1/11-Inch VGA CMOS Digital Image Sensor

MT9V013

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

Features

- DigitalClarity® CMOS imaging technology
- Low dark current
- Simple two-wire serial interface
- Auto black level calibration
- Support for external LED or xenon flash
- Programmable controls: gain, frame size/rate, exposure, left–right and top–bottom image reversal, window size, panning, and skip 2x
- Data interface: single lane serial module industry processor interface (MIPI)
- On-chip PLL (phase-locked loop oscillator)
- Superior low-light performance

Applications

- Cellular phones
- PC cameras
- PDAs

General Description

Aptina's MT9V013 is a 1/11-inch VGA-format CMOS active-pixel digital image sensor with a pixel array of 640 H x 480 V (648 H x 488 V 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 MT9V013 digital image sensor features DigitalClarity—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.

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

Table 1: Key Performance Parameters

Parameter		Value
Optical format		1/11-inch VGA (4:3)
Active imager size		1.43mm (H) x 1.07mm (V) 1.79mm diagonal (calculated from 648 x 488)
Active pixels		648H x 488V
Pixel size		2.2µm x 2.2µm
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS)
Maximum data rate/ master clock		14 Mp/s at 14 MHz system clock
Frame rate	VGA (640 x 480)	Programmable up to 30 fps
ADC resolution		10-bit, on-chip
Responsivity		1.1 V/lux-s
Dynamic range		64dB
SNR_{MAX}		> 36.5dB
Supply voltage	Analog	2.50–3.10V (2.80V nominal)
	Digital, DigitalPHY	1.7–1.9V (1.80V nominal)
	Digital I/O	1.7–1.9V (1.80V nominal)
Power consumption		68mW at 30 fps
Operating temperature		–30°C to +70°C
Packaging		Die

Ordering Information

Table 2: Available Part Numbers

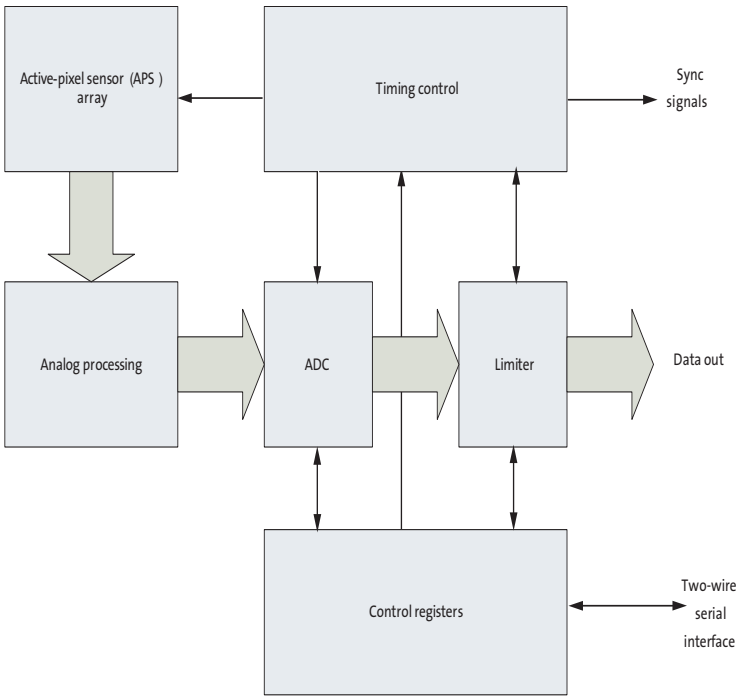
Part Number	Description
MT9V013D00STCM	Bare die

Table of Contents

Functional Overview

The MT9V013 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 MHz and 27 MHz. The maximum pixel rate is 14 Mp/s at 14 MHz clock. A block diagram of the sensor is shown in Figure 1.

Figure 1: Block Diagram



The core of the sensor is a VGA 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 are 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 and red pixels or blue and 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:

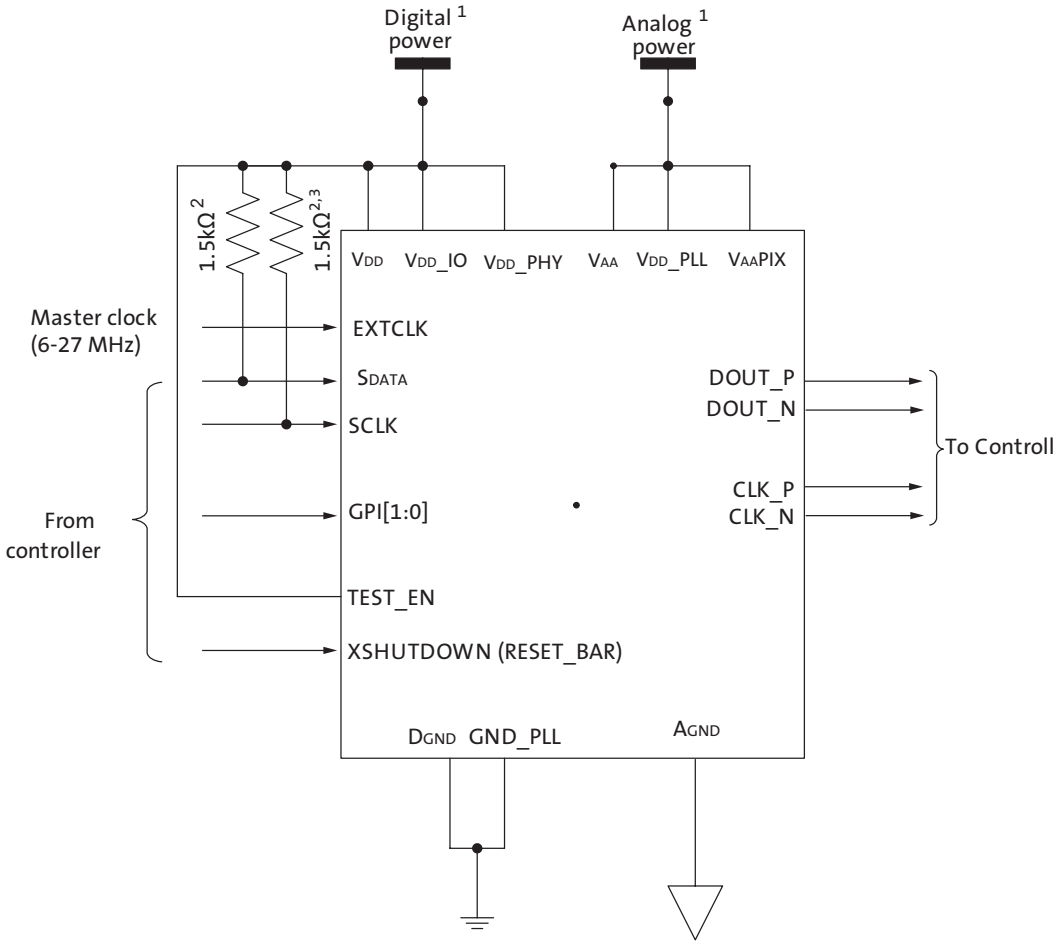
- A sensor core that provides array control and data path corrections. The output of the sensor core is a serial pixel data stream.
- A limiter and a serializer that are required to complete the sensor.

