaps Cb and Cr channels. In RGB mode, swaps R and B. This bit is subject to synchronous update.
	Output Format Config. (context A shadow register). Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
151 0x097	**15:0**	**0x0000**	**output_format_B (RW)**
	15:9	X	Reserved
	8	0x0000	Turn on processed Bayer mode Chip output data in Bayer fomat. As a result datarate decreses in two times.
	7:6	0x0000	RGB output format RGB output format: 00 = 16-bit RGB565 01 = 15-bit RGB555 10 = 12-bit RGB444x 11 = 12-bit RGBx444
	5	0x0000	RGB/YUV output RGB/YUV output 1=output RGB (see R0x332E [7:6]) 0=output YUV
	4	0x0000	Use CCIR656 codes when bypassing FIFO 1=use CCIR656 codes when bypassing FIFO 0xAB = frame start 0x80 = line start 0x9D = line end 0xB6 = frame end
	3	0x0000	Monochrome output Monochrome output.
	2	0x0000	Progressive Bayer Progressive Bayer
	1	0x0000	Swaps chrominance byte with luminance byte in YUV output. In RGB mode, swaps odd and even bytes. This bit is subject to synchronous update.
	0	0x0000	Swap Channels In YUV output mode, swaps Cb and Cr channels. In RGB mode, swaps R and B. This bit is subject to synchronous update.
	Output Format Config. (context B shadow register). Changes take effect only after REFRESH command.		
153 0x099	**15:0**	**0x6440**	**spec_effects_A (RW)**
	15:8	0x0064	Solarization threshhold
	7	X	Reserved
	6	0x0001	Dither Luma only 1= dither only in luma channel 0 = dither in all color channels
	5:3	0x0000	Bit width of dither Bit width of dither (valid values 1, 2, 3, and 4; no dither if value = 0, 5, 6, or 7)
	2:0	0x0000	Special effect selection bits 0 - disabled 1 - monochrome 2 - sepia 3 - negative 4 - solarization with unmodified UV 5 - solarization with -UV
	Special effects selection (context A shadow register). Changes take effect only after REFRESH command.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
155 0x09B	**15:0**	**0x6440**	**spec_effects_B (RW)**
	15:8	0x0064	Solarization threshhold
	7	X	Reserved
	6	0x0001	Dither Luma only 1= dither only in luma channel 0 = dither in all color channels
	5:3	0x0000	Bit width of dither Bit width of dither (valid values 1, 2, 3, and 4; no dither if value = 0, 5, 6, or 7)
	2:0	0x0000	Special effect selection bits 0 - disabled 1 - monochrome 2 - sepia 3 - negative 4 - solarization with unmodified UV 5 - solarization with -UV
	Special effects selection (context B shadow register). Changes take effect only after REFRESH command.		
157 0x09D	**15:0**	**0x0000**	**y_rgb_offset_A (RW)**
	Y/RGB Offset setting (context A shadow register). Changes take effect only after REFRESH command.		
158 0x09E	**15:0**	**0x0000**	**y_rgb_offset_B (RW)**
	Y/RGB Offset setting (context B shadow register). Changes take effect only after REFRESH command.		
159 0x09F	**15:0**	**0x0000**	**gam_shadow_0 (RW)**
	Contrast of the gamma table values for next frame readout. Value is calculated and temporary.		
160 0x0A0	**15:0**	**0x0014**	**gam_shadow_1 (RW)**
	See gam_shadow_0.		
161 0x0A1	**15:0**	**0x0022**	**gam_shadow_2 (RW)**
	See gam_shadow_0.		
162 0x0A2	**15:0**	**0x003A**	**gam_shadow_3 (RW)**
	See gam_shadow_0.		
163 0x0A3	**15:0**	**0x005D**	**gam_shadow_4 (RW)**
	See gam_shadow_0.		
164 0x0A4	**15:0**	**0x0076**	**gam_shadow_5 (RW)**
	See gam_shadow_0.		
165 0x0A5	**15:0**	**0x0088**	**gam_shadow_6 (RW)**
	See gam_shadow_0.		
166 0x0A6	**15:0**	**0x0096**	**gam_shadow_7 (RW)**
	See gam_shadow_0.		

Table 35: 7: Mode Variables (continued)

Reg. #	Bits	Default	Name
167 0x0A7	15:0	0x00A3	gam_shadow_8 (RW)
	See gam_shadow_0.		
168 0x0A8	15:0	0x00AF	gam_shadow_9 (RW)
	See gam_shadow_0.		
169 0x0A9	15:0	0x00BA	gam_shadow_10 (RW)
	See gam_shadow_0.		
170 0x0AA	15:0	0x00C4	gam_shadow_11 (RW)
	See gam_shadow_0.		
171 0x0AB	15:0	0x00CE	gam_shadow_12 (RW)
	See gam_shadow_0.		
172 0x0AC	15:0	0x00D7	gam_shadow_13 (RW)
	See gam_shadow_0.		
173 0x0AD	15:0	0x00E0	gam_shadow_14 (RW)
	See gam_shadow_0.		
174 0x0AE	15:0	0x00E8	gam_shadow_15 (RW)
	See gam_shadow_0.		
175 0x0AF	15:0	0x00F0	gam_shadow_16 (RW)
	See gam_shadow_0.		
176 0x0B0	15:0	0x00F8	gam_shadow_17 (RW)
	See gam_shadow_0.		
177 0x0B1	15:0	0x00FF	gam_shadow_18 (RW)
	See gam_shadow_0.		

Table 36: 11: HG Variables

Reg. #	Bits	Default	Name
0 0x000	15:0	0xE913	vmt (RW)
	Reserved.		
2 0x002	15:0	0x0008	DLevelBufferSpeed (RW)
	Response speed, 1–32; 32-maximum speed.		
3 0x003	15:0	0x0001	scaleGFactor (RW)
	Scale factor for histogram window size.		

Table 36: 11: HG Variables (continued)

Reg. #	Bits	Default	Name
4 0x004	15:0	0x0040	**maxDLevel (RW)**
	Maximum subtracted offset. Set to "0" to disable subtraction.		
5 0x005	15:0	0x0000	**percent (RW)**
	Percent of pixels to keep black; 10-bits data / 4. Setting >0 clips black.		
6 0x006	15:0	0x0000	**lowerLimit1 (RW)**
	Offset for bin 0, divided by 4 on 10-bit scale.		
7 0x007	15:0	0x0002	**binSize1 (RW)**
	Bin width, 0–4LSB, 1–8LSB, 2–16LSB, 7–512LSB on a 10-bit scale.		
8 0x008	15:0	0x00C0	**lowerLimit2 (RW)**
	Offset for bin 0, divided by 4 on 10-bit scale.		
9 0x009	15:0	0x0004	**binSize2 (RW)**
	Bin width, 0–4LSB, 1–8LSB, 2–16LSB, 7–512LSB on a 10-bit scale.		
10 0x00A	15:0	0x0000	**DLevel (RW)**
	Current subtracted offset.		
11 0x00B	15:0	0x0000	**DLevel_buf (RW)**
	Buffered current offset.		
13 0x00D	15:0	0x000A	**factorHI (RW)**
	Reserved		
14 0x00E	15:0	0x0000	**percentHI (RW)**
	Reserved		
15 0x00F	15:0	0x0000	**positionHI (RO)**
	Reserved		
17 0x011	15:0	0x0000	**Nl (RW)**
	Outlier pixel count for L counter		
19 0x013	15:0	0x0000	**Nu (RW)**
	Outlier pixel count for U counter		

