1/3.5-Inch 3.1Mp CMOS Digital Image Sensor

MT9T014

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

Features

- DigitalClarity® CMOS imaging technology
- Low dark current
- Simple two-wire serial interface
- Auto black level calibration
- Support for external mechanical shutter
- Support for external LED or xenon flash
- High frame rate preview mode with arbitrary down-size scaling from maximum resolution
- Programmable controls: gain, frame size/rate, exposure, left–right and top–bottom image reversal, window size and panning
- Data Interfaces: parallel and CCP2 compliant sub-low-voltage differential signalling (sub-LVDS)
- SMIA-compatible
- On-chip phase-locked loop (PLL) oscillator
- Bayer-pattern down-size scaler
- Integrated lens shading correction
- One-time programmable (OTP) memory for storing module information
- Superior low-light performance

Applications

- Cellular phones
- Digital still cameras
- PC cameras
- PDAs

General Description

The Micron Imaging MT9T014 is a 1/3.5-inch QXGA-format CMOS active-pixel digital image sensor with a pixel array of 2048H x 1536V (2064H x 1552V including border pixels). It incorporates sophisticated on-chip camera functions such as windowing, mirroring, column and row skip modes. It is programmable through a simple two-wire serial interface and has very low power consumption.

Table 1: Key Performance Parameters

Parameter		Value
Optical format		1/3.5-inch QXGA (4:3)
Active imager size		4.096mm(H) x 3.072mm(V) 5.12mm diagonal
Active pixels		2064H x 1552V
Pixel size		2.0 x 2.0µm
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS) and support for external mechanical shutter
Maximum data rate		64 Mp/s at parallel interface 640 Mb/s at CCP2 interface
Frame rate	QXGA (2048 x 1536)	Programmable up to 15.06 fps
	XGA (1024 x 768)	Programmable up to 47.15 fps
ADC resolution		10-bit, on-chip (61dB)
Responsivity		0.8/lux-sec
Dynamic range		68.7dB
SNR_{MAX}		38.8dB
Supply voltage	Analog	2.40–3.10V (2.80V nominal)
	Digital	1.70–1.90V (1.80V nominal)
	I/O	1.70–1.90V or 2.40–3.10V
Power consumption		TBD
Operating temperature		–30°C to +70°C
Packaging		Bare die

Ordering Information

Table 2: Available Part Numbers

Part Number	Description
MT9T014D00STCPC16DC1	Bare die

Table of Contents

List of Figures

List of Tables

General Description

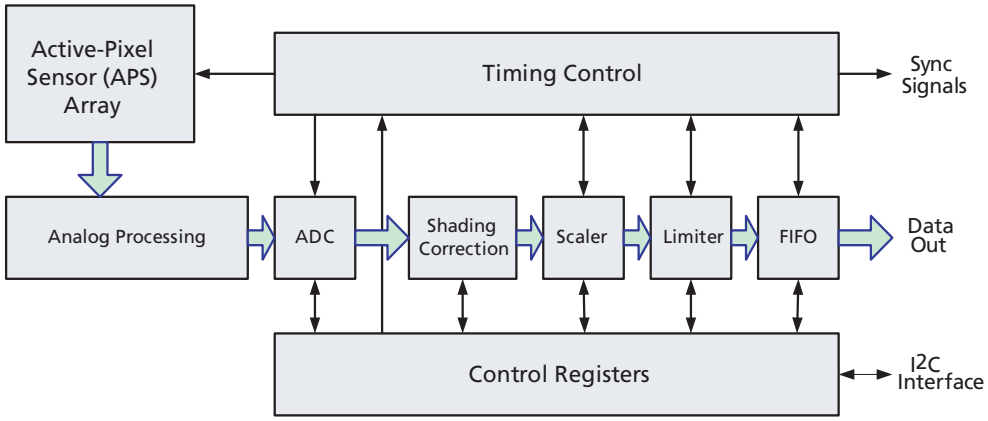
The MT9T014 digital image sensor features DigitalClarity—Micron's breakthrough low-noise CMOS imaging technology that achieves near-CCD image quality (based on signal-to-noise ratio and low-light sensitivity) while maintaining the inherent size, cost, and integration advantages of CMOS.

When operated in its default mode, the sensor generates a QXGA image at 15 frames per second (fps). An on-chip analog-to-digital converter (ADC) generates a 10-bit value for each pixel.

Functional Overview

The MT9T014 is a progressive-scan sensor that generates a stream of pixel data at a constant frame rate. It uses an on-chip, phase-locked loop (PLL) to generate all internal clocks from a single master input clock running between 6 and 48 MHz. The maximum pixel rate is 64 Mp/s, corresponding to a pixel clock rate of 64 MHz. A block diagram of the sensor is shown in Figure 1.

Figure 1: Block Diagram



The core of the sensor is a 3.1Mp 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 ADC output passes through a digital processing signal chain (which provides further data path corrections and applies digital gain).

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

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

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

The control registers, timing and control, and digital processing functions shown in Figure 1 are partitioned into three logical parts:

- A sensor core which provides array control and data path corrections. The output of the sensor core is a 10-bit parallel pixel data stream qualified by an output data clock (PIXCLK), together with LINE_VALID and FRAME_VALID signals.
- A digital shading correction block to compensate for color/brightness shading introduced by the lens or CRA curve mismatch.
- Additional functionality is provided to support the SMIA standard. This includes a horizontal and vertical image scaler, a limiter, a data compressor, an output FIFO, and a serializer.

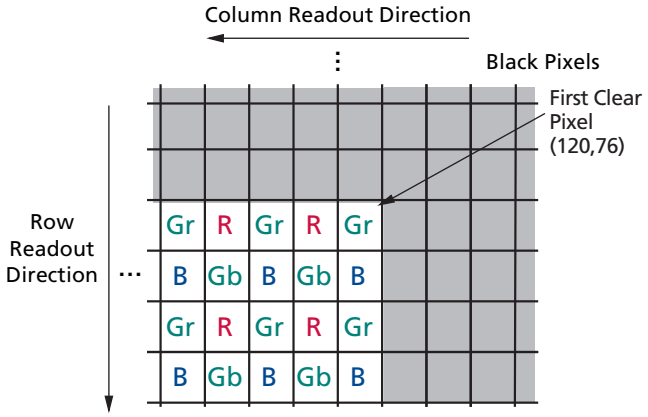
The output FIFO is present to prevent data bursts by keeping the data rate continuous. Programmable slew rates are also available to reduce the effect of electromagnetic interference from the output interface.

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

Pixel Array

The sensor core uses a Bayer color pattern, as shown in Figure 2. The even-numbered rows contain green and red pixels; odd-numbered rows contain blue and green pixels. Even-numbered columns contain green and blue pixels; odd-numbered columns contain red and green pixels.

Figure 2: Pixel Color Pattern Detail (Top Right Corner)

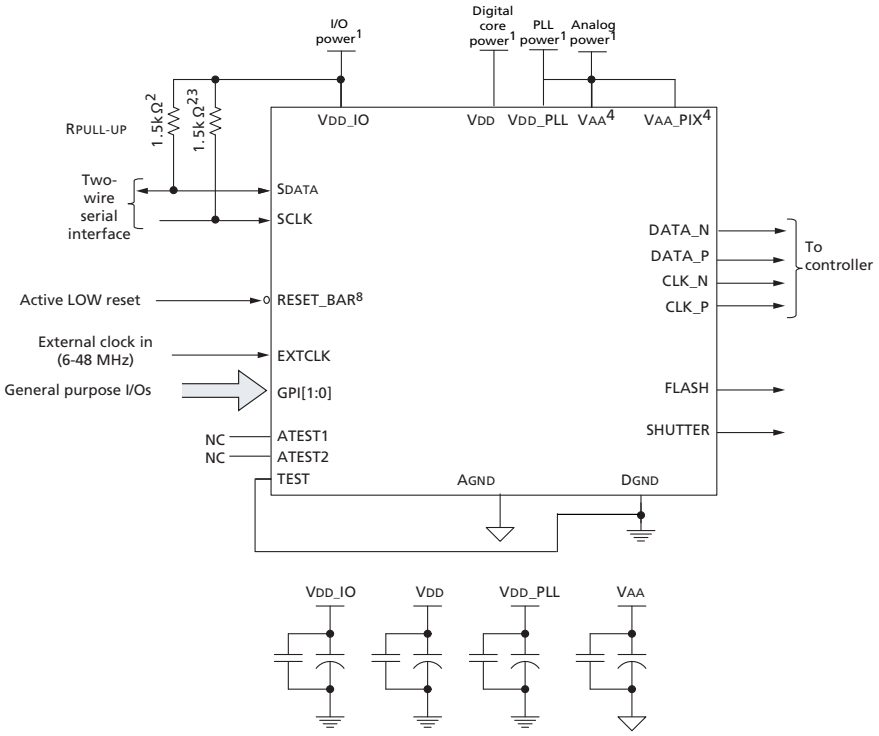


Operating Modes

By default, the MT9T014 powers up as a SMIA-compatible sensor with the serial pixel data interface enabled. A typical configuration in this mode is shown in Figure 3. The MT9T014 can also be configured to operate with a parallel pixel data interface. A typical configuration in this mode is shown in Figure 4 on page 10. These two operating modes are described in "Control of the Signal Interface" on page 65.

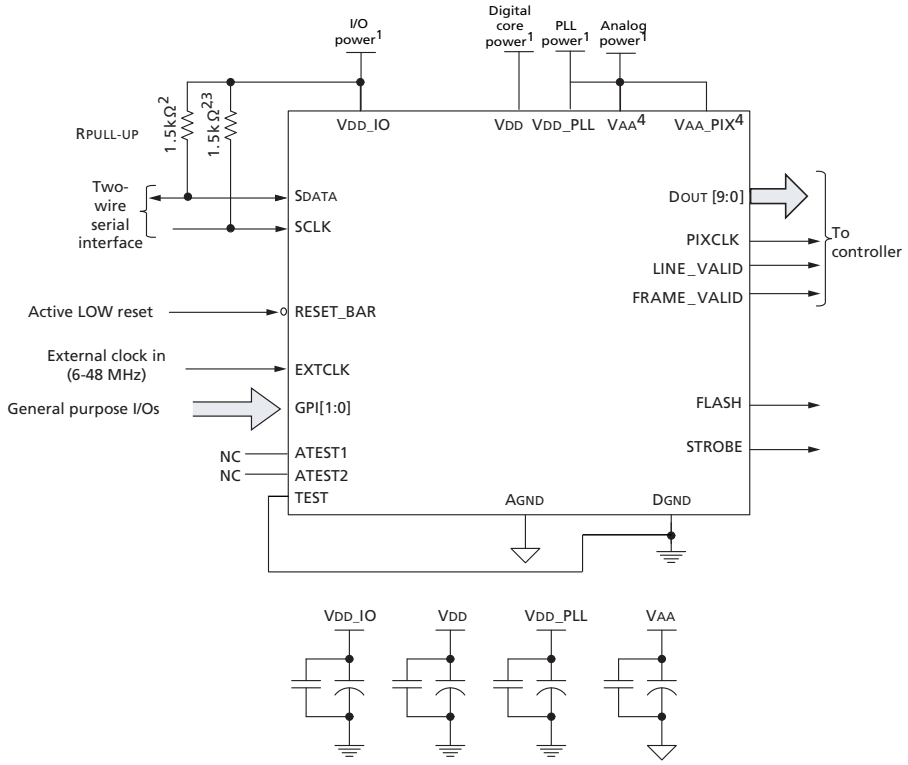
For low-noise operation, the MT9T014 requires separate power supplies for analog and digital. Incoming digital and analog ground conductors can be tied together next to the die. Both power supply rails should be decoupled to ground using capacitors as close as possible to the die. The use of inductance filters is not recommended on the power supplies or output signals.

Figure 3: Typical Configuration: Serial Pixel Data Interface



Notes:
1. All power supplies should be adequately decoupled.
2. A resistor value of 1.5kΩ is recommended, but may be greater for slower two-wire speed.
3. This pull-up resistor is not required if the controller drives a valid logic level on SCLK at all times.
4. VAA and VAA_PIX must be tied together.
5. The serial interface output pads can be left unconnected if the parallel output interface is used.
6. 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.
7. TEST must be tied to DGND.
8. Also referred to as XSHUTDOWN.
9. VPP, which can be used during the module manufacturing process, is not shown in Figure 3. This pad is left unconnected during normal operation.

Figure 4: Typical Configuration: Parallel Pixel Data Interface



Notes:
1. All power supplies should be adequately decoupled.
2. A resistor value of 1.5kΩ is recommended, but may be greater for slower two-wire speed.
3. This pull-up resistor is not required if the controller drives a valid logic level on SCLK at all times.
4. VAA and VAAPIX must be tied together.
5. 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.
6. The serial output pads can be left unconnected if the parallel output interface is used.
7. Test must be tied to GND.
8. VPP, which can be used during the module manufacturing process, is not shown in Figure 4. This pad is left unconnected during normal operation.

Signal Descriptions

Table 3 provides signal descriptions for MT9T014 die. For pad location and aperture information, refer to the MT9T014 die data sheet.

Table 3: Signal Descriptions

Pad Name	Pad Type	Description
EXTCLK	Input	Master clock input. PLL input clock. 6–48 MHz.
RESET_BAR (XSHUTDOWN)	Input	Asynchronous active LOW reset. When asserted, data output stops and all internal registers are restored to their factory default settings.
SCLK	Input	Serial clock for access to control and status registers.
GPI[1:0]	Input	General purpose inputs. After reset, these pads are powered-down by default; this means that it is not necessary to bond to these pads. Any of these pads can be configured to provide hardware control of the standby, output enable, SADDR select, and shutter trigger functions.
TEST	Input	Enable manufacturing test modes. Wire to digital GND for functional operation.
ATEST1	Input	Left floating during normal operation.
ATEST2	Input	Left floating during normal operation.
SDATA	I/O	Serial data for reads from and writes to control and status registers.
DATA_P	Output	Differential CCP2 (sub_lvds) serial clock/strobe (negative).
DATA_N	Output	Differential CCP2 (sub_lvds) serial clock/strobe (negative).
CLK_P	Output	Differential CCP2 (sub_lvds) serial clock/strobe (negative).
CLK_N	Output	Differential CCP2 (sub_lvds) serial clock/strobe (negative).
LINE_VALID	Output	LINE_VALID output. Qualified by PIXCLK.
FRAME_VALID	Output	FRAME_VALID output. Qualified by PIXCLK.
DOUT[9:0]	Output	Parallel pixel data output. Qualified by PIXCLK.
PIXCLK	Output	Pixel clock. Used to qualify the LINE_VALID, FRAME_VALID, and DOUT[9:0] outputs.
FLASH	Output	Flash output. Synchronization pulse for external light source.
SHUTTER	Output	Control for external mechanical shutter.
VAA	Supply	Power supply used to program the one-time programmable (OTP) memory. Disconnect pad when programming or when feature is not used.
VAA1, VAA2, VAA3, VAA4	Supply	Analog power supply.
VAA_PIX1, VAA_PIX2	Supply	Analog power supply for the pixel array.
AGND1, AGND2, AGND3, AGND4	Supply	Analog ground.
VDD1, VDD2, VDD3	Supply	Digital power supply.
VDD_IO1, VDD_IO2, VDD_IO3, VDD_IO4	Supply	I/O power supply.
DGND1, DGND2, DGND3 DGND4, DGND5, DGND6	Supply	Common ground for digital and I/O.
VDD_PLL	Supply	PLL power supply.

Output Data Format

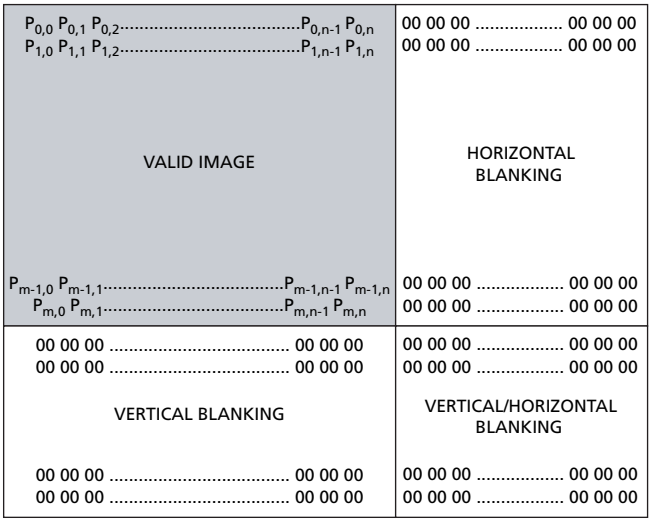
CSI Serial Pixel Data Interface

The MT9T014 serial pixel data interface implements data/clock and data/strobe signaling in accordance with the CCP2 specification. The RAW8 and RAW10 image data formats are supported.

Parallel Pixel Data Interface

MT9T014 image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking, as shown in Figure 5. 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 5: Spatial Illustration of Image Readout



Output Data Timing (Parallel Pixel Data Interface)

MT9T014 output data is synchronized with the PIXCLK output. When LINE_VALID is HIGH, one pixel value is output on the 10-bit DOUT output every PIXCLK period. The pixel clock runs at the calculated frequency based on the sensor's master input clock and internal PLL configuration, and rising edges on the PIXCLK signal occur one-half of a pixel clock period after transitions on LINE_VALID, FRAME_VALID, and DOUT (see Figure 6 on page 13). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking period. The MT9T014 can be programmed to delay the PIXCLK edge relative to the DOUT transitions. This can be achieved by programming the corresponding bits in the row_speed register.

Figure 6: Pixel Data Timing Example

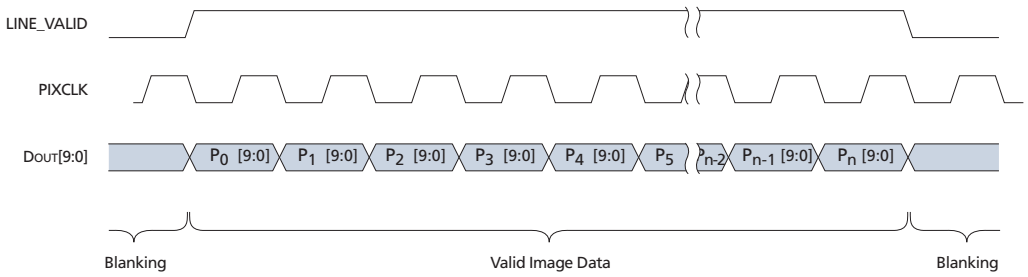


Figure 7: Row Timing and FRAME_VALID/LINE_VALID Signals

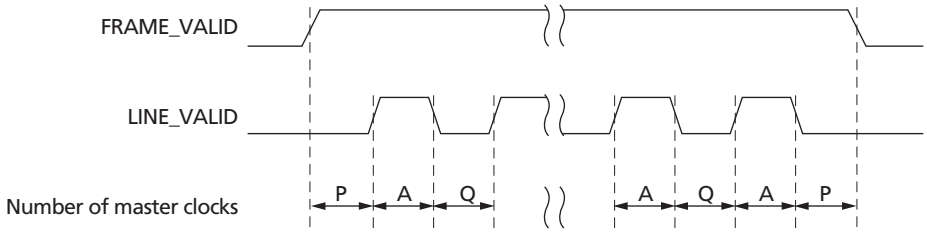


Table 4: Row Timing

Parameter	Name	Equation	Default Timing at 64 MHz
PIXCLK_PERIOD	Pixel clock period	R0x3016–7[2:0] / vt_pix_clk_freq_mhz	1 pixel clock = 15.625ns
S	Skip (subsampling) factor	For x_odd_inc = y_odd_inc = 3, S = 2. Otherwise, S = 1	1
A	Active data time	(x_addr_end - x_addr_start + x_odd_inc) * PIXCLK_PERIOD/S	2048 pixel clocks = 32.00µs
P	Frame start/end blanking	6 * PIXCLK_PERIOD	6 pixel clocks = 93.75ns
Q	Horizontal blanking	(line_length_pck - A) * PIXCLK_PERIOD	574 pixel clocks = 8.97µs
A + Q	Row time	line_length_pck * PIXCLK_PERIOD	2622 pixel clocks = 40.97µs
N	Number of rows	(y_addr_end - y_addr_start + y_odd_inc)/S	1,536 rows
V	Vertical blanking	([frame_length_lines - N] * [A+Q]) + Q - (2*P)	223432 pixel clocks = 3.49ms
T	Frame valid time	(N * (A + Q)) - Q + (2*P)	4026830 pixel clocks = 69.92ms
F	Total frame time	line_length_pck * frame_length_lines * PIXCLK_PERIOD	4250262 pixel clocks = 66.41ms

The sensor timing (Table 4) is shown in terms of pixel clock and master clock cycles (see Figure 6). The default settings for the on-chip PLL generate a 64 MHz pixel clock given a 20 MHz input clock to the MT9T014. Equations for calculating the frame rate are given in “Frame Rate Control” on page 84.

Two-Wire Serial Register Interface

The two-wire serial interface bus enables read/write access to control and status registers within the MT9T014. This interface is designed to be compatible with the SMIA 1.0 Part2: CCP2 Specification camera control interface (CCI) which uses the electrical characteristics and transfer protocols of the I^2C specification. The protocols described in the I^2C specification allow the slave device to drive SCLK LOW; the MT9T014 uses SCLK as an input only and therefore never drives it LOW.

Protocol

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

1. a (repeated) start condition
2. a slave address/data direction byte
3. an (a no) acknowledge bit
4. a message byte
5. a stop condition

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

Start Condition

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

Stop Condition

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

Data Transfer

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

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

Slave Address/Data Direction Byte

Bits [7:1] of this byte represent the device slave address and bit [0] indicates the data transfer direction. A "0" in bit [0] indicates a write, and a "1" indicates a read. The default slave addresses used by the MT9T014 are 0x20 (write address) and 0x21 (read address) in accordance with the SMIA specification. Alternate slave addresses of 0x30 (write address) and 0x31 (read address) can be selected by enabling and asserting the SADDR signal through the GPI pad.

An alternate slave address can also be programmed via R0x31FC.

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 SMIA CCI.

Acknowledge Bit

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

No-Acknowledge Bit

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

Typical Sequence

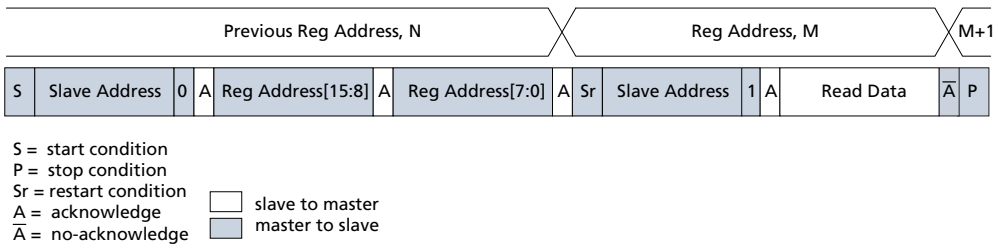
A typical read or write sequence begins by the master generating a start condition on the bus. After the start condition, the master sends the 8-bit slave address/data direction byte. The last bit indicates whether the request is for a read or a write, where a "0" indicates a write and a "1" indicates 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 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 (Figure 8) 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 8 shows how the internal register address maintained by the MT9T014 is loaded and incremented as the sequence proceeds.

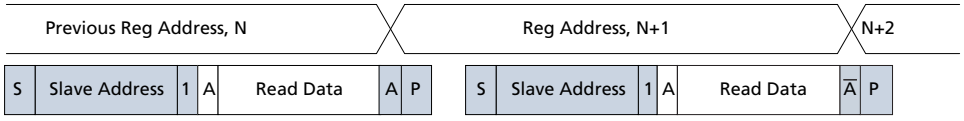
Figure 8: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 9) performs a read using the current value of the MT9D012 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.

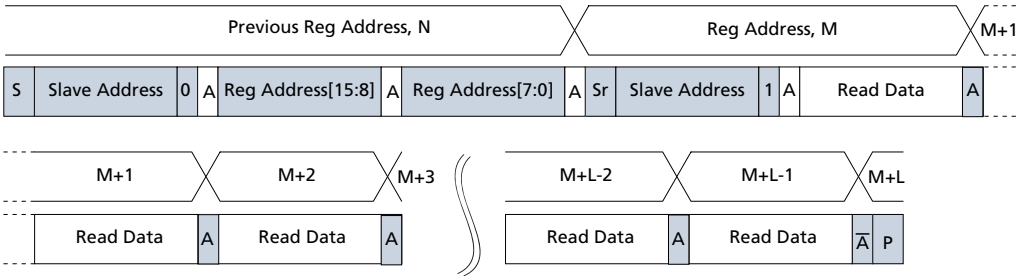
Figure 9: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 10) starts in the same way as the single READ from random location (Figure 8). 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.

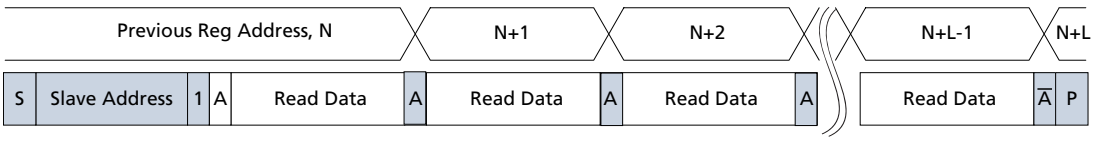
Figure 10: Sequential READ, Start from Random Location



Sequential READ, Start from Current Location

This sequence (Figure 11) starts in the same way as the single READ from current location (Figure 9 on page 16). 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.

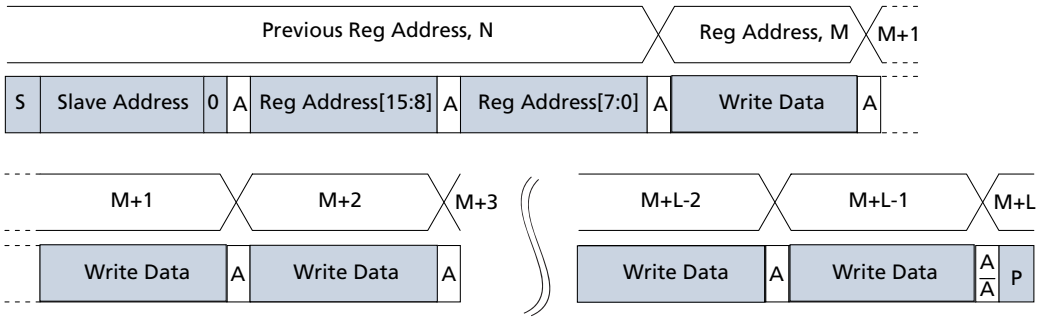
Figure 11: Sequential READ, Start from Current Location



Single WRITE to Random Location

This sequence (Figure 12) 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.

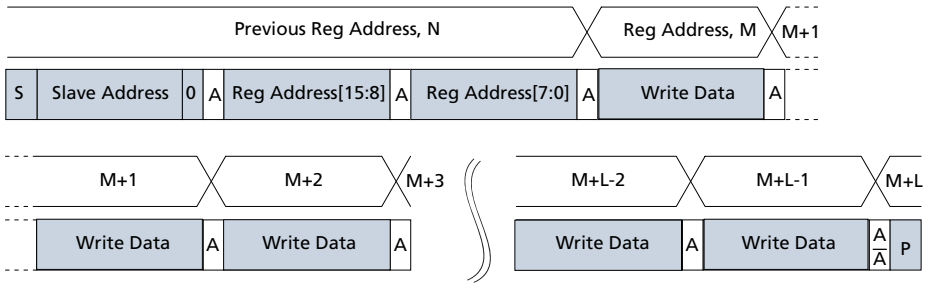
Figure 12: Single WRITE to Random Location



Sequential WRITE, Start at Random Location

This sequence (Figure 13 on page 17) starts in the same way as the single WRITE to random location (Figure 12). Instead of generating a stop condition after the first byte of data has been transferred, the master continues to perform byte writes until *L* bytes have been written. The WRITE is terminated by the master generating a stop condition.

Figure 13: Sequential Write, Start at Random Location



Registers

The MT9T014 provides a 32-bit register address space accessed through a serial interface ("Single READ from Random Location" on page 15). Each register location is 8 or 16 bits in size.

The address space is divided into the five major regions shown in Table 5.

Table 5: Address Space Regions

Address Range	Description
0x0000–0x0FFF	Configuration registers (read-only and read-write dynamic registers)
0x1000–0x1FFF	Parameter limit registers (read-only static registers)
0x2000–0x2FFF	Image statistics registers (none currently defined)
0x3000–0x3FFF	Manufacturer-specific registers (read-only and read-write dynamic registers)
0x4000–0xFFFF	Reserved (undefined)

Register Notation

The underlying mechanism for reading and writing registers provides byte write capability. However, it is convenient to consider some registers as multiple adjacent bytes. The MT9T014 uses 8-bit, 16-bit, and 32-bit registers, all implemented as 1 or more bytes at naturally aligned, contiguous locations in the address space.

In this document, registers are described either by address or by name. When registers are described by address, the size of the registers is explicit. For example, R0x3024 is an 8-bit register at address 0x3024, and R0x3000–1 is a 16-bit register at address 0x3000-0x3001. When registers are described by name, the size of the register is implicit. It is necessary to refer to the register table to determine that model_id is a 16-bit register.

Register Aliases

A consequence of the internal architecture of the MT9T014 is that some registers are decoded at multiple addresses. Some registers in "configuration space" are also decoded in "manufacturer specific space." In order to provide unique names for all registers, the name of the register within manufacturer specific register space has a trailing underscore. For example, R0x0000–1 is model_id, and R0x3000–1 is model_id_ (see the register table for more examples). The effect of reading or writing a register through any of its aliases is identical.

Bit Fields

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

Bit Field Aliases

In addition to the register aliases described above, some register fields are aliased in multiple places. For example, R0x0100 (mode_select) only has one operational bit, R0x0100[0]. This bit is aliased to R0x301A–B[2]. The effect of reading or writing a bit field through any of its aliases is identical.

Byte Ordering

Registers that occupy more than one byte of address space are shown with the lowest address in the highest-order byte lane to match the byte-ordering on the SMIA bus. For example, the model_id register is R0x0000–1. In the register table the default value is shown as 0x2600. This means that a read from address 0x0000 would return 0x26, and a read from address 0x0001 would return 0x00. When reading this register as two 8-bit transfers on the serial interface, the 0x26 will appear on the serial interface first, followed by the 0x00.

Address Alignment

All register addresses are aligned naturally. Registers that occupy 2 bytes of address space are aligned to even 16-bit addresses, and registers that occupy 4 bytes of address space are aligned to 16-bit addresses that are an integer multiple of 4.

Bit Representation

For clarity, 32-bit hex numbers are shown with an underscore between the upper and lower 16 bits. For example: 0x3000_01AB.

Data Format

Most registers represent an unsigned binary value or set of bit fields. For all other register formats, the format is stated explicitly at the start of the register description. The notation for these formats is shown in Table 6.