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 MT9V013 powers up as a MIPI 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



Notes:
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.

Signal Descriptions

Table 3 provides signal descriptions for MT9V013 die. For pad location and aperture information, refer to the MT9V013 die data sheet.

Table 3: Signal Descriptions

Pad Name	Pad Type	Description
EXTCLK	Input	Master clock input. 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_EN	Input	Enable manufacturing test modes. Wire to VDD for functional operation.
SDATA	I/O	Serial data for READS from and WRITES to control and status registers.
DATA_P	Output	Differential CSI (sub-LVDS) serial data (positive).
DATA_N	Output	Differential CSI (sub-LVDS) serial data (negative).
CLK_P	Output	Differential CSI (sub-LVDS) serial clock/data (positive).
CLK_N	Output	Differential CSI (sub-LVDS) serial clock/data (negative).
FLASH	Output	Flash output. Synchronization pulse for external light source.
VAA	Supply	Analog power supply.
VAA_PIX	Supply	Analog power supply for the pixel array.
AGND	Supply	Analog ground.
VDD	Supply	Digital power supply.
DGND1, DGND2, GND_PLL	Supply	Common ground for digital and I/O.
VDD_PLL	Supply	PLL power supply.
VDD_IO	Supply	Digital I/O supply.
VDD_PHY	Supply	Digital PHY supply.

Output Data Format

Serial Pixel Data Interface

The MT9V013 serial pixel data interface signaling is implemented according to the MIPI specification. The RAW10 image data format is supported.

Frame Timing

This section describes the frame timing of the MT9V013. This frame timing represents the limits that are possible for the output of the MT9V013.

The timing is described in terms of the FRAME_VALID (FV), LINE_VALID (LV), DOUT, and PIXCLK signals that exist on the interface to the sensor. Since the MT9V013 only provides serial interfaces for pixel data, these signals do not exist on the pads of the MT9V013. However, understanding of this timing is essential for correct programming of the MT9V013.

These figures are based on a PIXCLK period of 71.43ns (one 10-bit pixel every 71.43ns). This corresponds to a frequency of approximately 14 MHz.

The behavior of the interface is shown in Figure 3 and Figure 4 on page 10. The interface timing is shown in Table 3 on page 8. The interface timing uses the minimum blanking times, and therefore shows the maximum frame rate achievable. Specifically:

$$\text{frame\_length\_lines(min)} = \text{y\_add\_end} - \text{y\_addr\_start} + 1 + \text{min\_frame\_blanking\_lines} = 488 + 19 = 507 \quad \text{(EQ 1)}$$

$$\text{line\_length\_pck(min)} = \text{x\_addr\_end} - \text{x\_addr\_start} + 1 + \text{min\_line\_blanking\_lines} = 648 + 194 = 842 \quad \text{(EQ 2)}$$

Using these figures, the maximum frame rate is *1/30,493,038ns = 32.79 fps*.

The REG SYNC point shown in Figure 4 on page 10 is the point at which frame-synchronized register values are transferred from the pending registers to the live registers.