Table 37: 0: GPIO

Reg. #	Bits	Default	Name
4208 0x1070	**15:0**	**0x0000**	**GPIO Data (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Data Shows the state of GPIO pads 3:0 and controls those configured as outputs. Bit b (b = 8, ...,11) shows/controls the state of the pad GPIO(b - 8). If certain GPIO pads are floating during STANDBY, the corresponding bits should be tied to "0" converting the pads to output to reduce standby current. See technical note TN0934 "Standby Sequence" for more details.
	7:0	0x0000	AF_GPIO_DATA Shows the state of AF_GPIO pads 7:0 and controls those configured as outputs. Bit b (b = 0, ...,7) shows/controls the state of the pad AF_GPIO(b). If certain AF_GPIO pads are floating during STANDBY, the corresponding bits should be tied to "0" converting the pads to output to reduce standby current. See technical note TN0934 "Standby Sequence" for more details.
	Shows the state of the GPIO pads and controls those configured as outputs.		
4210 0x1072	**15:0**	**0x0000**	**GPIO Output Toggle (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Output Toggle Writing "1" to bit b (b = 8,...,11) toggles the GPIO(b-8) output.
	7:0	0x0000	AF GPIO Output Toggle Writing "1" to bit b (b = 0,...,7) toggles the AF_GPIO(b) output.
	GPIO Output Toggle		
4212 0x1074	**15:0**	**0x0000**	**GPIO Output Set (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Output Set Writing "1" to bit b (b=8,...,11) sets the GPIO(b - 8) output HIGH.
	7:0	0x0000	AF_GPIO Output Set Writing "1" to bit b (b = 0,...,7) sets the AF_GPIO(b) output HIGH.
	This register sets the desired output GPIO/AF_GPIO to HIGH.		
4214 0x1076	**15:0**	**0x0000**	**GPIO Output Clear (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Output Clear Writing "1" to bit b (b = 8,...,11) sets the GPIO(b - 8) output LOW.
	7:0	0x0000	AF_GPIO Output Clear Writing "1" to bit b (b = 0,...,7) sets the AF_GPIO(b) output LOW.
	This register sets the desired output GPIO/AF_GPIO to LOW.		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4216 0x1078	**15:0**	**0x0FFF**	**GPIO Direction (RW)**
	15:12	X	Reserved
	11:8	0x000F	GPIO Direction Controls the direction of GPIO pads 11:8. If bit b (b = 8,...,11) is set to "1," the pad GPIO(b - 8) is an input; otherwise it is an output. Upon power-up or reset all GPIO pads become inputs. If certain GPIO pads are floating during STANDBY, the corresponding bits should be tied to "0" converting the pads to output to reduce standby current. See technical note TN0934 "Standby Sequence" for more details.
	7:0	0x00FF	AF_GPIO Direction Controls the direction of AF_GPIO pads 7:0. If bit b (b = 0,...,7) is set to "1," the pad AF_GPIO(b) is an input; otherwise it is an output. Upon power-up or reset all AF_GPIO pads become inputs. If certain AF_GPIO pads are floating during STANDBY, the corresponding bits should be tied to "0" converting the pads to output to reduce standby current. See technical note TN0934 "Standby Sequence" for more details.
	Controls the direction of GPIO/AF_GPIO pads.		
4218 0x107A	**15:0**	**0x0000**	**GPIO Dir. Reverse (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Direction Reverse Writing "1" to bit b (b = 8,...,11) reverses the direction of GPIO(b - 8).
	7:0	0x0000	AF_GPIO Direction Reverse Writing "1" to bit b (b = 0,...,7) reverses the direction of AF_GPIO(b).
	This register changes the direction of the desired GPIO/AF_GPIO from input to output or vice versa.		
4220 0x107C	**15:0**	**0x0000**	**GPIO Dir. In (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Direction Input Writing "1" to bit b (b = 8,...,11) configures the pad GPIO(b - 8) as an input.
	7:0	0x0000	AF_GPIO Direction Input Writing "1" to bit b (b = 0,...,7) configures the pad AF_GPIO(b) as an input.
	This register configures the desired GPIO/AF_GPIO pads to inputs.		
4222 0x107E	**15:0**	**0x0000**	**GPIO Dir. Out (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO Direction Output Writing "1" to bit b (b = 8,...,11) configures the pad GPIO(b - 8) as an output.
	7:0	0x0000	AF_GPIO Direction Output Writing "1" to bit b (b = 0,...,7) configures the pad AF_GPIO(b) as an output.
	This register configures the desired GPIO/AF_GPIO pads to outputs.		
4224 0x1080	**15:0**	**0x0000**	**1st Subperiod at AF_GPIO-1 (RW)**
	First subperiodof waveform output at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of first subperiod of waveform output at AF_GPIO0 (in 16-bit counter mode).		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4225 0x1081	15:0	0x0000	**1st Subperiod at AF_GPIO-0 (RW)**
	First subperiod of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4226 0x1082	15:0	0x0000	**2nd Subperiod at AF_GPIO-1 (RW)**
	Second subperiod of waveform output at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of second subperiod of the waveform output at AF_GPIO0 (in 16-bit counter mode).		
4227 0x1083	15:0	0x0000	**2nd Subperiod at AF_GPIO-0 (RW)**
	Second subperiod of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4228 0x1084	15:0	0x0000	**3rd Subperiod at AF_GPIO-1 (RW)**
	Third subperiod of waveform output at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of third subperiod of waveform output at AF_GPIO0 (in 16-bit counter mode).		
4229 0x1085	15:0	0x0000	**3rd Subperiod at AF_GPIO-0 (RW)**
	Third subperiod of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4230 0x1086	15:0	0x0000	**4th Subperiod at AF_GPIO-1 (RW)**
	Fourth subperiod of waveform output at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of fourth subperiod of waveform output at AF_GPIO0 (in 16-bit counter mode).		
4231 0x1087	15:0	0x0000	**4th Subperiod at AF_GPIO-0 (RW)**
	Fourth subperiod of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4232 0x1088	15:0	0x0000	**5th Subperiod at AF_GPIO-1 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of 5th subperiod of waveform output at AF_GPIO0 (in 16-bit counter mode).		
4233 0x1089	15:0	0x0000	**5th Subperiod at AF_GPIO-0 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4234 0x108A	15:0	0x0000	**Duration at AF_GPIO-1 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of the duration of waveform generated at AF_GPIO0 (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing “0”makes the duration infinite. When read, this registers returns the number of waveform periods left to be generated at AF_GPIO1 (in 8-bit counter mode) or bits 15:8 of the number remaining at AF_GPIO0 (in 16-bit counter mode). To get all 16 bits of the latter number correct, one must read this register before AF_GPIO_WG_N0.		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4235 0x108B	15:0	0x0000	**Duration at AF_GPIO-0 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this register returns the number of waveform periods left to be generated at AF_GPIO0 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4236 0x108C	15:0	0x0000	**1st Subperiod at AF_GPIO-3 (RW)**
	First subperiod of waveform generated at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of first subperiod of waveform output at AF_GPIO2 (in 16-bit counter mode).		
4237 0x108D	15:0	0x0000	**1st Subperiod at AF_GPIO-2 (RW)**
	First subperiod of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4238 0x108E	15:0	0x0000	**2nd Subperiod at AF_GPIO-3 (RW)**
	Second subperiod of waveform output at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of second subperiod of waveform output at AF_GPIO2 (in 16-bit counter mode).		
4239 0x108F	15:0	0x0000	**2nd Subperiod at AF_GPIO-2 (RW)**
	Second subperiod of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4240 0x1090	15:0	0x0000	**3rd Subperiod at AF_GPIO-3 (RW)**
	Third subperiod of waveform generated at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of third subperiod of waveform output at AF_GPIO2 (in 16-bit counter mode).		
4241 0x1091	15:0	0x0000	**3rd Subperiod at AF_GPIO-2 (RW)**
	Third subperiod of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4242 0x1092	15:0	0x0000	**4th Subperiod at AF_GPIO-3 (RW)**
	Fourth subperiod of waveform output at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of fourth subperiod of waveform output at AF_GPIO2 (in 16-bit counter mode).		
4243 0x1093	15:0	0x0000	**4th Subperiod at AF_GPIO-2 (RW)**
	Fourth subperiod of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4244 0x1094	15:0	0x0000	**5th Subperiod at AF_GPIO-3 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of fifth subperiod of waveform output at AF_GPIO2 (in 16-bit counter mode).		
4245 0x1095	15:0	0x0000	**5th Subperiod at AF_GPIO-2 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4246 0x1096	15:0	0x0000	**Duration at AF_GPIO-3 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of the duration of waveform generated at AF_GPIO2 (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this registers returns the number of waveform periods left to be generated at AF_GPIO3 (in 8-bit counter mode) or bits 15:8 of the number remaining at AF_GPIO2 (in 16-bit counter mode). To get all 16 bits of the latter number right, one must read this register before AF_GPIO_WG_N2.		