Table 6: Data Formats

Name	Description
FIX16	Signed fixed-point, 16-bit number: two's complement number, 8 fractional bits. Examples: 0x0100 = 1.0, 0x8000 = –128, 0xFFFF = –0.0039065
UFIX16	Unsigned fixed-point, 16-bit number: 8.8 format. Examples: 0x0100 = 1.0, 0x280 = 2.5
FLP32	Signed floating-point, 32-bit number: IEEE 754 format. Example: 0x4280_0000 = 64.0

Register Behavior

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

Double-Buffered Registers

Some sensor settings cannot be changed during frame readout. For example, changing R0x0344–5 (x_addr_start) partway through frame readout would result in inconsistent row lengths within a frame. To avoid this, the MT9T014 double buffers many registers by implementing a "pending" and a "live" version. Reads and writes access the pending register. The live register controls the sensor operation.

The value in the pending register is transferred to a live register at a fixed point in the frame timing, called frame start. Frame start is defined as the point at which the first dark row is read out internally to the sensor. In the register tables the "Sync'd" column shows which registers or register fields are double-buffered in this way.

Using grouped_parameter_hold

Register grouped_parameter_hold (R0x0104) can be used to inhibit transfers from the pending to the live registers. When the MT9T014 is in streaming mode, this register should be written to "1" before making changes to any group of registers where a set of changes is required to take effect simultaneously. When this register is written to "0," all transfers from pending to live registers take place on the next frame start.

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

– An external auto exposure algorithm might want to change both gain and integration time between two frames. If the next frame starts between these operations, it will have the new gain, but not the new integration time, which would return a frame with the wrong brightness that might lead to a feedback loop with the AE algorithm resulting in flickering.

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 have changed during the frame.

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

By default, bad frames are not masked. If the masked bad frame option is enabled, both LINE_VALID and FRAME_VALID are inhibited for these frames so that the vertical blanking time between frames is extended by the frame time.

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

N—No. Changing the register value will not produce a bad frame.
Y—Yes. Changing the register value might produce a bad frame.
YM—Yes; but the bad frame will be masked out when mask_corrupted_frames (R0x0105) is set to "1."

Changes to Integration Time

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

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

Changes to Gain Settings

Usually, when the gain settings are changed, the gain is updated on the next frame start. 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. In this case, a new gain should not be set during the extra frame delay. There is an option to turn off the extra frame delay by setting reset_register[14] bit.

Embedded Data

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

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

- The integration time is delayed by one further frame, so that the value corresponds to the integration time used for the image data in the frame. See “Changes to Integration Time” on page 20.
- The PLL timing registers are not double-buffered, since the result of changing them in streaming mode is undefined. Therefore, the pending and live values for these registers are equivalent.

Register Map

Table 7 shows the locations used within the address space. Locations that are not shown in the table are reserved for future use; they return 0x00 on read, but should not be read from or written to in order to maintain compatibility with future designs. Locations that are shown as "Reserved" should not be accessed. The default read values of these registers is subject to change. The effect of writing to reserved registers is undefined and may include the possibility of causing permanent electrical damage to the sensor.

Register List and Default Values

Table 7: Register List and Default Values - SMIA Configuration

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x0000	MODEL_ID	dddd dddd dddd dddd	5635 (0x1603)
R0x0002	REVISION_NUMBER	???? ????	0 (0x0000)
R0x0003	MANUFACTURER_ID	???? ????	6 (0x0006)
R0x0004	SMIA_VERSION	???? ????	10 (0x000A)
R0x0005	FRAME_COUNT	???? ????	255 (0x00FF)
R0x0006	PIXEL_ORDER	0000 00??	0 (0x0000)
R0x0008	DATA_PEDESTAL	0000 00dd dddd dddd	42 (0x002A)
R0x0040	FRAME_FORMAT_MODEL_TYPE	???? ????	1 (0x0001)
R0x0041	FRAME_FORMAT_MODEL_SUBTYPE	???? ????	18 (0x0012)
R0x0042	FRAME_FORMAT_DESCRIPTOR_0	???? ???? ???? ????	22528 (0x5800)
R0x0044	FRAME_FORMAT_DESCRIPTOR_1	???? ???? ???? ????	4098 (0x1002)
R0x0046	FRAME_FORMAT_DESCRIPTOR_2	???? ???? ???? ????	22016 (0x5600)
R0x0048	FRAME_FORMAT_DESCRIPTOR_3	???? ???? ???? ????	0 (0x0000)
R0x004A	FRAME_FORMAT_DESCRIPTOR_4	???? ???? ???? ????	0 (0x0000)
R0x004C	FRAME_FORMAT_DESCRIPTOR_5	???? ???? ???? ????	0 (0x0000)
R0x004E	FRAME_FORMAT_DESCRIPTOR_6	???? ???? ???? ????	0 (0x0000)
R0x0050	FRAME_FORMAT_DESCRIPTOR_7	???? ???? ???? ????	0 (0x0000)
R0x0052	FRAME_FORMAT_DESCRIPTOR_8	???? ???? ???? ????	0 (0x0000)
R0x0054	FRAME_FORMAT_DESCRIPTOR_9	???? ???? ???? ????	0 (0x0000)
R0x0056	FRAME_FORMAT_DESCRIPTOR_10	???? ???? ???? ????	0 (0x0000)
R0x0058	FRAME_FORMAT_DESCRIPTOR_11	???? ???? ???? ????	0 (0x0000)
R0x005A	FRAME_FORMAT_DESCRIPTOR_12	???? ???? ???? ????	0 (0x0000)
R0x005C	FRAME_FORMAT_DESCRIPTOR_13	???? ???? ???? ????	0 (0x0000)
R0x005E	FRAME_FORMAT_DESCRIPTOR_14	???? ???? ???? ????	0 (0x0000)
R0x0080	ANALOGUE_GAIN_CAPABILITY	???? ???? ???? ????	1 (0x0001)
R0x0084	ANALOGUE_GAIN_CODE_MIN	???? ???? ???? ????	8 (0x0008)
R0x0086	ANALOGUE_GAIN_CODE_MAX	???? ???? ???? ????	127 (0x007F)
R0x0088	ANALOGUE_GAIN_CODE_STEP	???? ???? ???? ????	1 (0x0001)
R0x008A	ANALOGUE_GAIN_TYPE	???? ???? ???? ????	0 (0x0000)
R0x008C	ANALOGUE_GAIN_M0	???? ???? ???? ????	1 (0x0001)
R0x008E	ANALOGUE_GAIN_C0	???? ???? ???? ????	0 (0x0000)
R0x0090	ANALOGUE_GAIN_M1	???? ???? ???? ????	0 (0x0000)

Table 7: Register List and Default Values - SMIA Configuration (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x0092	ANALOGUE_GAIN_C1	???? ???? ???? ????	8 (0x0008)
R0x00C0	DATA_FORMAT_MODEL_TYPE	???? ????	1 (0x0001)
R0x00C1	DATA_FORMAT_MODEL_SUBTYPE	???? ????	3 (0x0003)
R0x00C2	DATA_FORMAT_DESCRIPTOR_0	???? ???? ???? ????	2570 (0x0A0A)
R0x00C4	DATA_FORMAT_DESCRIPTOR_1	???? ???? ???? ????	2056 (0x0808)
R0x00C6	DATA_FORMAT_DESCRIPTOR_2	???? ???? ???? ????	2568 (0x0A08)
R0x00C8	DATA_FORMAT_DESCRIPTOR_3	???? ???? ???? ????	0 (0x0000)
R0x00CA	DATA_FORMAT_DESCRIPTOR_4	???? ???? ???? ????	0 (0x0000)
R0x00CC	DATA_FORMAT_DESCRIPTOR_5	???? ???? ???? ????	0 (0x0000)
R0x00CE	DATA_FORMAT_DESCRIPTOR_6	???? ???? ???? ????	0 (0x0000)
R0x0100	MODE_SELECT	0000 000d	0 (0x0000)
R0x0101	IMAGE_ORIENTATION	0000 00dd	0 (0x0000)
R0x0103	SOFTWARE_RESET	0000 000d	0 (0x0000)
R0x0104	GROUPED_PARAMETER_HOLD	0000 000d	0 (0x0000)
R0x0105	MASK_CORRUPTED_FRAMES	0000 000d	0 (0x0000)
R0x0110	CCP2_CHANNEL_IDENTIFIER	0000 0ddd	0 (0x0000)
R0x0111	CCP2_SIGNALLING_MODE	0000 000d	1 (0x0001)
R0x0112	CCP_DATA_FORMAT	dddd dddd dddd dddd	2570 (0x0A0A)
R0x0120	GAIN_MODE	0000 000d	0 (0x0000)
R0x0200	FINE_INTEGRATION_TIME	00dd dddd dddd dddd	445 (0x01BD)
R0x0202	COARSE_INTEGRATION_TIME	dddd dddd dddd dddd	16 (0x0010)
R0x0204	ANALOGUE_GAIN_CODE_GLOBAL	0000 0000 0ddd dddd	8 (0x0008)
R0x0206	ANALOGUE_GAIN_CODE_GREENR	0000 0000 0ddd dddd	8 (0x0008)
R0x0208	ANALOGUE_GAIN_CODE_RED	0000 0000 0ddd dddd	8 (0x0008)
R0x020A	ANALOGUE_GAIN_CODE_BLUE	0000 0000 0ddd dddd	8 (0x0008)
R0x020C	ANALOGUE_GAIN_CODE_GREENB	0000 0000 0ddd dddd	8 (0x0008)
R0x020E	DIGITAL_GAIN_GREENR	0000 0ddd 0000 0000	256 (0x0100)
R0x0210	DIGITAL_GAIN_RED	0000 0ddd 0000 0000	256 (0x0100)
R0x0212	DIGITAL_GAIN_BLUE	0000 0ddd 0000 0000	256 (0x0100)
R0x0214	DIGITAL_GAIN_GREENB	0000 0ddd 0000 0000	256 (0x0100)
R0x0300	VT_PIX_CLK_DIV	0000 0000 000d dddd	10 (0x000A)
R0x0302	VT_SYS_CLK_DIV	0000 0000 000d dddd	1 (0x0001)
R0x0304	PRE_PLL_CLK_DIV	0000 0000 00dd dddd	2 (0x0002)
R0x0306	PLL_MULTIPLIER	0000 0000 dddd dddd	80 (0x0050)
R0x0308	OP_PIX_CLK_DIV	0000 0000 000d dddd	10 (0x000A)
R0x030A	OP_SYS_CLK_DIV	0000 0000 000d dddd	1 (0x0001)
R0x0340	FRAME_LENGTH_LINES	dddd dddd dddd dddd	1621 (0x0655)
R0x0342	LINE_LENGTH_PCK	00dd dddd dddd dddd	2622 (0x0A3E)
R0x0344	X_ADDR_START	0000 dddd dddd dddd	8 (0x0008)
R0x0346	Y_ADDR_START	0000 0ddd dddd dddd	8 (0x0008)
R0x0348	X_ADDR_END	0000 dddd dddd dddd	2055 (0x0807)
R0x034A	Y_ADDR_END	0000 0ddd dddd dddd	1543 (0x0607)
R0x034C	X_OUTPUT_SIZE	0000 dddd dddd ddd0	2048 (0x0800)
R0x034E	Y_OUTPUT_SIZE	0000 0ddd dddd ddd0	1536 (0x0600)

Table 7: Register List and Default Values - SMIA Configuration (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x0380	X_EVEN_INC	0000 0000 0000 000?	1 (0x0001)
R0x0382	X_ODD_INC	0000 0000 0000 0ddd	1 (0x0001)
R0x0384	Y_EVEN_INC	0000 0000 0000 000?	1 (0x0001)
R0x0386	Y_ODD_INC	0000 0000 0000 0ddd	1 (0x0001)
R0x0400	SCALING_MODE	0000 0000 0000 00dd	0 (0x0000)
R0x0402	SPATIAL_SAMPLING	0000 0000 0000 000d	0 (0x0000)
R0x0404	SCALE_M	0000 0000 dddd dddd	16 (0x0010)
R0x0406	SCALE_N	0000 0000 ???? ????	16 (0x0010)
R0x0500	COMPRESSION_MODE	0000 0000 0000 000?	1 (0x0001)
R0x0600	TEST_PATTERN_MODE	0000 000d 0000 0ddd	0 (0x0000)
R0x0602	TEST_DATA_RED	0000 00dd dddd dddd	0 (0x0000)
R0x0604	TEST_DATA_GREENR	0000 00dd dddd dddd	0 (0x0000)
R0x0606	TEST_DATA_BLUE	0000 00dd dddd dddd	0 (0x0000)
R0x0608	TEST_DATA_GREENB	0000 00dd dddd dddd	0 (0x0000)
R0x060A	HORIZONTAL_CURSOR_WIDTH	0000 0ddd dddd dddd	0 (0x0000)
R0x060C	HORIZONTAL_CURSOR_POSITION	0000 0ddd dddd dddd	0 (0x0000)
R0x060E	VERTICAL_CURSOR_WIDTH	0000 dddd dddd dddd	0 (0x0000)
R0x0610	VERTICAL_CURSOR_POSITION	0000 dddd dddd dddd	0 (0x0000)

Table 8: Register List and Default Values - SMIA Parameter Limits

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x1000	INTEGRATION_TIME_CAPABILITY	0000 0000 0000 000?	1 (0x0001)
R0x1004	COARSE_INTEGRATION_TIME_MIN	dddd dddd dddd dddd	0 (0x0000)
R0x1006	COARSE_INTEGRATION_TIME_MAX_MARGIN	dddd dddd dddd dddd	1 (0x0001)
R0x1008	FINE_INTEGRATION_TIME_MIN	dddd dddd dddd dddd	445 (0x01BD)
R0x100A	FINE_INTEGRATION_TIME_MAX_MARGIN	dddd dddd dddd dddd	207 (0x00CF)
R0x1080	DIGITAL_GAIN_CAPABILITY	???? ???? ???? ????	1 (0x0001)
R0x1084	DIGITAL_GAIN_MIN	???? ???? ???? ????	256 (0x0100)
R0x1086	DIGITAL_GAIN_MAX	???? ???? ???? ????	1792 (0x0700)
R0x1088	DIGITAL_GAIN_STEP_SIZE	???? ???? ???? ????	256 (0x0100)
R0x1100	MIN_EXT_CLK_FREQ_MHZ_1	???? ???? ???? ????	16576 (0x40C0)
R0x1102	MIN_EXT_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1104	MAX_EXT_CLK_FREQ_MHZ_1	???? ???? ???? ????	16856 (0x41D8)
R0x1106	MAX_EXT_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1108	MIN_PRE_PLL_CLK_DIV	???? ???? ???? ????	1 (0x0001)
R0x110A	MAX_PRE_PLL_CLK_DIV	???? ???? ???? ????	64 (0x0040)
R0x110C	MIN_PLL_IP_FREQ_MHZ_1	???? ???? ???? ????	16384 (0x4000)
R0x110E	MIN_PLL_IP_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)

Table 8: Register List and Default Values - SMIA Parameter Limits (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x1110	MAX_PLL_IP_FREQ_MHZ_1	???? ???? ???? ????	16696 (0x4138)
R0x1112	MAX_PLL_IP_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1114	MIN_PLL_MULTIPLIER	???? ???? ???? ????	1 (0x0001)
R0x1116	MAX_PLL_MULTIPLIER	???? ???? ???? ????	254 (0x00FE)
R0x1118	MIN_PLL_OP_FREQ_MHZ_1	???? ???? ???? ????	17184 (0x4320)
R0x111A	MIN_PLL_OP_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x111C	MAX_PLL_OP_FREQ_MHZ_1	???? ???? ???? ????	17460 (0x4434)
R0x111E	MAX_PLL_OP_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1120	MIN_VT_SYS_CLK_DIV	???? ???? ???? ????	1 (0x0001)
R0x1122	MAX_VT_SYS_CLK_DIV	???? ???? ???? ????	32 (0x0020)
R0x1124	MIN_VT_SYS_CLK_FREQ_MHZ_1	???? ???? ???? ????	16768 (0x4180)
R0x1126	MIN_VT_SYS_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1128	MAX_VT_SYS_CLK_FREQ_MHZ_1	???? ???? ???? ????	17440 (0x4420)
R0x112A	MAX_VT_SYS_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x112C	MIN_VT_PIX_CLK_FREQ_MHZ_1	???? ???? ???? ????	16640 (0x4100)
R0x112E	MIN_VT_PIX_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1130	MAX_VT_PIX_CLK_FREQ_MHZ_1	???? ???? ???? ????	17024 (0x4280)
R0x1132	MAX_VT_PIX_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1134	MIN_VT_PIX_CLK_DIV	???? ???? ???? ????	4 (0x0004)
R0x1136	MAX_VT_PIX_CLK_DIV	???? ???? ???? ????	10 (0x000A)
R0x1140	MIN_FRAME_LENGTH_LINES	dddd dddd dddd dddd	87 (0x0057)
R0x1142	MAX_FRAME_LENGTH_LINES	dddd dddd dddd dddd	65535 (0xFFFF)
R0x1144	MIN_LINE_LENGTH_PCK	dddd dddd dddd dddd	826 (0x033A)
R0x1146	MAX_LINE_LENGTH_PCK	dddd dddd dddd dddd	16383 (0x3FFF)
R0x1148	MIN_LINE_BLANKING_PCK	dddd dddd dddd dddd	570 (0x023A)
R0x114A	MIN_FRAME_BLANKING_LINES	dddd dddd dddd dddd	85 (0x0055)
R0x1160	MIN_OP_SYS_CLK_DIV	???? ???? ???? ????	1 (0x0001)
R0x1162	MAX_OP_SYS_CLK_DIV	???? ???? ???? ????	32 (0x0020)
R0x1164	MIN_OP_SYS_CLK_FREQ_MHZ_1	???? ???? ???? ????	17024 (0x4280)
R0x1166	MIN_OP_SYS_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1168	MAX_OP_SYS_CLK_FREQ_MHZ_1	???? ???? ???? ????	17440 (0x4420)
R0x116A	MAX_OP_SYS_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x116C	MIN_OP_PIX_CLK_DIV	???? ???? ???? ????	8 (0x0008)
R0x116E	MAX_OP_PIX_CLK_DIV	???? ???? ???? ????	10 (0x000A)
R0x1170	MIN_OP_PIX_CLK_FREQ_MHZ_1	???? ???? ???? ????	16640 (0x4100)
R0x1172	MIN_OP_PIX_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1174	MAX_OP_PIX_CLK_FREQ_MHZ_1	???? ???? ???? ????	17024 (0x4280)
R0x1176	MAX_OP_PIX_CLK_FREQ_MHZ_2	???? ???? ???? ????	0 (0x0000)
R0x1180	X_ADDR_MIN	???? ???? ???? ????	0 (0x0000)
R0x1182	Y_ADDR_MIN	???? ???? ???? ????	0 (0x0000)
R0x1184	X_ADDR_MAX	???? ???? ???? ????	2063 (0x080F)
R0x1186	Y_ADDR_MAX	???? ???? ???? ????	1551 (0x060F)
R0x11C0	MIN_EVEN_INC	???? ???? ???? ????	1 (0x0001)
R0x11C2	MAX_EVEN_INC	???? ???? ???? ????	1 (0x0001)

Table 8: Register List and Default Values - SMIA Parameter Limits (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x11C4	MIN_ODD_INC	???? ???? ???? ????	1 (0x0001)
R0x11C6	MAX_ODD_INC	???? ???? ???? ????	3 (0x0003)
R0x1200	SCALING_CAPABILITY	???? ???? ???? ????	2 (0x0002)
R0x1204	SCALER_M_MIN	???? ???? ???? ????	16 (0x0010)
R0x1206	SCALER_M_MAX	???? ???? ???? ????	128 (0x0080)
R0x1208	SCALER_N_MIN	???? ???? ???? ????	16 (0x0010)
R0x120A	SCALER_N_MAX	???? ???? ???? ????	16 (0x0010)
R0x1300	COMPRESSION_CAPABILITY	???? ???? ???? ????	1 (0x0001)
R0x1400	MATRIX_ELEMENT_REDINRED	dddd dddd dddd dddd	256 (0x0100)
R0x1402	MATRIX_ELEMENT_GREENINRED	dddd dddd dddd dddd	0 (0x0000)
R0x1404	MATRIX_ELEMENT_BLUEINRED	dddd dddd dddd dddd	0 (0x0000)
R0x1406	MATRIX_ELEMENT_REDINGREEN	dddd dddd dddd dddd	0 (0x0000)
R0x1408	MATRIX_ELEMENT_GREENINGREEN	dddd dddd dddd dddd	256 (0x0100)
R0x140A	MATRIX_ELEMENT_BLUEINGREEN	dddd dddd dddd dddd	0 (0x0000)
R0x140C	MATRIX_ELEMENT_REDINBLUE	dddd dddd dddd dddd	0 (0x0000)
R0x140E	MATRIX_ELEMENT_GREENINBLUE	dddd dddd dddd dddd	0 (0x0000)
R0x1410	MATRIX_ELEMENT_BLUEINBLUE	dddd dddd dddd dddd	256 (0x0100)

Table 9: Register List and Default Values - Manufacturer-Specific

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x3000	MODEL_ID_	dddd dddd dddd dddd	5635 (0x1603)
R0x3002	Y_ADDR_START_	0000 0ddd dddd dddd	8 (0x0008)
R0x3004	X_ADDR_START_	0000 dddd dddd dddd	8 (0x0008)
R0x3006	Y_ADDR_END_	0000 0ddd dddd dddd	1543 (0x0607)
R0x3008	X_ADDR_END_	0000 dddd dddd dddd	2055 (0x0807)
R0x300A	FRAME_LENGTH_LINES_	dddd dddd dddd dddd	1621 (0x0655)
R0x300C	LINE_LENGTH_PCK_	00dd dddd dddd dddd	2622 (0x0A3E)
R0x3010	FINE_CORRECTION	00dd dddd dddd dddd	176 (0x00B0)
R0x3012	COARSE_INTEGRATION_TIME_	dddd dddd dddd dddd	16 (0x0010)
R0x3014	FINE_INTEGRATION_TIME_	00dd dddd dddd dddd	445 (0x01BD)
R0x3016	ROW_SPEED	0000 0ddd 0ddd 0ddd	273 (0x0111)
R0x3018	EXTRA_DELAY	00dd dddd dddd dddd	0 (0x0000)
R0x301A	RESET_REGISTER	dd0d 0ddd dddd dddd	24 (0x0018)
R0x301C	MODE_SELECT_	0000 000d	0 (0x0000)
R0x301D	IMAGE_ORIENTATION_	0000 00dd	0 (0x0000)
R0x301E	DATA_PEDESTAL_	0000 00dd dddd dddd	42 (0x002A)
R0x3021	SOFTWARE_RESET_	0000 000d	0 (0x0000)
R0x3022	GROUPED_PARAMETER_HOLD_	0000 000d	0 (0x0000)

Table 9: Register List and Default Values - Manufacturer-Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x3023	MASK_CORRUPTED_FRAMES_	0000 000d	0 (0x0000)
R0x3024	PIXEL_ORDER_	0000 00??	0 (0x0000)
R0x3026	GPI_STATUS	dddd dddd dddd ????	65523 (0xFFF3)
R0x3028	ANALOGUE_GAIN_CODE_GLOBAL_	0000 0000 0ddd dddd	8 (0x0008)
R0x302A	ANALOGUE_GAIN_CODE_GREENR_	0000 0000 0ddd dddd	8 (0x0008)
R0x302C	ANALOGUE_GAIN_CODE_RED_	0000 0000 0ddd dddd	8 (0x0008)
R0x302E	ANALOGUE_GAIN_CODE_BLUE_	0000 0000 0ddd dddd	8 (0x0008)
R0x3030	ANALOGUE_GAIN_CODE_GREENB_	0000 0000 0ddd dddd	8 (0x0008)
R0x3032	DIGITAL_GAIN_GREENR_	0000 0ddd 0000 0000	256 (0x0100)
R0x3034	DIGITAL_GAIN_RED_	0000 0ddd 0000 0000	256 (0x0100)
R0x3036	DIGITAL_GAIN_BLUE_	0000 0ddd 0000 0000	256 (0x0100)
R0x3038	DIGITAL_GAIN_GREENB_	0000 0ddd 0000 0000	256 (0x0100)
R0x303A	SMIA_VERSION_	???? ????	10 (0x000A)
R0x303B	FRAME_COUNT_	???? ????	255 (0x00FF)
R0x303C	FRAME_STATUS	0000 0000 0000 00??	0 (0x0000)
R0x3040	READ_MODE	dd0d ddd0 dddd dddd	36 (0x0024)
R0x3046	FLASH	??dd dddd d000 0000	1536 (0x0600)
R0x3048	FLASH_COUNT	0000 00dd dddd dddd	8 (0x0008)
R0x3056	GREEN1_GAIN	0000 ddd? dddd dddd	528 (0x0210)
R0x3058	BLUE_GAIN	0000 ddd? dddd dddd	528 (0x0210)
R0x305A	RED_GAIN	0000 ddd? dddd dddd	528 (0x0210)
R0x305C	GREEN2_GAIN	0000 ddd? dddd dddd	528 (0x0210)
R0x305E	GLOBAL_GAIN	0000 ddd0 dddd dddd	528 (0x0210)
R0x306A	DATAPATH_STATUS	0000 0000 00?d dddd	0 (0x0000)
R0x306E	DATAPATH_SELECT	dddd dd00 d00d 0000	36864 (0x9000)
R0x3070	TEST_PATTERN_MODE_	0000 000d 0000 0ddd	0 (0x0000)
R0x3072	TEST_DATA_RED_	0000 00dd dddd dddd	0 (0x0000)
R0x3074	TEST_DATA_GREENR_	0000 00dd dddd dddd	0 (0x0000)
R0x3076	TEST_DATA_BLUE_	0000 00dd dddd dddd	0 (0x0000)
R0x3078	TEST_DATA_GREENB_	0000 00dd dddd dddd	0 (0x0000)
R0x30A0	X_EVEN_INC_	0000 0000 0000 000?	1 (0x0001)
R0x30A2	X_ODD_INC_	0000 0000 0000 0ddd	1 (0x0001)
R0x30A4	Y_EVEN_INC_	0000 0000 0000 000?	1 (0x0001)
R0x30A6	Y_ODD_INC_	0000 0000 0000 0ddd	1 (0x0001)
R0x30B6	DARK_GREEN1_AVERAGE	0000 0000 ???? ????	0 (0x0000)
R0x30B8	DARK_BLUE_AVERAGE	0000 0000 ???? ????	0 (0x0000)
R0x30BA	DARK_RED_AVERAGE	0000 0000 ???? ????	0 (0x0000)
R0x30BC	DARK_GREEN2_AVERAGE	0000 0000 ???? ????	0 (0x0000)
R0x30C2	CALIB_GREEN1	0000 000d dddd dddd	0 (0x0000)
R0x30C4	CALIB_BLUE	0000 000d dddd dddd	0 (0x0000)
R0x30C6	CALIB_RED	0000 000d dddd dddd	0 (0x0000)
R0x30C8	CALIB_GREEN2	0000 000d dddd dddd	0 (0x0000)
R0x3160	GLOBAL_SEQ_TRIGGER	0000 00?? 0000 0ddd	0 (0x0000)
R0x3162	GLOBAL_RST_END	dddd dddd dddd dddd	0 (0x0000)

Table 9: Register List and Default Values - Manufacturer-Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x3164	GLOBAL_SHUTTER_START	dddd dddd dddd dddd	0 (0x0000)
R0x3166	GLOBAL_READ_START	dddd dddd dddd dddd	0 (0x0000)
R0x31E8	HORIZONTAL_CURSOR_POSITION_	0000 0ddd dddd dddd	0 (0x0000)
R0x31EA	VERTICAL_CURSOR_POSITION_	0000 dddd dddd dddd	0 (0x0000)
R0x31EC	HORIZONTAL_CURSOR_WIDTH_	0000 0ddd dddd dddd	0 (0x0000)
R0x31EE	VERTICAL_CURSOR_WIDTH_	0000 dddd dddd dddd	0 (0x0000)
R0x31FC	I2C_IDS	dddd dddd dddd dddd	12320 (0x3020)
R0x3600	P_GR_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3602	P_GR_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3604	P_GR_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3606	P_GR_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3608	P_GR_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R0x360A	P_RD_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R0x360C	P_RD_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R0x360E	P_RD_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3610	P_RD_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3612	P_RD_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3614	P_BL_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3616	P_BL_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3618	P_BL_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R0x361A	P_BL_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R0x361C	P_BL_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R0x361E	P_GB_P0Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3620	P_GB_P0Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3622	P_GB_P0Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3624	P_GB_P0Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3626	P_GB_P0Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3640	P_GR_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3642	P_GR_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3644	P_GR_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3646	P_GR_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3648	P_GR_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R0x364A	P_RD_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R0x364C	P_RD_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R0x364E	P_RD_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3650	P_RD_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3652	P_RD_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3654	P_BL_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3656	P_BL_P1Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3658	P_BL_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R0x365A	P_BL_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R0x365C	P_BL_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R0x365E	P_GB_P1Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3660	P_GB_P1Q1	dddd dddd dddd dddd	0 (0x0000)

Table 9: Register List and Default Values - Manufacturer-Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x3662	P_GB_P1Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3664	P_GB_P1Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3666	P_GB_P1Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3680	P_GR_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3682	P_GR_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3684	P_GR_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3686	P_GR_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3688	P_GR_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R0x368A	P_RD_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R0x368C	P_RD_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R0x368E	P_RD_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3690	P_RD_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3692	P_RD_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3694	P_BL_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3696	P_BL_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3698	P_BL_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R0x369A	P_BL_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R0x369C	P_BL_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R0x369E	P_GB_P2Q0	dddd dddd dddd dddd	0 (0x0000)
R0x36A0	P_GB_P2Q1	dddd dddd dddd dddd	0 (0x0000)
R0x36A2	P_GB_P2Q2	dddd dddd dddd dddd	0 (0x0000)
R0x36A4	P_GB_P2Q3	dddd dddd dddd dddd	0 (0x0000)
R0x36A6	P_GB_P2Q4	dddd dddd dddd dddd	0 (0x0000)
R0x36C0	P_GR_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R0x36C2	P_GR_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R0x36C4	P_GR_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R0x36C6	P_GR_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R0x36C8	P_GR_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R0x36CA	P_RD_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R0x36CC	P_RD_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R0x36CE	P_RD_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R0x36D0	P_RD_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R0x36D2	P_RD_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R0x36D4	P_BL_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R0x36D6	P_BL_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R0x36D8	P_BL_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R0x36DA	P_BL_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R0x36DC	P_BL_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R0x36DE	P_GB_P3Q0	dddd dddd dddd dddd	0 (0x0000)
R0x36E0	P_GB_P3Q1	dddd dddd dddd dddd	0 (0x0000)
R0x36E2	P_GB_P3Q2	dddd dddd dddd dddd	0 (0x0000)
R0x36E4	P_GB_P3Q3	dddd dddd dddd dddd	0 (0x0000)
R0x36E6	P_GB_P3Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3700	P_GR_P4Q0	dddd dddd dddd dddd	0 (0x0000)

Table 9: Register List and Default Values - Manufacturer-Specific (continued)

1 = read-only, always 1; 0 = read-only, always 0; d = programmable; ? = read-only, dynamic; u = undefined after reset

Register # Hex	Name	Data Format (Binary)	Default Value Dec (Hex)
R0x3702	P_GR_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3704	P_GR_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3706	P_GR_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3708	P_GR_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R0x370A	P_RD_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R0x370C	P_RD_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R0x370E	P_RD_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3710	P_RD_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3712	P_RD_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R0x3714	P_BL_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3716	P_BL_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3718	P_BL_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R0x371A	P_BL_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R0x371C	P_BL_P4Q4	dddd dddd dddd dddd	0 (0x0000)
R0x371E	P_GB_P4Q0	dddd dddd dddd dddd	0 (0x0000)
R0x3720	P_GB_P4Q1	dddd dddd dddd dddd	0 (0x0000)
R0x3722	P_GB_P4Q2	dddd dddd dddd dddd	0 (0x0000)
R0x3724	P_GB_P4Q3	dddd dddd dddd dddd	0 (0x0000)
R0x3726	P_GB_P4Q4	dddd dddd dddd dddd	0 (0x0000)

Register Descriptions

Table 10: Register Description - SMIA Configuration

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x0000	15:0	0x1603	MODEL_ID (R/W)	N	N
	This register is an alias of R0x3000-1. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x0002	7:0	0x0000	REVISION_NUMBER (RO)	N	N
	Module/silicon revision number.				
R0x0003	7:0	0x0006	MANUFACTURER_ID (RO)	N	N
	Manufacturer ID assigned to Micron Imaging. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x0004	7:0	0x000A	SMIA_VERSION (RO)	N	N
	This register is an alias of R0x303A. Read-only.				
R0x0005	7:0	0x00FF	FRAME_COUNT (RO)	Y	N
	This register is an alias of R0x303B. Read-only.				
R0x0006	7:0	0x0000	PIXEL_ORDER (RO)	N	N
	This register is an alias of R0x3024. Read-only.				
R0x0008	15:0	0x002A	DATA_PEDESTAL (R/W)	N	Y
	This register is an alias of R0x301E-F. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x0040	7:0	0x0001	FRAME_FORMAT_MODEL_TYPE (RO)	N	N
	Type 1. 2-byte Generic Frame Format Description. Read-only.				
R0x0041	7:0	0x0012	FRAME_FORMAT_MODEL_SUBTYPE (RO)	N	N
	Number of descriptors: 1 X (column) descriptor and two Y (row) descriptors. Read-only.				
R0x0042	15:0	0x5800	FRAME_FORMAT_DESCRIPTOR_0 (RO)	Y	N
	X descriptor: Bits[11:0] of this register reflect the current value of x_output_size[11:0]. Upper 4 bits is the pixel code; 5=Visible Pixel Data. Read-only, dynamic.				
R0x0044	15:0	0x1002	FRAME_FORMAT_DESCRIPTOR_1 (RO)	Y	N
	Y descriptor: In normal operation, returns 0x1002 to indicate that 2 rows of embedded data are present in the output image. If embedded data is disabled (by selecting the PN9 test pattern using R0x3070-1) this register will return 0x1000. Read-only.				
R0x0046	15:0	0x5600	FRAME_FORMAT_DESCRIPTOR_2 (RO)	Y	N
	Y descriptor: Bits[11:0] of this register reflect the current value of y_output_size[11:0]. Upper 4 bits is the pixel code; 5=Visible Pixel Data. Read-only, dynamic.				
R0x0048	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_3 (RO)	N	N
	Read-only.				
R0x004A	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_4 (RO)	N	N
	Read-only.				
R0x004C	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_5 (RO)	N	N
	Read-only.				
R0x004E	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_6 (RO)	N	N
	Read-only.				
R0x0050	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_7 (RO)	N	N
	Read-only.				
R0x0052	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_8 (RO)	N	N
	Read-only.				
R0x0054	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_9 (RO)	N	N
	Read-only.				