Figure 3: Pixel Data Timing Example

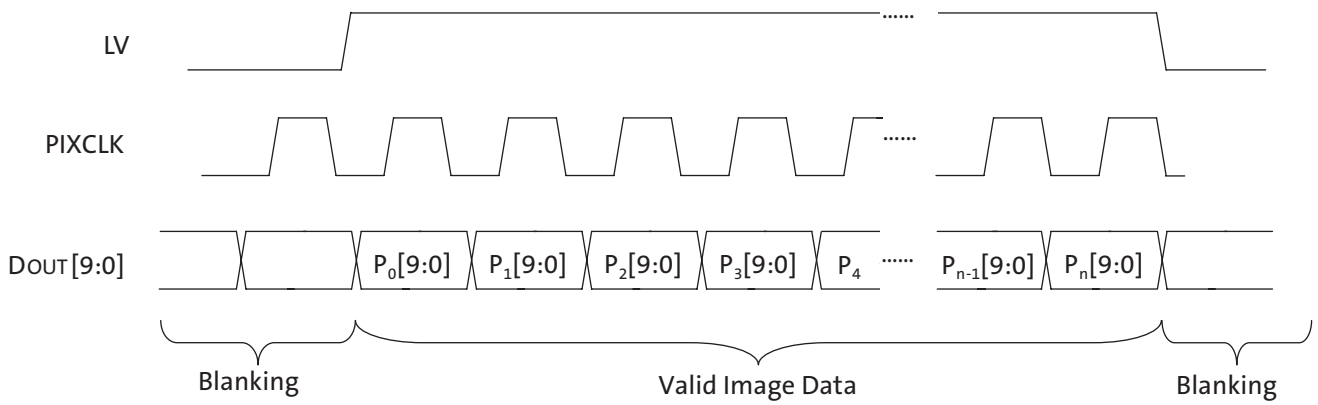


Figure 4: Row Timing and FV / LV Signals

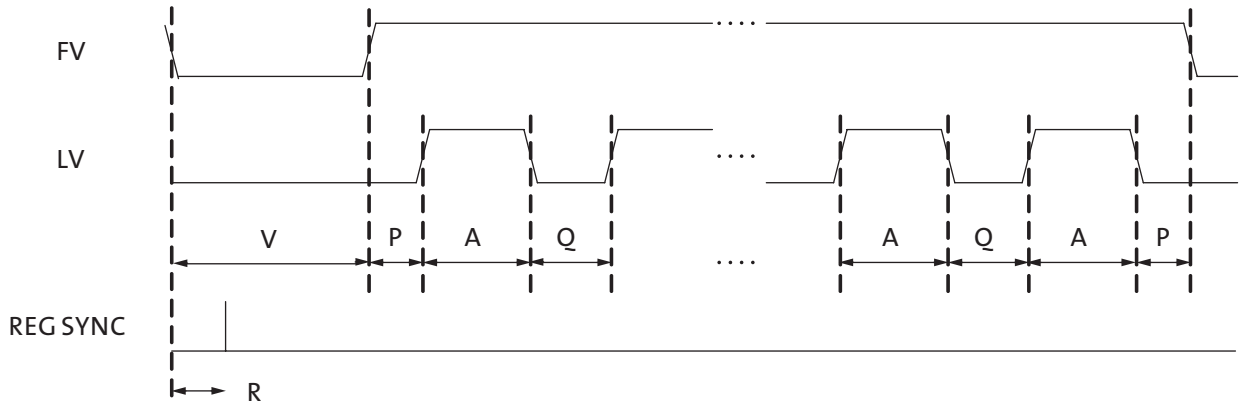


Table 4: Frame Timing

Parameter	Name	Equation	Default Timing[1]
PIXCLK_PERIOD	Pixel clock period	vt_pix_clk_freq_mhz	1 pixel clock = 71.43ns
S	Skip (subsampling) factor	if x_odd_inc = y_odd_inc = 3, S =2 otherwise, S = 1	1
A	Active data time	(x_addr_end – x_addr_start + 1) * PIXCLK_PERIOD / S	648 pixel clocks = 46.29?s
P	Frame start/end blanking	6 PIXCLK_PERIOD	6 pixel clocks = 429ns
Q	Horizontal blanking	line_length_pck * PIXCLK_PERIOD – A	194 pixel clocks = 13.86?s
A + Q	Row time	line_length_pck * PIXCLK_PERIOD	842 pixel clocks = 60.14?s
V	Vertical blanking	((frame_length_lines – N) * (A+Q)) + Q – (2 * P)	16,180 pixel clocks = 1155.74?s
N	Number of rows	(y_addr_end – y_addr_start + 1) / S	488 rows
	Frame valid time	(N * (A + Q)) – Q + (2 * P)	410,714 pixel clocks = 29.34ms
F	Total frame time	line_length_pck * frame_length_lines * PIXCLK_PERIOD	426,894 pixel clocks = 30.49ms
R	Register synchronization point	nominal minimum: (line_length_pck * 2) * PIXCLK_PERIOD	1684 pixel clocks = 120.29?s

Notes: 1. The defaults shown here are for an oversized image: 648 x 488, representing a 4-pixel border on each edge. The hardware defaults used by the sensor chip will be set up for an image size of 640 x 480.

Two-Wire Serial Register Interface

The two-wire serial interface bus enables read/write access to control and status registers within the MT9V013. This interface is designed to be compatible with the MIPI Alliance Standard for Camera Serial Interface 2 (CSI-2) 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 and SCLK are pulled up to VDD_IO 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 MT9V013 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:

1. a (repeated) start condition
2. a slave address/data direction byte
3. an acknowledge bit or 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 MT9V013 are 0x6C (WRITE address) and 0x6D (READ address) in accordance with the MIPI specification. Alternate slave addresses of 0x6E(WRITE address) and 0x6F (READ address) can be selected by assigning one of the GPI inputs to perform the SADDR function.

An alternate slave address can be programmed through registers. See Table 5 on page 1.

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 MIPI and CSI-2.

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.

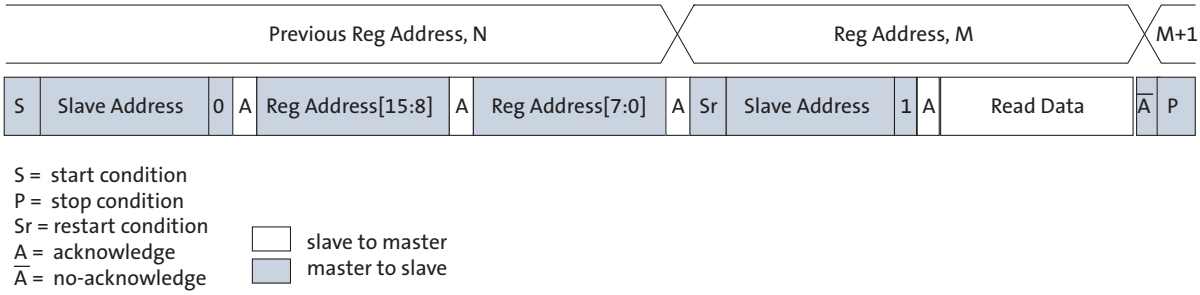
If the request was a WRITE, the master then transfers the 16-bit register address to which a WRITE should take place. This transfers takes place as two 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the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

If the request was a READ, the master sends the 8-bit write slave address/data direction byte and 16-bit register address, just as in the WRITE request. The master then generates a start or restart 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 5) 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 5 shows how the internal register address maintained by the MT9V013 is loaded and incremented as the sequence proceeds.

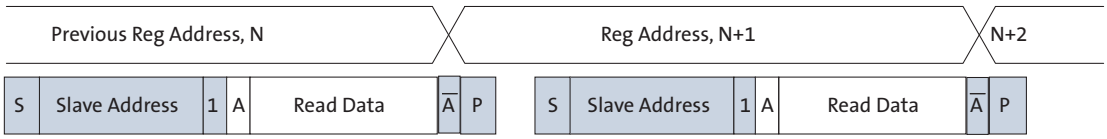
Figure 5: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 6) performs a READ using the current value of the MT9V013 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.

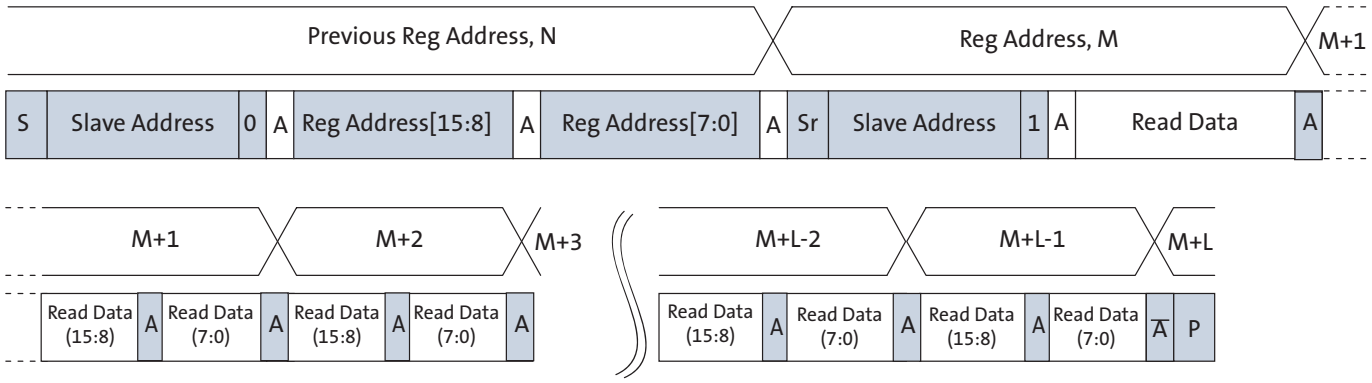
Figure 6: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 7) starts in the same way as the single READ from random location (Figure 5 on page 13). 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.

