1/5-Inch 1.3Mp CMOS Digital Image Sensor

MT9M019 Data Sheet

For the latest MT9M019 data sheet, refer to Aptina's Web site at www.aptina.com

Features

- Low dark current
- Simple two-wire serial interface
- Auto black level calibration
- 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
- SMIA-compatible
- Data interface: CCP2-compliant sub-low-voltage differential signalling (sub-LVDS)
- On-chip phase-locked loop (PLL) oscillator
- Bayer pattern downsize scaler
- Superior low-light performance

Applications

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

General Description

The Aptina® MT9M019 is a 1/5-inch SXGA-format CMOS active-pixel digital image sensor with a pixel array of 1280H x 1024V (1288H x 1032V including border pixels). It incorporates sophisticated on-chip camera functions such as windowing, mirroring, column and row skip modes, and snapshot mode. It is programmable through a simple two-wire serial interface and has very low power consumption.

The MT9M019 digital image sensor features Aptina'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.

Table 1: Key Performance Parameters

Parameter		Value
Optical format		1/5-inch SXGA (5:4)
Active imager size		2.83mm(H) x 2.27(V) 3.63mm diagonal (calculated from 1288 x 1032)
Active pixels		1288H x 1032V
Pixel size		2.2 x 2.2µm
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS)
Maximum data rate/ master clock		64 Mp/s at 64 MHz system clock
Frame rate	SXGA (1280 x 1024)	Programmable up to 30 fps
	VGA (640 x 480)	Programmable up to 60 fps
ADC resolution		10-bit, on-chip (61dB)
Responsivity		1.14V/lux-sec
Dynamic range		67.27dB
SNR_{MAX}		36.4dB
Supply voltage	Analog	2.4–3.1V (2.80V nominal)
	Digital	1.7–1.9V (1.80V nominal)
Power consumption		190mW
Operating temperature		–30°C to +70°C
Packaging		Die
Chief Ray Angle		24.77° at 85% image height

When operated in its default mode, the sensor generates a SXGA image at 30 frames per second (fps). An on-chip analog-to-digital converter (ADC) generates a 10-bit value for each pixel.

Ordering Information

Table 2: Available Part Numbers

Part Number	Description
MT9M019D00STCC14BC1	Bare die

Table of Contents

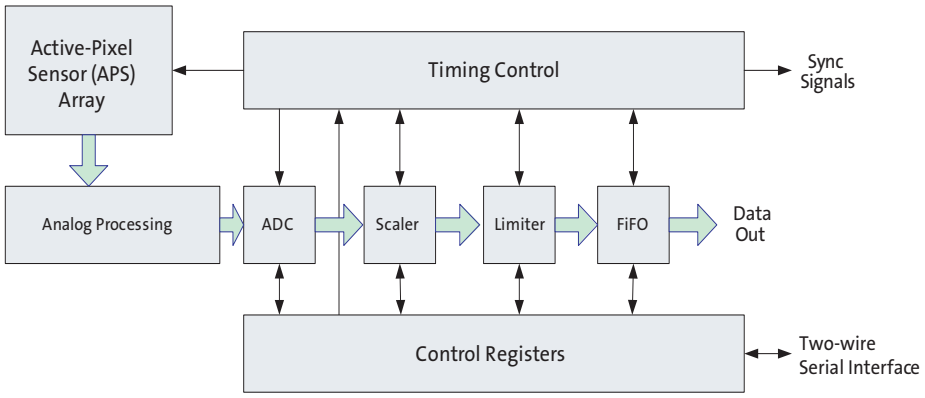
List of Figures

List of Tables

Functional Overview

The MT9M019 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 27 MHz. The maximum pixel rate is 64 Mp/s, corresponding to a system 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 1.3Mp 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 two logical parts:

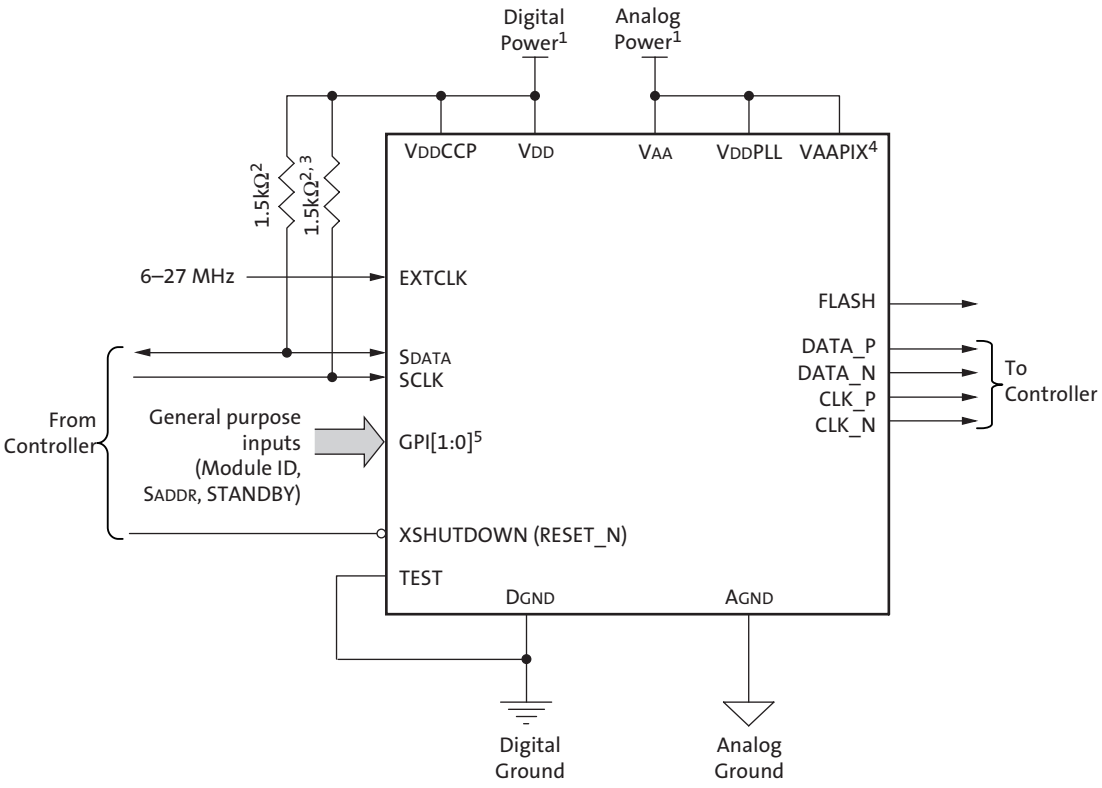
- A sensor core that provides array control and data path corrections. The output of the sensor core is a serial CCP2-compliant pixel data stream.
- Additional functionality is required to support the SMIA standard. This includes a horizontal and vertical image scaler, a limiter, a data compressor, an output FIFO, and a serializer.

A flash output strobe is provided to allow an external xenon or LED light source to synchronize with the sensor exposure time.

Operating Modes

By default, the MT9M019 powers up as a SMIA-compatible sensor with the serial pixel data interface enabled. A typical configuration in this mode is shown in Figure 2.

Figure 2: Typical Configuration: Serial Pixel Data Interface



Note:
1. All power supplies should be adequately decoupled.
2. Resistor value 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. The GPI pads can serve multiple features that can be reconfigured. The function will vary by application.

Signal Descriptions

Table 3 provides signal descriptions for MT9M019 die. For pad location and aperture information, refer to the MT9M019 die data sheet.

Table 3: Signal Descriptions

Pad Name	Pad Type	Description
EXTCLK	Input	Master clock input. PLL input clock. 6–27 MHz.
RESET_N (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 and SADDR functions.
TEST	Input	Enable manufacturing test modes. Wire to digital GND for functional operation.
SDATA	I/O	Serial data for reads from and writes to control and status registers.
DATA_P	Output	Differential CCP2 (sub-LVDS) serial data (positive).
DATA_N	Output	Differential CCP2 (sub-LVDS) serial data (negative).
CLK_P	Output	Differential CCP2 (sub-LVDS) serial clock/strobe (positive).
CLK_N	Output	Differential CCP2 (sub-LVDS) serial clock/strobe (negative).
FLASH	Output	Flash output. Synchronization pulse for external light source.
VAA1, VAA2, VAA3, VAA4	Supply	Analog power supply.
VAAPIX1, VAAPIX2	Supply	Analog power supply for the pixel array.
AGND1, AGND2, AGND3, AGND4	Supply	Analog ground.
VDD1, VDD2, VDDCCP	Supply	Digital power supply.
DGND1, DGND2, DGND3	Supply	Common ground for digital and I/O.
VDDPLL	Supply	PLL power supply.

Output Data Format

CCP2 Serial Pixel Data Interface

The MT9M019 serial pixel data interface implements data/clock and data/strobe signalling in accordance with the CCP2 specification. The RAW8 and RAW10 image data formats are supported.

Two-Wire Serial Register Interface

The two-wire serial interface bus enables read/write access to control and status registers within the MT9M019. 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 interface protocol uses a master/slave model in which a master controls one or more slave devices. The sensor acts as a slave device. The master generates a clock (SCLK) that is an input to the sensor and used to synchronize transfers. Data is transferred between the master and the slave on a bidirectional signal (SDATA). SDATA is pulled up to VDD off-chip by a 1.5KΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The protocols described in the I^2C specification enable the slave device to drive SCLK LOW. However, the MT9M019 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, as follows:

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

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

Start Condition

A start condition is defined as a HIGH-to-LOW transition on SDATA while SCLK an 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 MT9M019 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 asserting the SADDR input signal through GPI inputs or register programmable.

An alternate slave address can be programmed through R0x30FC–D.

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 I^2C specification and is defined as part of the SMIA CCI.

Acknowledge Bit

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

No-Acknowledge Bit

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

Typical 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.

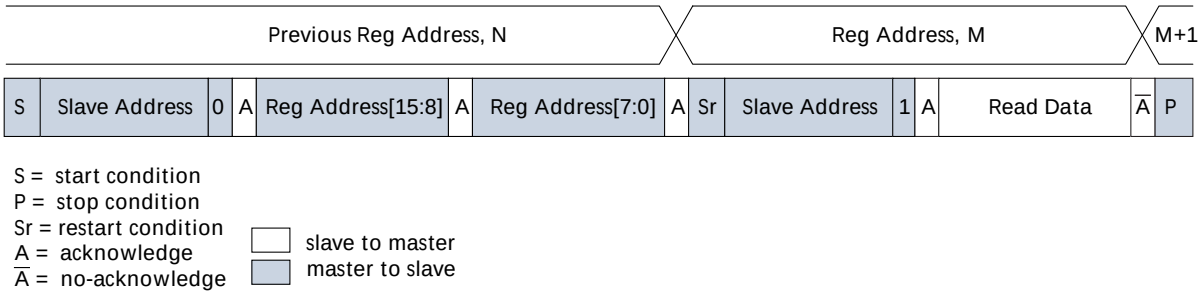
If the request was a write, the master then transfers the 16-bit register address to which the write should take place. This transfer takes place as two 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the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

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

Single READ from Random Location

This sequence (Figure 3 on page 11) 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 3 shows how the internal register address maintained by the MT9M019 is loaded and incremented as the sequence proceeds.