Table 10: Register Description - SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x0056	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_10 (RO)	N	N
	Read-only.				
R0x0058	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_11 (RO)	N	N
	Read-only.				
R0x005A	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_12 (RO)	N	N
	Read-only.				
R0x005C	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_13 (RO)	N	N
	Read-only.				
R0x005E	15:0	0x0000	FRAME_FORMAT_DESCRIPTOR_14 (RO)	N	N
	Read-only.				
R0x0080	15:0	0x0001	ANALOGUE_GAIN_CAPABILITY (RO)	N	N
	Indicates the provision of separate (per-color) analog gain control. The sensor supports both global and separate (per-color) analog gain control. Read-only.				
R0x0084	15:0	0x0008	ANALOGUE_GAIN_CODE_MIN (RO)	N	N
	Minimum gain code. Read-only.				
R0x0086	15:0	0x007F	ANALOGUE_GAIN_CODE_MAX (RO)	N	N
	Maximum gain code. Read-only.				
R0x0088	15:0	0x0001	ANALOGUE_GAIN_CODE_STEP (RO)	N	N
	Gain code step size. Read-only.				
R0x008A	15:0	0x0000	ANALOGUE_GAIN_TYPE (RO)	N	N
	Indicates support for analog gain coding type 0 (baseline SMIA). Read-only.				
R0x008C	15:0	0x0001	ANALOGUE_GAIN_M0 (RO)	N	N
	Constants for the gain equation. Read-only.				
R0x008E	15:0	0x0000	ANALOGUE_GAIN_C0 (RO)	N	N
	Constants for the gain equation. Read-only.				
R0x0090	15:0	0x0000	ANALOGUE_GAIN_M1 (RO)	N	N
	Constants for the gain equation. Read-only.				
R0x0092	15:0	0x0008	ANALOGUE_GAIN_C1 (RO)	N	N
	Constants for the gain equation. Read-only.				
R0x00C0	7:0	0x0001	DATA_FORMAT_MODEL_TYPE (RO)	N	N
	Indicates the use of 2-byte data format. Read-only.				
R0x00C1	7:0	0x0003	DATA_FORMAT_MODEL_SUBTYPE (RO)	N	N
	Indicates the number of data format descriptors. Read-only.				
R0x00C2	15:0	0x0A0A	DATA_FORMAT_DESCRIPTOR_0 (RO)	N	N
	Indicates support for RAW10, uncompressed data format. Read-only.				
R0x00C4	15:0	0x0808	DATA_FORMAT_DESCRIPTOR_1 (RO)	N	N
	Indicates support for RAW8 data format in which the two LSB of each 10-bit pixel data value are discarded. Read-only.				
R0x00C6	15:0	0x0A08	DATA_FORMAT_DESCRIPTOR_2 (RO)	N	N
	Indicates support for RAW8 data format in which each 10-bit pixel data value is compressed to an 8-bit value. Read-only.				
R0x00C8	15:0	0x0000	DATA_FORMAT_DESCRIPTOR_3 (RO)	N	N
	Read-only.				
R0x00CA	15:0	0x0000	DATA_FORMAT_DESCRIPTOR_4 (RO)	N	N
	Read-only.				

Table 10: Register Description - SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x00CC	15:0	0x0000	DATA_FORMAT_DESCRIPTOR_5 (RO)	N	N
	Read-only.				
R0x00CE	15:0	0x0000	DATA_FORMAT_DESCRIPTOR_6 (RO)	N	N
	Read-only.				
R0x0100	7:0	0x0000	MODE_SELECT (R/W)	Y	N
	This register field is an alias of R0x301A-B[2].				
R0x0101	7:0	0x0000	IMAGE_ORIENTATION (R/W)		
	7:2	X	Reserved		
	1	0x0000	IMAGE_ORIENTATION_VERTICAL_FLIP This register field is an alias of R0x3040-1[1].	Y	YM
	0	0x0000	IMAGE_ORIENTATION_HORIZONTAL_MIRROR This register field is an alias of R0x3040-1[0].	Y	YM
R0x0103	7:0	0x0000	SOFTWARE_RESET (R/W)	N	Y
	This register field is an alias of R0x301A-B[0].				
R0x0104	7:0	0x0000	GROUPED_PARAMETER_HOLD (R/W)	N	N
	This register field is an alias of R0x301A-B[15].				
R0x0105	7:0	0x0000	MASK_CORRUPTED_FRAMES (R/W)	N	Y
	This register field is an alias of R0x301A-B[9].				
R0x0110	7:0	0x0000	CCP2_CHANNEL_IDENTIFIER (R/W)	Y	N
	Set the DMA channel identifier within the CCP2 embedded sychronization codes.				
R0x0111	7:0	0x0001	CCP2_SIGNALLING_MODE (R/W)	Y	N
	0 = Use Data/Clock signalling on the CCP2 serial interface. 1 = Use Data/Strobe signalling on the CCP2 serial interface.				
R0x0112	15:0	0x0A0A	CCP_DATA_FORMAT (R/W)	Y	N
	[7:0] = The bit-width of the compressed pixel data [15:8] = The bit-width of the uncompressed pixel data The value in this register must match one of the valid data_format_descriptor registers (R0x00C2-R0x00C7).				
R0x0120	7:0	0x0000	GAIN_MODE (R/W)	N	N
	This read/write bit has no function.				
R0x0200	15:0	0x01BD	FINE_INTEGRATION_TIME (R/W)	Y	N
	Integration time programmed in units of pck. This register is an alias of R0x3014-5.				
R0x0202	15:0	0x0010	COARSE_INTEGRATION_TIME (R/W)	Y	N
	Integration time programmed in units of line_length_pck. This register is an alias of R0x3012-3.				
R0x0204	15:0	0x0008	ANALOGUE_GAIN_CODE_GLOBAL (R/W)	Y	N
	This register is an alias of R0x3028-9.				
R0x0206	15:0	0x0008	ANALOGUE_GAIN_CODE_GREENR (R/W)	Y	N
	This register is an alias of R0x302A-B.				
R0x0208	15:0	0x0008	ANALOGUE_GAIN_CODE_RED (R/W)	Y	N
	This register is an alias of R0x302C-D.				
R0x020A	15:0	0x0008	ANALOGUE_GAIN_CODE_BLUE (R/W)	Y	N
	This register is an alias of R0x302E-F.				
R0x020C	15:0	0x0008	ANALOGUE_GAIN_CODE_GREENB (R/W)	Y	N
	This register is an alias of R0x3030-1.				
R0x020E	15:0	0x0100	DIGITAL_GAIN_GREENR (R/W)	Y	N
	This register is an alias of R0x3032-3.				

Table 10: Register Description - SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x0210	15:0	0x0100	DIGITAL_GAIN_RED (R/W)	Y	N
	This register is an alias of R0x3034-5.				
R0x0212	15:0	0x0100	DIGITAL_GAIN_BLUE (R/W)	Y	N
	This register is an alias of R0x3036-7.				
R0x0214	15:0	0x0100	DIGITAL_GAIN_GREENB (R/W)	Y	N
	This register is an alias of R0x3038-9.				
R0x0300	15:0	0x000A	VT_PIX_CLK_DIV (R/W)	N	Y
	Clock divisor applied to video timing system clock to generate video timing pixel clock.				
R0x0302	15:0	0x0001	VT_SYS_CLK_DIV (R/W)	N	N
	Clock divisor applied to PLL output clock to generate video timing system clock.				
R0x0304	15:0	0x0002	PRE_PLL_CLK_DIV (R/W)	N	Y
	Clock divisor applied to EXTCLK to generate PLL input clock.				
R0x0306	15:0	0x0050	PLL_MULTIPLIER (R/W)	N	Y
	Clock multiplier applied to PLL input clock.				
R0x0308	15:0	0x000A	OP_PIX_CLK_DIV (R/W)	N	Y
	Clock divisor applied to the output system clock to generate the output pixel clock. This value must correspond to the number of bits per output pixel, as given by the LSB of CCP_data_format.				
R0x030A	15:0	0x0001	OP_SYS_CLK_DIV (R/W)	N	Y
	Clock divisor applied to PLL output clock to generate output system clock. Read-only.				
R0x0340	15:0	0x0655	FRAME_LENGTH_LINES (R/W)	Y	YM
	This register is an alias of R0x300A-B.				
R0x0342	15:0	0x0A3E	LINE_LENGTH_PCK (R/W)	Y	YM
	This register is an alias of R0x300C-D.				
R0x0344	15:0	0x0008	X_ADDR_START (R/W)	Y	N
	This register is an alias of R0x3004-5.				
R0x0346	15:0	0x0008	Y_ADDR_START (R/W)	Y	YM
	This register is an alias of R0x3002-5.				
R0x0348	15:0	0x0807	X_ADDR_END (R/W)	Y	N
	This register is an alias of R0x3008-9.				
R0x034A	15:0	0x0607	Y_ADDR_END (R/W)	Y	YM
	This register is an alias of R0x3006-7.				
R0x034C	15:0	0x0800	X_OUTPUT_SIZE (R/W)	Y	N
	Set X output size of displayed image. Bit[0] is read-only 0. The default value of this register is set to be consistent with the default values of x_addr_end and x_addr_start.				
R0x034E	15:0	0x0600	Y_OUTPUT_SIZE (R/W)	Y	N
	Set Y output size of the displayed image. Bit[0] is read-only 0. The default value of this registers set to be consistent with the default values of y_addr_end and y_addr_start. The output image will have two additional rows containing embedded data, in accordance with the frame format descriptors.				
R0x0380	15:0	0x0001	X_EVEN_INC (RO)	N	N
	Read-only. The fixed value of 1 constrains subsampling operation to use adjacent pixels of a pixel quad.				
R0x0382	15:0	0x0001	X_ODD_INC (R/W)	Y	YM
	This register field is an alias of R0x3040-1[7:5].				
R0x0384	15:0	0x0001	Y_EVEN_INC (RO)	N	N
	Read-only. The fixed value of 1 constrains subsampling operation to use adjacent pixels of a pixel quad.				

Table 10: Register Description - SMIA Configuration (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x0386	15:0	0x0001	Y_ODD_INC (R/W)	Y	YM
	This register field is an alias of R0x3040-1[4:2].				
R0x0400	15:0	0x0000	SCALING_MODE (R/W)	Y	N
	0 = Disable scaler 1 = Enable horizontal scaling 2 = Enable horizontal and vertical scaling 3 = Reserved				
R0x0402	15:0	0x0000	SPATIAL_SAMPLING (R/W)	Y	N
	0 = Bayer sampling 1 = Co-sited sampling				
R0x0404	15:0	0x0010	SCALE_M (R/W)	Y	N
	Scale factor M.				
R0x0406	15:0	0x0010	SCALE_N (RO)	N	N
	Scale factor N. Read-only.				
R0x0500	15:0	0x0001	COMPRESSION_MODE (RO)	N	Y
	0x0001 = 10-bit to 8-bit compression uses the DPCM/PCM Simple Predictor algorithm. Read-only. This register controls the algorithm that is to be used for compression. The sensor only supports a single algorithm and therefore this register is read-only. This register does not control whether data compression is enabled; that is controlled by the ccp_data_format register (R0x0012-3).				
R0x0600	15:0	0x0000	TEST_PATTERN_MODE (R/W)	N	Y
	This register is an alias of R0x3070-1.				
R0x0602	15:0	0x0000	TEST_DATA_RED (R/W)	N	Y
	This register is an alias of R0x3072-3.				
R0x0604	15:0	0x0000	TEST_DATA_GREENR (R/W)	N	Y
	This register is an alias of R0x3074-5.				
R0x0606	15:0	0x0000	TEST_DATA_BLUE (R/W)	N	Y
	This register is an alias of R0x3076-7.				
R0x0608	15:0	0x0000	TEST_DATA_GREENB (R/W)	N	Y
	This register is an alias of R0x3078-8.				
R0x060A	15:0	0x0000	HORIZONTAL_CURSOR_WIDTH (R/W)	N	N
	This register is an alias of Reg0x31EC-D.				
R0x060C	15:0	0x0000	HORIZONTAL_CURSOR_POSITION (R/W)	N	N
	This register is an alias of R0x31E8-9.				
R0x060E	15:0	0x0000	VERTICAL_CURSOR_WIDTH (R/W)	N	N
	This register is an alias of R0x31EE-F.				
R0x0610	15:0	0x0000	VERTICAL_CURSOR_POSITION (R/W)	N	N
	This register is an alias of R0x31EA-B.				

Table 11: Register Description - SMIA Parameter Limits

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x1000	15:0	0x0001	INTEGRATION_TIME_CAPABILITY (RO)	N	N
	Indicates the provision of coarse and fine integration time control. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1004	15:0	0x0000	COARSE_INTEGRATION_TIME_MIN (R/W)	N	N
	The minimum coarse integration time. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1006	15:0	0x0001	COARSE_INTEGRATION_TIME_MAX_MARGIN (R/W)	N	N
	The maximum value of coarse_integration_time is (frame_length_lines - coarse_integration_time_max_margin). Read-only. Can be made read/write by clearing R0x301A-B[3]. This limit can be broken. The result will be a graceful degradation of frame rate. When this happens, (coarse_integration_time - coarse_integration_time_max_margin) replaces (frame_length_lines) in the frame rate equation.				
R0x1008	15:0	0x01BD	FINE_INTEGRATION_TIME_MIN (R/W)	N	N
	The minimum fine integration time. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x100A	15:0	0x00CF	FINE_INTEGRATION_TIME_MAX_MARGIN (R/W)	N	N
	The minimum fine integration time is (line_length_pck - fine_integration_time_max_margin). Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1080	15:0	0x0001	DIGITAL_GAIN_CAPABILITY (RO)	N	N
	Indicates the provision of separate (per-color) digital gain control. Read-only.				
R0x1084	15:0	0x0100	DIGITAL_GAIN_MIN (RO)	N	N
	UFIX16. Minimum value of digital gain is 1.0. Read-only.				
R0x1086	15:0	0x0700	DIGITAL_GAIN_MAX (RO)	N	N
	UFIX16. Maximum value of digital gain is 7.0. Read-only.				
R0x1088	15:0	0x0100	DIGITAL_GAIN_STEP_SIZE (RO)	N	N
	UFIX16. Step size for digital gain is 1.0. Read-only.				
R0x1100	15:0	0x40C0	MIN_EXT_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum external clock frequency into PLL is 6.0 MHz. Read-only.				
R0x1102	15:0	0x0000	MIN_EXT_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum external clock frequency into PLL is 6.0 MHz. Read-only.				
R0x1104	15:0	0x41D8	MAX_EXT_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum external clock frequency into PLL is 27.0 MHz. Read-only.				
R0x1106	15:0	0x0000	MAX_EXT_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum external clock frequency into PLL is 27.0 MHz. Read-only.				
R0x1108	15:0	0x0001	MIN_PRE_PLL_CLK_DIV (RO)	N	N
	Minimum clock divisor applied to PLL input clock. Read-only.				
R0x110A	15:0	0x0040	MAX_PRE_PLL_CLK_DIV (RO)	N	N
	Maximum clock divisor applied to PLL input clock. Read-only. The value in this register reflects the programmable range of the register. The maximum useful value (given max_ext_clk_freq and min_pll_ip_freq) is 13.				

Table 11: Register Description - SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x110C	15:0	0x4000	MIN_PLL_IP_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum clock frequency into the PFD of the PLL is 2.0 MHz. Read-only.				
R0x110E	15:0	0x0000	MIN_PLL_IP_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum clock frequency into the PFD of the PLL is 2.0 MHz. Read-only.				
R0x1110	15:0	0x4138	MAX_PLL_IP_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum clock frequency into the PFD of the PLL is 11.5 MHz. Read-only.				
R0x1112	15:0	0x0000	MAX_PLL_IP_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum clock frequency into the PFD of the PLL is 11.5 MHz. Read-only.				
R0x1114	15:0	0x0001	MIN_PLL_MULTIPLIER (RO)	N	N
	The value in this register describes the minimum value you can write to pll_multiplier. The smallest useful value of pll_multiplier (given min_pll_op_freq and max_pll_ip_freq) is 13.				
R0x1116	15:0	0x00FE	MAX_PLL_MULTIPLIER (RO)	N	N
	The value in this register describes the maximum value you can write to pll_multiplier.				
R0x1118	15:0	0x4320	MIN_PLL_OP_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum output frequency supported by the PLL is 160.0 MHz. Read-only.				
R0x111A	15:0	0x0000	MIN_PLL_OP_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum output frequency supported by the PLL is 160.0 MHz. Read-only.				
R0x111C	15:0	0x4434	MAX_PLL_OP_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum output frequency supported by the PLL is 720.0 MHz. Read-only.				
R0x111E	15:0	0x0000	MAX_PLL_OP_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum output frequency supported by the PLL is 720.0 MHz. Read-only.				
R0x1120	15:0	0x0001	MIN_VT_SYS_CLK_DIV (RO)	N	N
	The minimum value allowed for vt_sys_clk_div if in Profile 0 mode (R0x306e-f[7]=0). In Profile 1/2 mode (R0x306e-f[7]=1), the vt_sys_clk_div value must be 2.				
R0x1122	15:0	0x0020	MAX_VT_SYS_CLK_DIV (RO)	N	N
	The maximum value allowed for vt_sys_clk_div if in Profile 0 mode (R0x306e-f[7]=0). In Profile 1/2 mode (R0x306e-f[7]=1), the vt_sys_clk_div value must be 2.				
R0x1124	15:0	0x4180	MIN_VT_SYS_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. This register is static read-only and shows the behavior of the default Profile. Profile 0 mode: Minimum frequency for vt_sys_clk is 16.0 MHz. Profile 1/2 mode: Minimum frequency for vt_sys_clk is 32.0 MHz.				
R0x1126	15:0	0x0000	MIN_VT_SYS_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. See min_vt_sys_clk_freq_mhz_1.				

Table 11: Register Description - SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x1128	15:0	0x4420	MAX_VT_SYS_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. The value in this register is static read-only, and shows the maximum frequency of vt_sys_clk for the default Profile mode. Profile 0: Maximum frequency for the vt_sys_clk is 640.0 MHz. Profile 1/2: Maximum frequency for the vt_sys_clk is 320.0 MHz.				
R0x112A	15:0	0x0000	MAX_VT_SYS_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. See max_vt_sys_clk_freq_mhz_1.				
R0x112C	15:0	0x4100	MIN_VT_PIX_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum frequency for video timing pix_clk is 8.0 MHz. Read-only.				
R0x112E	15:0	0x0000	MIN_VT_PIX_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum frequency for video timing pix_clk is 8.0 MHz. Read-only.				
R0x1130	15:0	0x4280	MAX_VT_PIX_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum frequency for video timing pix_clk is 64.0 MHz. Read-only.				
R0x1132	15:0	0x0000	MAX_VT_PIX_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum frequency for video timing pix_clk is 64.0 MHz. Read-only.				
R0x1134	15:0	0x0004	MIN_VT_PIX_CLK_DIV (RO)	N	N
	Minimum divisor for the video timing pix_clk. Read-only.				
R0x1136	15:0	0x000A	MAX_VT_PIX_CLK_DIV (RO)	N	N
	Maximum divisor for the video timing pix_clk. Read-only.				
R0x1140	15:0	0x0057	MIN_FRAME_LENGTH_LINES (R/W)	N	N
	Minimum frame length. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1142	15:0	0xFFFF	MAX_FRAME_LENGTH_LINES (R/W)	N	N
	Maximum frame length. The maximum frame length is only constrained by the size of the read/write field in the frame_length_lines register (16-bits). Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1144	15:0	0x033A	MIN_LINE_LENGTH_PCK (R/W)	N	N
	Minimum line length. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1146	15:0	0x3FFF	MAX_LINE_LENGTH_PCK (R/W)	N	N
	Maximum line length. The maximum line length is only constrained by the size of the read/write field in the line_length_pck register. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1148	15:0	0x023A	MIN_LINE_BLANKING_PCK (R/W)	N	N
	Minimum line blanking time. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x114A	15:0	0x0055	MIN_FRAME_BLANKING_LINES (R/W)	N	N
	Minimum frame blanking time. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1160	15:0	0x0001	MIN_OP_SYS_CLK_DIV (RO)	N	N
	Minimum divisor for the output sys_clk. Read-only.				
R0x1162	15:0	0x0020	MAX_OP_SYS_CLK_DIV (RO)	N	N
	Maximum divisor for the output sys_clk. Read-only.				
R0x1164	15:0	0x4280	MIN_OP_SYS_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum frequency for output sys_clk is 64.0 MHz. Read-only.				
R0x1166	15:0	0x0000	MIN_OP_SYS_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum frequency for output sys_clk is 64.0 MHz. Read-only.				

Table 11: Register Description - SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x1168	15:0	0x4420	MAX_OP_SYS_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum frequency for output sys_clk is 640.0 MHz. Read-only.				
R0x116A	15:0	0x0000	MAX_OP_SYS_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum frequency for output sys_clk is 640.0 MHz. Read-only.				
R0x116C	15:0	0x0008	MIN_OP_PIX_CLK_DIV (RO)	N	N
	Minimum divisor for output pix_clk. Read-only. Programmable op_pix_clk_div range = 8 (8-bit operation) or 10 (10-bit operation).				
R0x116E	15:0	0x000A	MAX_OP_PIX_CLK_DIV (RO)	N	N
	Maximum divisor for output pix_clk. Read-only. Programmable op_pix_clk_div range = 8 (8-bit operation) or 10 (10-bit operation).				
R0x1170	15:0	0x4100	MIN_OP_PIX_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Minimum frequency for output pix_clk is 8.0 MHz. Read-only.				
R0x1172	15:0	0x0000	MIN_OP_PIX_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Minimum frequency for output pix_clk is 8.0 MHz. Read-only.				
R0x1174	15:0	0x4280	MAX_OP_PIX_CLK_FREQ_MHZ_1 (RO)	N	N
	FLP32. Maximum frequency for output pix_clk is 64.0 MHz. Read-only.				
R0x1176	15:0	0x0000	MAX_OP_PIX_CLK_FREQ_MHZ_2 (RO)	N	N
	FLP32. Maximum frequency for output pix_clk is 64.0 MHz. Read-only.				
R0x1180	15:0	0x0000	X_ADDR_MIN (RO)	N	N
	Minimum value for x_addr_start, x_addr_end. Read-only.				
R0x1182	15:0	0x0000	Y_ADDR_MIN (RO)	N	N
	Minimum value for y_addr_start, y_addr_end. Read-only.				
R0x1184	15:0	0x080F	X_ADDR_MAX (RO)	N	N
	Maximum value for x_addr_start, x_addr_end. Read-only.				
R0x1186	15:0	0x060F	Y_ADDR_MAX (RO)	N	N
	Maximum value for y_addr_start, y_addr_end. Read-only.				
R0x11C0	15:0	0x0001	MIN_EVEN_INC (RO)	N	N
	Minimum value for increment of even X/Y addresses when subsampling is enabled. Read-only.				
R0x11C2	15:0	0x0001	MAX_EVEN_INC (RO)	N	N
	Maximum value for increment of even X/Y addresses when subsampling is enabled. Read-only.				
R0x11C4	15:0	0x0001	MIN_ODD_INC (RO)	N	N
	Minimum value for increment of odd X/Y addresses when subsampling is enabled. Read-only.				
R0x11C6	15:0	0x0003	MAX_ODD_INC (RO)	N	N
	Maximum value for increment of odd X/Y addresses when subsampling is enabled. Read-only. This set of 4 registers declares the capability for the subsampling mode that was called "skip2" on earlier Micron Imaging sensors.				
R0x1200	15:0	0x0002	SCALING_CAPABILITY (RO)	N	N
	Indicates the provision of a full (horizontal and vertical) scaler. Read-only.				
R0x1204	15:0	0x0010	SCALER_M_MIN (RO)	N	N
	Indicates the minimum M value for the scaler. Read-only.				
R0x1206	15:0	0x0080	SCALER_M_MAX (RO)	N	N
	Indicates the maximum M value for the scaler. Read-only.				
R0x1208	15:0	0x0010	SCALER_N_MIN (RO)	N	N
	Indicates the minimum N value for the scaler. Read-only.				

Table 11: Register Description - SMIA Parameter Limits (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x120A	15:0	0x0010	SCALER_N_MAX (RO)	N	N
	Indicates the maximum N value for the scaler. Read-only.				
R0x1300	15:0	0x0001	COMPRESSION_CAPABILITY (RO)	N	N
	Indicates the capability for performing 10-bit to 8-bit pixel data compression. Read-only.				
R0x1400	15:0	0x0100	MATRIX_ELEMENT_REDINRED (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1402	15:0	0x0000	MATRIX_ELEMENT_GREENINRED (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1404	15:0	0x0000	MATRIX_ELEMENT_BLUEINRED (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1406	15:0	0x0000	MATRIX_ELEMENT_REDINGREEN (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1408	15:0	0x0100	MATRIX_ELEMENT_GREENINGREEN (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x140A	15:0	0x0000	MATRIX_ELEMENT_BLUEINGREEN (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x140C	15:0	0x0000	MATRIX_ELEMENT_REDINBLUE (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x140E	15:0	0x0000	MATRIX_ELEMENT_GREENINBLUE (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x1410	15:0	0x0100	MATRIX_ELEMENT_BLUEINBLUE (R/W)	N	N
	Color-correction matrix. Read-only. Can be made read/write by clearing R0x301A-B[3].				