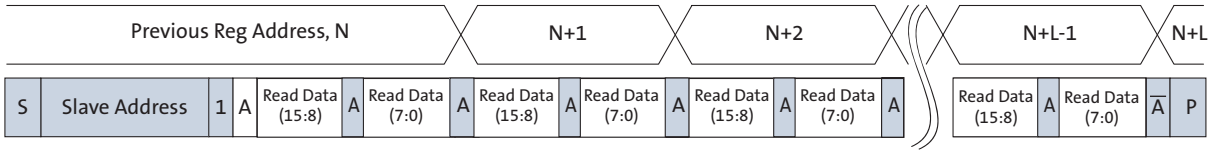
Figure 7: Sequential READ, Start from Random Location



Sequential READ, Start from Current Location

This sequence (Figure 8) starts in the same way as the single READ from current location (Figure 6 on page 13). 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.

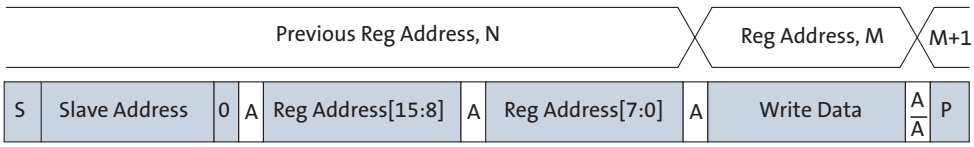
Figure 8: Sequential READ, Start from Current Location



Single WRITE to Random Location

This sequence (Figure 9) 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.

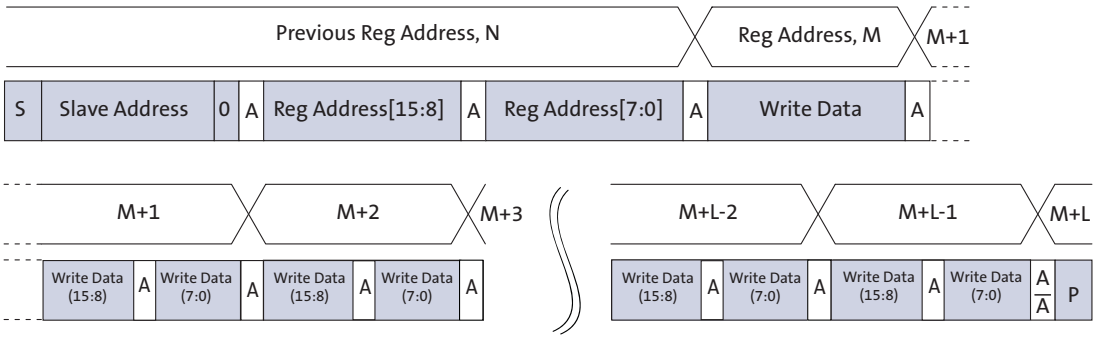
Figure 9: Single WRITE to Random Location



Sequential WRITE, Start at Random Location

This sequence (Figure 10) starts in the same way as the single WRITE to random location (Figure 9 on page 14). 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 10: Sequential WRITE, Start at Random Location