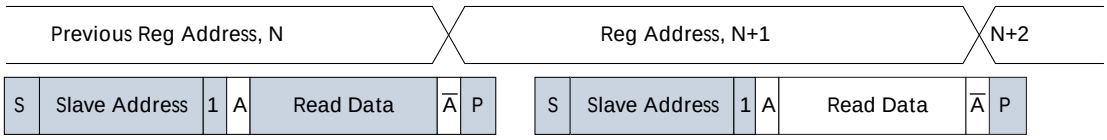
Figure 3: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 4) performs a read using the current value of the MT9M019 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.

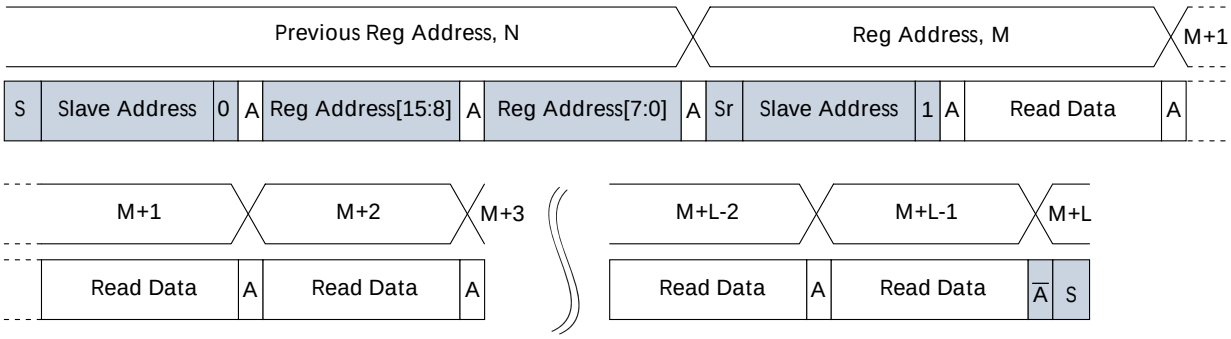
Figure 4: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 5) starts in the same way as the single READ from random location (Figure 3). 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.

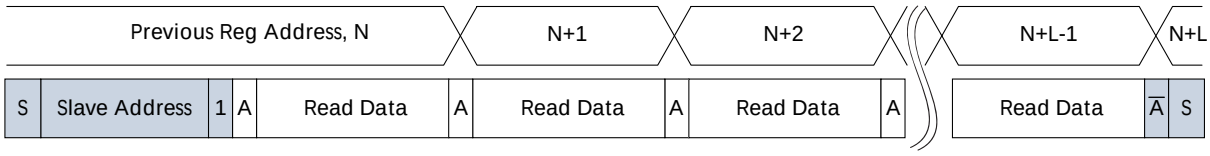
Figure 5: Sequential READ, Start from Random Location



Sequential READ, Start from Current Location

This sequence (Figure 6) starts in the same way as the single READ from current location (Figure 4 on page 11). 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.

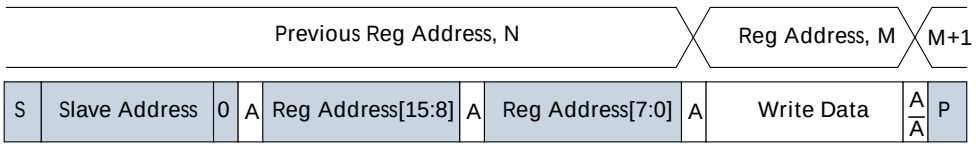
Figure 6: Sequential READ, Start from Current Location



Single WRITE to Random Location

This sequence (Figure 7) 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.

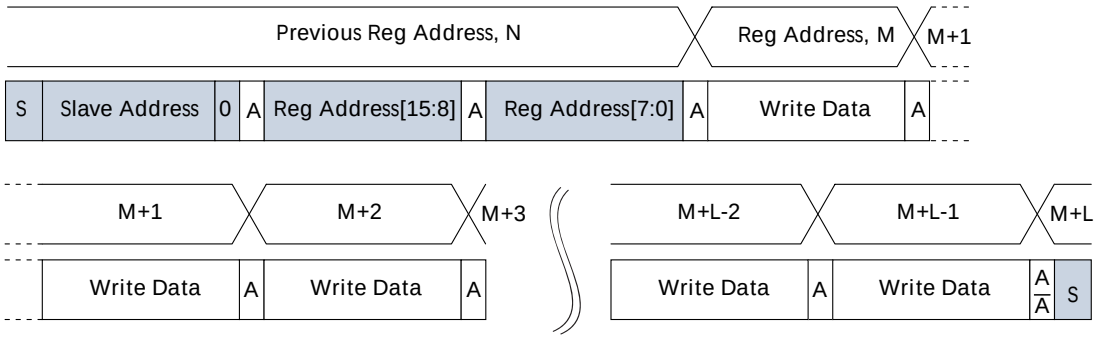
Figure 7: Single WRITE to Random Location



Sequential WRITE, Start at Random Location

This sequence (Figure 8) starts in the same way as the single WRITE to random location (Figure 7). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte WRITEs until "L" bytes have been written. The WRITE is terminated by the master generating a stop condition.

Figure 8: Sequential WRITE, Start at Random Location



Programming Restrictions

The SMIA specification imposes a number of programming restrictions. An implementation naturally imposes additional restrictions. Table 5 shows a list of programming rules that must be adhered to for correct operation of the MT9M019. It is recommended that these rules are encoded into the device driver stack, either implicitly or explicitly.

Table 4: Definitions for Programming Rules

Name	Definition
xskip	if (x_odd_inc == 1) xskip = 1 else xskip = 2
yskip	if (y_odd_inc == 1) yskip = 1 else yskip = 2

Table 5: 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 + 1)/ xskip) + min_line_blanking_pck		SMIA
frame_length_lines	((y_addr_end - y_addr_start + 1)/ 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+ 1)	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 + 1)	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
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

Table 5: Programming Rules (continued)

Parameter	Minimum Value	Maximum Value	Origin
scale_n	scaler_n_min	scaler_n_max	SMIA
x_output_size	256 (this is enforced in hardware: values lower than this are treated as 256)	1288	Minimum from SMIA FS[1] 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 "Subsampling" on page 29.

Note: 1. SMIA FS = SMIA Functional Specifications.

Output Size Restrictions

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

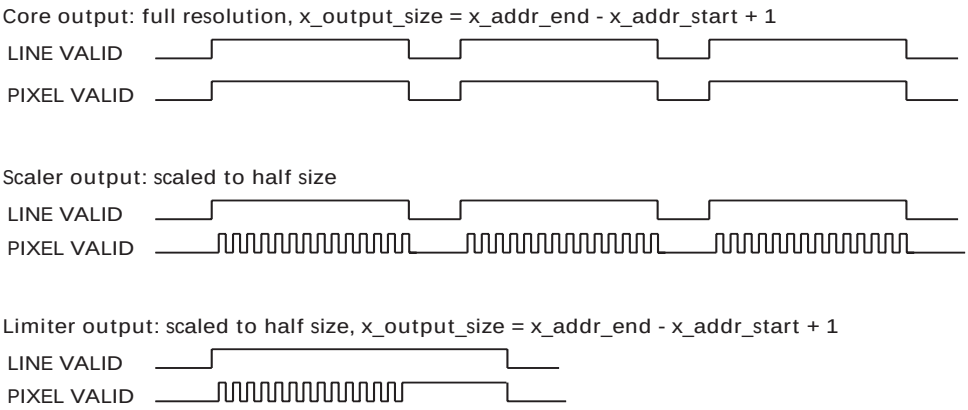
This restriction can be met by rounding up x_output_size to an appropriate multiple. Any extra pixels in the output image as a result of this rounding contain UNDEFINED pixel data but are guaranteed not to cause false synchronization on the CCP2 data stream.

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 MT9M019 will not operate properly 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 9.

Figure 9: Effect of Limiter on SMIA Data Path



In Figure 9, three different stages in the SMIA data path 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.

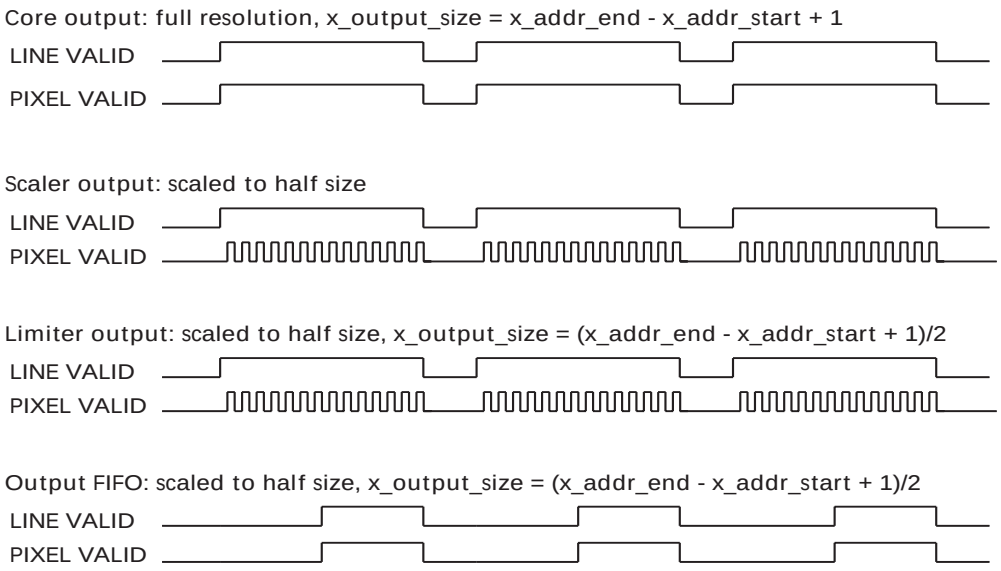
The second stage is the output of the scaler, when the scaler is set to reduce the image size by one-half in each dimension. The effect of the scaler is to combine groups of pixels. Therefore, the row time remains the same but only half the pixels out of the scaler are valid. This is signalled by transitions 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, as shown Figure 9 on page 16, the MT9M019 will cease to generate output frames.

A correct configuration is shown in Figure 10. This figure shows 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 9 on page 16 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 10: 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 *1,854 * 1,109 = 2,056,086* pixel periods. This value includes vertical and horizontal blanking times, so that the full-size image 1280 x 1026 (1,024 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 *2,056,086/64e6 = 32.1263* milliseconds, giving rise to a frame rate of 31.13 fps.