Table 12: Register Description - Manufacturer-Specific

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3000	15:0	0x1603	MODEL_ID_ (R/W)	N	N
	Model ID. Read-only. Can be made read/write by clearing R0x301A-B[3].				
R0x3002	15:0	0x0008	Y_ADDR_START_ (R/W)	Y	YM
	The first row of visible pixels to be read out (not counting any dark rows that may be read). To move the image window, set this register to the starting Y value.				
R0x3004	15:0	0x0008	X_ADDR_START_ (R/W)	Y	N
	The first column of visible pixels to be read out (not counting any dark columns that may be read). To move the image window, set this register to the starting X value.				
R0x3006	15:0	0x0607	Y_ADDR_END_ (R/W)	Y	YM
	The last row of visible pixels to be read out.				
R0x3008	15:0	0x0807	X_ADDR_END_ (R/W)	Y	N
	The last column of visible pixels to be read out.				
R0x300A	15:0	0x0655	FRAME_LENGTH_LINES_ (R/W)	Y	YM
	The number of complete lines (rows) in the output frame. This includes visible lines and vertical blanking lines.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x300C	15:0	0x0A3E	LINE_LENGTH_PCK_ (R/W)	Y	YM
	The number of pixel clock periods in one line (row) time. This includes visible pixels and horizontal blanking time.				
R0x3010	15:0	0x00B0	FINE_CORRECTION (R/W)	N	Y
	Fine integration time correction factor. This is an offset that is applied to the programmed value of fine_integration_time such that the actual integration time matches the integration time equation. This register should not be modified under normal operation, but must be modified when the row_speed register is changed from its default value or binning is enabled.				
R0x3012	15:0	0x0010	COARSE_INTEGRATION_TIME_ (R/W)	Y	N
	Integration time specified in multiples of line_length_pck_.				
R0x3014	15:0	0x01BD	FINE_INTEGRATION_TIME_ (R/W)	Y	N
	Integration time specified as a number of pixel clocks.				
R0x3016	15:0	0x0111	ROW_SPEED (R/W)		
	15:11	X	Reserved		
	10:8	0x0001	ROW_SPEED_OPCLK_SPEED Slows down the output pixel clock frequency relative to the system clock frequency. A programmed value of N gives an output pixel clock period of N system clocks. Only values 1, 2 and 4 are supported. A value of 0 is illegal: it causes the clock to stop. The resulting output pixel clock frequency (as controlled by op_sys_clk_div, op_pix_clk_div and this register) must not be faster than the video pixel clock frequency (as controlled by vt_sys_clk_div, vt_pix_clk_div and row_speed[2:0]).	N	N
	7	X	Reserved		
	6:4	0x0001	ROW_SPEED_OPCLK_DELAY Reserved.	N	N
	3	X	Reserved		
	2:0	0x0001	ROW_SPEED_PIXCLK_SPEED Slows down the internal pixel clock frequency relative to the system clock frequency. A programmed value of N gives a pixel clock period of N system clocks. Only values 1, 2 and 4 are supported. A value of 0 is illegal: it causes the clock to stop.	Y	YM
R0x3018	15:0	0x0000	EXTRA_DELAY (R/W)	Y	N
	Extra blanking inserted between frames. A programmed value of N increases the vertical blanking time by N pixel clock periods. Can be used to get a more exact frame rate. Changing this register from the default will affect the integration times of parts of the image when the integration time is less than 1 frame and more than 0 rows. This is a consequence of the Electronic Rolling Shutter (ERS). The difference in integration time between the affected part and the non-affected part equals N pixel clock periods.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x301A	15:0	0x0018	RESET_REGISTER (R/W)		
	15	0x0000	RESET_REGISTER_GROUPED_PARAMETER_HOLD 0 = Update of many of the registers is synchronized to frame start. 1 = Inhibit register updates; register changes will remain pending until this bit is returned to 0. When this bit is returned to 0, all pending register updates will be made on the next frame start.	N	N
	14:13	X	Reserved		
	12	0x0000	RESET_REGISTER_SMIA_SERIALISER_DIS This bit disables the SMIA high-speed serialiser and differential output buffers.	N	N
	11	X	Reserved		
	10	0x0000	RESET_REGISTER_RESTART_BAD 1 = a restart is forced any time a bad frame is detected. This can shorten the delay when waiting for a good frame, since the delay for masking out a bad frame will be the integration time rather than the full-frame time.	N	N
	9	0x0000	RESET_REGISTER_MASK_BAD 0 = The sensor will produce bad (corrupted) frames as a result of some register changes. 1 = Bad (corrupted) frames are masked within the sensor by extending the vertical blanking time for the duration of the bad frame.	N	N
	8	0x0000	RESET_REGISTER_GPI_EN 0 = the primary input buffers associated with the General Purpose Input (GPI) pins are powered down and they cannot be used. 1 = the input buffers are enabled and can be read through R0x3026-7.	N	N
	7	0x0000	RESET_REGISTER_PARALLEL_EN 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.	N	N
	6	0x0000	RESET_REGISTER_DRIVE_PINS This bit is “Don't Care” unless bit[7]=1. 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 R0x3026-7). 1 = The parallel data interface is driven.	N	N
	5	0x0000	Reserved		
	4	0x0001	RESET_REGISTER_STDBY_EOF 0 = Transition to standby is synchronized to the end of a sensor row readout (held-off until LINE_VALID has fallen). 1 = Transition to standby is synchronized to the end of a frame.	N	Y

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x301A	3	0x0001	RESET_REGISTER_LOCK_REG Many SMIA registers that are specified as read-only are actually implemented as read/write registers. Clearing this bit allows such registers to be written.	N	N
	2	0x0000	RESET_REGISTER_STREAM Setting this bit places the sensor in streaming mode. Clearing this bit places the sensor in a low power mode. The result of clearing this bit depends upon the operating mode of the sensor. Entry and exit from streaming mode can also be controlled from the signal interface.	Y	N
	1	0x0000	RESET_REGISTER_RESTART This bit always reads as 0. Setting this bit causes the sensor to truncate the current frame at the end of the current row and start resetting (integrating) the first row. The delay before the first valid frame is read out is equal to the integration time.	N	Y
	0	0x0000	RESET_REGISTER_RESET This bit always reads as 0. Setting this bit initiates a reset sequence: the frame being generated will be truncated.	N	Y
R0x301C	7:0	0x0000	MODE_SELECT_ (R/W)	Y	N
	This bit is an alias of R0x301A-B[2].				
R0x301D	7:0	0x0000	IMAGE_ORIENTATION_ (R/W)		
	7:2	X	Reserved		
	1	0x0000	IMAGE_ORIENTATION_VERT_FLIP This bit is an alias of R0x3040-1[1].	Y	YM
	0	0x0000	IMAGE_ORIENTATION_HORIZ_MIRROR This bit is an alias of R0x3040-1[0].	Y	YM
R0x301E	15:0	0x002A	DATA_PEDESTAL_ (R/W)	N	Y
	Constant offset that is added to the ADC output for all visible pixels in order to set the black level to a value greater than 0. Read-only. Can be made read/write by clearing R0x301A-B[3]. The data_pedestal can be overridden by the functionality in the test_raw_mode register.				
R0x3021	7:0	0x0000	SOFTWARE_RESET_ (R/W)	N	Y
	This bit is an alias of R0x301A-B[0].				
R0x3022	7:0	0x0000	GROUPED_PARAMETER_HOLD_ (R/W)	N	N
	This bit is an alias of R0x301A-B[15].				
R0x3023	7:0	0x0000	MASK_CORRUPTED_FRAMES_ (R/W)	N	N
	This bit is an alias of R0x301A-B[9].				
R0x3024	7:0	0x0000	PIXEL_ORDER_ (RO)	N	N
	00 = First row is GreenR/Red, first pixel is GreenR 01 = First row is GreenR/Red, first pixel is Red 02 = First row is Blue/GreenB, first pixel is Blue 03 = First row is Blue/GreenB, first pixel is GreenB The value in this register changes as a function of R0x3040-1[1:0].				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3026	15:0	0xFFF3	GPI_STATUS (R/W)		
	15:13	0x0007	GPI_STATUS_STANDBY_PIN_SELECT Associate the standby function with an active HIGH input pin 0 = associate with GPI0 1 = associate with GPI1 2-6 = RESERVED 7 = standby function cannot be controlled by any pin Must be set to 7 if reset_register[8]=0.	N	N
	12:10	0x0007	GPI_STATUS_OE_N_PIN_SELECT Associate the output-enable function with an active LOW input pin 0 = associate with GPI0 1 = associate with GPI12-6 = RESERVED 7 = output-enable function is not controlled by any pin Must be set to 7 if reset_register[8]=0.	N	N
	9:7	0x0007	GPI_STATUS_TRIGGER_PIN_SELECT Associate the trigger function with an active HIGH input pin 0 = associate with GPI0 1 = associate with GPI1 2-6 = RESERVED 7 = trigger function is not controlled by any pin Must be set to 7 if R0x301A-B[8]=0.	N	N
	6:4	0x0007	GPI_STATUS_SADDR_PIN_SELECT Associate the SADDR function with an active HIGH input pin 0 = associate with GPI0 1 = associate with GPI1 2-6 = RESERVED 7 = SADDR function is not controlled by any pin Must be set to 7 if R0x301A-B[8]=0.	N	N
	3	RO	Reserved		
	2	RO	Reserved		
	1	RO	GPI_STATUS_GPI1 Read-only. Return the current state of the GPI1 input pin. Invalid if R0x301A-B[8]=0.	N	N
	0	RO	GPI_STATUS_GPI0 Read-only. Return the current state of the GPI0 input pin. Invalid if R0x301A-B[8]=0.	N	N
R0x3028	15:0	0x0008	ANALOGUE_GAIN_CODE_GLOBAL_ (R/W)	Y	N
	Writing a gain code to this register is equivalent to writing that code to each of the 4 color-specific gain code registers. Reading from this register returns the value most recently written to the analogue_gain_code_greenR register.				
R0x302A	15:0	0x0008	ANALOGUE_GAIN_CODE_GREENR_ (R/W)	Y	N
	The gain code written to this register sets the gain for green pixels on red/green rows of the pixel array.				
R0x302C	15:0	0x0008	ANALOGUE_GAIN_CODE_RED_ (R/W)	Y	N
	The gain code written to this register sets the gain for red pixels.				
R0x302E	15:0	0x0008	ANALOGUE_GAIN_CODE_BLUE_ (R/W)	Y	N
	The gain code written to this register sets the gain for blue pixels.				
R0x3030	15:0	0x0008	ANALOGUE_GAIN_CODE_GREENB_ (R/W)	Y	N
	The gain code written to this register sets the gain for green pixels on blue/green rows of the pixel array.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3032	15:0	0x0100	DIGITAL_GAIN_GREENR_ (R/W)	Y	N
	Digital gain applied to green pixels on red/green rows of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x3056-7[11:9].				
R0x3034	15:0	0x0100	DIGITAL_GAIN_RED_ (R/W)	Y	N
	Digital gain applied to red pixels of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x305A-B[11:9].				
R0x3036	15:0	0x0100	DIGITAL_GAIN_BLUE_ (R/W)	Y	N
	Digital gain applied to blue pixels of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x3058-9[11:9].				
R0x3038	15:0	0x0100	DIGITAL_GAIN_GREENB_ (R/W)	Y	N
	Digital gain applied to green pixels on blue/green rows of the pixel array. The value is an unsigned 8.8 fixed-point format. Bits [10:8] are significant and are an alias of R0x305C-D[11:9].				
R0x303A	7:0	0x000A	SMIA_VERSION_ (RO)	N	N
	Return the value 10 to indicate an implementation of revision 1.0 of the SMIA specification. Read-only.				
R0x303B	7:0	0x00FF	FRAME_COUNT_ (RO)	Y	N
	In the soft standby state this counter is set to 0xFF. In streaming state this counter increments by 1 (modulo 255) at the start of each frame. The counter is incremented for both good frames and bad (corrupted) frames - its behavior is not affected by the state of R0x301A-B[9] (mask_corrupted_frames). After entry to the streaming state, the first frame will show a frame count of 0x01 in its embedded data. Read-only.				
R0x303C	15:0	0x0000	FRAME_STATUS (RO)		
	15:2	X	Reserved		
	1	RO	FRAME_STATUS_STANDBY This bit tells you whether the sensor is in standby state. Can be polled after standby is entered to see when the real low-power state is entered; which can happen at the end of row or frame depending on bit 0x301A[4]. The bit actually reflects the internal signal standby_gated.	N	N
	0	RO	FRAME_STATUS_FRAMESYNC Set on register write and reset on framesync. Acts as debug flag to verify that register writes completed before last framesync.	N	N

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3040	15:0	0x0024	READ_MODE (R/W)		
	15:14	0x0000	READ_MODE_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 11 = Reserved	N	N
	13	X	Reserved		
	12	0x0000	Reserved		
	11	0x0000	READ_MODE_X_BIN_EN 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.	N	N
	10	0x0000	READ_MODE_XY_BIN_EN 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.	N	N
	9	0x0000	READ_MODE_LOW_POWER Enables low power mode. This will automatically halve the pixel clock speed. Can not be used when pc_speed[2:0] = 4.	Y	YM
	8	X	Reserved		
	7:5	0x0001	READ_MODE_X_ODD_INC 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 (“skip 2x”).	Y	YM
	4:2	0x0001	READ_MODE_Y_ODD_INC 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 (“skip 2x”).	Y	YM
	1	0x0000	READ_MODE_VERT_FLIP 0 = Normal readout 1 = Readout is flipped (mirrored) vertically so that the row specified by y_addr_end_ is read out of the sensor first. Setting this bit will change the bayer pixel order (see R0x3024).	Y	YM
	0	0x0000	READ_MODE_HORIZ_MIRROR 0 = Normal readout 1 = Readout is mirrored horizontally so that the column specified by x_addr_end_ is read out of the sensor first. Setting this bit will change the bayer pixel order (see R0x3024).	Y	YM

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3046	15:0	0x0600	FLASH (R/W)		
	15	RO	FLASH_STROBE Reflects the current state of the FLASH output signal. Read-only.	N	N
	14	RO	FLASH_TRIGGERED Indicates that the FLASH output signal was asserted for the current frame. Read-only.	N	N
	13	0x0000	FLASH_XENON_FLASH Enable Xenon flash. When set, the FLASH output signal will assert for the programmed period (R0x3048-9) during vertical blanking. This is achieved by keeping the integration time equal to one frame, and the pulse width less than the vertical blanking time.	Y	N
	12:11	0x0000	FLASH_FRAME_DELAY Flash pulse delay measured in frames.	N	N
	10	0x0001	FLASH_END_OF_RESET 1 = In Xenon mode, the flash is triggered after resetting a frame. 0 = In Xenon mode, the flash is triggered after a frame readout.	N	N
	9	0x0001	FLASH_EVERY_FRAME 1 = Flash should be enabled every frame. 0 = Flash should be enabled for 1 frame only.	N	N
	8	0x0000	FLASH_LED_FLASH Enable LED flash. When set, the FLASH output signal will assert prior to the start of the resetting of a frame and will remain asserted until the end of the frame readout.	Y	Y
	7:0	X	Reserved		
R0x3048	15:0	0x0008	FLASH_COUNT (R/W)	N	N
	Length of flash pulse when Xenon flash is enabled. The value specifies the length in units of 256 x PIXCLK cycle increments (by default, PIXCLK = system_clock). When the Xenon count is set to its maximum value (0x3FF), the flash pulse will automatically be truncated prior to the readout of the first row, giving the longest pulse possible.				
R0x3056	15:0	0x0210	GREEN1_GAIN (R/W)		
	15:12	X	Reserved		
	11:9	0x0001	GREEN1_GAIN_DIGITAL_GAIN Digital Gain. Legal values 1-7.	Y	N
	8	RO	Reserved		
	7	0x0000	GREEN1_GAIN_ANALOG_GAIN Analog gain = (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0010	GREEN1_GAIN_INITIAL_GAIN Initial gain = bits [6:0] * 1/16.	Y	N

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3058	15:0	0x0210	BLUE_GAIN (R/W)		
	15:12	X	Reserved		
	11:9	0x0001	BLUE_GAIN_DIGITAL_GAIN Digital Gain. Legal values 1-7.	Y	N
	8	RO	Reserved		
	7	0x0000	BLUE_GAIN_ANALOG_GAIN Analog gain = (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0010	BLUE_GAIN_INITIAL_GAIN Initial gain = bits [6:0] * 1/16.	Y	N
R0x305A	15:0	0x0210	RED_GAIN (R/W)		
	15:12	X	Reserved		
	11:9	0x0001	RED_GAIN_DIGITAL_GAIN Digital Gain. Legal values 1-7.	Y	N
	8	RO	Reserved		
	7	0x0000	RED_GAIN_ANALOG_GAIN Analog gain = (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0010	RED_GAIN_INITIAL_GAIN Initial gain = bits [6:0] * 1/16.	Y	N
R0x305C	15:0	0x0210	GREEN2_GAIN (R/W)		
	15:12	X	Reserved		
	11:9	0x0001	GREEN2_GAIN_DIGITAL_GAIN Digital Gain. Legal values 1-7.	Y	N
	8	RO	Reserved		
	7	0x0000	GREEN2_GAIN_ANALOG_GAIN Analog gain = (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0010	GREEN2_GAIN_INITIAL_GAIN Initial gain = bits [6:0] * 1/16.	Y	N
R0x305E	15:0	0x0210	GLOBAL_GAIN (R/W)	Y	N
	Writing a gain to this register is equivalent to writing that code to each of the 4 color-specific gain registers. Reading from this register returns the value most recently written to the green1_gain register.				
R0x306A	15:0	0x0000	DATAPATH_STATUS (R/W)		
	15:6	X	Reserved		
	5	RO	Reserved		
	4	0x0000	Reserved		

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x306A	3	0x0000	DATAPATH_STATUS_FRAMEOVER A fatal error occurred because a new frame started before the current frame had completed. The usual reason for this is that the Y_OUTPUT_SIZE has been set too large so that the sensor output is still padding the frame with rows of undefined pixel data when the next frame starts. An alternative reason is that clock ratio between the VT clock domain and the OP clock domain does not allow sufficient time for the OP domain to complete a frame before the next frame starts. The first step in avoiding this error is to set Y_OUTPUT_SIZE so that it matches the number of rows generated from the pixel array (Y_ADDR_END - Y_ADDR_START + 1, taking into account any sub-sampling/binning and any scaling). If the error remains, the next step is to increase FRAME_LENGTH_LINES.	N	N
	2	0x0000	DATAPATH_STATUS_LINEOVER A fatal error occurred because the sensor output was unable to generate a full line of pixel data in the time budget provided by the setting of LINE_LENGTH_PCK and the vt_pix_clk period. The usual reason for this is that the X_OUTPUT_SIZE has been set too large so that the sensor output is still padding the row with undefined pixel data when the next row starts. An alternative reason is that clock ratio between the VT clock domain and the OP clock domain does not allow sufficient time for the OP domain to complete a row before the next row starts. The first step in avoiding this error is to set X_OUTPUT_SIZE so that it matches the number of pixels generated from the pixel array (X_ADDR_END - X_ADDR_START + 1, taking into account any sub-sampling/ binning and any scaling). If the error remains, the next step is to increase LINE_LENGTH_PCK.	N	N
	1	0x0000	DATAPATH_STATUS_FIFO_OVERFLOW A fatal error occurred because the output FIFO overflowed. The FIFO is sized to accommodate a full-length line from the pixel array, so this error can only occur when X_OUTPUT_SIZE is unnecessarily large, or when the ratio between the VT and OP clock domains has been set incorrectly. The first step in avoiding this error is to set X_OUTPUT_SIZE so that it matches the number of rows generated from the pixel array (X_ADDR_END - X_ADDR_START + 1, taking into account any sub-sampling/binning and any scaling).	N	N
	0	0x0000	DATAPATH_STATUS_FIFO_UNDERFLOW This fatal error condition is flagged if the output FIFO detects a data underflow.	N	N

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
	This register flags fatal error conditions associated with incorrect configuration of the sensor. Once any bit in this register has been set, behavior of the sensor is undefined and a reset may be required to restore correct operation. All bits in this register are cleared automatically on the transition from the software standby system state to the streaming system state.				
R0x306E	15:0	0x9000	DATAPATH_SELECT (R/W)		
	15:13	0x0004	DATAPATH_SELECT_SLEW_RATE_CTRL_PARALLEL Selects the slew (edge) rate for the DOUT[9:0], SHUTTER, FRAME_VALID, LINE_VALID and FLASH outputs. Affects only SHUTTER and FLASH outputs when parallel data output is disabled. The value 7 results in the fastest edge rates on these signals. Slowing down the edge rate can reduce ringing and electro-magnetic emissions.	N	N
	12:10	0x0004	DATAPATH_SELECT_SLEW_RATE_CTRL_PIXCLK Selects the slew (edge) rate for the PIXCLK output. Has no effect when parallel data output is disabled. The value 7 results in the fastest edge rates on this signal. Slowing down the edge rate can reduce ringing and electromagnetic emissions.	N	N
	9:8	X	Reserved		
	7	0x0000	DATAPATH_SELECT_PROFILE SMIA Profile Mode. Should only be changed in standby, and with attention to other clock settings. 0 = Profile 0 1 = Profile 1/2 In Profile 0 mode, the clock dividers are constrained to use vt_sys_clk_div and vt_pix_clk_div only. The sensor will therefore look like a Profile 0 sensor. Some Profile 1/2 functionality may not work in Profile 0 mode.	N	Y
	6:5	X	Reserved		
	4	0x0000	DATAPATH_SELECT_TRUE_BAYER Enables true Bayer scaling mode.	N	N
	3:0	X	Reserved		
R0x3070	15:0	0x0000	TEST_PATTERN_MODE_ (R/W)	N	Y
	0 = Normal operation: Generate output data from pixel array 1 = Solid color test pattern. 2 = 100% color bar test pattern 3 = Fade to gray color bar test pattern 4 = PN9 Link integrity test pattern 256 = 10-bit marching 1 test pattern 257 = 8-bit marching 1 test pattern All other values are reserved.				
R0x3072	15:0	0x0000	TEST_DATA_RED_ (R/W)	N	Y
	The value for red pixels in the bayer data used for the solid color test pattern and the test cursors.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3074	15:0	0x0000	TEST_DATA_GREENR_ (R/W)	N	Y
	The value for green pixels in red/green rows of the bayer data used for the solid color test pattern and the test cursors.				
R0x3076	15:0	0x0000	TEST_DATA_BLUE_ (R/W)	N	Y
	The value for blue pixels in the bayer data used for the solid color test pattern and the test cursors.				
R0x3078	15:0	0x0000	TEST_DATA_GREENB_ (R/W)	N	Y
	The value for green pixels in blue/green rows of the bayer data used for the solid color test pattern and the test cursors.				
R0x30A0	15:0	0x0001	X_EVEN_INC_ (RO)	N	N
	Read-only.				
R0x30A2	15:0	0x0001	X_ODD_INC_ (R/W)	Y	YM
	This register field is an alias of R0x3040-1[7:5]				
R0x30A4	15:0	0x0001	Y_EVEN_INC_ (RO)	N	N
	Read-only.				
R0x30A6	15:0	0x0001	Y_ODD_INC_ (R/W)	Y	YM
	This register field is an alias of R0x3040-1[4:2]				
R0x30B6	15:0	0x0000	DARK_GREEN1_AVERAGE (RO)	N	N
	The frame averaged Green1 black level that is used in the offset calibration algorithm. Read-only.				
R0x30B8	15:0	0x0000	DARK_BLUE_AVERAGE (RO)	N	N
	The frame averaged blue black level that is used in the offset calibration algorithm. Read-only				
R0x30BA	15:0	0x0000	DARK_RED_AVERAGE (RO)	N	N
	The frame averaged red black level that is used in the offset calibration algorithm. Read-only.				
R0x30BC	15:0	0x0000	DARK_GREEN2_AVERAGE (RO)	N	N
	The frame averaged green2 black level that is used in the offset calibration algorithm. Read-only.				
R0x30C2	15:0	0x0000	CALIB_GREEN1 (R/W)	N	Y
	Analog calibration offset for green pixels on red/green rows, 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 R0x30C0-1[0] = 0, this register is read-only and returns the current value computed by the offset calibration algorithm. If R0x30C0-1[0] = 1, this register is read/write, and can be used to set the calibration offset manually.				
R0x30C4	15:0	0x0000	CALIB_BLUE (R/W)	N	Y
	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 R0x30C0-1[0] = 0, this register is read-only and returns the current value computed by the offset calibration algorithm. If R0x30C0-1[0] = 1, this register is read/write, and can be used to set the calibration offset manually.				
R0x30C6	15:0	0x0000	CALIB_RED (R/W)	N	Y
	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 R0x30C0-1[0] = 0, this register is read-only and returns the current value computed by the offset calibration algorithm. If R0x30C0-1[0] = 1, this register is read/write, and can be used to set the calibration offset manually.				
R0x30C8	15:0	0x0000	CALIB_GREEN2 (R/W)	N	Y
	Analog calibration offset for green pixels on blue/green rows, 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 R0x30C0-1[0] = 0, this register is read-only and returns the current value computed by the offset calibration algorithm. If R0x30C0-1[0] = 1, this register is read/write, and can be used to set the calibration offset manually.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3160	15:0	0x0000	GLOBAL_SEQ_TRIGGER (R/W)		
	15:10	X	Reserved		
	9	RO	GLOBAL_SEQ_TRIGGER_GRST_RD Read-Only. Global reset read sequence indicator.	N	N
	8	RO	GLOBAL_SEQ_TRIGGER_GRST_SEQ Read-only. Global reset sequence indicator.	N	N
	7:3	X	Reserved		
	2	0x0000	GLOBAL_SEQ_TRIGGER_GLOBAL_FLASH 0 = When a Global Reset sequence is triggered, the FLASH output will remain negated. 1 = When a Global Reset sequence is triggered, the FLASH output will pulse during the integration phase.	N	Y
	1	0x0000	GLOBAL_SEQ_TRIGGER_GLOBAL_BULB 0 = Shutter open is triggered from bit[0] and shutter close is timed from the trigger point. 1 = Shutter open and close are triggered from bit[0]. This corresponds to the shutter "B" setting on a traditional camera, where "B" originally stood for "Bulb" (the shutter setting used for synchronization with a magnesium foil flash bulb) and was later considered to stand for "Brief" (an exposure that was longer than the shutter could automatically accommodate).	N	Y
	0	0x0000	GLOBAL_SEQ_TRIGGER_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 signals as a trigger input.	N	Y
R0x3162	15:0	0x0000	GLOBAL_RST_END (R/W)	N	N
	Controls the duration of the global reset row reset phase. A value of N gives a duration of N * 512 / vt_pix_clk_freq_mhz.				
R0x3164	15:0	0x0000	GLOBAL_SHUTTER_START (R/W)	N	N
	Controls the delay before the assertion of the SHUTTER output during a global reset sequence. A value of N gives an assertion time of N * 512 / vt_pix_clk_freq_mhz timed from the end of row that was in progress when the global reset sequence was triggered.				
R0x3166	15:0	0x0000	GLOBAL_READ_START (R/W)	N	N
	Controls the delay before the start of the global reset readout phase (equivalent to the end of global reset integration phase). A value of N gives a delay of N * 512 / vt_pix_clk_freq_mhz. The integration time is given by (global_read_start - global_rst_end) * 512 / vt_pix_clk_freq_mhz.				
R0x31E8	15:0	0x0000	HORIZONTAL_CURSOR_POSITION_ (R/W)	N	N
	Specify the start column for the test cursor.				
R0x31EA	15:0	0x0000	VERTICAL_CURSOR_POSITION_ (R/W)	N	N
	Specify the start column for the test cursor.				
R0x31EC	15:0	0x0000	HORIZONTAL_CURSOR_WIDTH_ (R/W)	N	N
	Specify the width, in rows, of the horizontal test cursor. A width of 0 disables the cursor.				
R0x31EE	15:0	0x0000	VERTICAL_CURSOR_WIDTH_ (R/W)	N	N
	Specify the width, in columns, of the vertical test cursor. A width of 0 disables the cursor.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x31FC	15:0	0x3020	I2C_IDS (R/W)	N	N
	Programmable two-wire serial interface slave addresses				
R0x3600	15:0	0x0000	P_GR_P0Q0 (R/W)	N	N
	P0 coefficient for Q0 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3602	15:0	0x0000	P_GR_P0Q1 (R/W)	N	N
	P0 coefficient for Q1 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3604	15:0	0x0000	P_GR_P0Q2 (R/W)	N	N
	P0 coefficient for Q2 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3606	15:0	0x0000	P_GR_P0Q3 (R/W)	N	N
	P0 coefficient for Q3 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3608	15:0	0x0000	P_GR_P0Q4 (R/W)	N	N
	P0 coefficient for Q4 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x360A	15:0	0x0000	P_RD_P0Q0 (R/W)	N	N
	P0 coefficient for Q0 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x360C	15:0	0x0000	P_RD_P0Q1 (R/W)	N	N
	P0 coefficient for Q1 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x360E	15:0	0x0000	P_RD_P0Q2 (R/W)	N	N
	P0 coefficient for Q2 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3610	15:0	0x0000	P_RD_P0Q3 (R/W)	N	N
	P0 coefficient for Q3 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3612	15:0	0x0000	P_RD_P0Q4 (R/W)	N	N
	P0 coefficient for Q4 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3614	15:0	0x0000	P_BL_P0Q0 (R/W)	N	N
	P0 coefficient for Q0 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3616	15:0	0x0000	P_BL_P0Q1 (R/W)	N	N
	P0 coefficient for Q1 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3618	15:0	0x0000	P_BL_P0Q2 (R/W)	N	N
	P0 coefficient for Q2 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x361A	15:0	0x0000	P_BL_P0Q3 (R/W)	N	N
	P0 coefficient for Q3 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x361C	15:0	0x0000	P_BL_P0Q4 (R/W)	N	N
	P0 coefficient for Q4 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x361E	15:0	0x0000	P_GB_P0Q0 (R/W)	N	N
	P0 coefficient for Q0 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3620	15:0	0x0000	P_GB_P0Q1 (R/W)	N	N
	P0 coefficient for Q1 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3622	15:0	0x0000	P_GB_P0Q2 (R/W)	N	N
	P0 coefficient for Q2 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3624	15:0	0x0000	P_GB_P0Q3 (R/W)	N	N
	P0 coefficient for Q3 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3626	15:0	0x0000	P_GB_P0Q4 (R/W)	N	N
	P0 coefficient for Q4 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3640	15:0	0x0000	P_GR_P1Q0 (R/W)	N	N
	P1 coefficient for Q0 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3642	15:0	0x0000	P_GR_P1Q1 (R/W)	N	N
	P1 coefficient for Q1 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3644	15:0	0x0000	P_GR_P1Q2 (R/W)	N	N
	P1 coefficient for Q2 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3646	15:0	0x0000	P_GR_P1Q3 (R/W)	N	N
	P1 coefficient for Q3 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3648	15:0	0x0000	P_GR_P1Q4 (R/W)	N	N
	P1 coefficient for Q4 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x364A	15:0	0x0000	P_RD_P1Q0 (R/W)	N	N
	P1 coefficient for Q0 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x364C	15:0	0x0000	P_RD_P1Q1 (R/W)	N	N
	P1 coefficient for Q1 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x364E	15:0	0x0000	P_RD_P1Q2 (R/W)	N	N
	P1 coefficient for Q2 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3650	15:0	0x0000	P_RD_P1Q3 (R/W)	N	N
	P1 coefficient for Q3 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3652	15:0	0x0000	P_RD_P1Q4 (R/W)	N	N
	P1 coefficient for Q4 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3654	15:0	0x0000	P_BL_P1Q0 (R/W)	N	N
	P1 coefficient for Q0 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3656	15:0	0x0000	P_BL_P1Q1 (R/W)	N	N
	P1 coefficient for Q1 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3658	15:0	0x0000	P_BL_P1Q2 (R/W)	N	N
	P1 coefficient for Q2 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x365A	15:0	0x0000	P_BL_P1Q3 (R/W)	N	N
	P1 coefficient for Q3 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x365C	15:0	0x0000	P_BL_P1Q4 (R/W)	N	N
	P1 coefficient for Q4 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x365E	15:0	0x0000	P_GB_P1Q0 (R/W)	N	N
	P1 coefficient for Q0 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3660	15:0	0x0000	P_GB_P1Q1 (R/W)	N	N
	P1 coefficient for Q1 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3662	15:0	0x0000	P_GB_P1Q2 (R/W)	N	N
	P1 coefficient for Q2 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3664	15:0	0x0000	P_GB_P1Q3 (R/W)	N	N
	P1 coefficient for Q3 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3666	15:0	0x0000	P_GB_P1Q4 (R/W)	N	N
	P1 coefficient for Q4 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3680	15:0	0x0000	P_GR_P2Q0 (R/W)	N	N
	P2 coefficient for Q0 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3682	15:0	0x0000	P_GR_P2Q1 (R/W)	N	N
	P2 coefficient for Q1 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3684	15:0	0x0000	P_GR_P2Q2 (R/W)	N	N
	P2 coefficient for Q2 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3686	15:0	0x0000	P_GR_P2Q3 (R/W)	N	N
	P2 coefficient for Q3 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3688	15:0	0x0000	P_GR_P2Q4 (R/W)	N	N
	P2 coefficient for Q4 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x368A	15:0	0x0000	P_RD_P2Q0 (R/W)	N	N
	P2 coefficient for Q0 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x368C	15:0	0x0000	P_RD_P2Q1 (R/W)	N	N
	P2 coefficient for Q1 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x368E	15:0	0x0000	P_RD_P2Q2 (R/W)	N	N
	P2 coefficient for Q2 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3690	15:0	0x0000	P_RD_P2Q3 (R/W)	N	N
	P2 coefficient for Q3 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3692	15:0	0x0000	P_RD_P2Q4 (R/W)	N	N
	P2 coefficient for Q4 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3694	15:0	0x0000	P_BL_P2Q0 (R/W)	N	N
	P2 coefficient for Q0 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3696	15:0	0x0000	P_BL_P2Q1 (R/W)	N	N
	P2 coefficient for Q1 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3698	15:0	0x0000	P_BL_P2Q2 (R/W)	N	N
	P2 coefficient for Q2 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x369A	15:0	0x0000	P_BL_P2Q3 (R/W)	N	N
	P2 coefficient for Q3 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x369C	15:0	0x0000	P_BL_P2Q4 (R/W)	N	N
	P2 coefficient for Q4 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x369E	15:0	0x0000	P_GB_P2Q0 (R/W)	N	N
	P2 coefficient for Q0 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36A0	15:0	0x0000	P_GB_P2Q1 (R/W)	N	N
	P2 coefficient for Q1 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36A2	15:0	0x0000	P_GB_P2Q2 (R/W)	N	N
	P2 coefficient for Q2 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36A4	15:0	0x0000	P_GB_P2Q3 (R/W)	N	N
	P2 coefficient for Q3 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36A6	15:0	0x0000	P_GB_P2Q4 (R/W)	N	N
	P2 coefficient for Q4 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x36C0	15:0	0x0000	P_GR_P3Q0 (R/W)	N	N
	P3 coefficient for Q0 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x36C2	15:0	0x0000	P_GR_P3Q1 (R/W)	N	N
	P3 coefficient for Q1 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x36C4	15:0	0x0000	P_GR_P3Q2 (R/W)	N	N
	P3 coefficient for Q2 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x36C6	15:0	0x0000	P_GR_P3Q3 (R/W)	N	N
	P3 coefficient for Q3 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x36C8	15:0	0x0000	P_GR_P3Q4 (R/W)	N	N
	P3 coefficient for Q4 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x36CA	15:0	0x0000	P_RD_P3Q0 (R/W)	N	N
	P3 coefficient for Q0 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x36CC	15:0	0x0000	P_RD_P3Q1 (R/W)	N	N
	P3 coefficient for Q1 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x36CE	15:0	0x0000	P_RD_P3Q2 (R/W)	N	N
	P3 coefficient for Q2 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x36D0	15:0	0x0000	P_RD_P3Q3 (R/W)	N	N
	P3 coefficient for Q3 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x36D2	15:0	0x0000	P_RD_P3Q4 (R/W)	N	N
	P3 coefficient for Q4 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x36D4	15:0	0x0000	P_BL_P3Q0 (R/W)	N	N
	P3 coefficient for Q0 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x36D6	15:0	0x0000	P_BL_P3Q1 (R/W)	N	N
	P3 coefficient for Q1 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x36D8	15:0	0x0000	P_BL_P3Q2 (R/W)	N	N
	P3 coefficient for Q2 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x36DA	15:0	0x0000	P_BL_P3Q3 (R/W)	N	N
	P3 coefficient for Q3 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x36DC	15:0	0x0000	P_BL_P3Q4 (R/W)	N	N
	P3 coefficient for Q4 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x36DE	15:0	0x0000	P_GB_P3Q0 (R/W)	N	N
	P3 coefficient for Q0 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36E0	15:0	0x0000	P_GB_P3Q1 (R/W)	N	N
	P3 coefficient for Q1 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36E2	15:0	0x0000	P_GB_P3Q2 (R/W)	N	N
	P3 coefficient for Q2 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36E4	15:0	0x0000	P_GB_P3Q3 (R/W)	N	N
	P3 coefficient for Q3 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x36E6	15:0	0x0000	P_GB_P3Q4 (R/W)	N	N
	P3 coefficient for Q4 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3700	15:0	0x0000	P_GR_P4Q0 (R/W)	N	N
	P4 coefficient for Q0 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3702	15:0	0x0000	P_GR_P4Q1 (R/W)	N	N
	P4 coefficient for Q1 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3704	15:0	0x0000	P_GR_P4Q2 (R/W)	N	N
	P4 coefficient for Q2 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3706	15:0	0x0000	P_GR_P4Q3 (R/W)	N	N
	P4 coefficient for Q3 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x3708	15:0	0x0000	P_GR_P4Q4 (R/W)	N	N
	P4 coefficient for Q4 for Gr. P_GR_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gr pixels.				
R0x370A	15:0	0x0000	P_RD_P4Q0 (R/W)	N	N
	P4 coefficient for Q0 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x370C	15:0	0x0000	P_RD_P4Q1 (R/W)	N	N
	P4 coefficient for Q1 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x370E	15:0	0x0000	P_RD_P4Q2 (R/W)	N	N
	P4 coefficient for Q2 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3710	15:0	0x0000	P_RD_P4Q3 (R/W)	N	N
	P4 coefficient for Q3 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				
R0x3712	15:0	0x0000	P_RD_P4Q4 (R/W)	N	N
	P4 coefficient for Q4 for Rd. P_RD_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Rd pixels.				