Spectral Characteristics

Figure 11: Quantum Efficiency

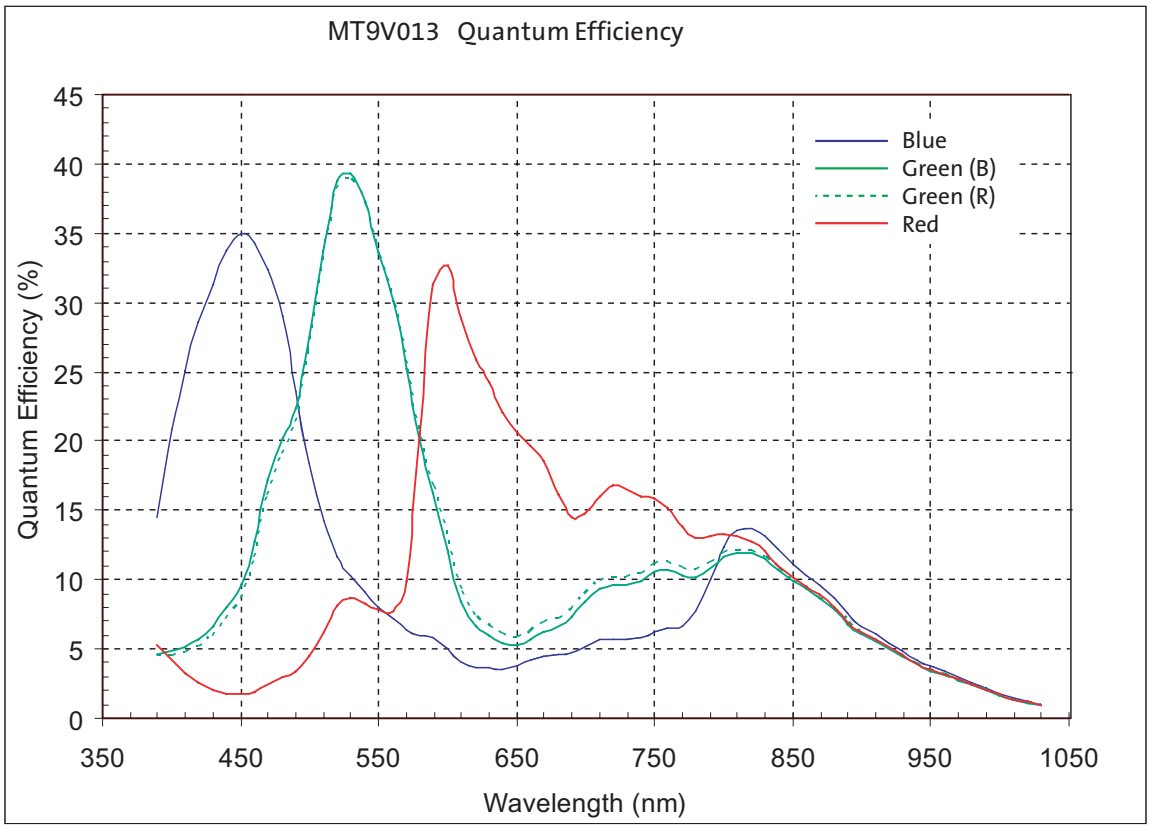


Figure 12: Chief Ray Angle (CRA) vs. Image Height

CRA vs. Image Height Plot

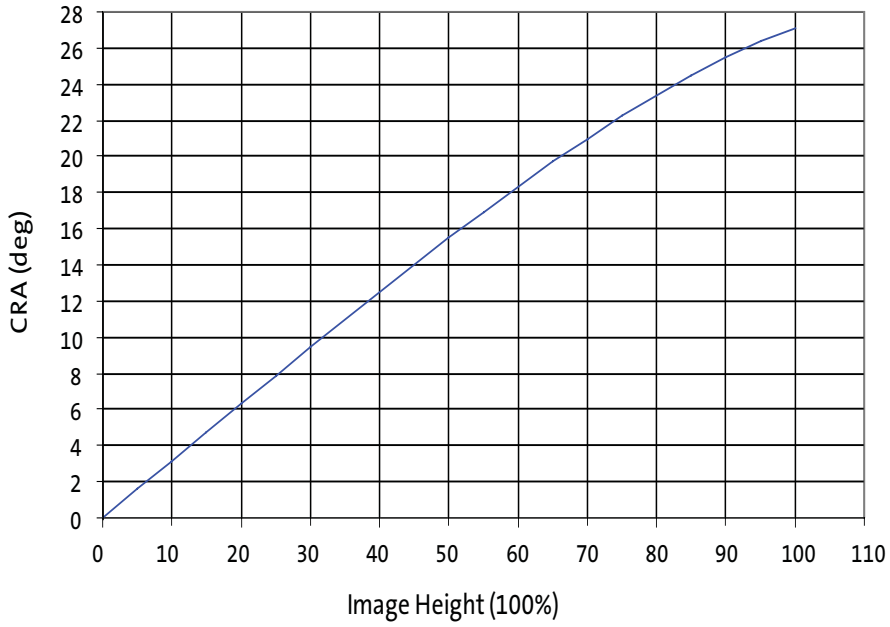


Image Height		CRA
(%)	(mm)	(deg)
0	0	0
5	0.044	1.59
10	0.088	3.17
15	0.132	4.75
20	0.176	6.33
25	0.220	7.90
30	0.264	9.47
35	0.308	11.01
40	0.352	12.53
45	0.396	14.04
50	0.440	15.52
55	0.484	16.96
60	0.528	18.35
65	0.572	19.70
70	0.616	21.00
75	0.660	22.23
80	0.704	23.40
85	0.748	24.48
90	0.792	25.48
95	0.836	26.34
100	0.880	27.10

Electrical Specifications

EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 5. 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 MT9V013 and the clock is stopped, the EXTCLK input to the MT9V013 must be driven to ground or to V_{DD_IO}. Failure to do this will result in excessive current consumption within the EXTCLK input receiver.

The EXTCLK input supports full and low amplitude signal. Low amplitude signal will allow lower swing of V_{DD_IO} to be supplied. If this is to be used, then it should be AC-coupled using a suitable capacitor. The value of the capacitor should be greater than 1nF.

Table 5: Electrical Characteristics (EXTCLK)

^{f}EXTCLKIN = 14 MHz; V_{DD} = V_{DD_IO} = V_{DD_PHY} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; Output load = 68.5pF; T_J = 70°C

Symbol	Parameter	Conditions	Min	Typ	Max	Units
^{f}EXTCLKIN	Input clock frequency		6	14	27	MHz
^{t}EXTCLKIN	Input clock period		166	71	37	ns
^{t}R	Input clock rise time		1	–	–	V/ns
^{t}F	Input clock fall time		1	–	–	V/ns
V_{IN_AC}	Input clock minimum voltage swing (AC coupled)		0.5	–	–	V (p-p)
V_{IN_DC}	Input clock maximum voltage (DC coupled)		–	–	V_{DD} + 0.5	V
^{f}CLKMAX(AC)	Input clock signaling frequency (low amplitude)	V_{IN} = V_{IN_AC} (MIN)	–	–	27	MHz
^{f}CLKMAX(DC)	Input clock signaling frequency (full amplitude)	V_{IN} = V_{DD}	–	–	27	MHz
	Clock duty cycle		45	50	55	%
^{t}JITTER	Input clock jitter		–	500	–	ps
^{t}LOCK	PLL VCO lock time	PLL enabled	–	0.2	1	ms
C_{IN}	Input pad capacitance		–	3.5	–	pF
I_{IH}	Input HIGH leakage current		–	–	10	µA
I_{IL}	Input LOW leakage current		–	–	–10	µA
V_{IH}	Input HIGH voltage		0.7 * V_{DD_IO}	–	V_{DD_IO} + 0.5	V
V_{IL}	Input LOW voltage		–0.5	–	0.3 * V_{DD_IO}	V

Flash Output

The electrical characteristics of the FLASH output are shown in Table 6.

Table 6: Electrical Characteristics for Flash Output

^{f}EXTCLKIN = 14 MHz; V_{DD} = V_{DD_IO} = V_{DD_PHY} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; C_{LOAD} = 68.5pF; T_J = 70°C

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{OH}	Output HIGH voltage	At specified I_{OH} of 8mA	V_{DD_IO} – 0.4	–	–	V
V_{OL}	Output LOW voltage	At specified I_{OL} of 8mA	–	–	0.4	V
I_{OH}	Output HIGH current	At specified V_{OH}, V_{DD_IO} = 1.8V	–	–	20	mA
I_{OL}	Output LOW current	At specified V_{OL} of 0.1V	–	–	–15	mA
I_{OL}	Output LOW current	At specified V_{OL} of 0.4V	–	–	–20	mA
I_{OZ}	Tri-state output leakage current		–10	–	10	µA
	Output slew (rising)	C_{LOAD} = 27pF		0.3		V/ns
	Output slew (falling)	C_{LOAD} = 27pF		0.4		V/ns

Two-Wire Serial Register Interface

The electrical characteristics of the two-wire serial register interface (SCLK, SDATA) are shown in Table 7. The SCLK and SDATA signals feature fail-safe input protection, Schmitt trigger input, and suppression of input pulses of less than 50ns duration.