Each pixel is 10 bits, by default. This data rate requires the serial interface to transmit 2,056,086 * 10 bits in 32.1263 milliseconds. That is, the serial data rate must be 10X the pixel rate—640 Mb/s.

The 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 31 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 2,056,086/20.8e6 = 98.85msec, giving rise to a frame rate of 10.12 fps.

Note: The output pixel rate cannot be greater than 20.8 MHz in CCP Class 0 mode, but the internal pixel rate (vt domain) which controls the frame rate can be modified to run faster by changing the horizontal blanking.

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.

Caution **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 (R0x306E–F).**

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

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

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, an external pull-up resistor should be connected to this signal.

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

SADDR in an input-only signal and must always be driven to a valid logic level for correct operation. In most applications this input will be hardwired to logic "0" if a GPI is used. There is no dedicated SADDR pin.

This interface is described in detail in "Two-Wire Serial Register Interface" on page 9.

Default Power-up State

At power-up and after a hard or soft reset, the reset state of the MT9M019 is under SMIA operation and the CCP2 high-speed serial interface.

Serial Pixel Data Interface

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

- DATAP
- DATAN
- CLKP
- CLKN

The signal pairs are driven differentially using sub-LVDS switching levels. This interface conforms to the SMIA 1.0 CCP2 requirements and supports both data/clock signalling and data/strobe signalling.

The DATAP, DATAN, CLKP, and CLKN 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.

Configuration of the Pixel Data Interface

Fields in R0x301A–B are used to configure the operation of the pixel data interface.

System States

The system states of the MT9M019 are represented as a state diagram in Figure 11 and described in subsequent sections. The effect of RESET_N on the system state and the configuration of the PLL in the different states are shown in Table 6.

The sensor's operation is broken down into three separate states: Hardware standby, software standby, and streaming. The transition between these states might take a certain amount of clock cycles, as outlined in Table 6.

Figure 11: MT9M019 System States

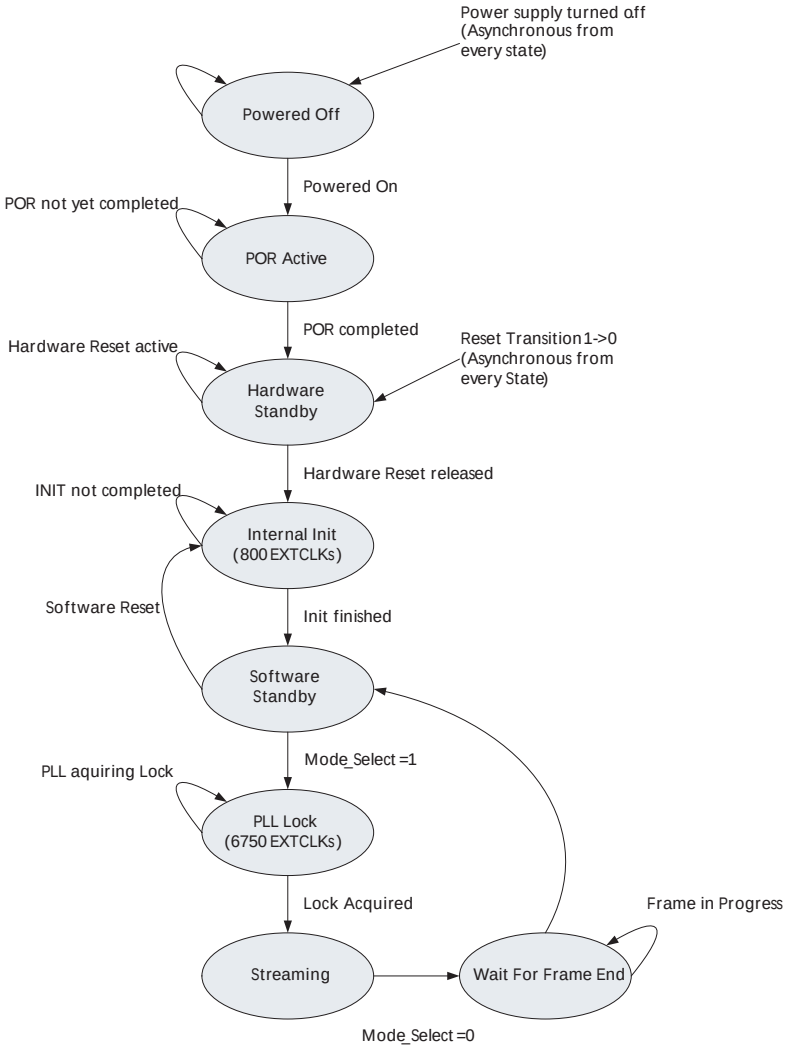


Table 6: PLL in System States

State	EXTCLKs	PLL
Powered off		VCO powered down
PLL lock	6750	VCO powering up and locking,
Streaming		VCO running, PLL output active
Wait for frame end		

Power-On Reset Sequence

When power is applied to the MT9M019, 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_N input.
2. A timeout of the internal power-on reset circuit.

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

The RESET_N signal is functionally equivalent to the SMIA-specified XSHUTDOWN signal.

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

While RESET_N is asserted (or the internal power-on reset circuit is active) the MT9M019 is in its lowest-powered, powered-up state; the internal PLL is disabled, the CCP2 serializer is disabled and internal clocks are gated off.

When the sensor leaves the hardware standby state, it performs an internal initialization sequence that takes 800 EXTCLK cycles. After this, it enters a low-power software standby state. While the initialization sequence is in progress, the MT9M019 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 so 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 (0x14 if R0x0000 is read).

When the sensor leaves software standby mode and enables the VCO, an internal delay will keep the PLL disconnected for 6,750 EXTCLKs so that the PLL can lock.

Soft Reset Sequence

The MT9M019 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 briefly enters the hardware standby state and then starts its internal initialization sequence. At this point, the behavior is exactly the same as for the power-on reset sequence.

Signal State During Reset

Table 7 on page 23 shows the state of the signal interface during hardware standby (RESET_N asserted) and the default state during software standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 7: Signal State During Reset

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Enabled. Must be driven to a valid logic level.	
RESET_N (XSHUTDOWN)	Input	Enabled. Must be driven to a valid logic level.	
SCL	Input	Enabled. Must be pulled up or driven to a valid logic level.	
SDA	I/O	Enabled as an input. Must be pulled up or driven to a valid logic level.	
FLASH	Output	High-Z.	Logic 0.
DATAP	Output	High-Z.	
DATAN	Output		
CLKP	Output		
CLKN	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 MT9M019 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 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

Other uses:

- SADDR pin (choose I^2C device address: R0x31FC–D[7:0] or R0x31FC–D[15:8])
- Module ID (the state of the two pins can be read back if different module versions tie the pins differently)
- Trigger (see the sections below)
- Standby functions (see the following sections)

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

Streaming/Standby Control

The MT9M019 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 8. Selection of a pin to use for the STANDBY function is described in "General Purpose Inputs" on page 23. The state diagram for transitions between soft standby and streaming states is shown in Figure 11 on page 21.

Table 8: 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

Clocking

The MT9M019 contains a phase-locked loop (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 register 0x306E–F[7] to 0 for profile 0 or to 1 for profile 1, 2.

Figure 12: MT9M019 SMIA Profile 1, 2 Clocking Structure

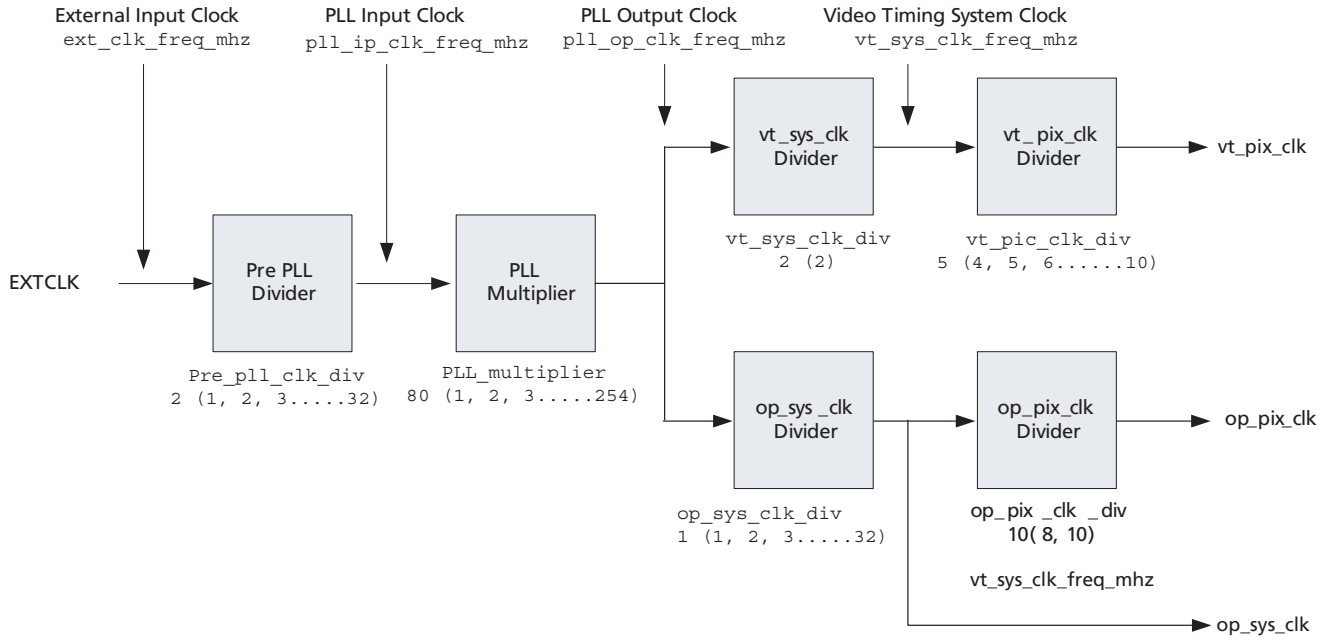


Figure 12 shows the different clocks and (in `courier` font) the names of the registers that contain or are used to control their values. The figure shows the default setting for each divider/multipler control register and the range of legal values for each divider/multiplier control register.

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.
- 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 are as shown in Table 9 and Table 10.