4247 0x1097	15:0	0x0000	**Duration at AF_GPIO-2 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this register returns the number of waveform periods left to be generated at AF_GPIO2 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4248 0x1098	15:0	0x0000	**1st Subperiod at AF_GPIO-5 (RW)**
	First subperiod of waveform generated at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of first subperiod of waveform output at AF_GPIO4 (in 16-bit counter mode).		
4249 0x1099	15:0	0x0000	**1st Subperiod at AF_GPIO-4 (RW)**
	First subperiod of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4250 0x109A	15:0	0x0000	**2nd Subperiod at AF_GPIO-5 (RW)**
	Second subperiod of waveform output at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of second subperiod of waveform output at AF_GPIO4 (in 16-bit counter mode).		
4251 0x109B	15:0	0x0000	**2nd Subperiod at AF_GPIO-4 (RW)**
	Second subperiod of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4252 0x109C	15:0	0x0000	**3rd Subperiod at AF_GPIO-5 (RW)**
	Third subperiod of waveform output at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of third subperiod of waveform output at AF_GPIO4 (in 16-bit counter mode).		
4253 0x109D	15:0	0x0000	**3rd Subperiod at AF_GPIO-4 (RW)**
	Third subperiod of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4254 0x109E	15:0	0x0000	**4th Subperiod at AF_GPIO-5 (RW)**
	Fourth subperiod of waveform output at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of fourth subperiod of waveform output at AF_GPIO4 (in 16-bit counter mode).		
4255 0x109F	15:0	0x0000	**4th Subperiod at AF_GPIO-4 (RW)**
	Fourth subperiod of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4256 0x10A0	15:0	0x0000	**5th Subperiod at AF_GPIO-5 (RW)**
	Fifth subperiod of waveform output at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of fifth subperiod of waveform output at AF_GPIO4 (in 16-bit counter mode).		
4257 0x10A1	15:0	0x0000	**5th Subperiod at AF_GPIO-4 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4258 0x10A2	15:0	0x0000	**Duration at AF_GPIO-5 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of the duration of waveform generated at AF_GPIO4 (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this registers returns the number of waveform periods left to be generated at AF_GPIO5 (in 8-bit counter mode) or bits 15:8 of the number remaining at AF_GPIO4 (in 16-bit counter mode). To get all 16 bits of the latter number right, one must read this register before AF_GPIO_WG_N4.		
4259 0x10A3	15:0	0x0000	**Duration at AF_GPIO-4 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this registers returns the number of waveform periods left to be generated at AF_GPIO4 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4260 0x10A4	15:0	0x0000	**1st Subperiod at AF_GPIO-7 (RW)**
	First subperiod of waveform generated at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of first subperiod of waveform output at AF_GPIO6 (in 16-bit counter mode).		
4261 0x10A5	15:0	0x0000	**1st Subperiod at AF_GPIO-6 (RW)**
	First subperiod of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4262 0x10A6	15:0	0x0000	**2nd Subperiod at AF_GPIO-7 (RW)**
	Second subperiod of waveform output at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of second subperiod of waveform output at AF_GPIO6 (in 16-bit counter mode).		
4263 0x10A7	15:0	0x0000	**2nd Subperiod at AF_GPIO-6 (RW)**
	Second subperiod of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4264 0x10A8	15:0	0x0000	**3rd Subperiod at AF_GPIO-7 (RW)**
	Third subperiod of waveform output at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of third subperiod of waveform output at AF_GPIO6 (in 16-bit counter mode).		
4265 0x10A9	15:0	0x0000	**3rd Subperiod at AF_GPIO-6 (RW)**
	Third subperiod of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4266 0x10AA	15:0	0x0000	**4th Subperiod at AF_GPIO-7 (RW)**
	Fourth subperiod of waveform output at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of fourth subperiod of waveform output at AF_GPIO6 (in 16-bit counter mode).		
4267 0x10AB	15:0	0x0000	**4th Subperiod at AF_GPIO-6 (RW)**
	Fourth subperiod of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4268 0x10AC	15:0	0x0000	**5th Subperiod at AF_GPIO-7 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of fifth subperiod of waveform output at AF_GPIO6 (in 16-bit counter mode).		
4269 0x10AD	15:0	0x0000	**5th Subperiod at AF_GPIO-6 (RW)**
	Fifth subperiod of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4270 0x10AE	15:0	0x0000	**Duration at AF_GPIO-7 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of the duration of waveform generated at AF_GPIO6 (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this registers returns the number of waveform periods left to be generated at AF_GPIO7 (in 8-bit counter mode) or bits 15:8 of the number remaining at AF_GPIO6 (in 16-bit counter mode). To get all 16 bits of the latter number right, one must read this register before AF_GPIO_WG_N6.		
4271 0x10AF	15:0	0x0000	**Duration at AF_GPIO-6 (RW)**
	Writing to this register sets the duration of waveform generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode). Finite duration is selected by writing the desired number of periods in the waveform. Writing "0" makes the duration infinite. When read, this register returns the number of waveform periods left to be generated at AF_GPIO6 (in 8-bit counter mode) or bits 7:0 of the same (in 16-bit counter mode).		
4272 0x10B0	15:0	0x0000	**Waveform Generator Config. (RW)**
	15:8	X	Reserved
	7:4	0x0000	Enable in 16-bit Counter Mode Clearing bit b (b = 4,...,7) enables the waveform generator to drive the pads AF_GPIO(2b - 8) and AF_GPIO(2b - 7) simultaneously. Counters needed to generate waveforms at these pads are put in the 8-bit mode. Setting bit b disconnects the waveform generator from the pad AF_GPIO(2b - 7) and allows it to generate a waveform at the pad AF_GPIO(2b - 8) using the 16-bit counter mode.
	3:0	0x0000	Enable in 8-bit Counter Mode Setting/clearing bit b (b = 0,...,3) enables/disables waveform generation at the pads AF_GPIO(2b) and AF_GPIO(2b + 1).
	Configures the GPIO Waveform Generator.		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4273 0x10B1	15:0	0x0000	**Chain Waveforms (RW)**
	By setting bits 6:0 in this register, one can link waveforms generated at different pads into a chain. The end of the first waveform in the chain coincides with the start of the second waveform, and so on. Specifically, for b = 0, 2, 4, the following is true: Setting bit b forces the waveform generator to start driving AF_GPIO(b) when it is done with AF_GPIO(b + 1) (in 8-bit counter mode) or AF_GPIO(b + 2) (in 16-bit counter mode). Setting bit (b + 1) forces the waveform generator in to start driving AF_GPIO(b + 1) when done with AF_GPIO(b + 2) (in 8-bit counter mode; the bit is ignored in 16-bit mode). In addition, setting bit 6 forces the waveforms generator to start driving AF_GPIO6 when done with AF_GPIO7 (in 8-bit counter mode; the bit is ignored in 16-bit mode).		
4274 0x10B2	15:0	0x0000	**Waveform Clock Dividers (RW)**
	15:8	X	Reserved
	7:4	0x0000	Divider 2 Holds d for divider 2.
	3:0	0x0000	Divider 1 Holds d for divider 1.
	The waveform generator has two clock dividers that enable it to generate waveforms at vastly different paces. Each divider divides the frequency of the AF_GPIO clock (typically 80 MHz) by a factor 2d + 1, where d is a 4-bit unsigned integer programmed into register AF_GPIO_WG_CLKDIV. Bits 3:0 of this register hold d for divider 1. Bits 7:4 of this register hold d for divider 2.		
4275 0x10B3	15:0	0x0000	**Clock Divider Selects (RW)**
	Cleared bit b (b = 0,…,7) tells the waveform generator to use clock divider 1 when generating waveforms at the pad AF_GPIO(b). Set bits select clock divider 2 for the corresponding pads.		
4276 0x10B4	15:0	0x0000	**Sync Waveform to Frame (RW)**
	If bit b (b=0,…,7) in AF_GPIO_WG_FRAME_SYNC is cleared, waveform generation at the AF_GPIO(b) output starts/resumes when bit b in AF_GPIO_WG_SUSPEND is cleared. Setting bit b in AF_GPIO_WG_FRAME_SYNC changes the conditions that must be met for waveform generation at AF_GPIO(b) can start/ resume. The clearing of bit b in AF_GPIO_WG_SUSPEND is still necessary, but no longer sufficient. After that bit is cleared, the waveform generator restarts on each falling edge of FRAME_VALID.		
4277 0x10B5	15:0	0x0000	**Waveform Resets (RW)**
	Setting bit b (b = 0,…,7) stops any ongoing waveform generation at AF_GPIO(b), and resets all counters used in it. The bit must be cleared before waveform generation can resume.		
4278 0x10B6	15:0	0x0000	**Waveform Suspends (RW)**
	Setting bit b (b = 0,…,7) suspends waveform generation at the pad AF_GPIO(b). Clearing the bit restarts it.		
4280 0x10B8	15:0	0x0000	**Enable Notification Signals (RW)**
	Setting bit b (b = 0,…,7) enables a notification signal (NS) at the end of waveform generation at AF_GPIO(b). Clearing the bit b enables a NS on next transition at AF_GPIO(b) whose sign matches the sign indicated by bit b in AF_GPIO_NS_EDGE. The pad may be configured as an output or input and the transition may be caused by the waveform generator, writing to AF_GPIO_DATA or external forcing.		