Table 12: Register Description - Manufacturer-Specific (continued)

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R0x3714	15:0	0x0000	P_BL_P4Q0 (R/W)	N	N
	P4 coefficient for Q0 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3716	15:0	0x0000	P_BL_P4Q1 (R/W)	N	N
	P4 coefficient for Q1 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x3718	15:0	0x0000	P_BL_P4Q2 (R/W)	N	N
	P4 coefficient for Q2 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x371A	15:0	0x0000	P_BL_P4Q3 (R/W)	N	N
	P4 coefficient for Q3 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x371C	15:0	0x0000	P_BL_P4Q4 (R/W)	N	N
	P4 coefficient for Q4 for Bl. P_BL_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Bl pixels.				
R0x371E	15:0	0x0000	P_GB_P4Q0 (R/W)	N	N
	P4 coefficient for Q0 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3720	15:0	0x0000	P_GB_P4Q1 (R/W)	N	N
	P4 coefficient for Q1 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3722	15:0	0x0000	P_GB_P4Q2 (R/W)	N	N
	P4 coefficient for Q2 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3724	15:0	0x0000	P_GB_P4Q3 (R/W)	N	N
	P4 coefficient for Q3 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				
R0x3726	15:0	0x0000	P_GB_P4Q4 (R/W)	N	N
	P4 coefficient for Q4 for Gb. P_GB_PpQq registers are read successively when the row (Q) coefficients are calculated during the horizontal blanking period before a row containing Gb pixels.				

Programming Restrictions

The SMIA specification imposes a number of programming restrictions. An implementation naturally imposes additional restrictions.Table 13 shows a list of programming rules that must be adhered to for correct operation of the MT9T014. It is recommended that these rules are encoded into the device driver stack—either implicitly or explicitly.

Table 13: Definitions for Programming Rules

Name	Definition
xskip	xskip = 1 if x_odd_inc = 1; xskip = 2 if x_odd_inc = 3; xskip = 4
yskip	yskip = 1 if y_odd_inc = 1; yskip = 2 if y_odd_inc = 3; yskip = 4

Table 14: Programming Rules

Parameter	Minimum Value	Maximum Value	Origin
coarse_integration_time	coarse_integration_time_min	frame_length_lines - coarse_integration_time_max_margin	SMIA
fine_integration_time	fine_integration_time_min	line_length_pck - fine_integration_time_max_margin	SMIA
digital_gain_*	digital_gain_min	digital_gain_max	SMIA
digital_gain_* is an integer multiple of digital_gain_step_size			SMIA
frame_length_lines	min_frame_length_lines	max_frame_length_lines	SMIA
line_length_pck	min_line_length_pck	max_line_length_pck	SMIA
line_length_pck	((x_addr_end - x_addr_start + x_odd_inc)/xskip) + min_line_blanking_pck		SMIA
frame_length_lines	((y_addr_end - y_addr_start + y_odd_inc)/yskip) + min_frame_blanking_lines		SMIA
x_addr_start	x_addr_min	x_addr_max	SMIA
x_addr_end	x_addr_start	x_addr_max	SMIA
(x_addr_end - x_addr_start+ x_odd_inc)	must be positive	must be positive	SMIA
x_addr_start[0]	0	0	SMIA
x_addr_end[0]	1	1	SMIA
y_addr_start	y_addr_min	y_addr_max	SMIA
y_addr_end	y_addr_start	y_addr_max	SMIA
(y_addr_end - y_addr_start + y_odd_inc)/	must be positive	must be positive	SMIA
y_addr_start[0]	0	0	SMIA
y_addr_end[0]	1	1	SMIA
x_even_inc	min_even_inc	max_even_inc	SMIA
x_even_inc[0]	1	1	SMIA
y_even_inc	min_even_inc	max_even_inc	SMIA
y_even_inc[0]	1	1	SMIA

Table 14: Programming Rules (continued)

Parameter	Minimum Value	Maximum Value	Origin
x_odd_inc	min_odd_inc	max_odd_inc	SMIA
x_odd_inc[0]	1	1	SMIA
y_odd_inc	min_odd_inc	max_odd_inc	SMIA
y_odd_inc[0]	1	1	SMIA
scale_m	scaler_m_min	scaler_m_max	SMIA
scale_n	scaler_n_min	scaler_n_max	SMIA
x_output_size	256	2064	Minimum from SMIA FS Section 5.2.2.5. Maximum is a consequence of the output FIFO size on this implementation.
x_output_size[0]	0 (this is enforced in hardware: bit[0] is read-only)	0	SMIA FS Section 5.2.2.2.
y_output_size	2	frame_length_lines	Minimum ensures 1 bayer row-pair. Maximum avoids output frame being longer than pixel array frame.
y_output_size[0]	0 (this is enforced in hardware: bit[0] is read-only)	0	SMIA FS Section 5.2.2.2
With subsampling, start and end pixels must be addressed (impact on x/y start/end addresses, function of image orientation bits)			SMIA FS Errata See "Subsampling" on page 79.

Output Size Restrictions

The SMIA CCP2 specification imposes the restriction that an output line shall be a multiple of 32 bits in length. This imposes an additional restriction on the legal values of x_output_size:

- When ccp_format[7:0] = 8 (RAW8 data), x_output_size must be a multiple of 4 (x_output_size[1:0] = 0).
- When ccp_format[7:0] = 10 (RAW10 data), x_output_size must be a multiple of 16 (x_output_size[3:0] = 0).

This restriction only applies when the serial pixel data path is in use. It can be met by rounding up x_output_size to an appropriate multiple. Any extra pixels in the output image as a result of this rounding contain undefined pixel data but are guaranteed not to cause false synchronization on the CCP2 data stream.

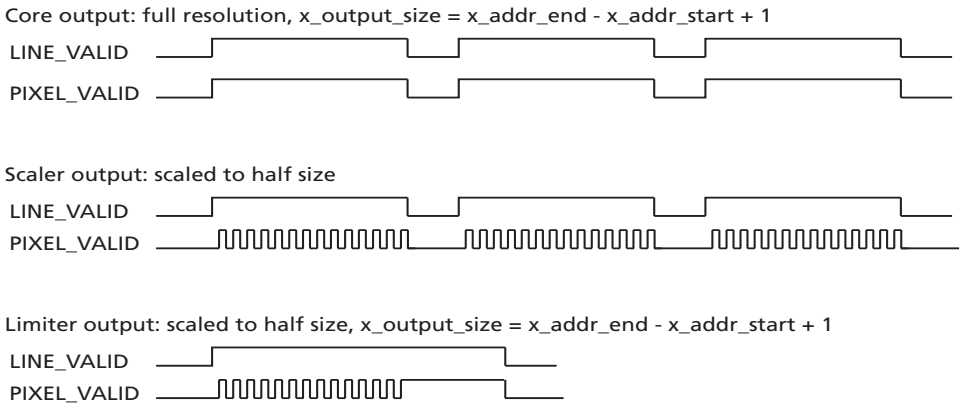
When the parallel pixel data path is in use, the only restriction on x_output_size is that it must be even (x_output_size[0] = 0), and this restriction is enforced in hardware.

When the serial pixel data path is in use, there is an additional restriction that x_output_size must be small enough such that the output row time (set by x_output_size, the framing and CRC overhead of 12 bytes, the ccp_signalling_mode and the output clock rate) must be less than the row time of the video array (set by line_length_pck and the video timing clock rate).

Effect of Scaler on Legal Range of Output Sizes

When the scaler is enabled, it is necessary to adjust the values of x_output_size and y_output_size to match the image size generated by the scaler. The MT9T014 will mis-operate if the x_output_size and y_output_size are significantly larger than the output image. To understand the reason for this, consider the situation where the sensor is operating at full resolution and the scaler is enabled with a scaling factor of 32 (half the number of pixels in each direction). This situation is shown in Figure 14 on page 62.

Figure 14: Effect of Limiter on the SMIA Data Path



In this figure, three different stages in the SMIA data path (see “Digital Data Path” on page 99) are shown. The first stage is the output of the sensor core. The core is running at full resolution and x_output_size is set to match the active array size. The LINE_VALID signal is asserted once per row and remains asserted for *N* pixel times. The PIXEL_VALID signal toggles with the same timing as LINE_VALID, indicating that all pixels in the row are valid.

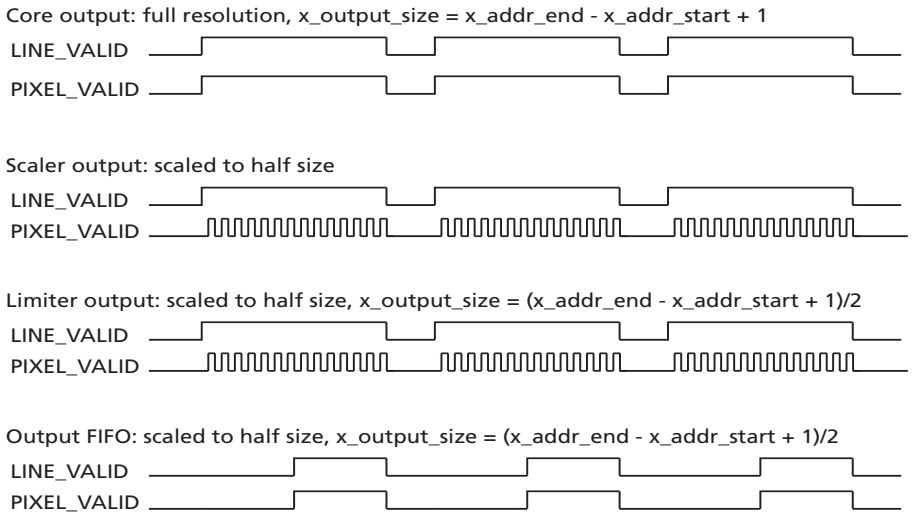
The second stage is the output of the scaler, when the scaler is set to reduce the image size by one-half in each dimension. The effect of the scaler is to combine groups of pixels. Therefore, the row time remains the same, but only half the pixels out of the scaler are valid. This is signalled by transitions in PIXEL_VALID. Overall, PIXEL_VALID is asserted for (*N*/2) pixel times per row.

The third stage is the output of the limiter when the x_output_size is still set to match the active array size. Because the scaler has reduced the amount of valid pixel data without reducing the row time, the limiter attempts to pad the row with (*N*/2) additional pixels. If this has the effect of extending LINE_VALID across the whole of the horizontal blanking time, the MT9T014 will cease to generate output frames.

A correct configuration is shown in Figure 15 on page 63, in addition to showing the x_output_size reduced to match the output size of the scaler. In this configuration, the output of the limiter does not extend LINE_VALID.

Figure 15 also shows the effect of the output FIFO, which forms the final stage in the SMIA data path. The output FIFO merges the intermittent pixel data back into a contiguous stream. Although not shown in this example, the output FIFO is also capable of operating with an output clock that is at a different frequency from its input clock.

Figure 15: Timing of SMIA Data Path



Effect of CCP2 Class on Legal Range of Output Sizes/Frame Rate

The pixel array readout rate is set by line_length_pck * frame_length_lines. With the default register values, one frame time takes 2622 * 1,621 = 4250262 pixel periods. This value includes vertical and horizontal blanking times so that the full-size image 2048 x 1538 (1,536 lines of pixel data, 2 lines of embedded information) forms a subset of these pixels.

When the internal clock is running at 64 MHz, this frame time corresponds to 5484544/64e6 = 64.41ms, giving rise to a frame rate of 15.06 fps.

Each pixel is 10 bits, by default. As a result, the serial data rate is required to transmit faster than the pixel rate. However, the SMIA CCP2 class 2 specifications has a maximum of 650 Mb/s, which cannot be exceeded.

The SMIA CCP2 specification shows that class 0 (data/clock) runs up to 208 Mb/s. Therefore, it is not possible to transmit full-resolution images at 15 fps using CCP2 class 0. Changing the ccp_data_format (to use 8 bits per pixel) reduces the bandwidth requirement, but is not enough to allow full-resolution operation.

The only way to get a full image out is to reduce the pixel clock rate until it is appropriate for the maximum CCP2 class 0 data rate. This requires the pixel rate to be reduced to 20.8 MHz. This has the side effect of reducing the frame rate. Repeating the calculation above, at 20.8 MHz internal clock, this corresponds to 5484544/20.8e6 = 0.2043 second, giving rise to a frame rate of 4.89 fps.

To use CCP2 class 0 with an internal clock of 64 MHz, it is necessary to reduce the amount of output data. This can be achieved by changing x_output_size, y_output_size so that less data comes out per frame. A change to the output size can be done in conjunction with windowing the image from the sensor (by adjusting x_addr_start, x_addr_end, y_addr_start, y_addr_end) or by enabling the scaler.

Output Data Timing

The output FIFO acts as a boundary between two clock domains. Data is written to the FIFO in the VT (video timing) clock domain. Data is read out of the FIFO in the OP (output) clock domain.

When the scaler is disabled, the data rate in the VT clock domain is constant and uniform during the active period of each pixel array row readout. When the scaler is enabled, the data rate in the VT clock domain becomes intermittent, corresponding to the data reduction performed by the scaler.

A key constraint when configuring the clock for the output FIFO is that the frame rate out of the FIFO must exactly match the frame rate into the FIFO. When the scaler is disabled, this constraint can be met by imposing the rule that the row time on the CCP2 data stream must be greater than or equal to the row time at the pixel array. The row time on the CCP2 data stream is calculated from the x_output_size and the ccp_data_format (8 or 10 bits per pixel), and must include the time taken in the CCP2 data stream for start of frame/row, end of row/frame and checksum symbols.

If this constraint is not met, the FIFO will either underrun or overrun. FIFO underrun or overrun is a fatal error condition that is signalled through the data path_status register (R0x306A).

Changing Registers while Streaming

The following registers should only be reprogrammed while the sensor is in software standby:

- ccp2_channel_identifier
- ccp2_signalling_mode
- ccp_data_format
- scale_m
- vt_pix_clk_div
- vt_sys_clk_div
- pre_pll_clk_div
- pll_multiplier
- op_pix_clk_div
- op_sys_clk_div
- Profile 0/1, 2 selection

Programming Restrictions when Using Global Reset

Interactions between the registers that control the global reset imposes some programming restrictions on the way in which they are used; these are discussed in "Global Reset" on page 87.

Control of the Signal Interface

This section describes the operation of the signal interface in all functional modes.

Serial Register Interface

The serial register interface uses the following signals:

- SCLK
- SDATA
- SADDR (via a GPI pad)

SCLK is an input-only signal and must always be driven to a valid logic level for correct operation; if the driving device can place this signal in High-Z state, an external pull-up resistor should be connected on this signal.

SDATA is a bidirectional signal. An external pull-up resistor should be connected on this signal.

SADDR is a signal which can be optionally enabled and controlled by a GPI pad to select an alternate slave address. These slave addresses can also be programmed via R0x31FC.

This interface is described in detail in "Two-Wire Serial Register Interface" on page 106.

Default Power-Up State

The MT9T014 provides interfaces for pixel data via the CCP2 high-speed serial interface described by the SMIA specification, and a parallel data interface.

At power up and after a hard or soft reset, the reset state of the MT9T014 is to enable the SMIA CCP2 high-speed serial interface.

SMIA CCP2 requirements and supports both data/clock signalling and data/strobe signalling.

The serial pixel data interface is enabled by default at power up and after reset.

The DATA_P, DATA_N, CLK_P, and CLK_N pads are turned off if the SMIA serial disable bit is asserted (R0x301A–B[12] = 1) or when the sensor is in the soft standby state.

In data/clock mode, the clock remains HIGH when no data is being transmitted. In data/strobe mode before frame start, clock is LOW and data is HIGH.

When the serial pixel data interface is used, the LINE_VALID, FRAME_VALID, PIXCLK, and DOUT[9:0] signals can be left unconnected.

Serial Pixel Data Interface

The serial pixel data interface uses the following output-only signal pairs:

- DATA_P
- DATA_N
- CLK_P
- CLK_N

The signal pairs are driven differentially using sub-LVDS switching levels.

This interface conforms to the SMIA CCP2 requirements and supports both data/clock signaling and data/strobe signaling.

The serial pixel data interface is enabled by default at power up and after reset. The DATA_P, DATA_N, CLK_P, and CLK_N pads are turned off if the SMIA serial disable bit is asserted (R0x301A–B[12] = 1) or when the sensor is in the soft standby state.

In data/clock mode, the clock remains HIGH when no data is being transmitted. In data/strobe mode before frame start, clock is LOW and data is HIGH.

When the serial pixel data interface is used, the LINE_VALID, FRAME_VALID, PIXCLK and DOUT[9:0] signals can be left unconnected.

Parallel Pixel Data Interface

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

- FRAME_VALID
- LINE_VALID
- PIXCLK
- DOUT[9:0]

When the parallel pixel data interface is in use, the DATA_P, DATA_N, CLK_P, and CLK_N signals can be left unconnected. Set reset_register[12] to disable the serializer while in parallel output mode.

Output Enable Control

When the parallel pixel data interface is enabled, its signals can be switched asynchronously between the driven and High-Z state under pin or register control, as shown in Table 15. Selection of a pin to use for the OE_N function is described in "General Purpose Inputs" on page 70.

Table 15: Output Enable Control

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

Configuration of the Pixel Data Interface

Fields in R0x301A are used to configure the operation of the pixel data interface. The supported combinations are shown in Table 16.

Table 16: Configuration of the Pixel Data Interface

Serializer Disable R0x301A–B[12]	Parallel Enable R0x301A–B[7]	Standby End-of-Frame R0x301A–B[4]	Description
0	0	1	Power up default. Serial pixel data interface and its clocks are enabled. Transitions to soft standby are synchronized to the end of frames on the serial pixel data interface.
1	1	0	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of the current row readout on the parallel pixel data interface.
1	1	1	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of frames on the parallel pixel data interface.

System States

The system states of the MT9T014 are represented as a state diagram in Figure 16 and described in subsequent sections. The effect of RESET_BAR on the system state and the configuration of the PLL in the different states are shown in Table 17 on page 69.

The sensors operation is broken down into three separate states: hardware standby, soft standby, and streaming. The transition between these states might take a certain amount of clock cycles as outlined in Table 17 on page 69.

Figure 16: MT9T014 System States

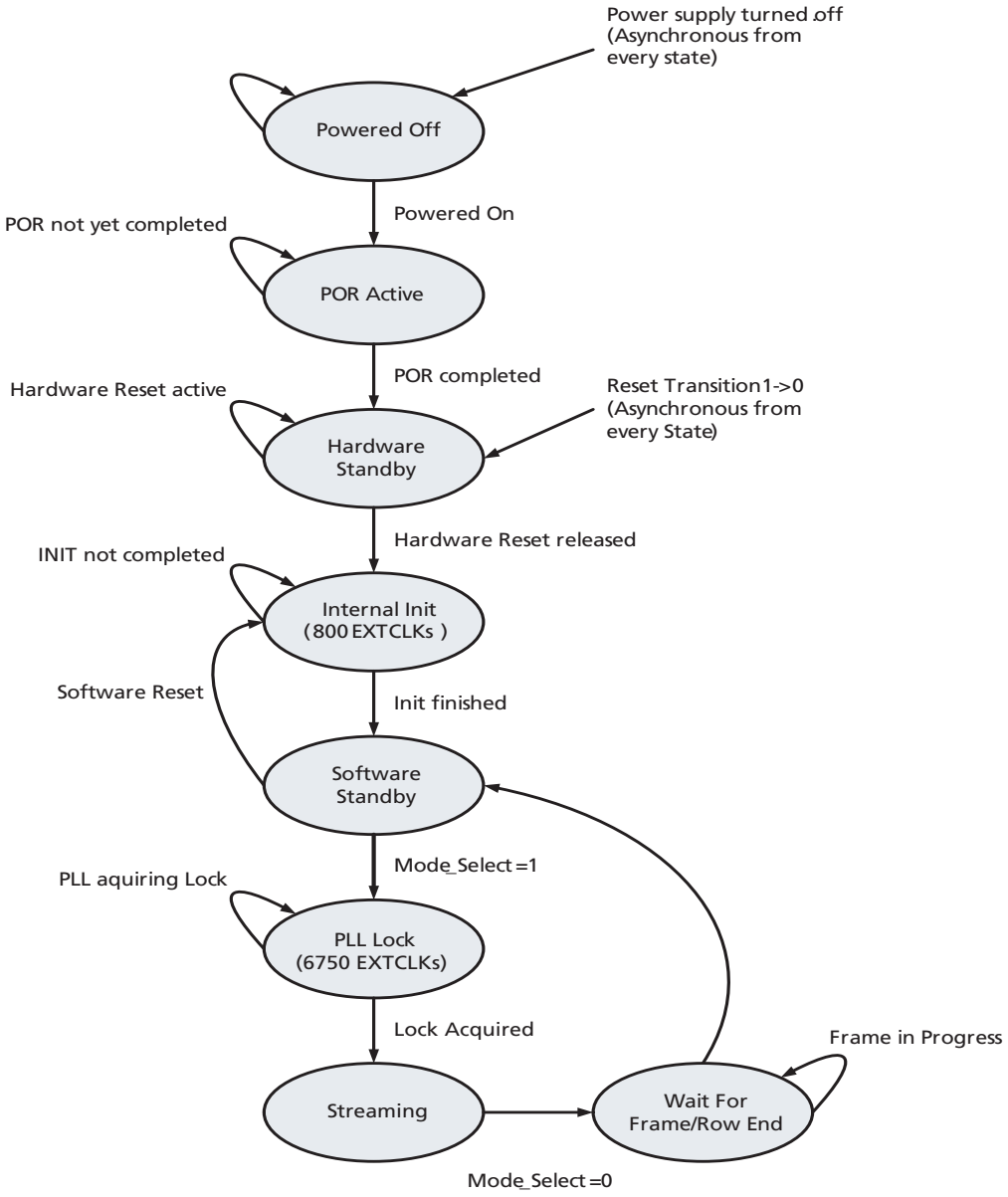


Table 17: PLL in System States

State	EXTCLKs	PLL
Powered off		
POR Active		
Hardware standby		
Internal Initialization	2400	VCO powered down
Software standby		
PLL Lock	6750	VCO powering up and locking, PLL output bypassed
Streaming		VCO running, PLL output active
Wait for frame end		

Power-On Reset Sequence

When power is applied to the MT9T014, it enters a low-power hardware standby state. Exit from this state is controlled by the later of two events:

1. The negation of the RESET_BAR input.
2. A timeout of the internal power-on reset circuit.

It is possible to hold RESET_BAR permanently negated and rely upon the internal power-on reset circuit.

When RESET_BAR is asserted it asynchronously resets the sensor, truncating any frame that is in progress.

When the sensor leaves the hardware standby state it performs an internal initialization sequence that takes 800 EXTCLK cycles. After this time it enters a low-power soft standby state. While the initialization sequence is in progress, the MT9T014 will not respond to read transactions on its two-wire serial interface. Therefore, a method to determine when the initialization sequence has completed is to poll a sensor register; for example, R0x0000. While the initialization sequence is in progress, the sensor will not respond to its device address and reads from the sensor will result in a NACK on the two-wire serial interface bus. When the sequence has completed, reads will return the operational value for the register (0x26 if R0x0000 is read).

When the sensor leaves soft standby mode and enables the VCO, an internal delay will keep the PLL disconnected for up to 6750 EXTCLKs so that the PLL can lock.

Soft Reset Sequence

The MT9T014 can be reset under software control by writing "1" to software_reset (R0x0103). A software reset asynchronously resets the sensor, truncating any frame that is in progress. The sensor starts the internal initialization sequence, while the PLL and analog blocks are turned off. At this point, the behavior is exactly the same as for the power-on reset sequence.

Signal State During Reset

Table 18 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during soft standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 18: Signal State During Reset

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Self biased. Can be left disconnected/floating.	
RESET_BAR (XSHUTDOWN)	Input	Enabled. Must be driven to a valid logic level.	
LINE_VALID	Output	High-Z. Can be left disconnected/floating.	
FRAME_VALID	Output		
DOUT[9:0]	Output		
PIXCLK	Output		
SCLK	Input	Enabled. Must be pulled up or driven to a valid logic level.	
SDATA	I/O	Enabled as an input. Must be pulled up or driven to a valid logic level.	
FLASH	Output	High-Z.	Logic 0.
SHUTTER	Output	High-Z.	Logic 0.
DATA_P	Output	High-Z.	
DATA_N	Output		
CLK_P	Output		
CLK_N	Output		
GPI[1:0]	Input	Powered down. Can be left disconnected/floating.	
TEST	Input	Enabled. Must be driven to a logic 0.	