Table 9: Valid Divisor Combinations (10 bits per pixel)

op_sys_clk_div	vt_pix_clk_div
1	4, 5
2, 4, 6, 8	4, 5, 6, 7, 8, 9, 10
10	5, 6, 7, 8, 9, 10
12	6, 7, 8, 9, 10
14	7, 8, 9, 10
16	8, 9, 10
18, 20	9, 10
22	10

Table 10: Valid Divisor Combinations (8 bits per pixel)

op_sys_clk_div	vt_pix_clk_div
1	4
2	4, 5, 6, 7 8
4, 6, 8, 10	4, 5, 6, 7, 8, 9, 10
12	5, 6, 7, 8 9, 10
14, 16	6, 7, 8, 9, 10
18	7, 8, 9, 10
20	8, 9, 10
22, 24	9, 10
26	10

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.
- op_pix_clk is used to load parallel pixel data from the output FIFO 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 the ccp_data_format register (R0x0112–3).
- 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:

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

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

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

Figure 13: MT9M019 SMIA Profile 0 Clocking Structure

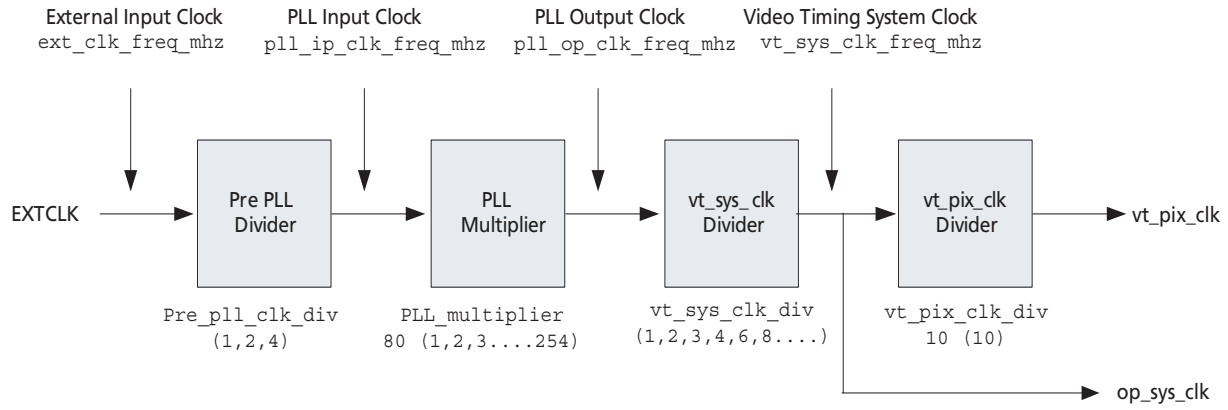


Figure 13 shows the different clocks and (in `courier` font) the names of the registers that contain or are used to control their values. Figure 13 shows the default setting for each divider/multipler control register and the range of legal values for each divider/multiplier control register.

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 MT9M019 is in the software 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 will delay entering streaming mode by 6,750 EXTCLKs so that the PLL can lock.

The effect of programming the PLL divisors while the MT9M019 is in the streaming state is UNDEFINED.

Influence of ccp_data_format

The ccp_data_format register (R0x0112–3) 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

The ccp2_signalling_mode register (R0x0111) 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 CCP2 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.

To disguise this behavior from the programmer, the actual pll multiplier is right-shifted by 1 bit relative to the programmed value when ccp2_signalling_mode selects data/strobe signalling.

Programming Example

This section provides one programming example which is the default settings. For this example, a table of register values is shown (for example, Table 11, Default Settings). The settings for the clock divisors show the "Programmed Value" and "Apparent Frequency." These values are consistent with MT9M019 SMIA Profile 0 Clocking Structure and the associated equations for the different clocks. The table also shows the "Effective Value" and "Actual Frequency." These values are implementation details that reflect the internal operation of the clocks; they are consistent with the descriptions given in “Influence of ccp_data_format” on page 27.

Example:

- 10 bits per-pixel data
- CCP2 Class 1/Class 2 signalling
- Highest possible frame rate at maximum resolution

To meet the requirement for the highest possible frame rate, vt_pix_clk should run at 64 MHz. For 10 bits per-pixel operation, op_sys_clk must run at 10x op_pix_clk. Therefore, op_sys_clk_div is set to 1 and op_pix_clk_div to 10, giving an overall divide-by-10 in the op clock domain. Since vt_sys_clk_div is fixed at 2, the same divide-by-10 is achieved in the vt clock domain by setting vt_pix_clk_div to 5. As a result, the PLL output frequency must be set to 64 * 10 = 640 MHz. There are various ways doing this, depending upon the frequency of EXTCLK.

The register settings for this example and the resulting clock frequencies are shown in Table 11. If the MT9M019 is programmed with these values (and all other registers are left at their default values), and is then put into streaming mode (mode_select=1) it will stream frames at full resolution (1,280 x 1,024 pixels) through its CCP2 interface at 31.1 fps.

Table 11: Default Settings

Register	Programmed Value	Effective Value	Clock	Apparent Frequency	Actual Frequency
ccp_data_format	0x0A0A	10 bits per pixel	–	–	–
ccp2_signalling_mode	1	Data/strobe signalling	–	––	–
–	–	–	EXTCLK	16 MHz	16 MHz
pre_pll_clk_div	2	2	pll_ip_clk	8 MHz	8 MHz
pll_multiplier	80	40	pll_op_clk	640 MHz	320 MHz
vt_sys_clk_div	2	1	vt_sys_clk	320 MHz	320 MHz
vt_pix_clk_div	5	5	vt_pix_clk	64 MHz	64 MHz
op_sys_clk_div	1	1	op_sys_clk	640 MHz	320 MHz
op_pix_clk_div	10	5	op_pix_clk	64 MHz	64 MHz

Clock Control

The MT9M019 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 MT9M019 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 two-wire serial interface continues to respond to read and write requests.

Features

Image Acquisition Mode

The MT9M019 supports electronic rolling shutter (ERS) mode. When the MT9M019 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 MT9M019 switches cleanly from the old integration time to the new while only generating frames with uniform integration.

Window Control

The sequencing of the pixel array is controlled by the x_addr_start, y_addr_start, x_addr_end, y_addr_end registers. When the SMIA data path is enabled, 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 1280H x 1024V image. A border of up to 4 pixels on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end and y_addr_end registers and then adjusting the x_output_size and y_output_size registers accordingly.

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. 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 (see R0x3024).

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. 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 (see R0x3024).

Subsampling

The MT9M019 supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the sensor and thereby allows the frame rate to be increased. Subsampling is enabled by setting x_odd_inc = 3 and/or y_odd_inc = 3.

This will skip two rows/columns during readout and is equivalent to the skip2 readout mode provided by earlier Aptina sensors. The effect of the different subsampling settings on the pixel array readout is shown in Figures 14 through 17.

Figure 14: Pixel Readout (no subsampling)

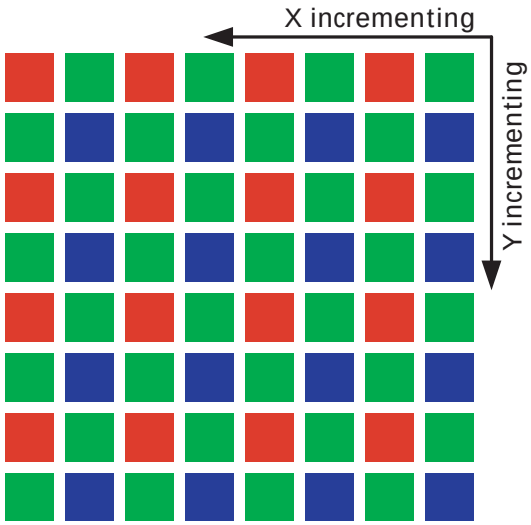


Figure 15: Pixel Readout (x_odd_inc = 3, y_odd_inc = 1)

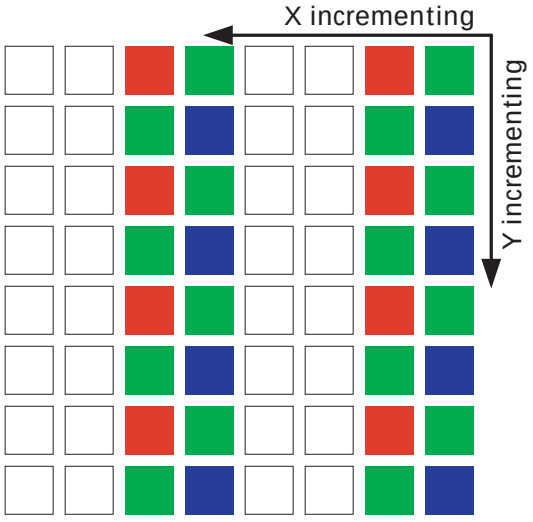


Figure 16: Pixel Readout (x_odd_inc = 1, y_odd_inc = 3)

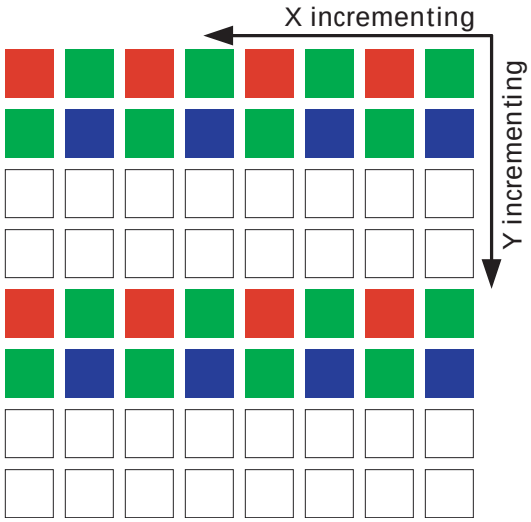
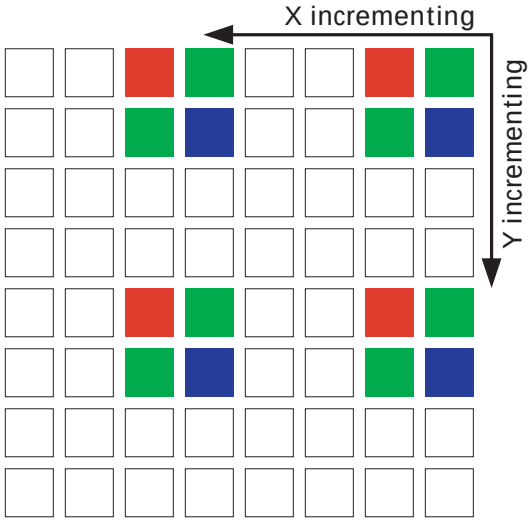


Figure 17: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)



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 and y_addr_end settings. The values for these registers are required to correspond with rows/columns that form part of the subsampling sequence. The adjustment should be made in accordance with the following rule:

remainder = (addr_end - addr_start + 1) AND 0x0002;

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

Table 12 shows the row address sequencing for normal and subsampled (with y_odd_inc = 3) readout. The same sequencing applies to column addresses for subsampled readout. There are two possible subsampling sequences (because the subsampling sequence only read half of the rows and columns) depending upon the alignment of the start address.