Table 37: 0: GPIO (continued)

Reg. #	Bits	Default	Name
4281 0x10B9	**15:0**	**0x0000**	**Trigger Edge Selects (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO NS Edge Select Bit b (b = 8,...,11) selects the sign of transitions on pad GPIO(b - 8) that triggers notification signals. Setting the bit selects rising edges, clearing it, the falling edges.
	7:0	0x0000	AF_GPIO NS Edge Select Bit b (b = 0,…,7) selects the sign of transitions on pad AF_GPIO(b) that triggers notification signals. Setting the bit selects rising edges, clearing it, the falling edges.
	Selects the sign of transitions on GPIO pads that triggers notification signals. Setting the bit selects rising edges, clearing it, the falling edges.		
4283 0x10BB	**15:0**	**0x0FFF**	**Trigger Mask (RW)**
	15:12	X	Reserved
	11:8	0x000F	Trigger Mask Setting bit b (b = 8,...,11) masks all notification signals caused by events on pad GPIO(b - 8). Masked signals do not cause the microcontroller to wake up. Clearing the bit enables waking up.
	7:0	0x00FF	Trigger Mask Setting bit b (b = 0,…,7) masks all notification signals caused by events on pad AF_GPIO(b). Masked signals do not cause microcontroller to wake up. Clearing the bit enables waking up.
	Trigger mask on notification signals.		
4285 0x10BD	**15:0**	**0x0000**	**Sync Waveform to STROBE (RW)**
	If bit b (b = 0,…,7) in AF_GPIO_WG_STROBE_SYNC is cleared, waveform generation at AF_GPIO(b) output starts/resumes when bit b in AF_GPIO_WG_SUSPEND is cleared. Setting bit b in AF_GPIO_WG_STROBE_SYNC changes the conditions that must be met for waveform generation on AF_GPIO(b) can start/ resume. The clearing of bit b in AF_GPIO_WG_SUSPEND is still necessary, but no longer sufficient. After that bit is cleared, the waveform generator restarts on each rising edge of STROBE.		
4286 0x10BE	**15:0**	**0x0000**	**Signals Pending (RW)**
	15:12	X	Reserved
	11:8	0x0000	GPIO notification signal status When bit b (b = 8,...,11) is set, it signals that some event on pad GPIO(b - 8) caused a notification signal. Writing “1” to the bit to clears it.
	7:0	0x0000	AF_GPIO notification signal status When bit b (b = 0,…,7) is set, it signals that some event on pad AF_GPIO(b) caused a notification signal. Writing “1” to the bit to clears it.
	GPIO notification signal (NS) status		

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. The sensor includes a power-on reset feature that initiates a reset upon VDD power up. Even though this feature is included on the device, it is advised that the user still manually assert a hard reset upon power up.

Reset

Two types of reset are available:

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

Hard Reset

Figure 34: Hard Reset

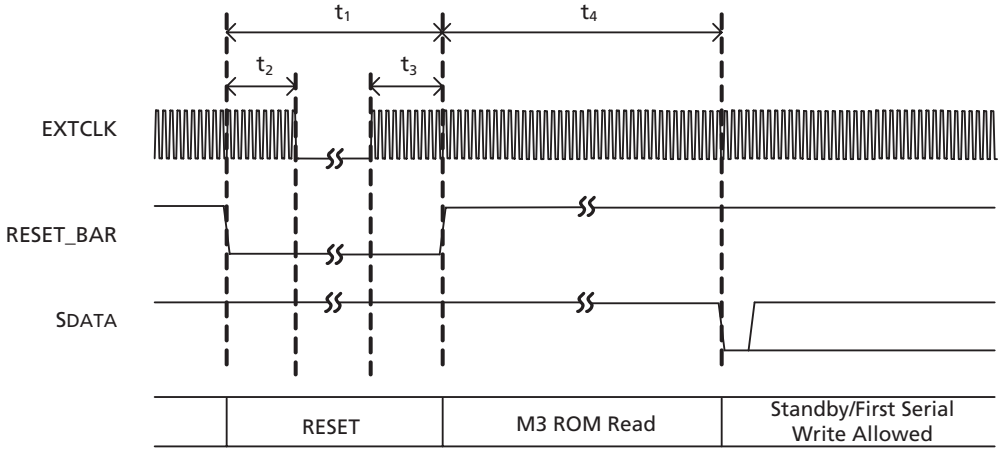


Table 38: Hard Reset

Definition	Condition	Symbol	Min	Typ	Max	Units
RESET_BAR Pulse Width		t1	30	–	–	EXTCLK Cycles
Active EXTCLK after RESET_BAR asserted		t2	10	–	–	
Active EXTCLK before RESET_BAR de-asserted		t3	10	–	–	
Max ROM read time		t4	–	–	6000	

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 6000 clock cycles before using the two-wire serial interface.

After hard reset, the output FIFO is configured for operation but disabled and all outputs are tri-stated. To enable outputs, user must assert R0x301A[6] via the serial interface.

Soft Reset

A soft reset sequence to the camera has the same affect as the hard reset and can be activated by the following steps:

1. Enable soft reset by setting R0x301A[0].
2. Wait 6000 clock cycles before using the two-wire serial interface.

Standby Modes

The MT9D112 supports three different standby modes:

- Hard standby mode
- Soft standby mode
- Shutdown mode

For both hard and soft standby, entry can be inhibited by setting R0x3202[3] = 0.