General Purpose Inputs

The MT9T014 provides two general purpose inputs. After reset, the input pads associated with these signals are powered down by default, allowing the pads to be left disconnected/floating.

The general purpose inputs are enabled by setting reset_register[8] (R0x301A–B). Once enabled, both of the inputs must be driven to valid logic levels by external signals. The state of the general purpose inputs can be read through gpi_status[1:0] (R0x3026–7).

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

- Output enable (see “Output Enable Control” on page 66)
- Trigger (see the sections below)
- Standby functions (see the following sections)
- SADDR selection (see “Serial Register Interface” on page 65).

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

Streaming/Standby Control

The MT9T014 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 19. Selection of a pin to use for the STANDBY function is described in "General Purpose Inputs" on page 70. The state diagram for transitions between soft standby and streaming states is shown in Figure 16 on page 68.

Table 19: Streaming/STANDBY

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

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 20. Selection of a pin to use for the TRIGGER function is described in "General Purpose Inputs" on page 70.

Table 20: Trigger Control

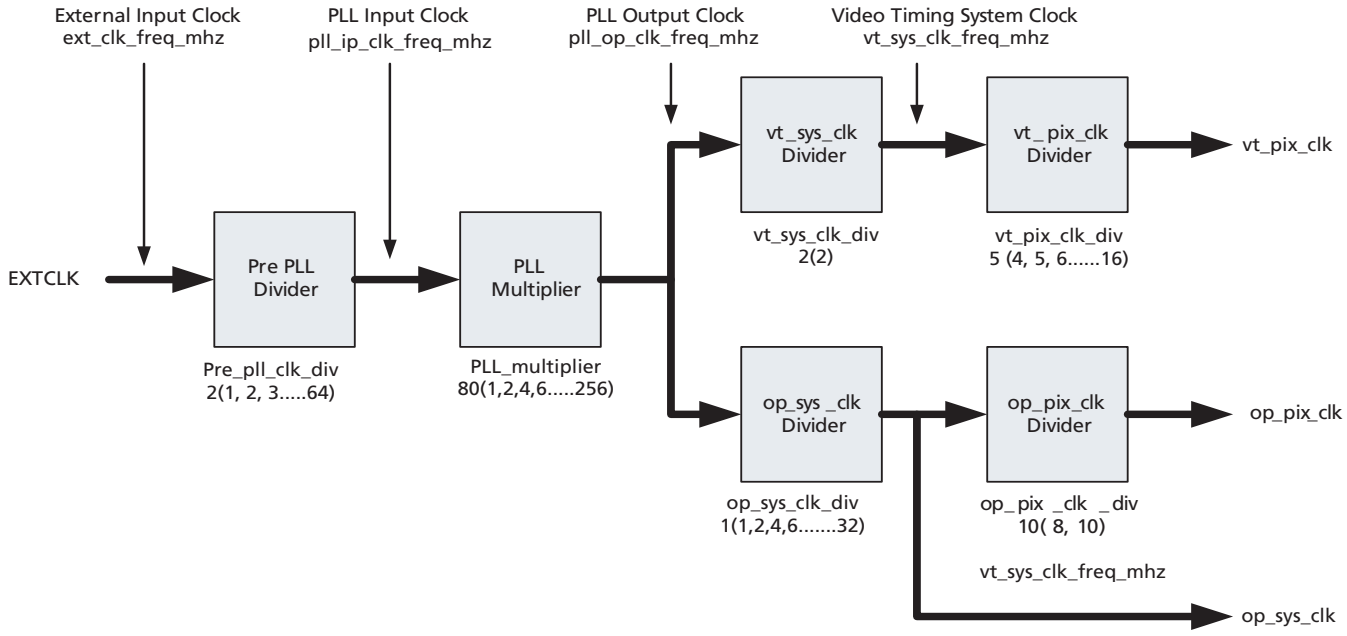
Trigger	Global Trigger R0x3160–1[0]	Description
Disabled	0	Idle
Disabled	1	Trigger
0	0	Idle
X	1	Trigger
1	X	Trigger

Clocking

The MT9T014 contains a PLL for timing generation and control. The PLL contains a prescaler to divide the input clock applied on EXTCLK, a VCO to multiply the prescaler output, and a set of dividers to generate the output clocks.

Both SMIA profile 0 clock scheme and profile 1, 2 are supported. The clocking scheme can be selected by either setting R0x306E–F[7] to 0 for profile 0 or to 1 for profile 1, 2.

Figure 17: MT9T014 SMIA Profile 1, 2 Clocking Structure



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

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

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

- The minimum/maximum frequency limits for the associated clock must be met.
- The minimum/maximum value for the divider/multiplier must be met.
- The value of pll_multiplier should be a multiple of 2.
- The op_pix_clk must never run faster than the vt_pix_clk to ensure that the CCP2 output data stream is contiguous.
- Given the maximum programmed line length, the minimum blanking time, the maximum image width, the available PLL divisor/multiplier values, and the requirement that the output line time (including the necessary blanking) must be output in a time equal to or less than the time defined by line_length_pck, the valid combinations of the clock divisors.

PLL input clock frequency range, after the pre-PLL divider stage, is 2.0–11.5 MHz.

The usage of the output clocks is shown below:

- vt_pix_clk is used by the sensor core to control the timing of the pixel array. The sensor core produces one 10-bit pixel each vt_pix_clk period. The line length (line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of the vt_pix_clk period.
- op_pix_clk is used to load parallel pixel data from the output FIFO (see Figure 41 on page 99) to the CCP2 serializer. The output FIFO generates one pixel each op_pix_clk period. The pixel is either 8-bit or 10-bit depending upon the output data format, controlled by R0x0112-3 (ccp_data_format).
- op_sys_clk is used to generate the serial data stream on the CCP2 output. The relationship between this clock frequency and the op_pix_clk frequency is dependent upon the output data format.

In Profile 1, 2, the output clock frequencies can be calculated as:

$$vt\_pix\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * vt\_sys\_clk\_div * vt\_pix\_clk\_div} \quad \text{(EQ 1)}$$

$$op\_pix\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * op\_sys\_clk\_div * op\_pix\_clk\_div} \quad \text{(EQ 2)}$$

$$op\_sys\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * op\_sys\_clk\_div} \quad \text{(EQ 3)}$$

Figure 18: MT9T014 SMIA Profile 0 Clocking Structure

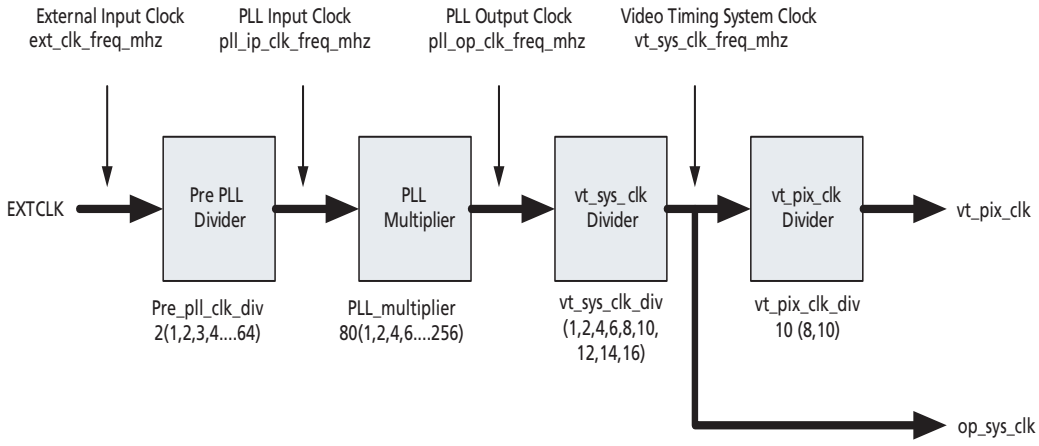


Figure 18 shows the different clocks and the names of the registers that contain or are used to control their values. Figure 18 also shows the default setting for each divider/multipler control register and the range of legal values for each divider/multiplier control register.

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

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

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

- The minimum/maximum frequency limits for the associated clock must be met.
- The minimum/maximum value for the divider/multiplier must be met.
- Given the maximum programmed line length, the minimum blanking time, the maximum image width, the available PLL divisor/multiplier values, and the requirement that the output line time (including the necessary blanking) must be output in a time equal to or less than the time defined by line_length_pck.

PLL input clock frequency range. is 2.0–11.5 MHz

The usage of the output clocks is shown below:

- vt_pix_clk is used by the sensor core to control the timing of the pixel array. The sensor core produces one 10-bit pixel each vt_pix_clk period. The line length (line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of the vt_pix_clk period.
- vt_sys_clk is also used to generate the serial data stream on the CCP2 output.

In Profile 0 the output clock frequencies can be calculated as:

$$vt\_pix\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * vt\_sys\_clk\_div * 10} \quad \text{(EQ 4)}$$

$$op\_pix\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * vt\_sys\_clk\_div * 10} \quad \text{(EQ 5)}$$

$$op\_sys\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * vt\_sys\_clk\_div} \quad \text{(EQ 6)}$$

Programming the PLL Divisors

The PLL divisors should be programmed while the MT9T014 is in the soft standby state. After programming the divisors, it is necessary to wait for the VCO lock time before enabling the PLL. The PLL is enabled by entering the STREAMING state.

An external timer needs to delay the entering of STREAMING mode by 1ms so that the PLL can lock.

The effect of programming the PLL divisors while the MT9T014 is in the streaming state is undefined.

Influence of ccp_data_format

R0x0112-3 (ccp_data_format) controls whether the pixel data interface will generate 10 bits per pixel or 8 bits per pixel. The raw output of the sensor core is 10 bits per pixel; the two 8-bit modes represent a compressed data mode and a mode in which the two least significant bits of the 10-bit data are discarded.

When the pixel data interface is generating 8 bits per-pixel, op_pix_clk_div must be programmed with the value 8. When the pixel data interface is generating 10 bits per-pixel, op_pix_clk_div must be programmed with the value 10.

Influence of ccp2_signalling_mode

R0x0111 (ccp2_signalling_mode) controls whether the serial pixel data interface uses data/strobe signalling or data/clock signalling.

When data/clock signalling is selected, the pll_multiplier supports both odd and even values.

When data/strobe signalling is selected, the pll_multiplier only supports even values; the least significant bit of the programmed value is ignored and treated as "0."

This behavior is a result of the implementation of the CCP serializer and the PLL. When the serializer is using data/strobe signalling, it uses both edges of the op_sys_clk, and therefore that clock runs at one half of the bit rate. All of the programmed divisors are set up to make this behavior invisible. For example, when the divisors are programmed to generate a PLL output of 640 MHz, the actual PLL output is 320 MHz, but both edges are used.

When the serializer is using data/clock signalling, it uses a single edge on the op_sys_clk, and therefore that clock runs at the bit rate.

In order to disguise this behavior from the programmer, the actual PLL multiplier is right-shifted by one bit relative to the programmed value when ccp2_signalling_mode selects data/strobe signalling.

Clock Control

The MT9T014 uses an aggressive clock-gating methodology to reduce power consumption. The clocked logic is divided into a number of separate domains, each of which is only clocked when required.

When the MT9T014 enters a low-power state, almost all of the internal clocks are stopped. The only exception is that a small amount of logic is clocked so that the two-wire serial interface continues to respond to read and write requests.

Features

Shading Correction (SC)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing fixed pattern signal gradients in images captured by image sensors. The cumulative result of all these factors is known as image shading. The MT9T014 has an embedded shading correction module that can be programmed to counter the shading effects on each individual Red, GreenB, GreenR, and Blue color signal.

The Correction Function

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

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

$$Pcorrected(row, col) = Psensor(row,col) * f(row,col) \quad \text{(EQ 7)}$$

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

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

One-Time Programmable (OTP) Memory

The MT9T014 has a six-byte OTP memory that can be utilized during module manufacturing to store specific information about the module. This feature provides system integrators and module manufacturers the ability to label and distinguish various module types based on lens, IR-cut filter, or other properties.

During the programming process, a dedicated pin for high voltage needs to be provided in order to perform the programming operation. This voltage (VPP) would need to be 8.5V ± 5 percent. Completion of the programming process will be communicated by a register through the two-wire serial interface.

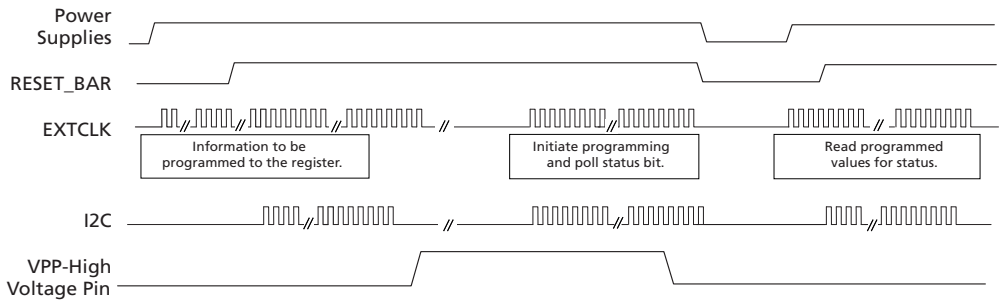
Since this programming pin needs to sustain a higher voltage than other input/output pins, having a dedicated high voltage pin (VPP) minimizes the design risk. If the module manufacturing process can probe the sensor at the die or PCB level (that is, supply all the power rails, clocks, two-wire serial interface signals), then this dedicated high voltage pin does not need to be assigned to the module connector pin-out. However, if the VPP pin needs to be bonded out as a pin on the module, the trace for VPP needs to carry a maximum of 200µA. This pin should be left floating, but may optionally be connect to analog ground.

The programming of the OTP memory requires the sensor to be fully powered and remain in software standby with its clock input applied. The information will be programmed through the use of the two-wire serial interface, and once the data is written to an internal register, the programming host machine will apply a high voltage to the programming pin, and send a program command the programming process. After the sensor has finished programming the OTP memory, a status bit will be set to indicate the end of the programming cycle, and the host machine can poll the setting of the status bit through the two-wire serial interface.

Reading the OTP memory data requires the sensor to be fully powered and operational with its clock input applied. The data can be read through a register from the two-wire serial interface. The steps below describe the steps of operational during the manufacturing process:

1. Apply power to all the power rails of the sensor.
2. Supply 8.5V to VPP pin.
3. Provide 12 MHz EXTCLK clock input.
4. Perform the proper reset sequence to the sensor.
5. Place the sensor in soft standby (sensor default state upon power-up).
6. Set burn duration to 16ms by programming R0x3054 = 0x50B6.
7. Write information to be programmed to R0x3800–5 on the sensor through the two-wire serial interface.
8. Set the control bit to R0x304A–B to initiate programming.
9. Poll R0x304A[1] for the completion of programming.
10. Remove high voltage.
11. Set EXTCLK to normal operating frequency (24 MHz).
12. Perform the proper reset sequence to the sensor.
13. Set control bit to R0x304A–B to initiate reading process.
14. Read data from R0x3800–5.

Figure 19: Sequence for Programming the Device



Notes:
1. Timing specifications NOT included in this sequence. Timing information will be present after characterization.
2. I^2C bus is pulled high when NOT in active state. This is NOT presented in the timing diagram.
3. Sensor powers up to soft standby and should remain in soft standby for the duration of the programming of the OTP memory.

Image Acquisition Modes

The MT9T014 supports two image acquisition modes:

- Electronic rolling shutter (ERS) mode. This is the normal mode of operation. When the MT9T014 is streaming, it generates frames at a fixed rate, and each frame is integrated (exposed) using the ERS. When the ERS is in use, timing and control logic within the sensor sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and subsequently reading that row, the pixels in the row integrate incident light. The integration (exposure) time is controlled by varying the time between row reset and row readout. For each row in a frame, the time between row reset and row readout is fixed, leading to a uniform integration time across the frame. When the integration time is changed (by using the two-wire serial interface to change register settings), the timing and control logic controls the transition from old to new integration time in such a way that the stream of output frames from the MT9T014 switches cleanly from the old integration time to the new while only generating frames with uniform integration.
- Global reset mode. This mode can be used to acquire a single image at the current resolution. In this mode, the pixel integration time is controlled by an external electromechanical shutter, and the MT9T014 provides control signals to interface to that shutter. The operation of this mode is described in detail in "Global Reset" on page 87.

The use of an external electromechanical shutter increases cost and may reduce ruggedness of the end application. The motivation for the use of an external electromechanical shutter is that it eliminates the visual artifacts associated with ERS operation. Visual artifacts arise in ERS operation, particularly at low frame rates, because an ERS image effectively integrates each row of the pixel array at a different point in time so that its operation is somewhat similar to that of a photo-finish machine at a race track.

Window Control

The sequencing of the pixel array is controlled by the x_addr_start, y_addr_start, x_addr_end, and y_addr_end registers. For both parallel and CCP interfaces, the output image size is controlled by the x_output_size and y_output_size registers.

Pixel Border

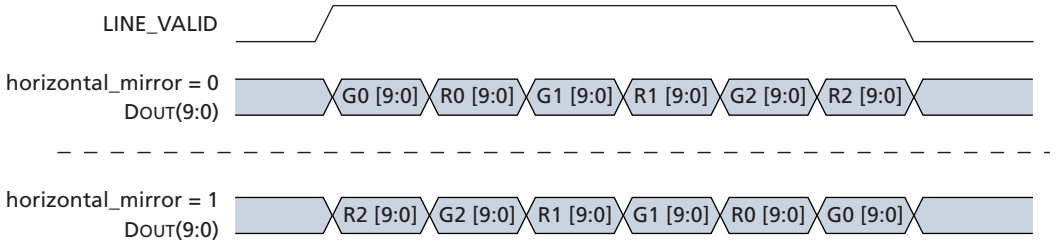
The default settings of the sensor provide a 2048H x 1536V image. A border of up to 8 pixels (4 in binning) on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end, y_addr_end, and x_output_size and y_output_size registers accordingly.

Readout Modes

Horizontal Mirror

When the horizontal_mirror bit is set in the image_orientation register, the order of pixel readout within a row is reversed, so that readout starts from x_addr_end and ends at x_addr_start. Figure 20 on page 79 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.

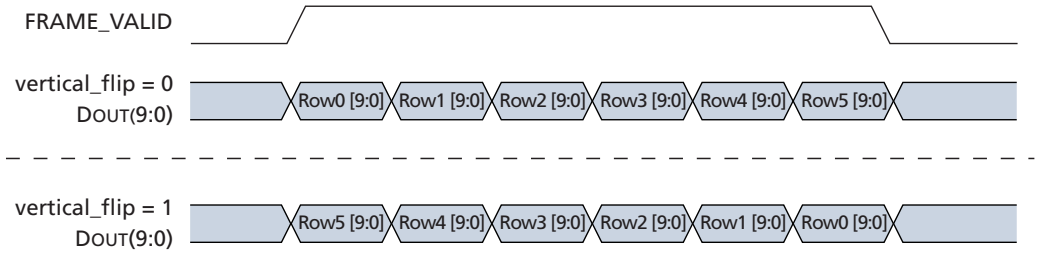
Figure 20: Effect of horizontal_mirror on Readout Order



Vertical Flip

When the vertical_flip bit is set in the image_orientation register, the order in which pixel rows are read out is reversed, so that row readout starts from y_addr_end and ends at y_addr_start. Figure 21 shows a sequence of 6 rows being read out with vertical_flip = 0 and vertical_flip = 1. Changing vertical_flip causes the bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.

Figure 21: Effect of vertical_flip on Readout Order



Subsampling

The MT9T014 supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the MT9T014 thereby allowing the frame rate to be increased. Subsampling is enabled by setting x_odd_inc and/or y_odd_inc. Values of 1 and 3 can be supported. Setting both of these variables to 3 reduces the amount of row and column data processed and is equivalent to the 2 x 2 skipping readout mode provided by the MT9T014. Figure 22 shows a sequence of 8 columns being read out with x_odd_inc = 3 and y_odd_inc = 1.

Figure 22: Effect of x_odd_inc = 3 on Readout Sequence

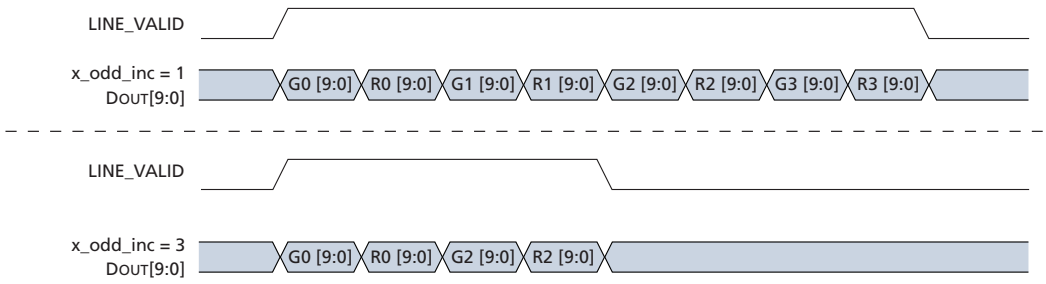


Figure 23: Pixel Readout (No Subsampling)

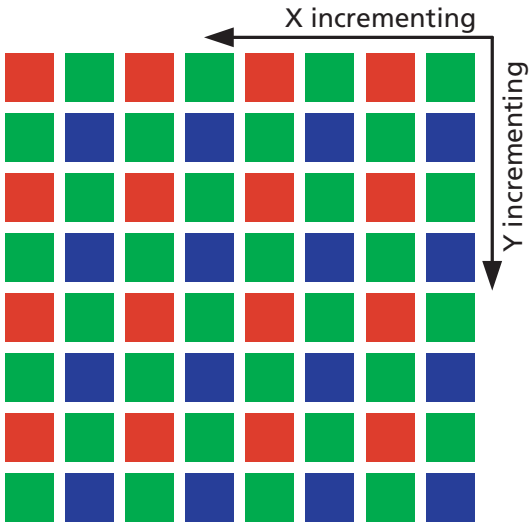
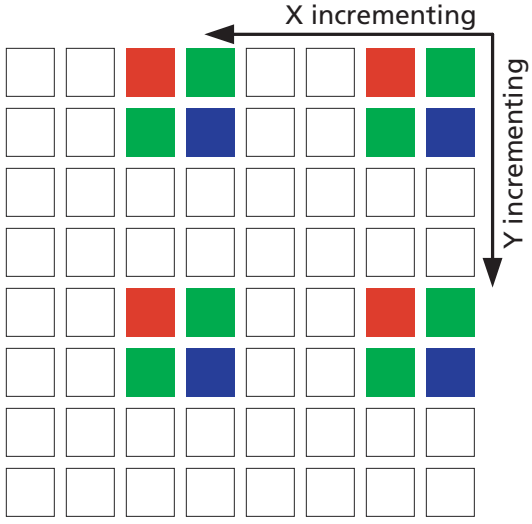


Figure 24: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)



Programming Restrictions when Subsampling

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

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

- x_addr_start must be a multiple of 2 for example 0, 4, 6, 8, and x_addr_start = 2 is not supported

When 2 x 2 skipping mode is enabled,

- (x_addr_end - x_addr_start + x_odd_inc) should be a multiple of 4
- (y_addr_end - y_addr_start + y_odd_inc) should be a multiple of 4

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

When 2 x 2 skipping mode is enabled

- columns/rows = (addr_end - addr_start + odd_inc) / 2

Example:

To achieve 2048 x 1536 full resolution without skipping, the recommended register settings are:

```
[full resolution starting address with (8,8)]
REG=0x0104, 1                   // GROUPED_PARAMETER_HOLD
REG=0x0382, 1                   // X_ODD_INC
REG=0x0386, 1                   // Y_ODD_INC
REG=0x0344, 8                   // X_ADDR_START
REG=0x0346, 8                   // Y_ADDR_START
REG=0x0348, 2055                // X_ADDR_END
REG=0x034A, 1543                // Y_ADDR_END
REG=0x034C, 2048                // X_OUTPUT_SIZE
REG=0x034E, 1536                // Y_OUTPUT_SIZE
REG=0x0104, 0                   // GROUPED_PARAMETER_HOLD
```

To achieve a 1024 x 786 resolution without 2 x 2 skipping, the recommended register settings are:

```
[2x2 skipping starting address with (8,8)]
REG=0x0104, 1                   // GROUPED_PARAMETER_HOLD
REG=0x0382, 3                   // X_ODD_INC
REG=0x0386, 3                   // Y_ODD_INC
REG=0x0344, 8                   // X_ADDR_START
REG=0x0346, 8                   // Y_ADDR_START
REG=0x0348, 2053                // X_ADDR_END
REG=0x034A, 1541                // Y_ADDR_END
REG=0x034C, 1024                // X_OUTPUT_SIZE
REG=0x034E, 768                 // Y_OUTPUT_SIZE
REG=0x0104, 0                   // GROUPED_PARAMETER_HOLD
```

Table 21 shows the row address sequencing for normal and subsampled readout. The same sequencing applies to column addresses for subsampled readout. There are two possible subsampling sequences for the rows (because the subsampling sequence only read half of the rows) depending upon the alignment of the start address. The row address sequencing during binning is also shown.

Table 21: Row Address Sequencing

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

Binning

The MT9T014 supports 2 x 1 and 2 x 2 analog 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 (odd_inc = 3 and y_odd_inc = 1 for x-binning, x_odd_inc = 3 and y_odd_inc = 3 for xy-binning) and setting the appropriate binning bit in read_mode (R0x3040–1). As for subsampling, x_addr_end and y_addr_end may require adjustment when binning is enabled as described in "Programming Restrictions when Subsampling" on page 80. Note that it is the first of the two columns/rows binned together that should be the end column/row in binning, so the requirements to the end address is exactly the same as in non-binning subsampling mode. The effect of the different subsampling settings is shown in Figure 25 and Figure 26 on page 83.

Figure 25: Pixel Readout (x_odd_inc = 3, y_odd_inc = 1, x_bin = 1)

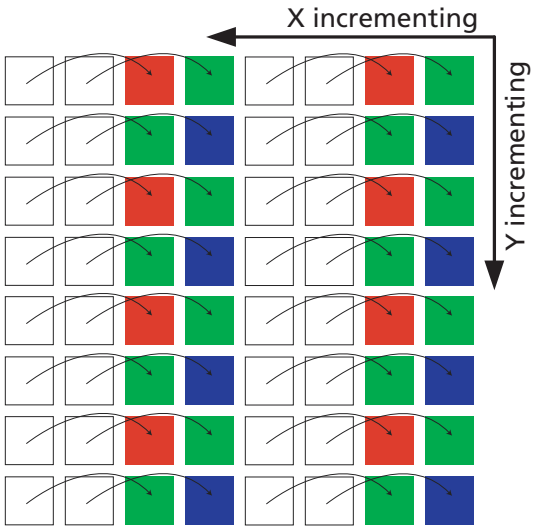
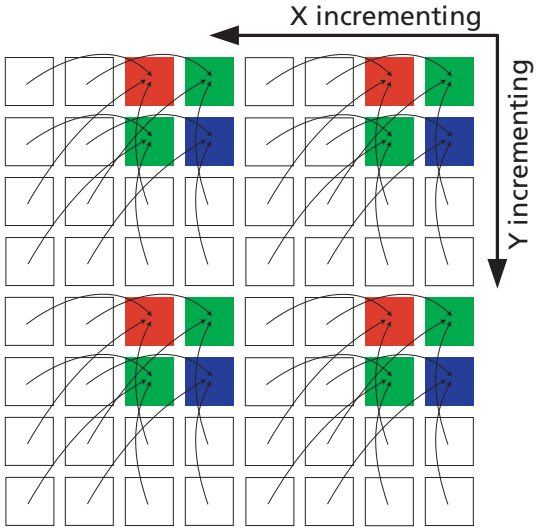


Figure 26: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3, xy_bin = 1)



Programming Restrictions when Binning

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. The recommended settings are shown in Table 22. None of these adjustments are required for x-binning.

Table 22: Register Adjustments Required for Binning Mode

Register	Type	Default (Normal Readout)	Recommended Setting During Binning	Notes
min_line_blanking_pck	read-only	0x023A	0x03A0	Read-only register for control software; does not affect operation of sensor.
min_line_length_pck	read-only	0x033A	0x0508	Read-only register for control software; does not affect operation of sensor.
fine_integration_time_min	read-only	0x01B1	0x0341	Read-only register for control software; does not affect operation of sensor.
fine_integration_time_max_margin	read-only	0x00CF	0x01C7	Read-only register for control software; does not affect operation of sensor.
fine_correction	read/write	0x00B0	0x013C	Affects operation of sensor.
fine_integration_time	read/write	0x3014	0x0341	Normal default is minimum value.

Since binning also requires subsampling to be enabled, the same restrictions apply to the setting of x_addr_end and y_addr_end (see

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

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

Frame Rate Control

The formula for calculating the frame rate of the MT9T014 are shown below:

$$\text{line\_length\_pck} = \left(\frac{\text{x\_addr\_end - x\_addr\_start + x\_odd\_inc}}{\text{subsampling factor}} + \text{min\_line\_blanking\_pck}\right) \quad \text{(EQ 8)}$$

$$\text{frame\_length\_lines} = \left(\frac{\text{y\_addr\_end - y\_addr\_start + y\_odd\_inc}}{\text{subsampling factor}} + \text{min\_frame\_blanking\_lines}\right) \quad \text{(EQ 9)}$$

$$\text{frame rate [FPS]} = \frac{(\text{vt\_pixel\_clock\_mhz} * 1\times10^{6})}{(\text{line\_length\_pck} * \text{frame\_length\_lines})} \quad \text{(EQ 10)}$$

Integration Time

The integration (exposure) time of the MT9T014 is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

$$\text{fine\_integration\_time\_min} <= \text{fine\_integration\_time} <= (\text{line\_length\_pck-fine\_integration\_time\_max\_margin}) \quad \text{(EQ 11)}$$

The limits for the coarse integration time are defined by:

$$coarse\_integration\_time\_min < = coarse\_integration\_time \quad \text{(EQ 12)}$$

If coarse_integration_time > (frame_lenth_lines-coarse_integration_time_max_margin), then the frame rate will be reduced.

The actual integration time is given by:

$$integration\_time\ [sec] = \frac{((coarse\_integration\_time * line\_length\_pck) + fine\_integration\_time)}{(vt\_pix\_clk\_freq\_mhz * 1 * 10^6)} \quad \text{(EQ 13)}$$

With a vt_pix_clk of 64 MHz, the maximum integration time that can be achieved without reducing the frame rate is given by:

$$Maximum\ integration\ time\ [sec] = \quad \text{(EQ 14)}$$

$$\frac{(((frame\_length\_lines - 1) * line\_length\_pck) + (line\_length\_pck - fine\_integration\_time\_max\_margin)}{(vt\_pix\_clk\_freq\_mhz * 1 * 10^6)}$$

$$= \frac{(((0x1621 - 1) * 0x2622) + 0x2415)}{(64\ MHz * 1 * 10^6)} = 66.41ms$$

It is fundamental to the operation of an ERS that it is not possible to set an integration time that is greater than the frame time. Setting an integration time that is greater than the frame time therefore increases the frame time beyond frame_length_lines in order to make longer exposure times available.