Table 12: Row Address Sequencing

Normal	Subsampled	Subsampled
0	0	–
1	1	–
2	–	2
3	–	3
4	4	–
5	5	–
6	–	6
7	–	7

Frame Rate Control

The formula for calculating the frame rate of the MT9M019 is shown below:

$$\text{line\_length\_pck} = \left(\frac{\text{x\_addr\_end} - \text{x\_addr\_start} + 1}{\text{subsampling factor}} + \text{min\_line\_blanking\_pck}\right) \quad \text{(EQ 7)}$$

$$\text{frame\_length\_lines} = \left(\frac{\text{y\_addr\_end} - \text{y\_addr\_start} + 1}{\text{subsampling factor}} + \text{min\_frame\_blanking\_lines}\right) \quad \text{(EQ 8)}$$

$$\text{frame rate [FPS]} = \frac{(\text{vt\_pixel\_clock\_mhz} * 1\times10^{6})}{(\text{line\_length\_pck} * \text{frame\_length\_lines})} \quad \text{(EQ 9)}$$

Frame Rates at Common Image Sizes

Table 13 shows the maximum frame rates that can be achieved at various common image sizes with a data rate limit of 640 Mb/s (CCP2). The frame rates are shown with subsampling enabled (x_odd_inc = 3, y_odd_inc = 3). The frame rates assume a pixel rate of 64 Mp/s (vt_pix_clk = 64 MHz) and the minimum line blanking of 574 pixels/ minimum frame blanking of 85 lines.

Table 13: Frame Rates

Programmed Image Size	Resulting Image Size		Resulting Frame Rate	
	Subsampling Disabled	Subsampling Enabled	Subsampling Disabled	Subsampling Enabled
1280 x 1024	1280 x 1024	640 x 512	31.12	88.30
1024 x 768	1024 x 768	512 x 384	46.95	125.65
640 x 512	640 x 512	320 x 256	88.3	209.93
352 x 288	352 x 288	176 x 144	185.29	372.63

Valid Data Signal Options

Integration Time

The integration (exposure) time of the MT9M019 is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

$$\text{fine\_integration\_time\_min} <= \text{fine\_integration\_time} <= (\text{line\_length\_pck-fine\_integration\_time\_max\_margin}) \quad \text{(EQ 10)}$$

The limits for the coarse integration time are defined by:

$$\text{coarse\_integration\_time\_min} <= \text{coarse\_integration\_time} \quad \text{(EQ 11)}$$

If *coarse_integration_time>(frame_length_lines-coarse_integration_time_max_margin)* then the frame rate will be reduced.

The actual integration time is given by:

$$\text{integration\_time [sec]} = \frac{(\text{coarse\_integration\_time}*\text{line\_length\_pck})"" + \text{fine\_integration\_time})}{(\text{vt\_pix\_clk\_freq\_mhz} \times 1 \times 10^6)} \quad \text{(EQ 12)}$$

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

$$\text{Maximum integration time [sec]} = \frac{(((0x4E4-1)*0x8EC)+0x7EC)}{(64\ \text{MHz} \times 1 \times 10^6)} = 44.721\ \text{ms} \quad \text{(EQ 13)}$$

Setting an integration time that is greater than the frame time therefore increases the frame time beyond frame_length_lines to make longer exposure times available. Note that the frame length is now set by *(coarse_integration_time - coarse_integration_time_max_margin)* rather than the *frame_length_lines*. This should be taken into account for all other formulae used in this document. 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.

Flash Control

The MT9M019 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 18, 19, and 20. The flash and flash_count registers allow the timing of the flash to be changed. The flash can be programmed to fire only once, be 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 20. Read-only bit flash[14] is set during frames that are correctly integrated; the state of this bit is shown in Figures 18, 19 and Figure 20 on page 35.

Figure 18: Xenon Flash Enabled

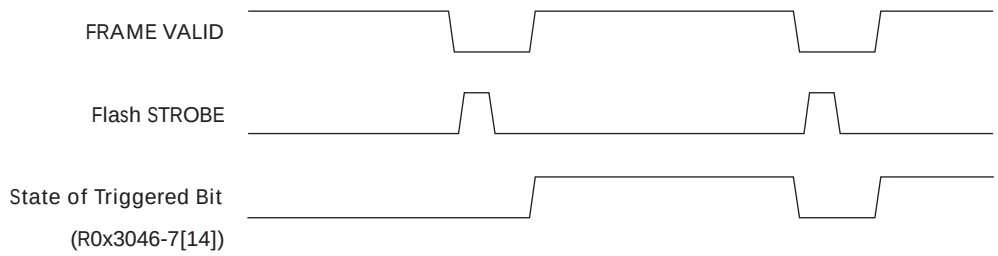
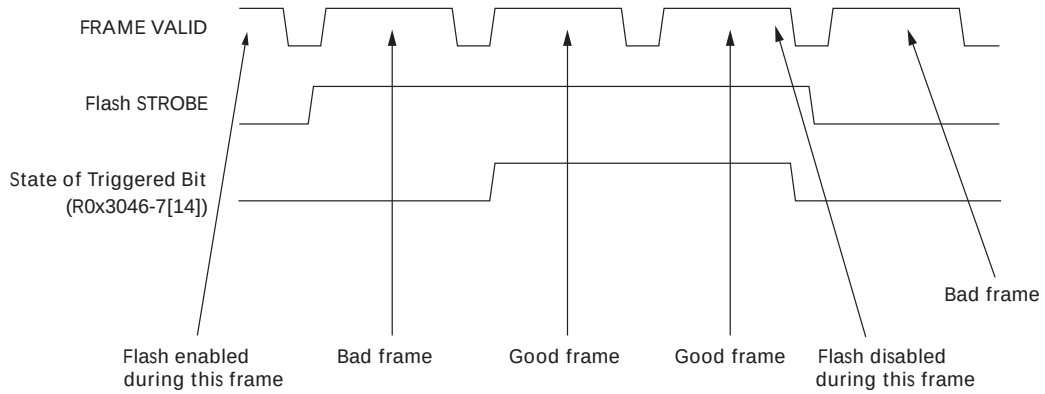
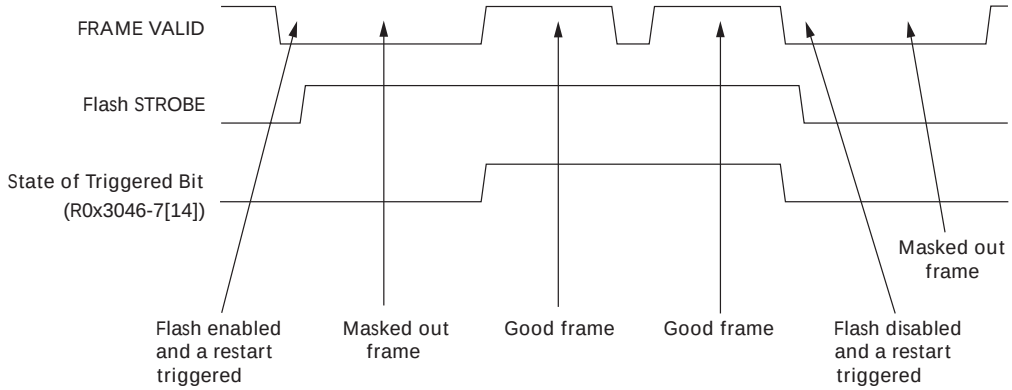


Figure 19: LED Flash Enabled



Note: Integration time = number of rows in a frame.

Figure 20: LED Flash Enabled Following Forced Restart



Low Power Mode

The sensor supports a low-power mode by programming a sequence of registers:
Set Read Mode register bit 9 (R0x3040–1[9]) = 1
Set Fine Correction register (R0x3010–1) = 0x52

Programming this sequence will change the min_line_pck. It is important that the *line_length_pck* formula in Table 5 on page 14 is upheld at all times.

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

Setting read _mode[9] bit will result in the following:

- Double the value of pc_speed[2:0] internally. This means halving the internal pixel clock frequency.
- Replace analog DAC settings with low power settings.

Please be aware of the following fine_integration_time limits:
fine_integration_time_min(R0x1008–9) = 0xF0
fine_integration_time_max_margin (R0x100A–B) = 0x7A

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

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

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

Any limitations related to changing the internal pixel clock speed will also apply to low power mode since it automatically changes the pixel clock speed. SMIA limiter registers therefore need to be reprogrammed by the host to match the new internal pixel clock frequency.

Analog Gain

The MT9M019 provides two mechanisms for setting the analog gain. The first uses the SMIA gain model. The second uses the traditional Aptina 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 MT9M019 provides per-color gain control. However, the MT9M019 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 green1/greenR gain register.

The read/write gain_mode register required by SMIA has no defined function in the SMIA specification. In the MT9M019 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:

$$\text{gain} = \frac{\text{analogue\_gain\_m0} \times \text{analogue\_gain\_code}}{\text{analogue\_gain\_c1}} = \frac{\text{analogue\_gain\_code\_<color>}}{8} \quad \text{(EQ 14)}$$

Aptina Gain Model

The Aptina 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 provide a 7-bit gain stage and a 2X gain stage. As a result, the step size varies depending upon whether the 2X gain stage is enabled. The analog gain is given by:

$$\text{gain} = (\text{<color>\_gain[7]} + 1) \times \frac{\text{<color>\_gain[6:0]}}{16} \quad \text{(EQ 15)}$$

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 Aptina 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 Aptina 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 Aptina 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, to provide the mapping with the lowest noise.

When the Aptina 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 Aptina 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 MT9M019 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 14.

Table 14: 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

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 MT9M019 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

Effect of Data Path Processing on Test Patterns

Test patterns 1–3 are introduced early in the pixel data path. As a result, they are affected by pixel processing that occurs within the data path. This includes:

- Noise cancellation
- Black pedestal adjustment
- Dark current compensation

These effects can be eliminated by the following register settings:

- R0x3044–5[10] = 0
- R0x30CA–B[0] = 1
- R0x30CC–D = 0
- R0x30CE–F = 0
- R0x30D0–1 = 0
- R0x30D2–3 = 0
- R0x30C0–1[0] = 0
- R0x30C2–3 = 0
- R0x30C4–5 = 0
- R0x30C6–7 = 0

- R0x30C8–9 = 0
- R0x301A–B[3] = 0 (enable writes to data pedestal)
- R0x301E–F = 0x0000 (set data pedestal to “0”)
- R0x31E0[0] = 0

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).