Hard Standby

Hard standby disables all digital logic; the sensor can only be woken up by de-asserting STANDBY. All registers and RAM content will be preserved. Hard standby can be performed in any sequencer state after all AE, AWB, AF, HG, and flicker calculations are finished and the chip is in sleep mode.

During standby, the sequencer does the following:

- Configures MIPI, PLL, sensor core, and AFM driver
- Save clock control registers
- Set standby done bit
- Disable internal clocks
- MCU goes to sleep mode

After leaving standby, the sequencer wakes up the MCU and performs the following sequence:

- Restore internal clocks
- Clear standby done bit
- Leave standby for MIPI, PLL, sensor core, and AFM driver

To enter standby:

a. Assert pin STANDBY=1
   MCU will run standby routine and will set bit R0x3204[0]= 1 when standby entry is completed. By default clock is gated off when standby state is entered; therefore, the status of R0x3204[0] cannot be polled, during which time, no registers or variables are accessible.
b. To ensure Standby mode is entered, wait until the two-wire serial interface becomes non-responsive.
c. Optionally, stop the EXTCLK to minimize the standby current.

To leave standby:

a. Provide EXTCLK 10 clock cycles before de-asserting STANDBY
b. De-assert STANDBY

Figure 35: Hard Standby

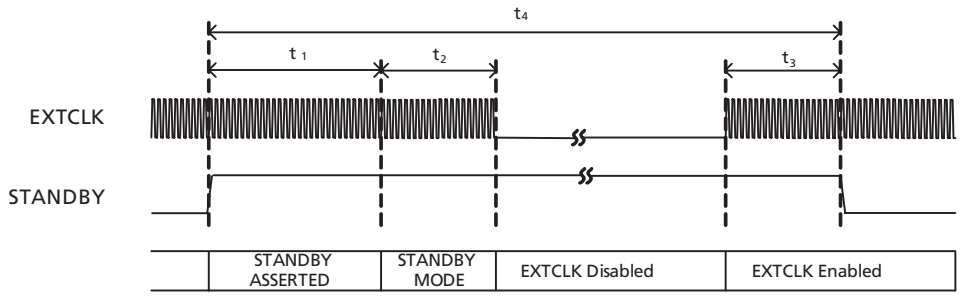


Table 39: Hard Standby

Definition	Condition	Symbol	Min	Typ	Max	Units
STANDBY asserted to Standby mode		t_1	TBD	–	–	EXTCLK Cycles
Active EXTCLK after STANDBY asserted		t_2	TBD	–	–	
Active EXTCLK before STANDBY de-asserted		t_3	TBD	–	–	
STANDBY Pulse Width		t_4	TBD	–	–	

Soft Standby

Soft standby can be enabled by register access and disables the sensor core and most of the digital logic. The serial interface is still active and the sensor can be woken up via commands. All register settings and RAM content will be preserved. Soft Standby can be performed in any sequencer state after all AE, AWB, AF, HG, and flicker calculations are finished and the chip is in sleep mode. The sequencer follows the same steps for soft standby that are performed for hard standby.

To enter standby:

a. Set R0x3202[3] = 1
b. Set R0x3202[0] = 1
c. To ensure Standby mode is entered, poll R0x3204[0] until it equals 1, which indicates that the standby entry is completed.
d. Stop the EXTCLK to minimize the standby current (this will disable the serial interface).

To leave standby:

a. Provide EXTCLK 10 clock cycles before b)
b. Set R0x3202[0] = 0

Figure 36: Soft Standby

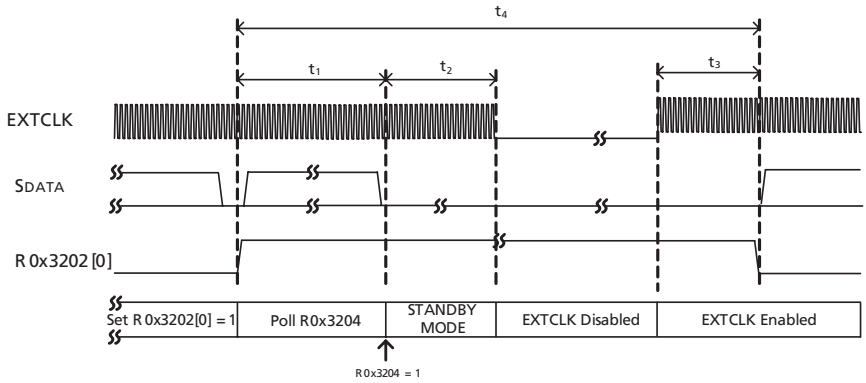


Table 40: Soft Standby

Definition	Condition	Symbol	Min	Typ	Max	Units
Stdby entry complete		t_1	–	–	TBD	EXTCLK Cycles
Active EXTCLK after soft stdby activates		t_2	TBD	–	–	
Active EXTCLK before soft stdby de-activates		t_3	TBD	–	–	
Minimum Stdby time		t_4	TBD	–	–	

The execution of standby will take place after the completion of the current line by default. In order to synchronize the execution of standby with the end of frame, set R0x3204[4] = 1.

Shutdown Mode

Low power standby mode uses the SHUTDOWN pin to shut down digital power (VDD) and ensure the lowest power consumption. All the two-wire serial interface settings and firmware variables including patches will be lost in this mode. Waking up from this mode is equivalent to power up.

To enter standby:

a. Follow Hard Standby or Soft Standby procedure to enter into standby and check for standby entry
b. Assert SHUTDOWN pin HIGH

To leave standby:

a. Assert SHUTDOWN pin LOW
b. De-assert STANDBY pin if hard standby is used
c. Follow power up procedure

Table 41: Pin States During Conditions

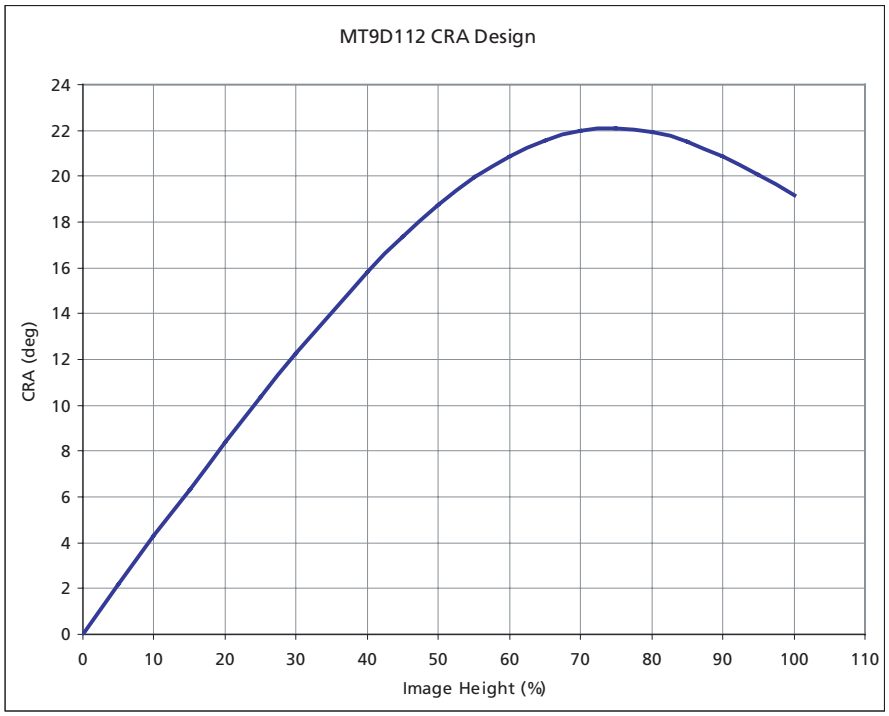
Pin	Reset	Post-reset	Standby	Standby w/ SHUTDOWN	Power Down
DOUT[7:0]	Z	Z	Z by default (configurable via OE_BAR or two-wire serial interface reg)	Z by default	X
PIXCLK	Z	Z	Z by default (configurable)	Z by default	X
LV	Z	Z	Z by default (configurable	Z by default	X
FV	Z	Z	Z by default (configurable	Z by default	X