Table 7: Two-Wire Serial Register Interface Electrical Characteristics

^{f}EXTCLKIN = 14 MHz; V_{DD} = V_{DD}_IO = VDD_PHY = 1.8V; V_{AA} = 2.8V; V_{AA}_PIX = 2.8V; V_{DD}PLL = 2.8V; Output load = 68.5pF; T_J = 70°C

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IL}	Input LOW voltage		–0.5	–	0.3 * V_{DD}_IO	V
I_{IN}	Input leakage current	No pull-up resistor; V_{IN} = V_{DD}_IO or DGND	–2	–	2	µA
V_{OL}	Output LOW voltage	At specified I_{OL} 3mA	0.11	–	0.275	V
I_{OL}	Output LOW current	At specified V_{OL} 0.1V	–	–	3	mA
C_{IN}	Input pad capacitance		–	–	6	pF
C_{LOAD}	Load capacitance		–	–	N/A	pF

Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLK_P, CLK_N, DATA_P, DATA_N) are shown in Table 8. To operate the serial pixel data interface within the electrical limits of the serial specification, VDD_PHY is restricted to operate in the range 1.7–1.9V.

Table 8: Electrical Characteristics

[f]CLKIN = 14 MHz; VDD = VDD_IO = VDD_PHY = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Output load = 56pF; TJ = 70°C

Symbol	Parameter	Min	Typ	Max	Unit
High Speed (HS)					
V_{OD}	HS transmit differential voltage	–	160	–	mV
V_{CMTX}	HS transmit static common-mode voltage	–	195	–	mV
$\lvert\Delta V_{OD}\rvert$	V_{OD} mismatch when output is Differential-1 or Differential-0	–	1	–	mV
$\lvert\Delta V_{CMTX(1,0)}\rvert$	V_{CMTX} mismatch when output is Differential-1 or Differential-0	–	1.3	–	mV
V_{OHHS}	HS output high voltage	–	170	–	mV
Z_{OS}	Single-ended output impedance	–	65	–	Ω
ΔZ_{OS}	Single-ended output impedance mismatch	–	6	–	%
$\Delta V_{CMTX(LF)}$	Common-level variation between 50–450 MHz	–	9	–	mV_{PEAK}
t_R	20–80% rise time	–	400	–	ps
t_F	20–80% fall time	–	420	–	ps
Low Power (LP)					
V_{OL}	Output low level	–	14	–	mV
V_{OH}	Output high level	–	1.2	–	V
Z_{OLP}	Output impedance of LP transmitter	–	100	–	Ω
T_{RLP}	15–85% rise time	–	7	–	ns
T_{FLP}	15–85% fall time	–	13	–	ns
$\Delta V/\Delta t_{SR}$	Slew rate (C_{LOAD} = 5–20pF)	–	265	–	mV/ns
$\Delta V/\Delta t_{SR}$	Slew rate (C_{LOAD} = 20–70pF)	–	150	–	mV/ns

Control Interface

The electrical characteristics of the control interface (RESET_BAR, SADDR [through GPI], TEST, GPI0, GPI1) are shown in Table 9.

Table 9: AC Electrical Characteristics (Control Interface)

^{f}EXTCLKIN = 14 MHz; V_{DD} = V_{DD}_IO =VDD_PHY = 1.8V; V_{AA} = 2.8V; V_{AA}_PIX = 2.8V; V_{DD}_PLL = 2.8V; Output load = 68.5pF; T_J = 70°C

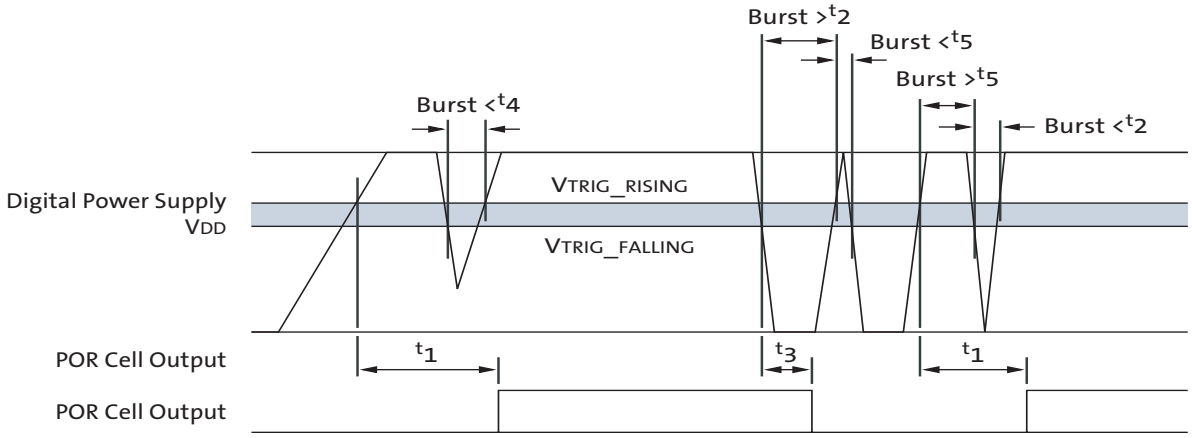
Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IH}	Input HIGH voltage		0.7 x V_{DD}_IO	–	V_{DD}_IO + 0.5	V
V_{IL}	Input LOW voltage		–0.5	–	0.3 x V_{DD}_IO	V
I_{IN}	Input leakage current	No pull-up resistor; V_{IN} = V_{DD}_IO or D_{GND}	–10	–	10	µA
C_{IN}	Input pad capacitance		–	6.5	–	pF

Power-On-Reset

Table 10: 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	19	µs
$^{t}2$	V_{DD} falling, crossing $V_{TRIG\_FALLING}$; internal reset active		–	0.5	1.0	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	–	
$^{t}4$	Minimum V_{DD} spike width below $V_{TRIG\_FALLING}$; considered to be a reset when POR cell output is LOW		–	1.0	–	µs
$^{t}5$	Minimum V_{DD} spike width above V_{TRIG_RISING}; considered to be a stable supply when POR cell output is LOW	While the POR cell output is LOW, all V_{DD} spikes above V_{TRIG_RISING} less than $^{t}5$ must be ignored	–	50	–	ns
V_{TRIG_RISING}	V_{DD} rising trigger voltage		1.15	1.4	1.55	V
$V_{TRIG\_FALLING}$	V_{DD} falling trigger voltage		1.00	1.25	1.45	V

Figure 13: Internal Power-On Reset



Operating Voltages

V_{AA} and V_{AA_PIX} must be at the same potential for correct operation of the MT9V013.