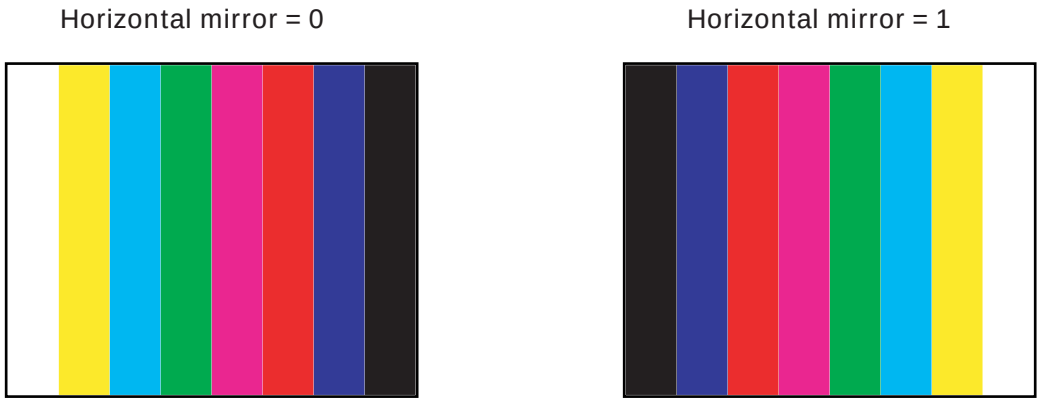
100 Percent Color Bars Test Pattern

In this test pattern, shown in Figure 21, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, black). Each bar is 160 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 * 160 = 1280 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 160 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 on the left side of the output image. Any pattern repeat occurs on 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 and scaling of this test pattern is UNDEFINED.

Figure 21: 100 Percent Color Bars Test Pattern



Fade-to-Gray Color Bars Test Pattern

In this test pattern, shown in Figure 22 on page 41, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, black). Each bar is 160 pixels wide and occupies 1,024 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 * 200 = 1600 pixels and vertically after 1,024 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.

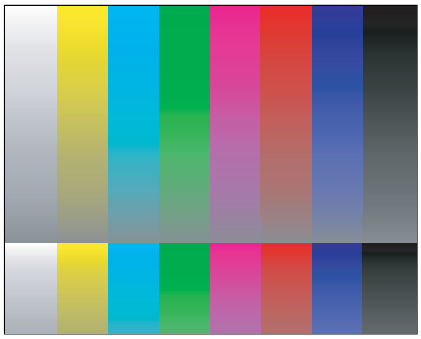
The image size is set by x_addr_start, x_addr_end, y_addr_start, y_addr_end and may be affected by the setting of x_output_size, y_output_size. The color-bar pattern starts at the column identified by x_addr_start. The number of colors that are visible in the output is dependent upon x_addr_end - x_addr_start and the setting of x_output_size.

The 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 on the left side of the output image. Any pattern repeat occurs on the right side of the output image regardless of the setting of horizontal_mirror.

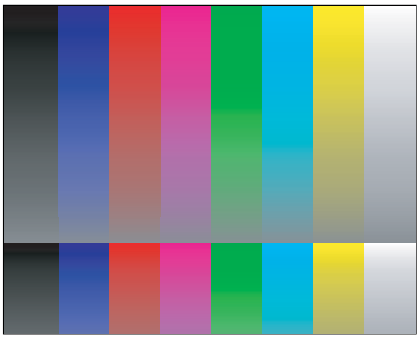
The effect of subsampling and scaling of this test pattern is UNDEFINED.

Figure 22: Fade-to-Gray Color Bars Test Pattern

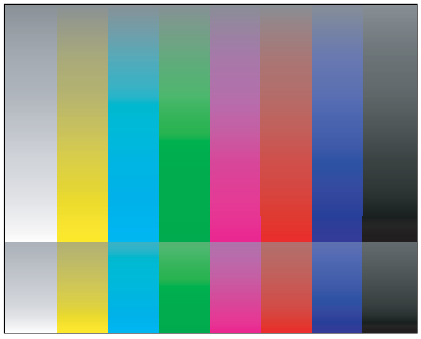
Horizontal mirror = 0, Vertical flip = 0



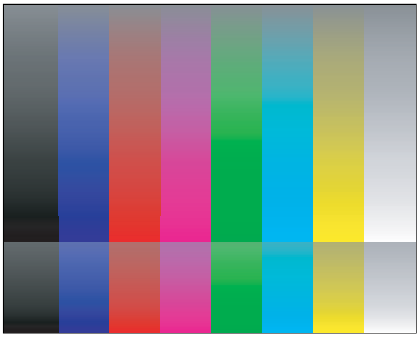
Horizontal mirror = 1, Vertical flip = 0



Horizontal mirror = 0, Vertical flip = 1



Horizontal mirror = 1, Vertical flip = 1



Note: Figures are for illustration purposes only.

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.

Test Cursors

The MT9M019 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 are programmable. 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.

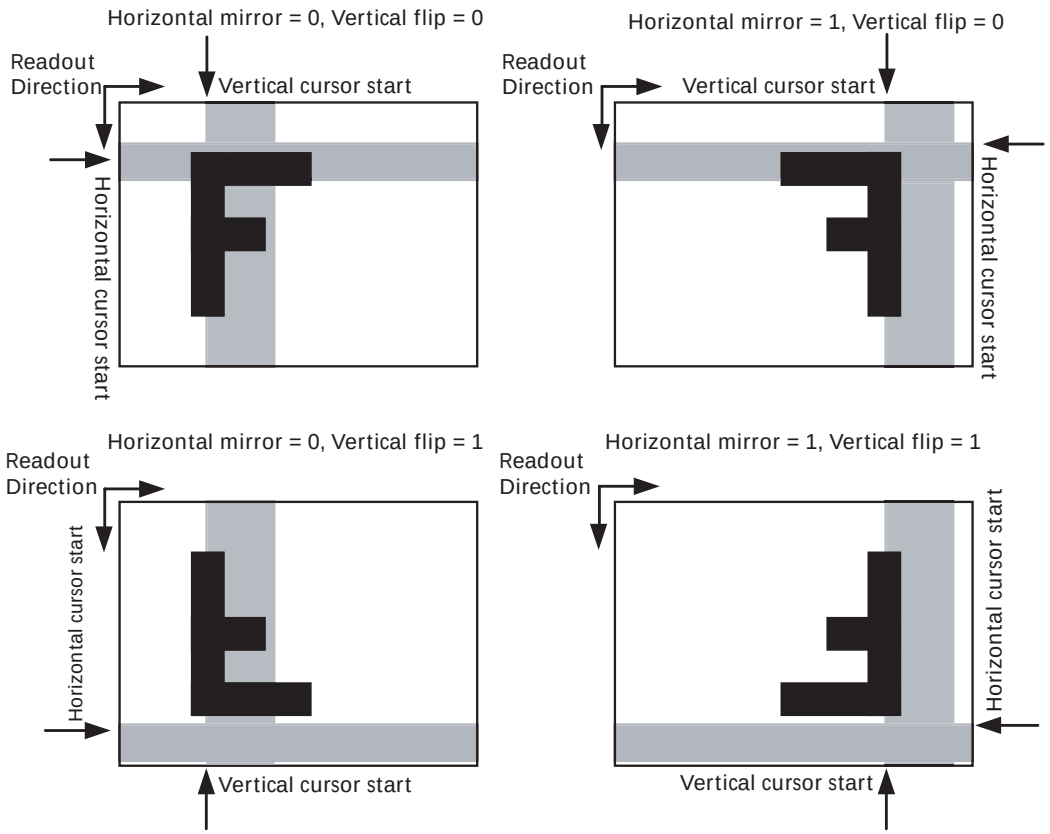
When vertical_cursor_position = 0x07FF, the vertical cursor operates in an automatic mode in which its position advances every frame. In this mode the cursor starts at the column associated with x_addr_start = 0 and advances by a step-size of 8 columns each frame, until it reaches the column associated 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 MT9M019 is shown in Figure 23 on page 43, where 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 23: Test Cursor Behavior – image_orientation



Digital Gain

Integer digital gains in the range 0–7 can be programmed. A digital gain of "0" sets all pixel values to "0" (the pixel data will simply represent the value applied by the pedestal block).

Pedestal

This block adds value from the data_pedestal_ register (R0x301E–F[9:0] to the incoming pixel value.

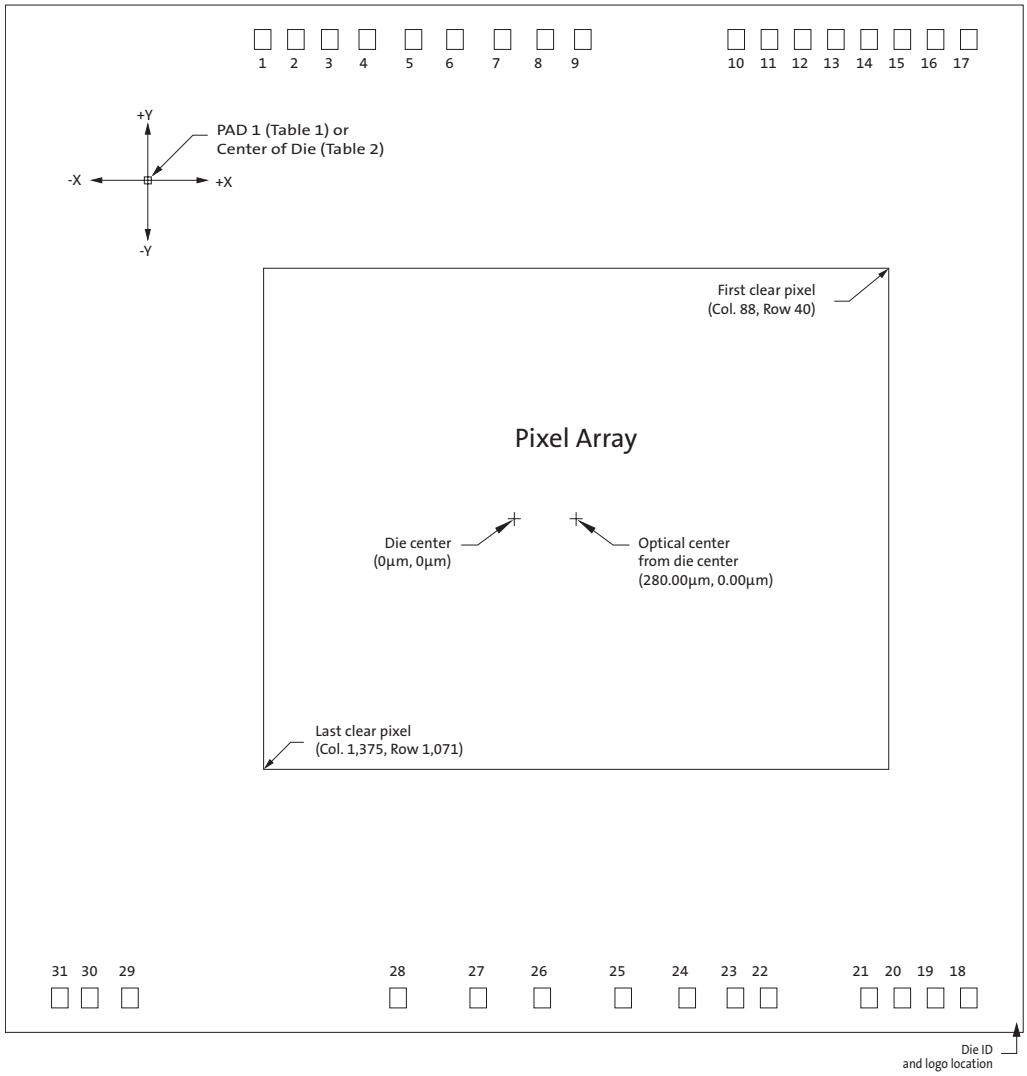
The data_pedestal_ register is read-only by default but can be made read/write by clearing the lock_reg bit in reset_register (R0x301A–B).