Table 41: Pin States During Conditions (Continued)

Pin	Reset	Post-reset	Standby	Standby w/ SHUTDOWN	Power Down
DATA_OUT_N	0	0	0	0	X
DATA_OUT_P	0	0	0	0	X
CLK_OUT_N	0	0	0	0	X
CLK_OUT_P	0	0	0	0	X
GPIO[3:0]	Z	Z	Depending on how system use them as DOUTPSB/Shutter/Flash/Module ID/Trigger/ OE_BAR	Depending on how system use them as DOUTPSB/Shutter/ Flash/Module ID/ Trigger/OE_BAR	X
AF_GPIO[7:0]	Z	Z			X
SADDR	Input	Input	Input	Input	
SDATA	Input	I/O	Input	Input	
SCLK	Input	Input	Input	Input	
ATEST	I/O (Analog pad)	I/O (Analog pad)	I/O (Analog pad)	I/O (Analog pad)	X

Spectral Characteristics

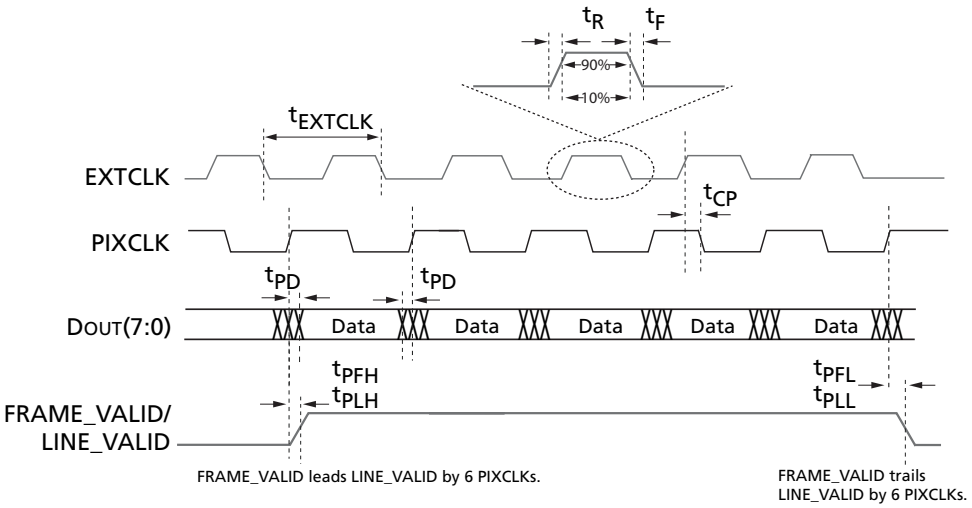
Figure 37: CRA versus Image Height



CRA versus. Image Height Plot	Image Height		CRA (Deg)
	(%)	(mm)	
	0	0	0
	5	0.110	2.17
	10	0.220	4.28
	15	0.330	6.34
	20	0.440	8.37
	25	0.550	10.34
	30	0.660	12.26
	35	0.770	14.09
	40	0.880	15.80
	45	0.990	17.37
	50	1.100	18.76
	55	1.210	19.95
	60	1.320	20.89
	65	1.430	21.57
	70	1.540	21.98
	75	1.650	22.10
	80	1.760	21.94
	85	1.870	21.53
	90	1.980	20.88
	95	2.090	20.06
	100	2.200	19.14

Electrical Specifications

Figure 38: I/O Timing Diagram



Note: PLL disabled for tCP.

Table 42: AC Electrical Characteristics

Setup Conditions: ^{f}EXTCLK = 6–54 MHz; V_{DD} = 1.8V; V_{DDIO} = 2.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DDPLL} = 2.8V; T_Junction = 25°C; C_{LOAD} = 15-20pF.

Symbol	Definition	Conditions	Min	Typ	Max	Units	Notes
^{f}EXTCLK	External clock frequency	PLL enabled/disabled	6	–	54	MHz	
^{f}INTCLK	Internal master clock frequency	PLL disabled	6	–	54	MHz	
^{f}INTCLK	Internal master clock frequency	PLL enabled	6	–	80	MHz	
^{f}PIXCLK	PIXCLK frequency	Default	6	–	80	MHz	
^{t}R	Input clock rise time	From 10% to 90% Vp-p	–	2	–	ns	
^{t}F	Input clock fall time	From 90% to 10% Vp-p	–	2	–	ns	
	Clock duty cycle		45	50	55	%	
^{t}JITTER	Input clock jitter		–	0.5	–	ns	1
^{t}CP	EXTCLK rising to PIXCLK falling propagation delay	PLL disabled	–	15	–	ns	
Output pin slew	Programmable slew = 7; V_{DD} = 1.8V Temp = 25°C	V_{DDIO} = 2.8V, C_{LOAD} = 30	–	TBD	–	ns	
		V_{DDIO} = 2.8V, C_{LOAD} = 15	–	TBD	–	ns	
	Programmable slew condition 1	V_{DDIO} = 2.8V,C_{LOAD} = 30	–	TBD	–	ns	
		V_{DDIO} = 2.8V, C_{LOAD} = 15	–	TBD	–	ns	
	Programmable slew condition 2	V_{DDIO} = 2.8V, C_{LOAD} = 30	–	TBD	–	ns	
		V_{DDIO} = 2.8V, C_{LOAD} = 15	–	TBD	–	ns	
	Programmable slew condition n	V_{DDIO} = 2.8V, C_{LOAD} = 30	–	TBD	–	ns	
		V_{DDIO} = 2.8V, C_{LOAD} = 15	–	TBD	–	ns	
^{t}PD	PIXCLK to data valid	Default	–	5	–	ns	
^{t}PFH	PIXCLK to FV HIGH	Default	–	6	–	ns	
^{t}PLH	PIXCLK to LV HIGH	Default	–	6	–	ns	
^{t}PFL	PIXCLK to FV LOW	Default	–	4	–	ns	
^{t}PLL	PIXCLK to LV LOW	Default	–	4	–	ns	
C_{IN}	Input pin capacitance		–	3.5	–	pF	

Note: 1. Value equal to jitter on tester.

Table 43: I/O Parameters

Symbol	Definition	Conditions	Min	Typ	Max	Units
V_{IH}	Input HIGH voltage	V_{DDIO} = 2.8V V_{DDIO} = 1.8V	2.4 1.4		V_{DDIO}+0.3 V_{DDIO}+0.3	V
V_{IL}	Input LOW voltage	V_{DDIO} = 2.8V V_{DDIO} = 1.8V	GND-0.3 GND-0.3		0.8 0.4	V
I_{IN}	Input leakage current	No pull-up resistor; V_{IN} = V_{DD} or D_{GND}	-15	±0.5	15	µA
V_{OH}	Output HIGH voltage	At specified I_{OH}	V_{DDIO}-0.4	–		V
V_{OL}	Output LOW voltage	At specified I_{OL}	–	–	0.4	V
I_{OH}	Output HIGH current	At specified V_{OH}	–	–	–	mA
I_{OL}	Output HIGH current	At specified V_{OL}	–	–	–	mA
I_{OZ}	Tri-state output leakage current	V_{IN} = V_{DDIO} or GND	-10	±0.5	10	µA

Table 44: DC Electrical Definitions and Characteristics

Setup Conditions: [f]EXTCLK = 6–54 MHz; V_{DD} = 1.8V; V_{DDIO} = 2.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DD}PLL = 2.8V; T_Junction = 25°C; Dark lighting conditions