Flash Control

The MT9T014 supports both Xenon and LED flash through the FLASH output signal. The timing of the FLASH signal with the default settings is shown in Figure 27, 28, and Figure 29 on page 86. The flash and flash_count registers allow the timing of the flash to be changed. The flash can be programmed to fire only once, delayed by a few frames when asserted, and (for Xenon flash) the flash duration can be programmed.

Enabling the LED flash will cause one bad frame, where several of the rows only have the flash on for part of their integration time. This can be avoided by forcing a restart (write reset_register[1] = 1) immediately after enabling the flash; the first bad frame will then be masked out as shown in Figure 29 on page 86. Read-only bit flash[14] is set during frames that are correctly integrated; the state of this bit is shown in the Figure 27 on page 86, Figure 28 on page 86, and Figure 29 on page 86.

Figure 27: Xenon Flash Enabled

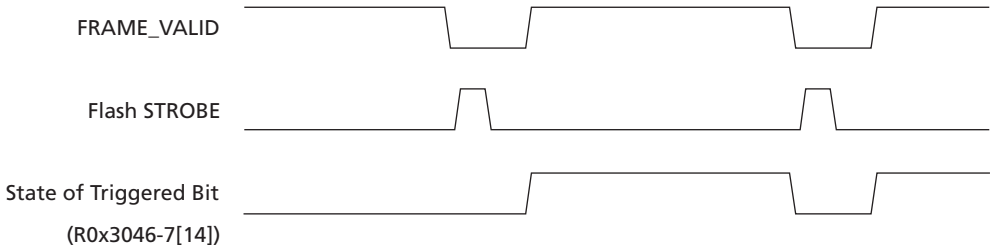
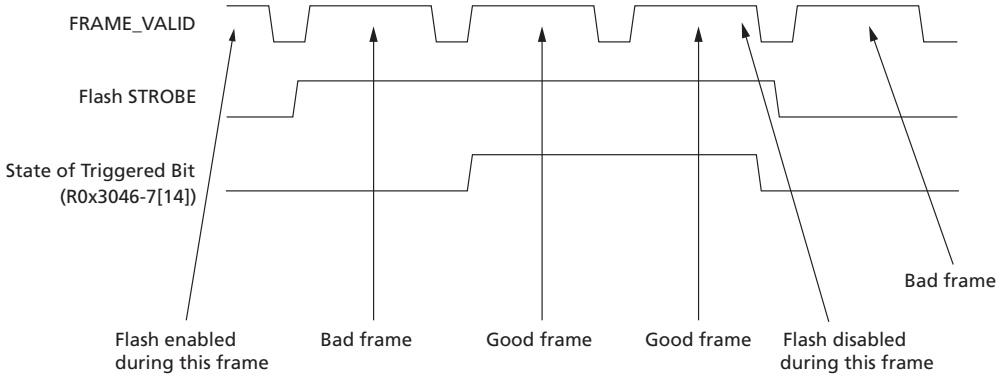
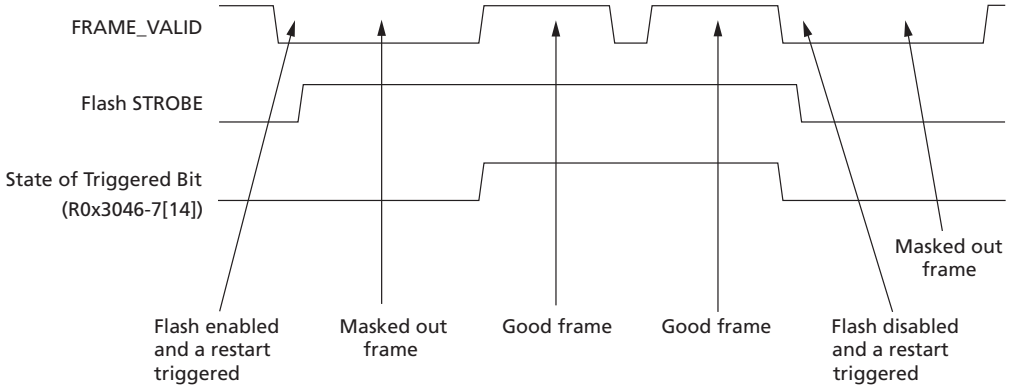


Figure 28: LED Flash Enabled



Notes: 1. Integration time = number of rows in a frame.
Bad frames will be masked when mask corrupted frames option is enabled.
An option to invert the flash output signal is also available.

Figure 29: LED Flash Enabled Following Forced Restart



Global Reset

Global reset mode allows the integration time of the MT9T014 to be controlled by an external electromechanical shutter. Global reset mode is generally used in conjunction with ERS mode. The ERS mode is used to provide viewfinder information, the sensor is switched into global reset mode to capture a single frame, and the sensor is then returned to ERS mode to restore viewfinder operation.

Global reset mode is designed for use in conjunction with the parallel pixel data interface. The SMIA specification only provides for operation in ERS mode. The MT9T014 does support the use of global reset mode in conjunction with the SMIA data path, but there are additional restrictions on its use.

Overview of Global Reset Sequence

The basic elements of the global reset sequence are described below:

1. By default, the sensor operates in ERS mode and the SHUTTER output signal is LOW. The electromechanical shutter must be open to allow light to fall on the pixel array. Integration time is controlled by the coarse_integration_time and fine_integration_time registers.
2. A global reset sequence is triggered.
3. All of the rows of the pixel array are placed in reset.
4. All of the rows of the pixel array are taken out of reset simultaneously. All rows start to integrate incident light. The electromechanical shutter may be open or closed at this time.
5. If the electromechanical shutter has been closed, it is opened.
6. After the desired integration time (controlled internally or externally to the MT9T014), the electromechanical shutter is closed.
7. A single output frame is generated by the sensor with the usual LINE_VALID, FRAME_VALID, PIXCLK, and DOUT timing. As soon as the output frame has completed (FRAME_VALID negates), the electromechanical shutter may be opened again.
8. The sensor automatically resumes operation in ERS mode.

This sequence is shown in Figure 30. The following sections expand on this figure to show how the timing of this sequence is controlled.

Figure 30: Overview of Global Reset Sequence

Entering and Leaving the Global Reset Sequence

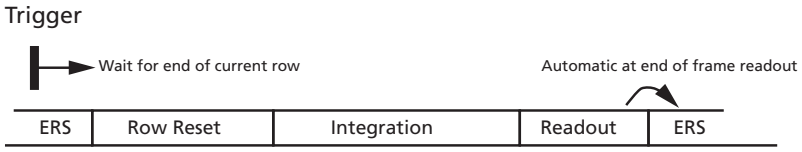
A global reset sequence can be triggered either by a register write to global_seq_trigger[0] (global trigger, to transition this bit from a 0 to a 1) or by a rising edge on a suitably-configured GPI input (see “Trigger Control” on page 71).

When a global reset sequence is triggered, the sensor waits for the end of the current row. When LINE_VALID negates for that row, FRAME_VALID is negated 6 PIXCLK periods later, potentially truncating the frame that was in progress.

The global reset sequence completes with a frame readout. At the end of this readout phase, the sensor automatically resumes operation in ERS mode. The first frame integrated with ERS will be generated after a delay of approximately ((min_frame_blanking + coarse_integration_time) * line_length_pck). This sequence is shown in Figure 31.

While operating in ERS mode, double-buffered registers ("Double-Buffered Registers" on page 19) are updated at the start of each frame in the usual way. During the global reset sequence, double-buffered registers are updated just before the start of the readout phase.

Figure 31: Entering and Leaving a Global Reset Sequence



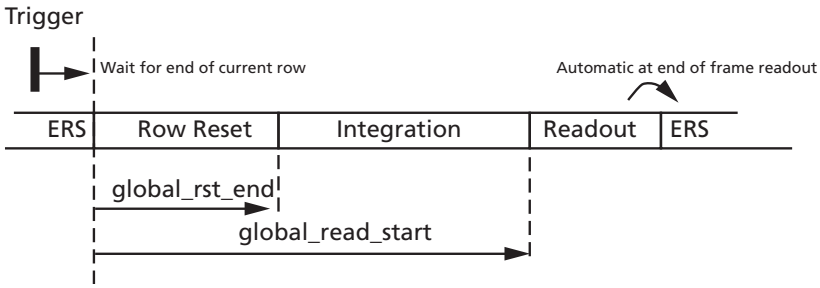
Programmable Settings

The registers global_rst_end and global_read_start allow the duration of the row reset phase and the integration phase to be controlled, as shown in Figure 32. The duration of the readout phase is determined by the active image size.

The recommended setting for global_rst_end is 0xA0 (preliminary). This allows sufficient time for all rows of the pixel array to be set to the correct reset voltage level. The row reset phase takes a finite amount of time due to the capacitance of the pixel array and the capability of the internal voltage booster circuit that is used to generate the reset voltage level.

As soon as the global_rst_end count has expired, all rows in the pixel array are taken out of reset simultaneously and the pixel array begins to integrate incident light.

Figure 32: Controlling the Reset and Integration Phases of the Global Reset Sequence

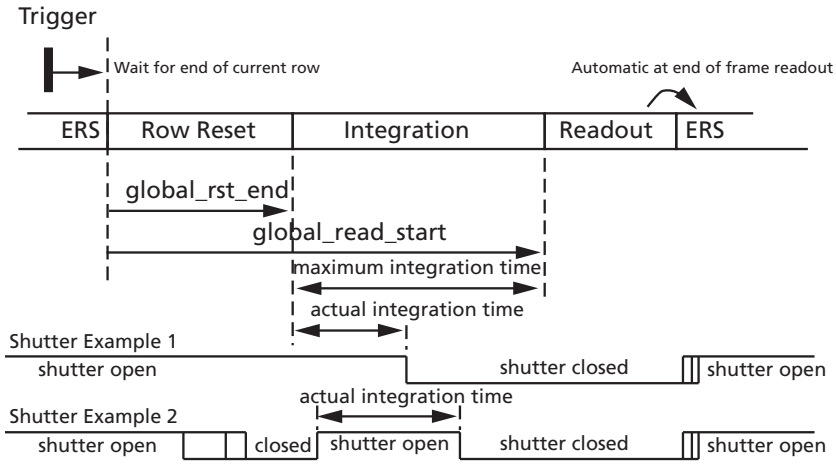


Control of the Electromechanical Shutter

Figure 33 shows two different ways in which a shutter can be controlled during the global reset sequence. In both cases, the maximum integration time is set by the difference between global_read_start and global_rst_end. In shutter example 1, the shutter is open during the initial ERS sequence and during the row reset phase. The shutter closes during the integration phase. The pixel array is integrating incident light from the start of the integration phase to the point at which the shutter closes. Finally, the shutter opens again after the end of the readout phase. In shutter example 2, the shutter is open during the initial ERS sequence and closes sometime during the row reset phase. The shutter both opens and closes during the integration phase. The pixel array is inte-

grating incident light for the part of the integration phase during which the shutter is open. As for the previous example, the shutter opens again after the end of the readout phase.

Figure 33: Control of the Electromechanical Shutter



It is essential that the shutter remains closed during the entire row readout phase (that is, until FRAME_VALID has negated for the frame readout); otherwise, some rows of data will be corrupted (over-integrated).

It is essential that the shutter closes before the end of the integration phase. If the row readout phase is allowed to start before the shutter closes, each row in turn will be integrated for one row-time longer than the previous row.

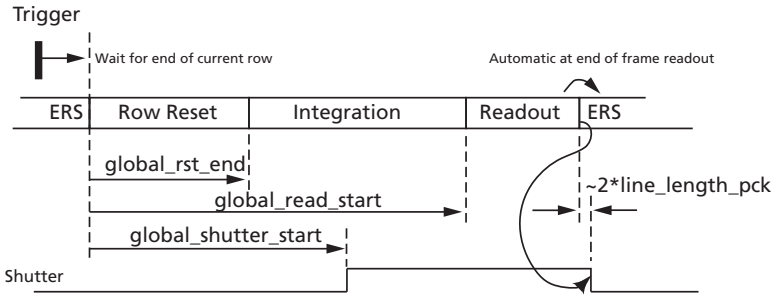
After FRAME_VALID negates to signal the completion of the readout phase, there is a time delay of approximately (10 * line_length_pck) before the sensor starts to integrate light-sensitive rows for the next ERS frame. It is essential that the shutter be opened at some point in this time window; otherwise, the first ERS frame will not be uniformly integrated.

The MT9T014 provides a SHUTTER output signal to control (or help the host system control) the electromechanical shutter. The timing of the SHUTTER output is shown in Figure 34. SHUTTER is negated by default. The point at which it asserts is controlled by the programming of global_shutter_start. At the end of the global reset readout phase, SHUTTER negates approximately (2 * line_length_pck) after the negation of FRAME_VALID.

The following programming restriction must be met for correct operation:

- global_read_start > global_shutter_start.

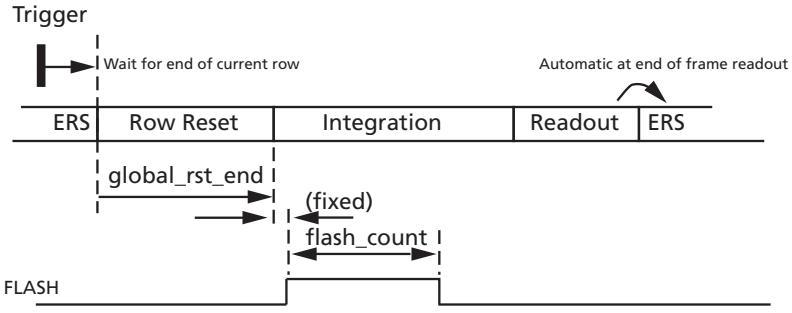
Figure 34: Controlling the SHUTTER Output



Using FLASH with Global Reset

If global_seq_trigger[2] = 1 (global flash enabled) when a global reset sequence is triggered, the FLASH output signal will be pulsed during the integration phase of the global reset sequence. The FLASH output will assert a fixed number of cycles after the start of the integration phase and will remain asserted for a time that is controlled by the value of the flash_count register, as shown Figure 35.

Figure 35: Using FLASH with Global Reset



External Control of Integration Time

If global_seq_trigger[1] = 1 (global bulb enabled) when a global reset sequence is triggered, the end of the integration phase is controlled by the level of trigger (global_seq_trigger[0] or the associated GPI input). This allows the integration time to be controlled directly by an input to the sensor and allows integration times that are longer than can be accommodated by the programming limits of the global_read_start register.

This operation corresponds to the shutter “B” setting on a traditional camera, where “B” originally stood for “Bulb” (the shutter setting used for synchronization with a magnesium foil flash bulb) and was later considered to stand for “Brief” (an exposure that was longer than the shutter could automatically accommodate).

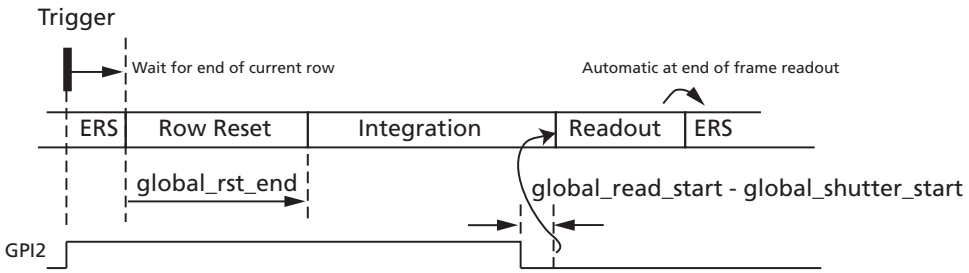
When the trigger is negated to end integration, the integration phase is extended by a further time given by (global_read_start - global_shutter_start). Usually this means that global_read_start should be set to (global_shutter_start + 1).

The operation of this mode is shown in Figure 36 on page 91. The figure shows the global reset sequence being triggered by the GPI2 input, but it could be triggered by any of the GPI inputs or by the setting and subsequence clearing of the global_seq_trigger[0] under software control.

The following programming restrictions must be met for correct operation of 'Bulb' exposures:

- global_read_start > global_shutter_start
- global_shutter_start > global_rst_end
- global_shutter_start must be smaller than the exposure time (that is, this counter must expire before the trigger is negated)

Figure 36: Global Reset Bulb



Retriggering the Global Reset Sequence

The trigger for the global reset sequence is edge sensitive; the global reset sequence cannot be retriggered until the global trigger bit (in the global_seq_trigger register) has been returned to "0," and the GPI (if any) associated with the trigger function has been negated.

The earliest time that the global reset sequence can be retriggered is the point at which the SHUTTER output negates; this occurs approximately (2 * line_length_pck) after the negation of FRAME_VALID for the global reset readout phase.

Using Global Reset with SMIA Data Path

When a global reset sequence is triggered, it usually results in the frame in progress being truncated (at the end of the current output line). The SMIA data path limiter function (see Figure 41 on page 99) attempts to extend (pad) all frames to the programmed value of y_output_size. If this padding is still in progress when the global reset readout phase starts, the SMIA data path will not detect the start of the frame correctly. Therefore, in order to use global reset with the SMIA data path, this timing scenario must be avoided. One possible way of doing this would be to synchronize (under software control) the assertion of trigger to an end-of-frame marker on the CCP serial data stream.

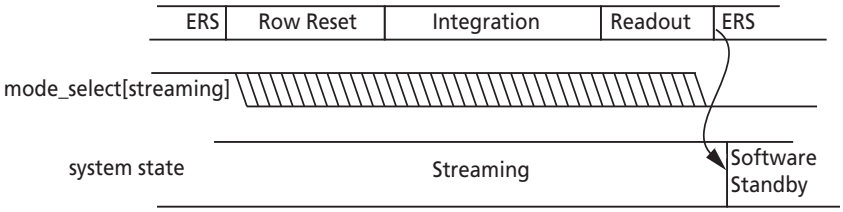
At the end of the readout phase of the global reset sequence, the sensor automatically resumes operation in ERS mode.

The frame that is read out of the sensor during the global reset readout phase has exactly the same format as any other frame out of the serial pixel data interface, including the addition of 2 lines of embedded data. The values of the coarse_integration_time and fine_integration_time registers within the embedded data match the programmed values of those registers and do *not* reflect the integration time used during the global reset sequence.

Global Reset and Soft Standby

If the mode_select[stream] bit is cleared while a global reset sequence is in progress, the MT9T014 will remain in streaming state until the global reset sequence (including frame readout) has completed, as shown in Figure 37 on page 92.

Figure 37: Entering Soft Standby During a Global Reset Sequence



Analog Gain

The MT9T014 provides two mechanisms for setting the analog gain. The first uses the SMIA gain model; the second uses the traditional Micron Imaging gain model. The following sections describe both models, the mapping between the models, and the operation of the per-color and global gain control.

Using Per-color or Global Gain Control

The read-only analogue_gain_capability register returns a value of "1," indicating that the MT9T014 provides per-color gain control. However, the MT9T014 also provides the option of global gain control. Per-color and global gain control can be used interchangeably. A write to a global gain register is aliased as a write of the same data to the four associated color-dependent gain registers. A read from a global gain register is aliased to a read of the associated greenB/greenR gain register.

The read/write gain mode register required by SMIA has no defined function in the SMIA specification.In the MT9T014 this register has no side-effects on the operation of the gain; per-color and global gain control can be used interchangeably regardless of the state of the gain_mode register.

SMIA Gain Model

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

- analogue_gain_code_global
- analogue_gain_code_greenR
- analogue_gain_code_red
- analogue_gain_code_blue
- analogue_gain_code_greenB

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

$$gain = \frac{analogue\_gain\_m0 \times analogue\_gain\_code}{analogue\_gain\_c1} = \frac{analogue\_gain\_code\_<color>}{8} \quad \text{(EQ 15)}$$

Micron Imaging Gain Model

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

- global_gain
- green1_gain
- red_gain
- blue_gain
- green2_gain

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

$$\text{gain} = (\text{<color>\_gain[7]} + 1) \times \frac{\text{<color>\_gain[6:0]}}{16} \quad \text{(EQ 16)}$$

As a result of the 2X gain stage, many of the possible gain settings can be achieved in two different ways. For example, red_gain = 0x90 provides the same gain as red_gain = 0x20. The first example uses the 2X gain stage and the second example does not. In all cases, the preferred setting is the setting that enables the 2X gain stage, since this will result in lower noise.

Gain Code Mapping

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

When the SMIA gain model is in use and values have been written to the analogue_gain_code_<color> registers, the associated value in the Micron Imaging gain model can be read from the associated <color>_gain register. In cases where there is more than one possible mapping, the 2X gain stage is enabled in order to provide the mapping with the lowest noise.

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

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

Sensor Core Digital Data Path

Test Patterns

The MT9T014 supports a number of test patterns to facilitate system debug. Test patterns are enabled using test_pattern_mode (R0x0600–1). The test patterns are listed in Table 23.

Table 23: Test Patterns

test_pattern_mode	Description
0	Normal operation: no test pattern
1	Solid color
2	100% color bars
3	Fade-to-gray color bars
4	PN9 link integrity pattern
256	Walking 1s (10-bit)
257	Walking 1s (8-bit)

Test patterns 0–3 replace pixel data in the output image (the embedded data rows are still present). Test pattern 4 replaces all data in the output image (the embedded data rows are omitted and test pattern data replaces the pixel data).

For all of the test patterns, the MT9T014 registers must be set appropriately to control the frame rate and output timing. This includes:

- All clock divisors
- x_addr_start
- x_addr_end
- y_addr_start
- y_addr_end
- frame_length_lines
- line_length_pck
- x_output_size
- y_output_size

The MT9T014 will disable this automatically when test patterns are activated. The test cursor is now added to the end of the data path.

Solid Color Test Pattern

In this mode, all pixel data is replaced by fixed Bayer pattern test data. The intensity of each pixel is set by its associated test data register (test_data_red, test_data_greenR, test_data_blue, test_data_greenB).

100 Percent Color Bars Test Pattern

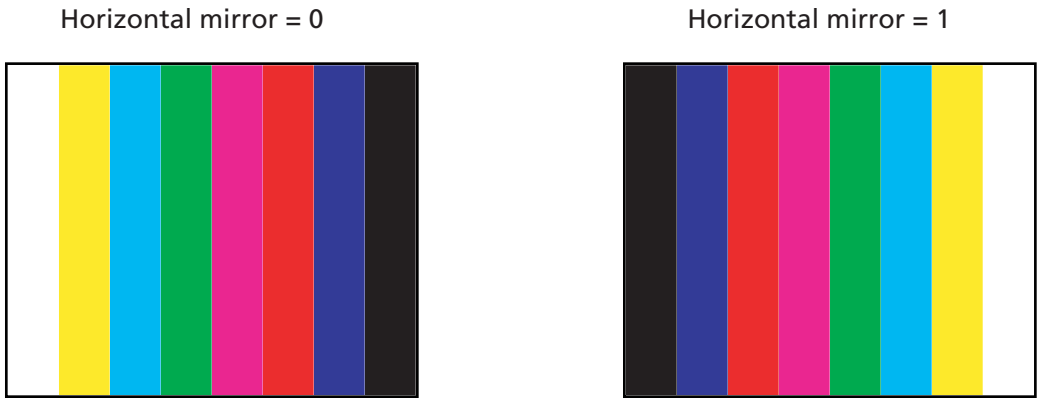
In this test pattern, shown in Figure 38, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, and black). Each bar is 256 pixels wide and occupies the full height of the output image. Each color component of each bar is set to either 0 (fully off) or 0x3FF (fully on for 10-bit data). The pattern repeats after 8*256 = 2048 pixels. The image size is set by x_addr_start, x_addr_end, y_addr_start, y_addr_end and may be affected by the setting of x_output_size, y_output_size. The color-bar pattern starts at the column identified by

x_addr_start. The number of colors that are visible in the output is dependent upon x_addr_end - x_addr_start and the setting of x_output_size. The width of each color-bar is fixed at 256 pixels.

The effect of setting horizontal_mirror in conjunction with this test pattern is that the order in which the colors are generated is reversed. The black bar appears at the left side of the output image. Any pattern repeat occurs at the right side of the output image regardless of the setting of horizontal_mirror. The state of vertical_flip has no effect on this test pattern.

The effect of subsampling, binning, and scaling of this test pattern is undefined.

Figure 38: 100 Percent Color Bars Test Pattern



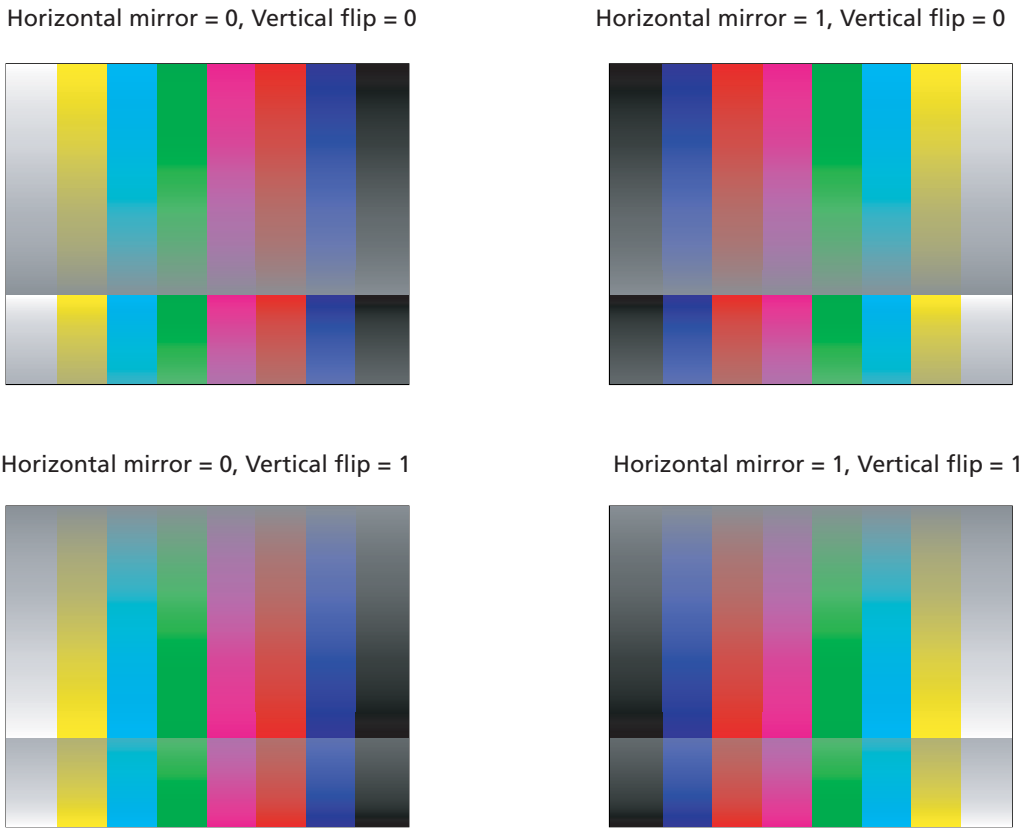
Fade-to-Gray Color Bars Test Pattern

In this test pattern, shown in Figure 39 on page 96, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, and black). Each bar is 256 pixels wide and occupies 1024 rows of the output image. Each color bar fades vertically from full intensity at the top of the image to 50 percent intensity (mid-gray) on the 1024th row. Each color bar is divided into a left and a right half, in which the left half fades smoothly and the right half fades in quantized steps every 8 pixels for a given color. Due to the Bayer pattern of the colors this means that the level changes every 16 rows. The pattern repeats horizontally after 8*256 = 2048 pixels and vertically after 1024 rows (Using 10-bit data, the fade-to-gray pattern goes from 100 to 50 percent or from 0 to 50 percent for each color component, so only half of the 2^{10} states of the 10-bit data are used. However, to get all of the gray levels, each state must be held for two rows, hence the vertical size of 2^{10} / 2 * 2 = 1024). The image size is set by x_addr_start, x_addr_end, y_addr_start, and y_addr_end and may be affected by the setting of x_output_size and y_output_size. The color-bar pattern starts at the column identified by x_addr_start. The number of colors that are visible in the output is dependent upon x_addr_end - x_addr_start and the setting of x_output_size. The width of each color-bar is fixed at 256 pixels.

The effect of setting horizontal_mirror or vertical_flip in conjunction with this test pattern is that the order in which the colors are generated is reversed. The black bar appears at the left side of the output image. Any pattern repeat occurs at the right side of the output image regardless of the setting of horizontal_mirror.

The effect of subsampling, binning, and scaling of this test pattern is undefined.

Figure 39: Fade-to-Gray Color Bars Test Pattern



PN9 Link Integrity Pattern

This test pattern provides a 512-bit pseudo-random test sequence to test the integrity of the serial pixel data output stream. The polynomial $x^9 + x^5 + 1$ is used. The polynomial is initialized to 0x1FF at the start of each frame.

When this test pattern is enabled:

- The embedded data rows are disabled, and the value of frame_format_decriptor_1 changes from 0x1002 to 0x1000 to indicate that no rows of embedded data are present.
- The whole output frame, bounded by the limits programmed in x_output_size and y_output_size, is filled with data from the PN9 sequence.
- The output data format is (effectively) forced into RAW10 mode regardless of the state of the data_format register.

This polynomial generates the following sequence of 10-bit values: 0x1FF, 0x378, 0x1A1, 0x336, 0x385, and so on. On the parallel pixel data output, these values are presented 10-bits per PIXCLK. On the serial pixel data output, these values are streamed out sequentially without performing the RAW10 packing to bytes that normally occurs on this interface.

Walking "1" Test Pattern

The main purpose of the walking "1" test pattern is to detect stuck-at bits at the parallel interface, DOUT[9:0]. During active data period, no more than one "1"-logic high-would appear at the parallel interface at any given time. Each value in the pattern would appear for two consecutive PIXCLK. The resulting pattern would have the sequence of:

RAW10:
0x000, 0x000, 0x001, 0x001, 0x002, 0x002, 0x004, 0x004, 0x008, 0x008, 0x010, 0x010, …, 0x3FF, 0x3FF

RAW8:
0x00, 0x00, 0x01, 0x01, 0x02, 0x02, 0x04, 0x04, 0x08, 0x08, 0x10, 0x10, …, 0xFF, 0xFF

The walking "1" test pattern is not active during the blanking periods, hence the output would reset to a value of 0x0. When the active period starts again, the pattern would re-start from the beginning. The behavior of this test pattern is the same between full resolution and subsampling mode. RAW10 and RAW8 walking "1" modes are enabled by different test pattern codes.

Test Cursors

The MT9T014 supports one horizontal and one vertical cursor, allowing a "cross hair" to be superimposed on the image or on test patterns 1–3.