Table 11: DC Electrical Definitions and Characteristics

^{f}EXTCLKIN = 14 MHz; V_{DD} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; Output Load = 68.5pF; T_J = 70°C

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{DD}	Core digital voltage		1.7	1.8	1.9	V
V_{DD_IO}	I/O digital voltage		1.7	1.8	1.9	V
V_{DD_PHY}	PHY digital voltage	Serial pixel data interface	1.7	1.8	1.9	V
V_{AA}	Analog voltage		2.5	2.8	3.1	V
V_{AA_PIX}	Pixel supply voltage		2.5	2.8	3.1	V
V_{DD_PLL}	PLL supply voltage		2.5	2.8	3.1	V
I_{DD}	Digital operating current	Streaming, full resolution	–	1.4	–	mA
I_{DD_PHY}	PHY digital operating current	Streaming, full resolution	–	5	–	mA
I_{AA}	Analog operating current	Streaming, full resolution	–	11	–	mA
I_{AA_PIX}	Pixel supply current	Streaming, full resolution	–	0.6	–	mA
I_{DD_PLL}	PLL supply current	Streaming, full resolution	–	8.5	–	mA
	Hardware standby (clock off)	Analog 3.1V	–	–	8	µA
		Digital 1.9V	–	–	5	µA
	Hardware standby (clock on: 6 MHz)	Analog 3.1V	–	–	30	µA
		Digital 1.9V	–	–	150	µA
	Software standby (clock off)	Analog 3.1V	–	–	8	µA
		Digital 1.9V	–	–	5	µA
	Software standby (clock on: 6 MHz)	Analog 3.1V	–	–	30	µA
		Digital 1.9V	–	–	150	µA

Absolute Maximum Ratings

Caution **Stresses greater than those listed in Table 12 may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect reliability.**

Table 12: Absolute Maximum Values

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DD_MAX}	Core digital voltage		–	–	2.4	V
V_{DD_MAX}	I/O digital voltage		–	–	2.4	V
$V_{DD\_PHY\_MAX}$	PHY digital voltage		–	–	2.4	V
V_{AA_MAX}	Analog voltage		–	–	4	V
$V_{AA\_PIX\_MAX}$	Pixel supply voltage		–	–	4	V
$V_{DD\_PLL\_MAX}$	PLL supply voltage		–	–	4	V
I_{DD_MAX}	Digital operating current	Worst case current	–	–	2.5	mA
$I_{DD\_IO\_MAX}$	I/O digital operating current	Worst case current	–	–	0.1	mA
$I_{DD\_PHY\_MAX}$	PHY digital operating current	Worst case current	–	–	7	mA
I_{AA_MAX}	Analog operating current	Worst case current	–	–	14	mA
$I_{AA\_PIX\_MAX}$	Pixel supply current	Worst case current	–	–	1	mA
$I_{DD\_PLL\_MAX}$	PLL supply current	Worst case current	–	–	14	mA
T_{OP}	Operating temperature	Measure at Junction	–30	–	70	°C
T_{STG}[1]	Storage temperature		–40	–	85	°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.

Two-Wire Serial Bus Timing

Figure 14 and Table 13 on page 27 describe the timing for the two-wire serial interface.