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

Mechanical Specifications

Figure 24 shows the die outline, including the readout orientation, the optically-active area, and the offset of the optical center from the die center.

Figure 24: Die Outline



Spectral Characteristics

Figure 25: Quantum Efficiency

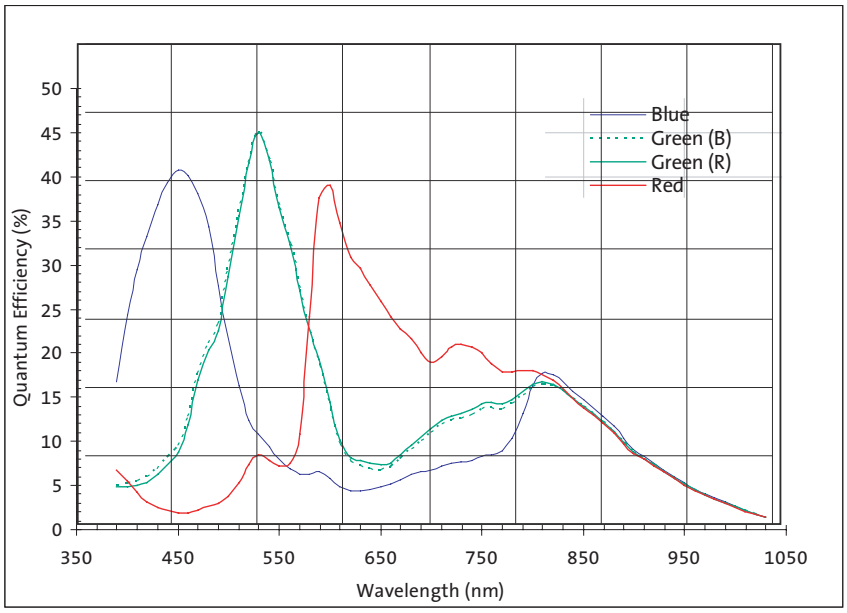
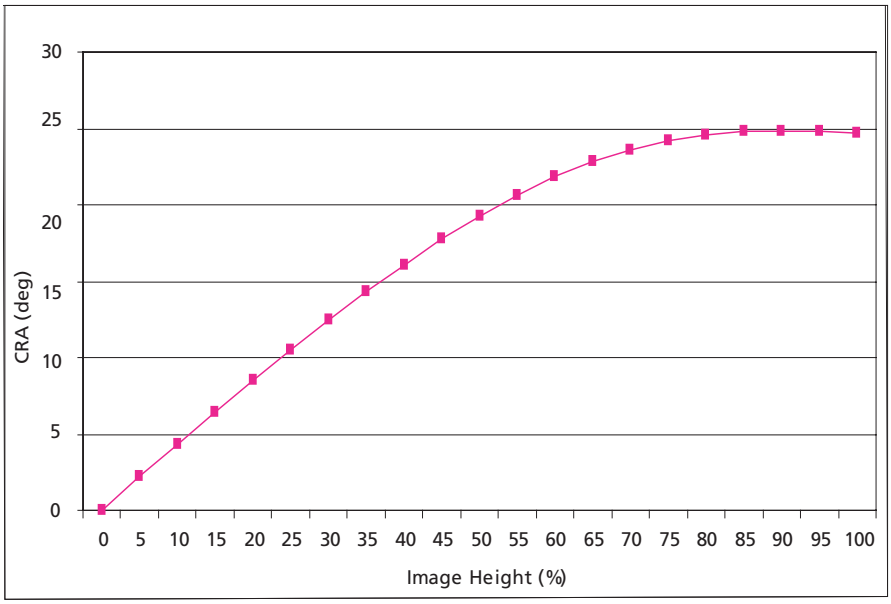


Figure 26: CRA vs. Image Height

CRA vs. Image Height Plot	Image Height (%)	Image Height (mm)	CRA (Deg)
	0	0	0
	5	0.090	2.18
	10	0.180	4.32
	15	0.270	6.41
	20	0.361	8.48
	25	0.451	10.49
	30	0.541	12.45
	35	0.631	14.33
	40	0.721	16.11
	45	0.811	17.77
	50	0.902	19.29
	55	0.992	20.64
	60	1.082	21.82
	65	1.172	22.80
	70	1.262	23.58
	75	1.352	24.16
	80	1.442	24.55
	85	1.533	24.77
	90	1.623	24.84
	95	1.713	24.82
	100	1.803	24.75

Electrical Specifications

EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 15. The EXTCLK input supports either an AC-coupled sine-wave input clock or a DC-coupled square-wave input clock.

If EXTCLK is AC-coupled to the MT9M019 and the clock is stopped, the EXTCLK input to the MT9M019 must be driven to ground or to VDDIO. Failure to do this will result in excessive current consumption within the EXTCLK input receiver.

Table 15: Electrical Characteristics (EXTCLK)

^{f}EXTCLK = 16 MHz; VDD = 1.8V; VDDCCP (VDDIO) = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Output load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
^{f}EXTCLKIN1	Input clock frequency	PLL enabled	6	16	27	MHz
^{t}EXTCLKIN1	Input clock period	PLL enabled	166	62.5	37	ns
^{t}R	Input clock rise time		0.03	–	1	V/ns
^{t}F	Input clock fall time		0.03	–	1	V/ns
VIN_AC	Input clock minimum voltage swing (AC coupled)		0.5	–	–	V (p-p)
VIN_DC	Input clock maximum voltage (DC coupled)		VDD	–	VDD + 0.1	V
^{f}CLKMAX(AC)	Input clock signalling frequency (low amplitude)	VIN = VIN_AC (MIN)	–	–	27	MHz
^{f}CLKMAX(DC)	Input clock signalling frequency (full amplitude)	VIN = VDD	–	–		
	Clock duty cycle		45	50	55	%
^{t}JITTER	Input clock jitter		–	–	300	ps
^{t}CP	EXTCLK to PIXCLK propagation delay		–		–	ns
		PLL enabled	–	5	–	
^{t}LOCK	PLL VCO lock time	PLL enabled	–	0.5	1	ms
CIN	Input pad capacitance		–	2.5	–	pF
IIH	Input HIGH leakage current		–	2.5	10	µA
IIL	Input LOW leakage current		–	2.5	–10	µA

Two-Wire Serial Register Interface

The electrical characteristics of the two-wire serial register interface (SCLK, SDATA) are shown in Table 16. The SCLK and SDATA signals feature fail-safe input protection, Schmitt trigger input, and suppression of input pulses of less than 50ns duration.

Table 16: Two-Wire Serial Register Interface Electrical Characteristics

^{f}EXTCLK = 16 MHz; V_{DD} = 1.8V; V_{DDCCP} (V_{DDIO}) = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DDPLL} = 2.8V; Output load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	Note
V_{IH}	Input HIGH voltage		0.7 x V_{DD}	–	V_{DD} + 0.5	V	
V_{IL}	Input LOW voltage		–0.5	–	0.3V x V_{DD}	V	
I_{IN}	Input leakage current	No pull-up resistor; V_{IN} = V_{DD} or DGND	–	10	–	µA	
V_{OH}	Output HIGH voltage	At specified I_{OH} 8mA, V_{DD} = 1.8V	–	V_{DD}	–	V	1
V_{OL}	Output LOW voltage	At specified I_{OL} 8mA, V_{DD} = 1.8V	0.16	–	0.35	V	1
I_{OH}	Output HIGH current	At specified V_{OH} MAX, V_{DD} = 1.8V	8.9	–	22.3	mA	
I_{OL}	Output LOW current	At specified V_{OL} 0.1V	2.6	–	5.1	mA	
I_{OL}	Output LOW current	At specified V_{OL} 0.4V	8.9	–	18.5	mA	
I_{OZ}	Tri-state output leakage current		–	–	1	µA	
C_{IN}	Input pad capacitance		–	–	6	pF	
C_{LOAD}	Load capacitance		–	–	N/A	pF	

Note: 1. The two-wire serial registers interface does not drive output to HIGH; the HIGH signal state is achieved through the pull-up resistors.

Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLKP, CLKN, DATAP, DATAN) are shown in Table 17.

To operate the serial pixel data interface within the electrical limits of the CCP2 specification, VDDCCP (V_{DDIO}) is restricted to operate in the range 1.7–1.9V.

Table 17: Electrical Characteristics (Serial Pixel Data Interface)

^{f}EXTCLK = 16 MHz; V_{DD} = 1.8V; V_{DDCCP} (V_{DDIO}) = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DDPLL} = 2.8V; Output load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
	Operating frequency		1	–	320	MHz
V_{CMF}	Fixed common mode voltage		0.8	0.9	1	V
Vod	Differential voltage swing		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	%
Vod	Rise time (20–80%)		300	330	400	ps

Table 17: Electrical Characteristics (Serial Pixel Data Interface) (continued)

[f]EXTCLK = 16 MHz; V_{DD} = 1.8V; V_{DD}CCP (V_{DD}IO) = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DD}PLL = 2.8V; Output load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Vod	Fall time (20–80%)		280	300	470	ps
	Differential skew		–	300	500	ps
	Channel-to-channel slew		–	–	200	ps
	Maximum data rate Data/strobe mode Data/clock mode		–	–	 640 208	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_N, SADDR [through GPI], TEST, GPI0, GPI1) are shown in Table 18.