Symbol	Definition	Conditions	Min	Typ	Max	Units	Note
V_{DD}	Core digital voltage		1.7	1.8	1.95	V	
V_{DDIO}	I/O digital voltage		1.7	1.8	1.95	V	
V_{DDIO}	I/O digital voltage		2.5	2.8	3.1	V	
V_{AA}	Analog voltage		2.5	2.8	3.1	V	
VAAPIX	Pixel supply voltage		2.5	2.8	3.1	V	
VDDPLL	PLL supply voltage		2.5	2.8	3.1	V	
VDD_AF	AF supply voltage		1.7	1.8	1.95	V	
VDD_AF	AF supply voltage		2.5	2.8	3.1	V	
IDD_1	Digital operating current	Low power preview	–	24	–	mA	
IAA_1	Analog operating current	Low power preview	–	21	–	mA	
IAA_PIX_1	Pixel supply current	Low power preview	–	0.5	–	mA	
IDD_PLL_1	PLL supply current	Low power preview with PLL enabled	–	9	–	mA	
	Total power consumption	Low power preview	–	200	–	mW	
IDD_2	Digital operating current	Full resolution	–	38	–	mA	
IAA_2	Analog operating current	Full resolution	–	36	–	mA	
IAA_PIX_2	Pixel supply current	Full resolution	–	0.3	–	mA	
IDD_PLL_2	PLL supply current	Full resolution with PLL enabled	–	9	–	mA	
	Total power consumption	Full resolution	–	350	–	mW	
STDBY_1	Standby current 1	Standby/EXTCLK Dis/ SHUTDOWN	–	10	–	µA	
STDBY_2	Standby current 2	Standby/EXTCLK Dis	–	100	–	µA	
STDBY_3	Standby current 3	Soft standby/EXTCLK Dis/SHUTDOWN	–	10	–	µA	

Table 45: Absolute Maximum Ratings

Symbol	Parameter	Rating		Unit
		Min	Max	
VDD_MAX	Core digital voltage	-0.3	2.4	V
VDD_IO_MAX	I/O digital voltage	-0.3	4.0	V
VAA_MAX	Analog voltage	-0.3	4.0	V
VAA_PIX_MAX	Pixel supply voltage	-0.3	4.0	V
VDD_PLL MAX	PLL supply voltage	-0.3	4.0	V
VIH_MAX	Input HIGH voltage	-0.3	VDDIO-0.3	V
VIL_MAX	Input LOW voltage	-0.3	VDDIO-0.3	V
VFSIO_MAX	Input HIGH voltage	TBD	TBD	
IDD_MAX	Digital operating current	TBD	TBD	mA
IDD_IO_MAX	I/O digital operating current	TBD	TBD	mA
IAA_MAX	Analog operating current	TBD	TBD	mA
IAA_PIX_MAX	Pixel supply current	TBD	TBD	mA
IDD_PLL_MAX	PLL supply current	TBD	TBD	mA
			–	
T_OP	Operating temperature (measure at junction)	-30	75	°C
T_STG[1]	Storage temperature	-40	85	°C

Note: 1. Stresses above those listed may cause permanent damage to the device. This is a stress rating only, and functional operation of the device at these or any other conditions above those indicated in the product specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Figure 39: Internal Power-on Reset

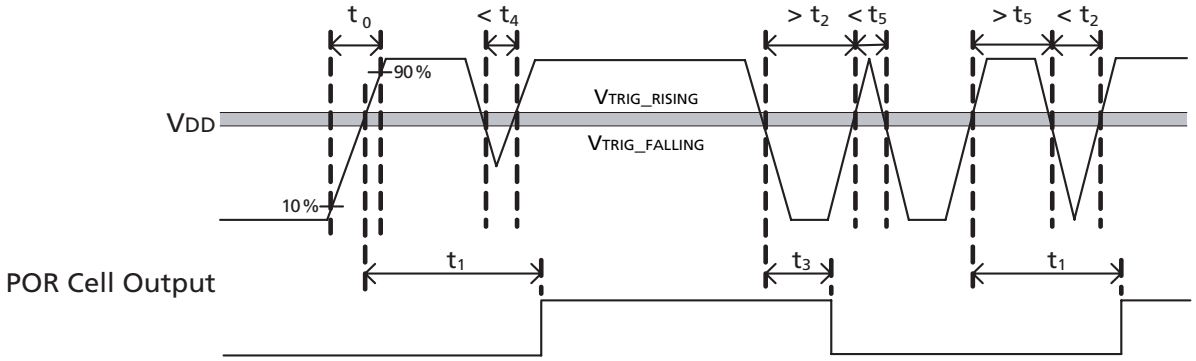
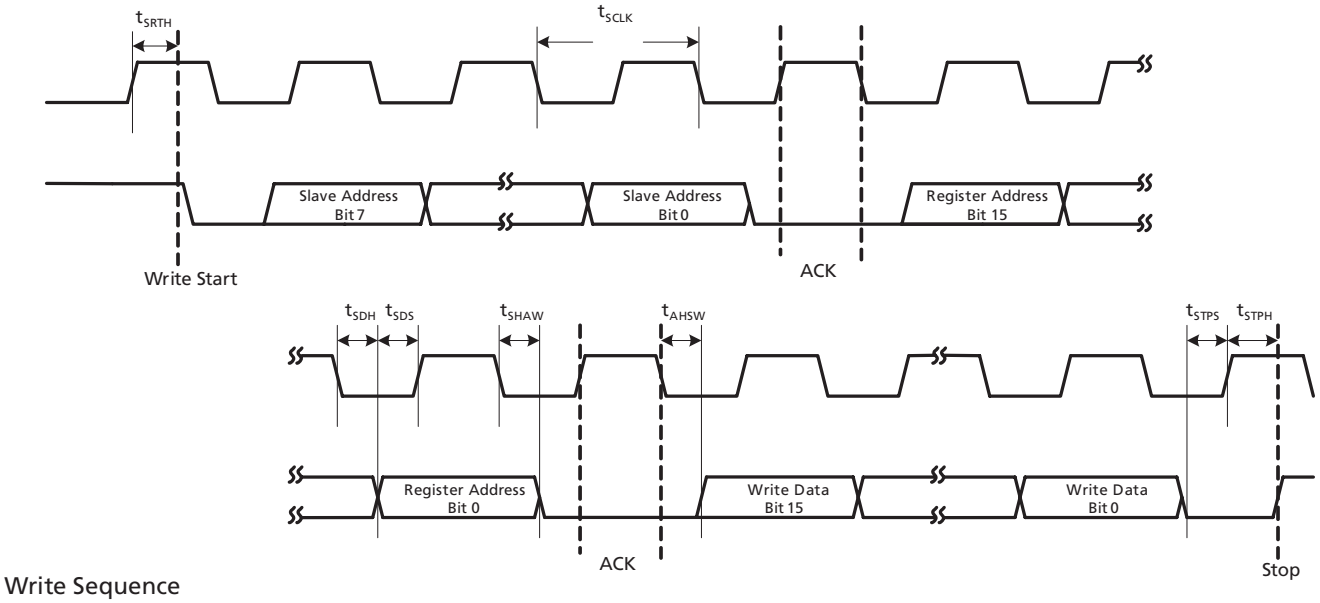