The position and width of each cursor is programmable in R0x31E8–0x31EE. Only even cursor positions and even cursor widths are supported (this is a consequence of the internal architecture of the pixel array). Each cursor can be inhibited by setting its width to "0."

The programmed cursor position corresponds to an absolute row or column in the pixel array. For example, setting horizontal_cursor_position to the same value as y_addr_start would result in a horizontal cursor being drawn starting on the first row of the image.

The cursors are opaque (they replace data from the imaged scene or test pattern). The color of each cursor is set by the values of the bayer components in the test_data_red, test_data_greenR, test_data_blue, and test_data_greenB registers. As a consequence, the cursors are the same color as test pattern 1 and are therefore invisible when test pattern 1 is selected.

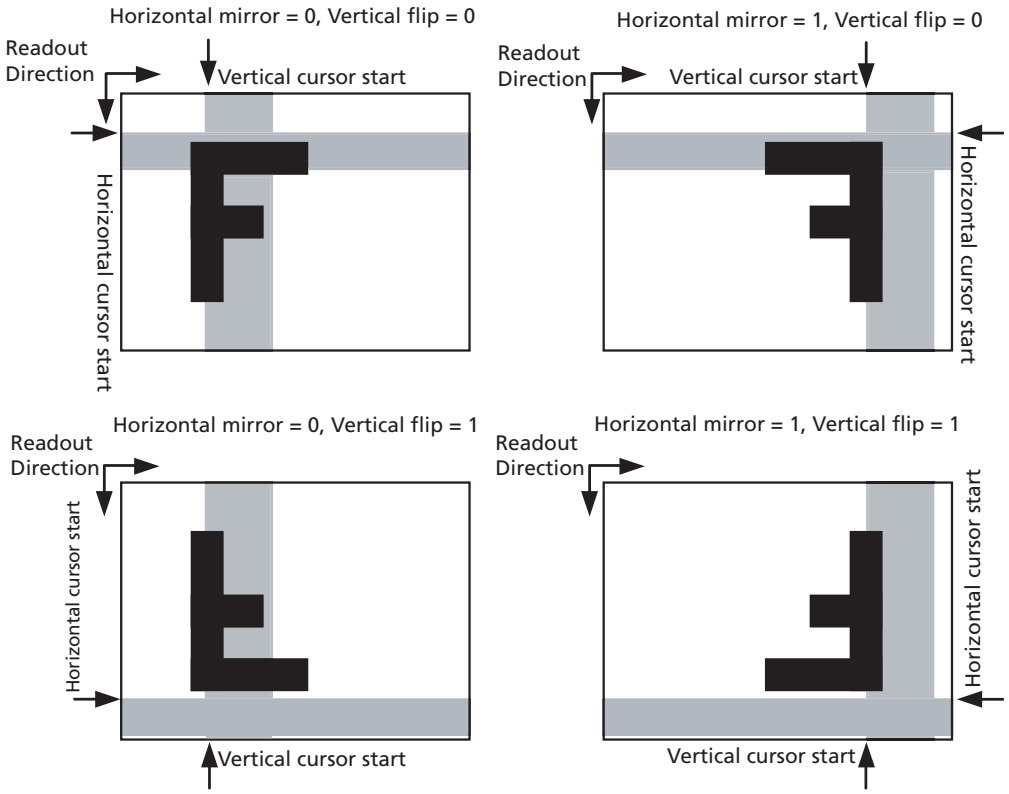
When vertical_cursor_position = 0x0FFF, the vertical cursor operates in an automatic mode in which its position advances every frame. In this mode the cursor starts at the column associated with x_addr_start=0 and advances by a step-size of 8 columns each frame until it reaches the column associated with x_addr_start = 2040, after which it wraps (256 steps). Note that the active pixel array is smaller than this, so in the last 56 steps the cursor will not be visible. The width and color of the cursor in this automatic mode are controlled in the usual way.

The effect of enabling the test cursors when the image_orientation register is non-zero is not defined by the SMIA specification. The behavior of the MT9T014 is shown in Figure 40 on page 98. In the figure, the test cursors are shown as translucent for clarity. In practice, they are opaque (they overlay the imaged scene). The manner in which the test cursors are affected by the value of image_orientation can be understood from the following implementation details:

- The test cursors are inserted early in the data path, so that they correlate to rows and to columns of the physical pixel array (rather than to x and to y coordinates of the output image).

- The drawing of a cursor starts when the pixel array row or column address matches the value of the associated cursor_position register. As a result, the cursor start position remains fixed relative to the rows and columns of the pixel array for all settings of image_orientation.
- The cursor generation continues until the appropriate cursor_width pixels have been drawn. The cursor width is generated from the start position and proceeds in the direction of pixel array readout. As a result, each cursor is reflected about an axis corresponding to its start position when the appropriate bit is set in the image_orientation register.

Figure 40: Test Cursor Behavior when image_orientation



Digital Gain

Integer digital gains in the range 1–7 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).

Pedestal

This block adds the value from R0x0008-9 (data_pedestal_) to the incoming pixel value.

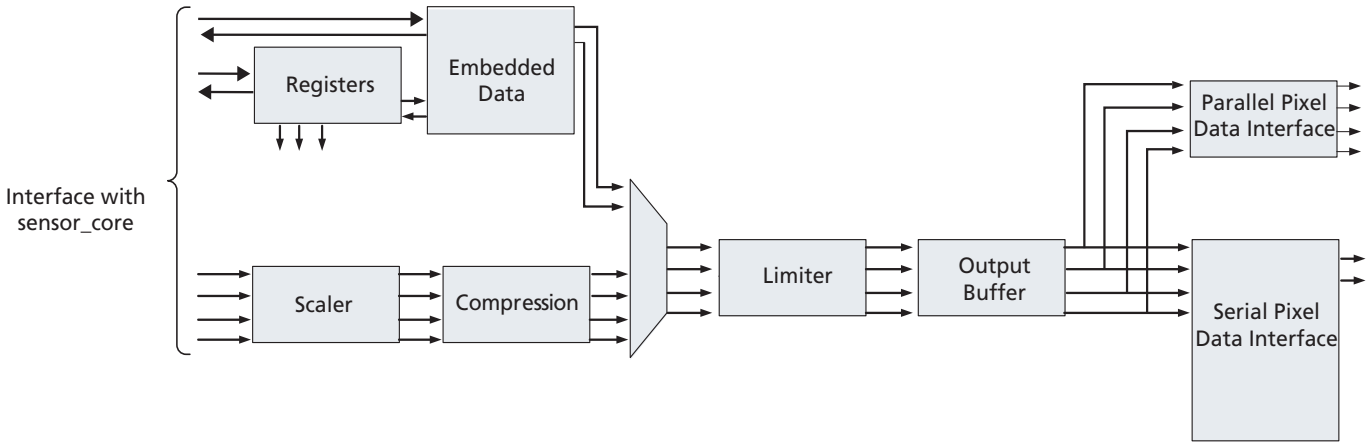
The data_pedestal register is read-only by default but can be made read/write by clearing the lock_reg bit in R0x301A–B.

The only way to disable the effect of the pedestal is to set it to “0.”

Digital Data Path

The digital data path after the sensor core is shown in Figure 41.

Figure 41: Data Path



Timing Specifications

Power-Up Sequence

The recommended power-up sequence for the MT9T014 is shown in Figure 42. The available power supplies—VDD_IO, VDD, VDD_PLL, VAA, VAA_PIX—can be turned on at the same time or have the separation specified below.

1. Turn on VDDIO power supply.
2. After 1–500ms, turn on VDD power supply.
3. After 1–500ms, turn on VDD_PLL and VAA/VAA_PIX power supplies.
4. After the last power supply is stable, enable EXTCLK.
5. Assert RESET_BAR for at least 1ms.
6. Wait 800 EXTCLKs for internal initialization into software standby.
7. Configure PLL, output, and image settings to desired values.
8. Set mode_select = 1 (R0x0100).
9. Wait 6750 EXTCLKs for the PLL to lock before streaming state is reached (enforced in hardware).

Figure 42: Power-Up Sequence

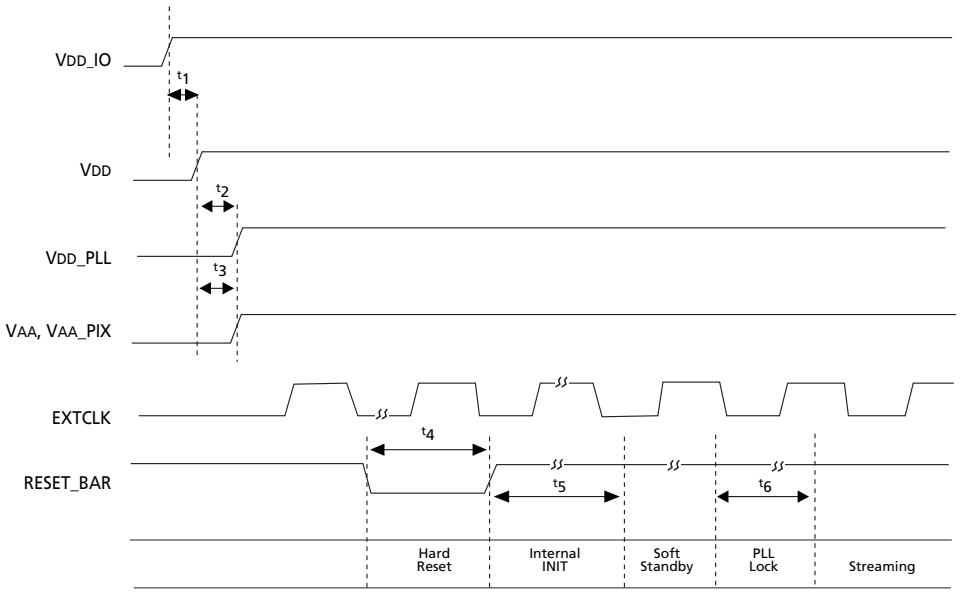


Table 24: Power-Up Sequence

Definition	Symbol	Min	Typ	Max	Unit
VDD_IO to VDD time	t_1	1	–	500	ms
VDD to VDD_PLL time	t_2	1	–	500	ms
VDD to VAA/VAA_PIX time	t_3	1	–	500	ms
Active hard reset	t_4	1	–	–	ms
Internal initialization	t_5	800	–	–	EXTCLKs
PLL lock time	t_6	6750	–	–	EXTCLKs

Power-Down Sequence

The recommended power-down sequence for the MT9T014 is shown in Figure 43. The available power supplies—VDD_IO, VDD, VDD_PLL, VAA, VAA_PIX—can be turned off at the same time or have the separation specified below.

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. Assert hard reset by setting RESET_BAR to a logic "0."
4. Turn off the VAA/VAA_PIX and VDD_PLL power supplies.
5. After 1–500ms, turn off VDD power supply.
6. After 1–500ms, turn off VDD_IO power supply.

Figure 43: Power-Down Sequence

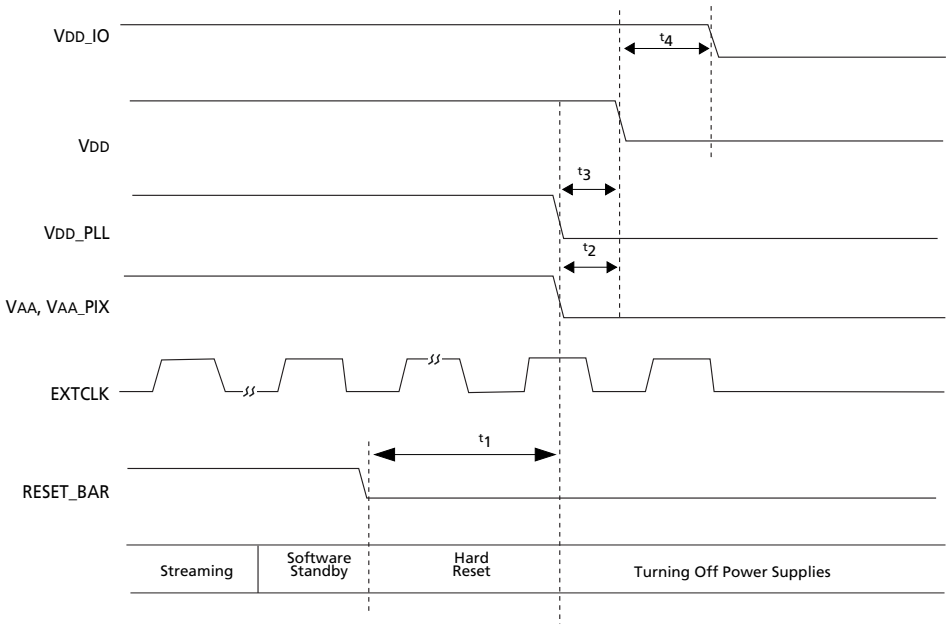


Table 25: Power-Down Sequence

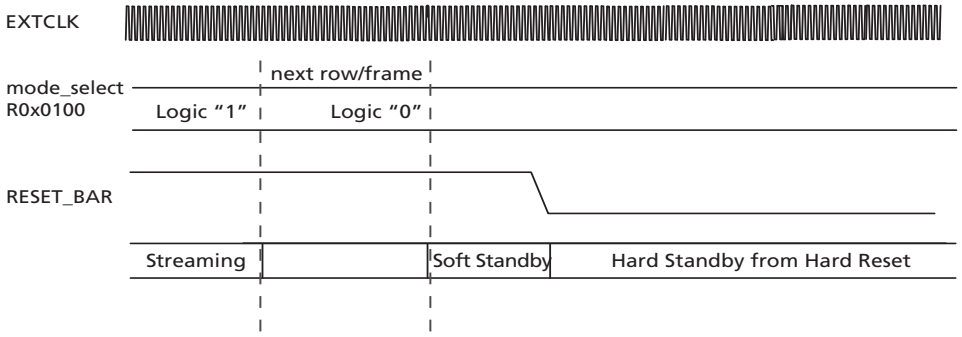
Definition	Symbol	Min	Typ	Max	Unit
Hard reset	$^{t}1$	1	–	–	ms
VDD/VAA/VAA_PIX to VDD time	$^{t}2$	0	–	500	ms
VDD_PLL to VDD time	$^{t}3$	1	–	500	ms
VDD to VDD_IO time	$^{t}4$	1	–	500	ms

Hard Standby and Hard Reset

The hard standby state is reached by the assertion of the RESET_BAR pad (hard reset). Register values are not retained by this action, and will be returned to their default values once hard reset is completed. The minimum power consumption is achieved by the hard standby state. The details of the sequence are described below and shown in Figure 44.

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. Assert RESET_BAR (active low) to reset the sensor.
4. The sensor remains in hard standby state if RESET_BAR remains in the logic “0” state.

Figure 44: Hard Standby and Hard Reset



Soft Standby and Soft Reset

The MT9T014 can reduce power consumption by switching to the soft standby state when the output is not needed. Register values are retained in the soft standby state. Once this state is reached, soft reset can be enabled optionally to return all register values back to the default. The details of the sequence are described below and shown in Figure 45.

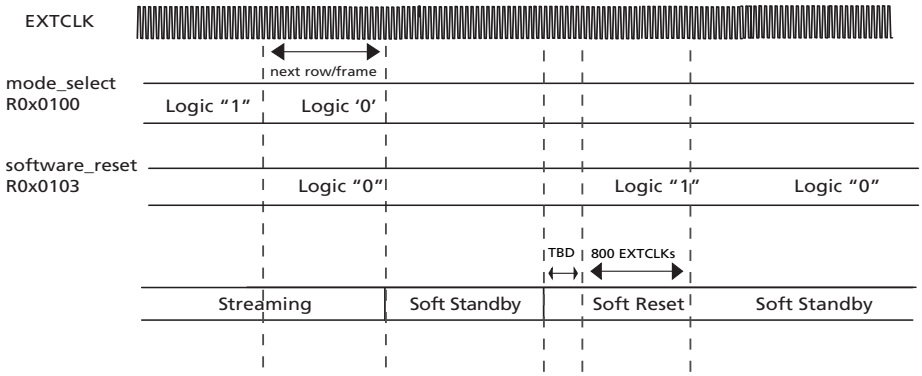
Soft Standby

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.

Soft Reset

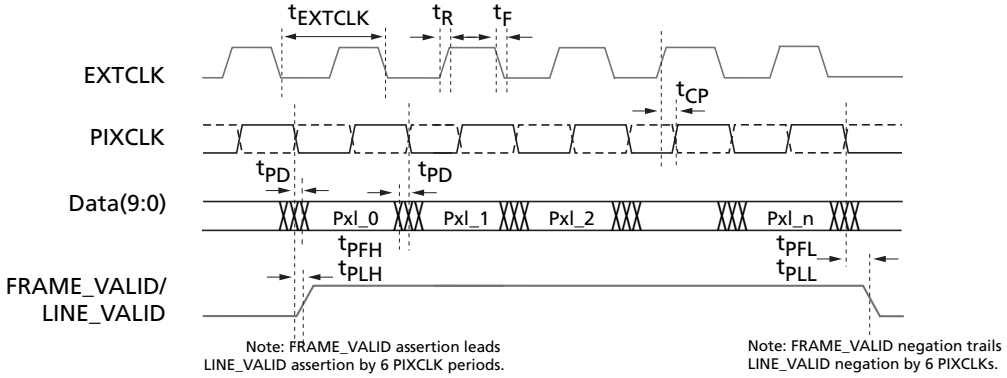
1. Follow the soft standby sequence list above.
2. Set software_reset = 1 (R0x0103) to start the internal initialization sequence.
3. After 800 EXTCLKs, the internal initialization sequence is completed and the current state returns to soft standby automatically. All registers, including software_reset, returns to their default values.

Figure 45: Soft Standby and Soft Reset



Electrical Specifications

Figure 46: Default Data Output Timing Diagram



EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 26 on page 104. The EXTCLK input supports either an AC-coupled sine-wave input clock or a DC-coupled square-wave input clock.

Table 26: Electrical Characteristics (EXTCLK)
V_{DD} = 1.8V; V_{DD_IO} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; Dark lighting conditions; Ambient temperature

Definition	Condition	Symbol	Min	Typ	Max	Unit
Input clock frequency		f_{EXTCLK}	6	16	27	MHz
Input clock period		t_{EXTCLK}	166	62.5	37	ns
Input clock amplitude (AC coupled sine wave)		V_{IN_AC}	0.5	–	–	V(p-p)
Input clock duty cycle			45	50	55	%
Input clock jitter		t_{JITTER}	–	–	300	ps
PLL VCO lock time		t_{LOCK}	–	800	–	EXTCLK cycles
Input pad capacitance			–	2.5	–	pF
Input HIGH leakage current	$V_{IN} = V_{DD\_IO}$	I_{IH}	–	2.5	–	µA
Input LOW leakage current	$V_{IN} = D_{GND}$	I_{IL}	–	2.5	–	µA
Input HIGH voltage (DC coupled)	At specified I_{IH}	V_{IH}	TBD	TBD	TBD	V
Input LOW voltage (DC coupled)	At specified I_{IL}	V_{IL}	TBD	TBD	TBD	V

Parallel Pixel Data Interface

The electrical characteristics of the parallel pixel data interface (FRAME_VALID, LINE_VALID, DOUT[9:0], PIXCLK, SHUTTER, and FLASH outputs) are shown in Table 27.

Table 27: Electrical Characteristics (Parallel Pixel Data Interface)
VDD = 1.8V; VDD_IO = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Dark lighting conditions; TA = 25°C, CLOAD = 30pF

Definition	Condition			Symbol	Min	Typ	Max	Units	Note
Output HIGH voltage	At specified IOH			VOH	TBD	TBD	TBD	V	
Output LOW voltage	At specified IOL			VOL	TBD	TBD	TBD	V	
Output HIGH current	Magnitude, at specified VOH			IOH	TBD	TBD	TBD	mA	
Output LOW current	Magnitude, at specified VOL			IOH	TBD	TBD	TBD	mA	
Tri-state output leakage current	VIN = VDD_IO or GND			IOZ	TBD	TBD	TBD	µA	
Output pin slew	Programmable Slew = 7	VDD_IO = 2.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
		VDD_IO = 1.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
	Programmable Slew = 4	VDD_IO = 2.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
		VDD_IO = 1.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
	Programmable Slew = 1	VDD_IO = 2.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
		VDD_IO = 1.8V	CLOAD = 30pF ±3pF	SR	TBD	TBD	TBD	V/ns	
			CLOAD = 15pF ±3pF	SR	TBD	TBD	TBD	V/ns	
PIXCLK frequency	default			^{f}PIXCLK	TBD	TBD	TBD	MHz	
EXTCLK to PIXCLK propagation delay	default			^{t}CP	TBD	TBD	TBD	ns	1
PIXCLK to data valid	default			^{t}PD	TBD	TBD	TBD	ns	1
PIXCLK to FRAME_VALID HIGH	default			^{t}PFH	TBD	TBD	TBD	ns	1

Table 27: Electrical Characteristics (Parallel Pixel Data Interface) (continued)
VDD = 1.8V; VDD_IO = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V;
Dark lighting conditions; TA = 25°C, CLOAD = 30pF

Definition	Condition			Symbol	Min	Typ	Max	Units	Note
PIXCLK to LINE_VALID HIGH	default			^{t}PLH	TBD	TBD	TBD	ns	1
PIXCLK to FRAME_VALID LOW	default			^{t}PFL	TBD	TBD	TBD	ns	1
PIXCLK to LINE_VALID LOW	default			^{t}PLL	TBD	TBD	TBD	ns	1

Note: Valid for CLOAD < 30pF on PIXCLK, DOUT, LINE_VALID, and FRAME_VALID pads. Loads must be matched as closely as possible.

Two-Wire Serial Register Interface

The electrical characteristics of the two-wire serial register interface (SCLK, SDATA) are shown in Table 28. The SCLK and SDATA signals feature fail-safe input protection, Schmitt trigger input, and suppression of input pulses of less than 50ns duration.

Table 28: Two-Wire Serial Register Interface Electrical Characteristics
VDD = 1.8V; VDD_IO = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V;
Dark lighting conditions; TA = 25°C; CLOAD 30pF

Definition	Condition	Symbol	Min	Typ	Max	Unit
Input HIGH voltage		VIH	0.7 x VDD	–	VDD + 0.5	V
Input LOW voltage		VIL	-0.5	–	0.3 x VDD	V
Input leakage current	No pull-up resistor; VIN = VDD_IO or DGND	IIN	–	10	–	µA
Output HIGH voltage	At specified IOH	VOH	–	VDD	–	V
Output LOW voltage	At specified IOL	VOL	0.16	–	0.35	V
Output HIGH current	At specified VOH	IOH	8.9	–	22.3	mA
Output LOW current	At specified VOL	IOL	8.9	–	18.5	mA
Tri-state output leakage current		IOZ	–	–	1	µA
Input pad capacitance		CIN	–	–	6	pF
Load capacitance		CLOAD	–	–	N/A	pF

Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLK_P, CLK_N, DATA_P, and DATA_N) are shown in Table 29.

In order to operate the serial pixel data interface within the electrical limits of the CCP2 specification, VDD_IO (I/O digital voltage) is restricted to operate in the range 1.7–1.9V.

Table 29: Electrical Characteristics (Serial Pixel Data Interface)
VDD = 1.8V; VDD_IO = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Dark lighting conditions; Ambient temperature

Definition	Symbol	Min	Typ	Max	Unit
Operating frequency		6	16	27	MHz
Fixed common mode voltage	VCMF	0.8	0.9	1.0	V
Differential voltage swing	VOD	100	155	200	mV
Drive current range		0.83	1.5	2	mA
Drive current variation		–	–	15	%
Output impedance		40	58	140	Ω
Output impedance mismatch		–	4	10	%
Clock duty cycle @ 416 MHz		45	50	55	%
Rise time (20–80%)	VOD	300	330	400	ps
Fall time (20–80%)	VOD	280	300	470	ps
Differential skew		–	300	500	ps
Channel-to-channel slew		–	–	200	ps
Maximum data rate Data/strobe mode Data/clock mode		–	–	 640 280	Mb/s
Power supply rejection ratio (PSRR) 0–100 MHz		30	–	–	dB
Power supply rejection ratio (PSRR) 100–1,000 MHz		10	–	–	dB

Control Interface

The electrical characteristics of the control interface (RESET_BAR, TEST, GPI0, and GPI1) are shown in Table 30.

Table 30: Electrical Characteristics (Control Interface)
VDD = 1.8V; VDD_IO = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Dark lighting conditions; Ambient temperature; CLOAD 15–30pF

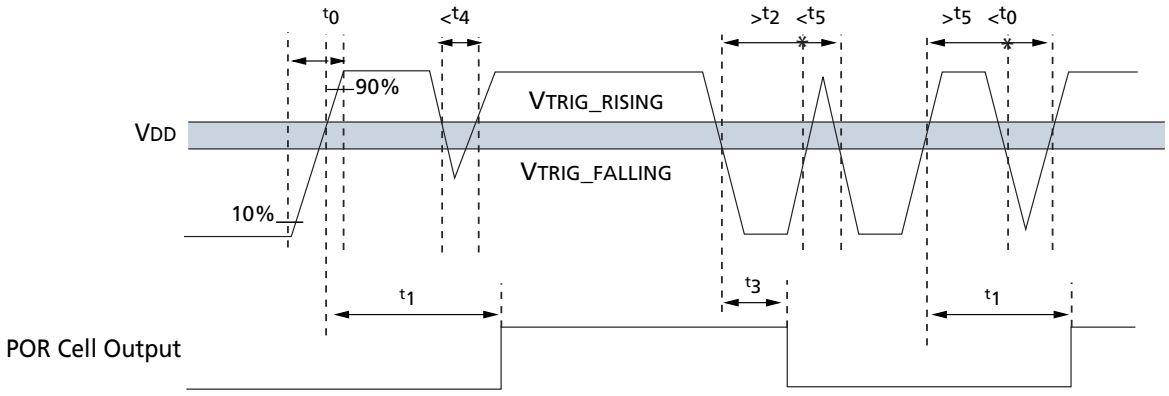
Definition	Condition	Symbol	Min	Typ	Max	Unit
Input HIGH voltage		VIH	0.7 x VDD	–	VDD + 0.5	V
Input LOW voltage		VIL	-0.3	–	0.3 x VDD	V
Input leakage current	No pull-up resistor; VIN = VDD_IO or DGND	IIN	–	<10	–	µA
Input pad capacitance		CIN	–	6.5	–	pF

Power-On Reset

Table 31: Power-On Reset Characteristics

Definition	Condition	Symbol	Min	Typ	Max	Unit
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	–	
Minimum VDD spike width below VTRIG_FALLING; considered to be a reset when POR cell output is LOW		$^{t}4$	–	0.5	–	µs
Minimum VDD spike width above VTRIG_RISING; considered to be a stable supply when POR cell output is LOW	While the POR cell output is LOW, all VDD spikes above VTRIG_RISING less than $^{t}5$ must be ignored	$^{t}5$	–	1	–	ns
VDD rising trigger voltage		VTRIG_RISING	1.12	–	1.55	V
VDD falling trigger voltage		VTRIG_FALLING	1.0	–	1.45	V

Figure 47: Internal Power-On Reset



Operating Voltages

V_{AA} and V_{AA_PIX} must be at the same potential for correct operation of the MT9T014.

Table 32: DC Electrical Definitions and Characteristics

V_{DD} = 1.8V; V_{DD_IO} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; Dark lighting conditions; C_{LOAD} 15–30pF

Definition	Condition	Symbol	Min	Typ	Max	Unit
Core digital voltage		V_{DD}	1.7	1.8	1.9	V
I/O digital voltage	Parallel pixel data interface	V_{DD_IO}	1.7	1.8	1.9	V
I/O digital voltage	Serial pixel (CCP2) data interface	V_{DD_IO}	TBD	TBD	TBD	V
Analog voltage		V_{AA}	2.4	2.8	3.1	V
Pixel supply voltage		V_{AA_PIX}	2.4	2.8	3.1	V
PLL supply voltage		V_{DD_PLL}	2.4	2.8	3.1	V
Digital operating current	Streaming, full resolution	I_{DD1}	–	TBD	TBD	mA
I/O digital operating current	Streaming, full resolution	I_{DDQ}	–	TBD	TBD	mA
Analog operating current	Streaming, full resolution	I_{AA}	–	TBD	TBD	mA
Pixel supply current	Streaming, full resolution	I_{AA_PIX}	–	TBD	TBD	mA
PLL supply current	Streaming, full resolution	I_{DD_PLL}	–	5	7	mA
Hard standby	Analog		0	–	1	µA
	Digital		0	–	9	µA
Soft standby (clock off)	Analog		0	–	1	µA
	Digital		0	–	49	µA
Soft standby (clock on (6 MHz))	Analog		0	–	5	µA
	Digital		0	–	45	µA

Absolute Maximum Ratings

Caution Stresses greater than those listed may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect reliability.

Table 33: Absolute Maximum Values

Definition	Condition	Symbol	Min	Typ	Max	Unit
Core digital voltage		V_{DD_MAX}	-0.3	–	1.9	V
I/O digital voltage		$V_{DD\_IO\_MAX}$	-0.3	–	3.1	V
Analog voltage		V_{AA_MAX}	-0.3	–	3.1	V
Pixel supply voltage		$V_{AA\_PIX\_MAX}$	-0.3	–	3.1	V
PLL supply voltage		$V_{DD\_PLL\_MAX}$	-0.3	–	3.1	V
Input HIGH voltage		V_{IH_MAX}	-0.3	–	V_{DD} + 0.5	V
Input LOW voltage		V_{IL_MAX}	0.7 x V_{DD}	–	0.3 x V_{DD}	V
Digital operating current	Worst case current	I_{DD_MAX}	–	–	80	mA
I/O digital operating current	Worst case current	I_{DDQ_MAX}	–	–	15	mA
Analog operating current	Worst case current	I_{AA_MAX}	–	–	70	mA
Pixel supply current	Worst case current	$I_{AA\_PIX\_MAX}$	–	–	3	mA
PLL supply current	Worst case current	$I_{DD\_PLL\_MAX}$	–	–	8	mA
Operating temperature	Measure at junction	T_{OP}	-30	–	70	°C
Storage temperature		T_{STG}	-40	–	125	°C

Notes: 1. This is a stress rating only, and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.

SMIA Specification Reference

The part itself and this documentation is based on the following SMIA reference documents:

–Functional Specification:

SMIA 1.0 Part 1: Functional Specification (Version 1.0 dated 30-June-2004)

SMIA 1.0 Part 1: Functional Specification ECR0001 (Version 1.0 dated 11-Feb-2005)

–Electrical Specification

SMIA 1.0 Part 2: CCP2 Specification (Version 1.0 dated 30-June-2004)

SMIA 1.0 Part 2: CCP2 Specification ECR0002 (Version 1.0 dated 11-Feb-2005)

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. A 05/15/2007

- Initial release.