Table 18: AC Electrical Characteristics (Control Interface)

[f]EXTCLK = 16 MHz; V_{DD} = 1.8V; V_{DD}CCP (V_{DD}IO) = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DD}PLL = 2.8V; Output load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IH}	Input HIGH voltage		0.7 x V_{DD}	–	V_{DD} + 0.5	V
V_{IL}	Input LOW voltage		–0.3	–	0.3 x V_{DD}	V
I_{IN}	Input leakage current	No pull-up resistor; V_{IN} = V_{DD} or DGND	–	<10	–	µA
C_{IN}	Input pad capacitance		–	6.5	–	pF

Power-On Reset

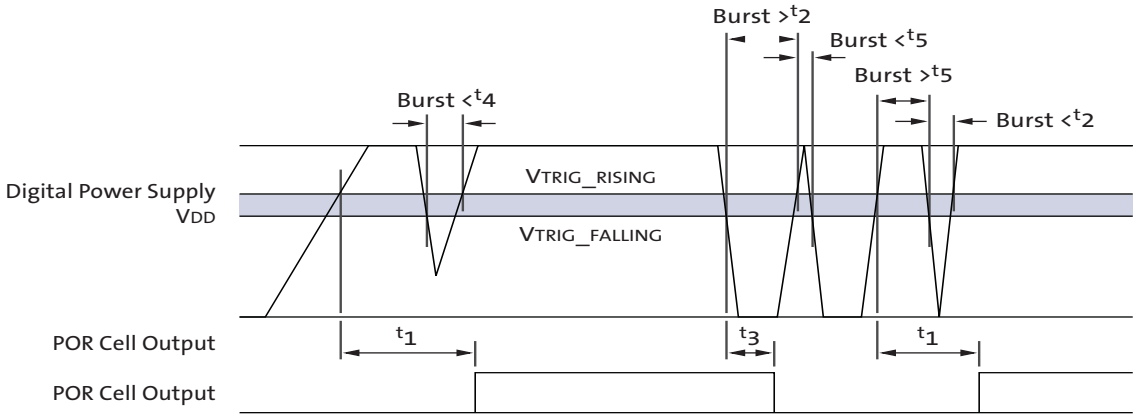
Table 19: Power-On Reset Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Unit
t_1	V_{DD} rising, crossing V_{TRIG_RISING}; Internal reset being released		7	10	15	µs
t_2	V_{DD} falling, crossing $V_{TRIG\_FALLING}$; Internal reset active		–	0.5	1	µs
t_3	Minimum V_{DD} spike width below $V_{TRIG\_FALLING}$; considered to be a reset when POR cell output is HIGH		–	0.5	–	µs
t_4	Minimum V_{DD} spike width below $V_{TRIG\_FALLING}$; considered to be a reset when POR cell output is LOW		–	1	–	µs

Table 19: Power-On Reset Characteristics (continued)

Symbol	Parameter	Condition	Min	Typ	Max	Unit
$^{t}5$	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	–	50	–	ns
VTRIG_RISING	VDD rising trigger voltage		1.12	–	1.55	V
VTRIG_FALLING	VDD falling trigger voltage		1.0	–	1.45	V

Figure 27: Internal Power-On Reset



Operating Voltages

VAA and VAAPIX must be at the same potential for correct operation of the MT9M019.

Table 20: DC Electrical Definitions and Characteristics

^{f}EXTCLK = 16 MHz; VDD = 1.8V; VDDCCP (VDDIO) = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V; Output Load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
VDD	Core digital voltage		1.7	1.8	1.9	V
VDDCCP (VDDIO)	I/O digital voltage	Serial pixel (CCP2) data interface	1.7	1.8	1.9	V
VAA	Analog voltage		2.4	2.8	3.1	V
VAAPIX	Pixel supply voltage		2.4	2.8	3.1	V
VDDPLL	PLL supply voltage		2.4	2.8	3.1	V
IDD1	Digital operating current	Streaming, full resolution	–	18	30	mA
IDD2	I/O digital operating current	Streaming, full resolution	–	10	12	mA
IAA	Analog operating current	Streaming, full resolution	–	42	50	mA
IAAPIX	Pixel supply current	Streaming, full resolution	–	1.4	1.9	mA
IDDPLL	PLL supply current	Streaming, full resolution	–	5	7	mA
	Hard standby (clock off)	Analog	0	–	10	µA
		Digital	10	–	50	µA
	Hard standby (clock on (6 MHz))	Analog	10	–	50	µA
		Digital	160	–	250	µA

Table 20: DC Electrical Definitions and Characteristics (continued)

[f]EXTCLK = 16 MHz; V_{DD} = 1.8V; V_{DD}CCP (V_{DD}IO) = 1.8V; V_{AA} = 2.8V; VAAPIX = 2.8V; V_{DD}PLL = 2.8V; Output Load = 68.5pF; Ambient temperature; 0 lux on sensor

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
	Soft standby (clock off)	Analog (VCO power down)	0	–	20	µA
		Digital (VCO power down)	10	–	50	µA
	Soft standby (clock on (6 MHz))	Analog (VCO power down)	10	–	50	µA
		Digital (VCO power down)	160	–	250	µA

Table 21: SMIA Characterization Data Table

Default power supply voltage: VDIG = 1.8 +/- 0.1V, VANA = 2.8 +/- 0.1V
Parameter at 23+/- 2°C ~ 40°C, at 40°C: except resolution and color accuracy testing

Sensor Dependency Parameters					
Parameters		Lower Limit	Typical	Upper Limit	Units (SMIA)
Sensitivity			0.07		1/(cdm-2*sec)
Photo Response Non-Uniformity (PRNU)			0.8		%
Module Response Non-Linearity	Integral Non-Linearity: 50%		0.00024		Codes/FSD
	Differential Non-Linearity: 50%		0.00458		ratio
SNR	10 cd/m^2		36.60		dB
	50 cd/m^2		36.61		dB
	100 cd/m^2		36.05		dB
	450 cd/m^2		36.31		dB
Maximum Illumination (Minimum Integration Time = 28?sec)			4.76E+05		cd/m^2
Minimum Illumination (Maximum Integration Time = 266ms)			2.64E-02		cd/m^2
Dynamic Range (MSR settings for SMIA test : ???)			66.05		dB
Vertical FPN	Level		3.5E-04		Codes/FSD
	Max		6.9E-04		Codes/FSD
Horizontal FPN	Level		8.1E-05		Codes/FSD
	Max		1.8E-04		Codes/FSD
Temporal Noise			-61.26285412		dB
Column Noise	Level		-86.04232908		dB
	Max		-84.68667414		dB
Row Noise	Level		-68.46697445		dB
	Max		-66.74710259		dB
Frame-to-Frame Flicker			8.43E-04		Codes/pedestal
Dark Signal			3.96E-04		(sec)-1
Dark Signal Non-Uniformity (DSNU)			8.99E-04		(sec)-1
Power Supply Rejection Ratio (PSRR): Value * : in case of SMIA test method which fixed amplitude noise					
	PSRR at 50Hz ~10 kHz	*	69.91		dB
	PSRR at ~1 MHz	*	42.16		dB
	PSRR at ~10 MHz	*	64.28		dB
Image Lag as wafer level test data			<0.05		%

Table 21: SMIA Characterization Data Table

Default power supply voltage: VDIG = 1.8 +/- 0.1V, VANA = 2.8 +/- 0.1V
Parameter at 23+/- 2°C ~ 40°C, at 40°C: except resolution and color accuracy testing

Sensor Dependency Parameters					
Parameters		**Lower Limit**	**Typical**	**Upper Limit**	**Units (SMIA)**
SMIA Color Accuracy	Mean		3.53		ΔE*ab
	Standard Deviation		2.16		ΔE*ab
	Maximum		10.09		ΔE*ab

Note: 1. All the tests were performed using a Rev 3 sensor at gain = 1 and at room temperature except for DSNU, which was performed at gain = 15.875 and at 50°C.

Table 22: Electrical Characteristics (FLASH)

[f]EXTCLKIN = 16 MHz; V_{DD} = 1.8V; V_{DDIO} = 1.8V; VAA = 2.8V; VAAPIX = 2.8V; VDDPLL = 2.8V;
Output Load = 68.5; Ambient temperature

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OH}	Output HIGH voltage	At specified I_{OH} 8mA	V_{DDIO} –0.5	–	–	V
V_{OL}	Output LOW voltage	At specified I_{OL} 8mA		–	0.4	V
I_{OH}	Output HIGH current	At specified V_{OH} MIN, V_{DDIO} = 1.8V	2.7	–	6.4	mA
I_{OH}	Output HIGH current	At specified V_{OH} MAX, V_{DDIO} = 1.8V	8.9	–	22.3	mA
I_{OL}	Output LOW current	At specified V_{OL} 0.1V	2.6	–	5.1	mA
I_{OL}	Output LOW current	At specified V_{OL} 0.4V	8.9	–	18.5	mA
I_{OZ}	Tri-state output leakage current		–	–	1	µA
	Ouput pin slew	Default, C_{LOAD} = 30pF	–	0.3818	–	V/ns

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 23: Absolute Maximum Values

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

Note: 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.

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)

Revision History

Rev. G 2/2/12

- Updated trademarks

Rev. F 5/26/10

- Updated to non-confidential

Rev. E 5/4/10

- Updated to Aptina template

Rev. D, Production 7/30/2007

- Updated Table 23, “Absolute Maximum Values,” on page 52
- Updated Data Sheet to Production status

Rev. C 2/26/2007

- Update Table 1, “Key Performance Parameters,” on page 1
- Remove Figure 27: “Internal Power-On Reset,” on page 49
- Update Table 20, “DC Electrical Definitions and Characteristics,” on page 49
- Add Table 21, “SMIA Characterization Data Table,” on page 50
- Add Table 22, “Electrical Characteristics (FLASH),” on page 51

Rev. B 12/06

- Update Table 1, “Key Performance Parameters,” on page 1
- Update Figure 2: “Typical Configuration: Serial Pixel Data Interface,” on page 7
- Update "Effect of CCP2 Class on Legal Range of Output Sizes/Frame Rate" on page 17
- Update Figure 11: “MT9M019 System States,” on page 21
- Update "Power-On Reset Sequence" on page 22
- Table 6, “PLL in System States,” on page 21
- Update "Power-On Reset Sequence" on page 22
- Update "General Purpose Inputs" on page 23
- Update Figure 12: “MT9M019 SMIA Profile 1, 2 Clocking Structure,” on page 24
- Add "Programming Example" on page 28
- Add Table 11, “Default Settings,” on page 28
- Update Table 12, “Row Address Sequencing,” on page 32
- Update Table 13, “Frame Rates,” on page 33
- Update "Frame Rates at Common Image Sizes" on page 33
- Update "Low Power Mode" on page 35
- Update "Aptina Gain Model" on page 36
- Update Table 13, “Frame Rates,” on page 33
- Update "Low Power Mode" on page 35
- Update Table 14, “Test Patterns,” on page 38
- Update "Effect of Data Path Processing on Test Patterns" on page 38
- Update "Fade-to-Gray Color Bars Test Pattern" on page 40
- Update "100 Percent Color Bars Test Pattern" on page 39
- Update "Test Cursors" on page 42
- Update "Pedestal" on page 43
- Remove Embedded Data Format and Control section
- Remove Table 23, “Embedded Data”
- Update Figure 24: “Die Outline,” on page 44

- Update Table 25, “Quantum Efficiency,” on page 45
- Add Figure 26: “CRA vs. Image Height,” on page 45
- Update Figure 27: “Internal Power-On Reset,” on page 49
- Update Table 15, “Electrical Characteristics (EXTCLK),” on page 46
- Update Table 17, “Electrical Characteristics (Serial Pixel Data Interface),” on page 47
- Update Table 18, “AC Electrical Characteristics (Control Interface),” on page 48
- Update Table 20, “DC Electrical Definitions and Characteristics,” on page 49

Rev. A 7/06

- Initial release

10 Eunos Road 8 13-40, Singapore Post Center, Singapore 408600 prodmktg@aptina.com www.aptina.com