Table 46: POR Parameters

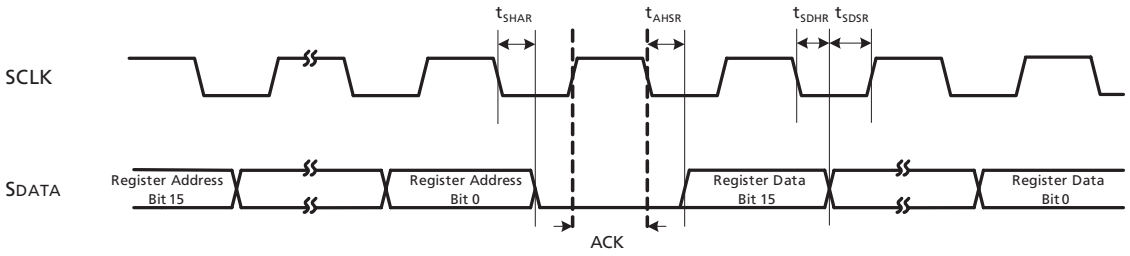
Definition	Condition	Symbol	Min	Typ	Max	Units
VDD ramp time		$^{t}0$	50	–	500	µs
VDD rising, crossing VTRIG_RISING Internal reset being released		$^{t}1$	7	10	15	µs
VDD falling, crossing VTRIG_FALLING; Internal reset active		$^{t}2$	–	0.5	1	µs
Minimum VDD spike width below VTRIG_FALLING; considered to be a reset when POR cell output is HIGH		$^{t}3$	–	0.5	–	µs
Minimum VDD spike width below VTRIG_FALLING considered to be a reset when POR cell output is LOW		$^{t}4$	–	1	–	µs
Minimum VDD spike width above VTRIG_RISING considered to be a stable supply when POR cell output is LOW		$^{t}5$	–	50	–	ns
VDD rising trigger voltage		VTRIG_RISING	1.12	–	1.55	V
VDD falling trigger voltage		VTRIG_FALLING	1.0	–	1.45	V

Note: The hard reset and POR signals are gated, therefore, M3 ROM read will begin after both have been de-asserted.

Figure 40: Two-Wire Serial Bus Timing Parameters



Write Sequence



Read Sequence – read waveforms start after the slave address of the device has been written.

Table 47: Two-wire Serial Bus Characteristics

Symbol	Definition	Conditions	Min	Typ	Max	Units
^{f}SCLK	Serial interface input clock frequency		100	–	400	kHz
	SCLK duty cycle		40	–	60	%
^{t}SRTH	Start hold time	WRITE/READ	4	–	^{t}EXTCLK	ns
^{t}SDH	SDATA hold	WRITE	4	–	^{t}EXTCLK	ns
^{t}SDS	SDATA setup	WRITE	4	–	^{t}EXTCLK	ns
^{t}SHAW	SDATA hold to ACK	WRITE	4	–	^{t}EXTCLK	ns
^{t}AHSW	ACK hold to SDATA	WRITE	4	–	^{t}EXTCLK	ns
^{t}STPS	Stop setup time	WRITE/READ	4	–	^{t}EXTCLK	ns
^{t}STPH	Stop hold time	WRITE/READ	4	–	^{t}EXTCLK	ns
^{t}SHAR	SDATA hold to ACK	READ	4	–	^{t}EXTCLK	ns
^{t}AHSR	ACK hold to SDATA	READ	4	–	^{t}EXTCLK	ns
^{t}SDHR	SDATA hold	READ	4	–	^{t}EXTCLK	ns
^{t}SDSR	SDATA setup	READ	4	–	^{t}EXTCLK	ns
SDATA Slew Rate	No programmable slew	Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
	Programmable slew condition 1	Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
	Programmable slew condition 2	Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
		Max VDD_IO C_LOAD = ?	–	–	–	
C_IN_SI	Serial interface input pin capacitance		–	–	6	pF
C_LOAD_SD	SDATA max load capacitance		–	–	15	pF
R_SD	SDATA pull-up resistor		–	1.5	–	KΩ

Note: Either the slave or master device can drive the SCLK line low—the interface protocol determines which device is allowed to drive the SCLK line at any given time.

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 B, Preliminary 2/06

- “Features” on page 1
- Table 1, “Key Performance Parameters,” on page 1
- “General Description” on page 7
- “MT9D112 Overview” on page 7
- Table 3, “Signal Description,” on page 9
- Figure 1: “Typical Configuration (connection),” on page 10
- “Sensor Core Description” on page 11
- “Pixel Array Structure” on page 12
- Figure 4: “Pixel Array,” on page 12
- Figure 5: “Pixel Color Pattern Detail (Top Right Corner),” on page 13
- “Default Readout Order” on page 13
- “Integration Time” on page 14
- “PLL Setup” on page 15
- "Horizontal Mirror" on page 16
- Table 4, “Frequency Parameters,” on page 16
- “Window Size” on page 16
- “Raw Data Timing” on page 22
- Table 6, “Row Timing Parameters,” on page 23
- Figure 20: “Test Patterns,” on page 25
- "Test Patterns" on page 25
- “First Black Level Subtraction and Digital Gain” on page 26
- “Contrast and Gamma Correction” on page 28
- “Color Kill” on page 30
- “Parallel and MIPI Output” on page 32
- “YUV/RGB Uncompressed Output” on page 32
- “Uncompressed 10-Bit Bypass Output” on page 33
- “Firmware Architecture” on page 36
- "Slave Address/Data Direction Byte" on page 44
- "Message Byte" on page 44
- “Protocol” on page 44
- “Data Transfer” on page 44
- “Typical Serial Transfer” on page 45
- Figure 28: “Single Read from Random Location,” on page 46
- Figure 29: “Single Read from Current Location,” on page 46
- Figure 30: “Sequential Read, Start from Random Location,” on page 46
- Figure 31: “Sequential Read, Start from Current Location,” on page 47
- “Registers” on page 49
- “Variables” on page 49
- "Double-buffered Registers" on page 50
- "Bad Frames" on page 51
- "Changes to Integration Time" on page 51
- "Changes to Gain Settings" on page 51
- Table 12, “0: Core Registers,” on page 52
- Table 14, “2: SOC2,” on page 58

- Table 15, “0: Monitor Variables,” on page 64
- Table 16, “1: Sequencer Variables,” on page 64
- Table 17, “2: AE Variables,” on page 66
- Table 18, “3: AWB Variables,” on page 67
- Table 19, “4: Flicker Varialbes,” on page 69
- Table 20, “5: Auto Focus Variables,” on page 71
- Table 21, “6: AFM Variables,” on page 71
- Table 22, “7: Mode Variables,” on page 72
- Table 23, “11: HG Variables,” on page 75
- Table 24, “0: GPIO,” on page 75
- Table 25, “0: Core Registers,” on page 78
- Table 26, “1: SOC1,” on page 91
- Table 28, “0: Monitor Variables,” on page 129
- Table 28, “0: Monitor Variables,” on page 129
- Table 29, “1: Sequencer Variables,” on page 130
- Table 30, “2: AE Variables,” on page 140
- Table 31, “3: AWB Variables,” on page 145
- Table 32, “4: Flicker Variables,” on page 151
- Table 33, “5: Auto Focus Variables,” on page 157
- Table 34, “6: AFM Variables,” on page 163
- Table 35, “7: Mode Variables,” on page 168
- Table 36, “11: HG Variables,” on page 182
- Table 37, “0: GPIO,” on page 184
- “Power-up” on page 193
- “Reset” on page 193
- “Hard Reset” on page 193
- “Standby Modes” on page 194
- “Soft Reset” on page 194
- “Hard Standby” on page 194
- “Soft Standby” on page 195
- Figure 35: “Hard Standby,” on page 195
- Table 39, “Hard Standby,” on page 195
- Figure 36: “Soft Standby,” on page 196
- “Shutdown Mode” on page 196
- Table 41, “Pin States During Conditions,” on page 196
- Figure 37: “CRA versus Image Height,” on page 198
- Figure 38: “I/O Timing Diagram,” on page 199
- Table 42, “AC Electrical Characteristics,” on page 200
- Table 43, “I/O Parameters,” on page 201
- Table 44, “DC Electrical Definitions and Characteristics,” on page 201
- Table 45, “Absolute Maximum Ratings,” on page 202
- Table 46, “POR Parameters,” on page 203
- Table 47, “Two-wire Serial Bus Characteristics,” on page 205

Rev A, Advance, ... 10/05

- Initial release