Figure 14: Two-Wire Serial Bus Timing Parameters

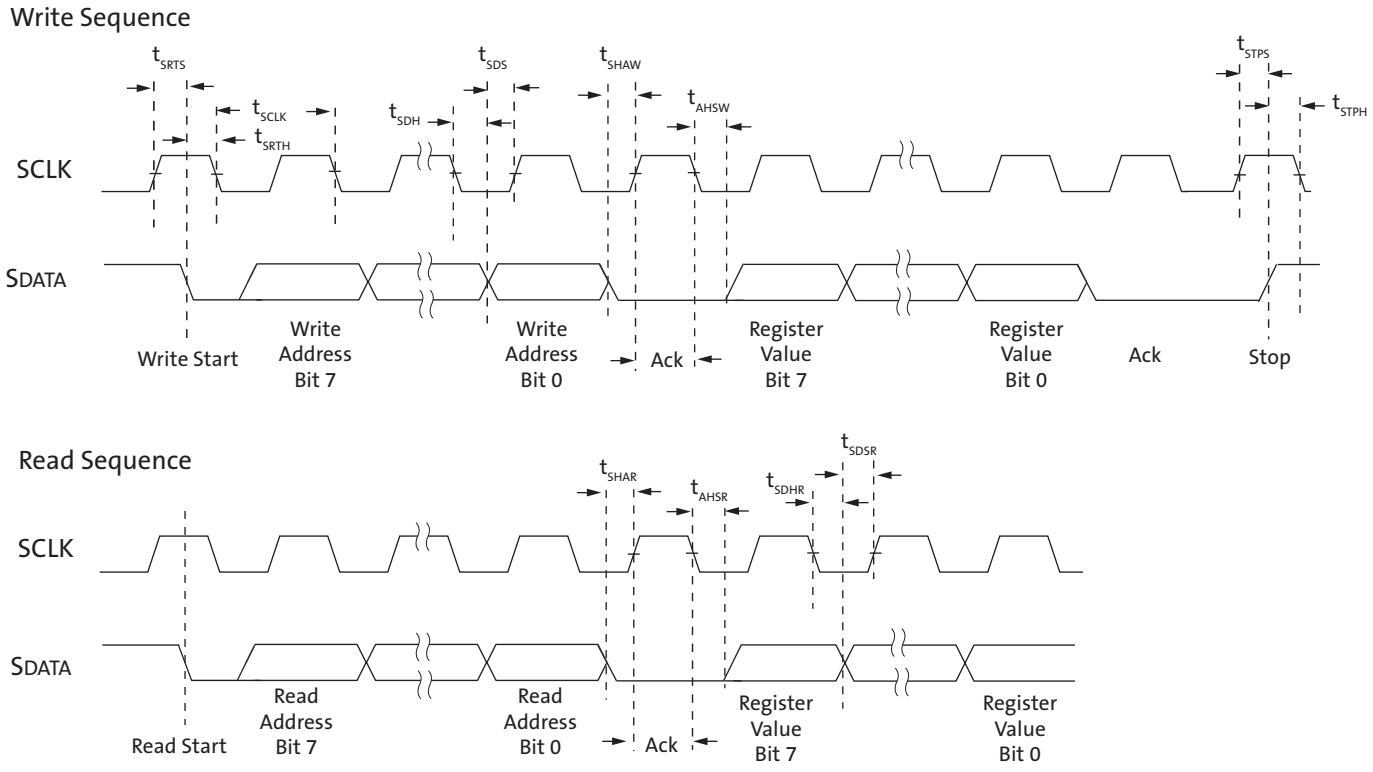


Table 13: Two-Wire Serial Bus Characteristics

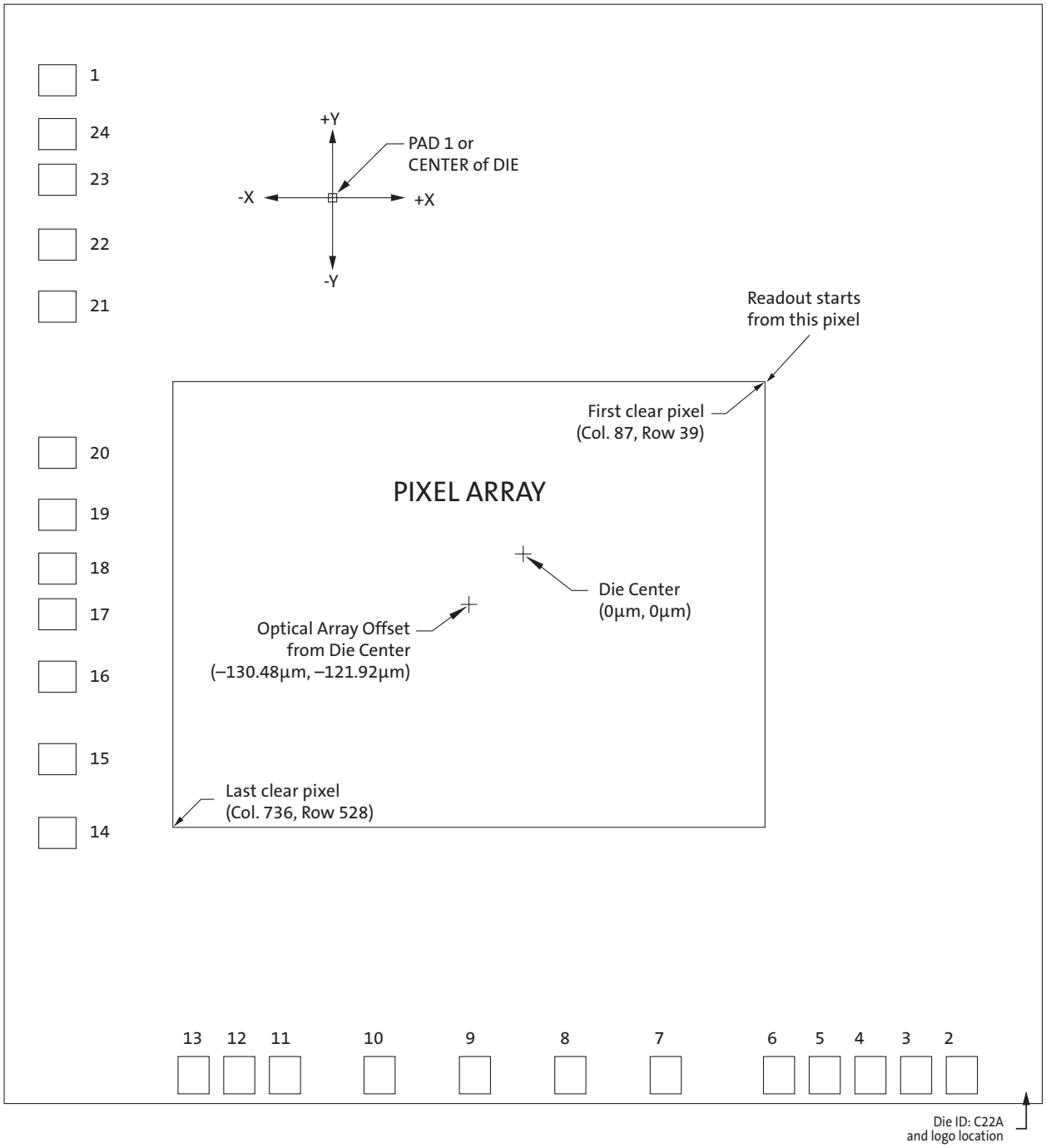
^{f}EXTCLKIN = 14 MHz; V_{DD} = 1.8V; V_{DD_IO} = 1.8V; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_PLL} = 2.8V; V_{DD_PHY} = NA; T_J = 70°C; C_{LOAD} = 68.5pF

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
^{f}SCLK	Serial interface input clock frequency		100	–	400	kHz
^{t}SCLK	Serial interface input clock period		2.5	–	10	?s
	SCLK duty cycle		45	50	50	%
^{t}r	SCLK/SDATA rise time		–	–	300	ns
^{t}SRTS	Start setup time	Master write to slave	600	–	–	
^{t}SRTH	Start hold time	Master write to slave	300	–	–	ns
^{t}SDH	SDATA hold	Master write to slave	300	–	650	ns
^{t}SDS	SDATA setup	Master write to slave	300	–	–	ns
^{t}SHAW	SDATA hold to ack	Master read from slave	150	–	–	ns
^{t}AHSW	Ack hold to SDATA	Master read from slave	150	–	–	ns
^{t}STPS	Stop setup time	Master write to slave	300	–	–	ns
^{t}STPH	Stop hold time	Master write to slave	600	–	–	ns
^{t}SHAR	SDATA hold to ack	Master write to slave	300	–	–	ns
^{t}AHSR	ACK hold to SDATA	Master write to slave	300	–	–	ns
^{t}SDHR	SDATA hold	Master read from slave	300	–	650	ns
^{t}SDSR	SDATA setup	Master read from slave	300	–	–	ns

Mechanical Specifications

Figure 15 shows the die outline, including the readout orientation, the optically-active area, and the offset of the optical center from the die center.

Figure 15: Die Outline



Specification References

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

- MIPI Alliance Standard for CSI-2 version 1.0
- MIPI Alliance Standard for D-PHY version 0.65

Programming Restrictions

Table 15 shows a list of programming rules that must be adhered to for correct operation of the MT9V013. It is recommended that these rules are encoded either implicitly or explicitly into the device driver stack.

Table 14: 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 15: Programming Rules

Parameter	Minimum Value	Maximum Value
coarse_integration_time	coarse_integration_time_min	frame_length_lines – coarse_integration_time_max_margin
fine_integration_time	fine_integration_time_min	line_length_pck – fine_integration_time_max_margin
digital_gain	digital_gain_min	digital_gain_max
digital_gain is an integer multiple of digital_gain_step_size		
frame_length_lines	min_frame_length_lines	max_frame_length_lines
line_length_pck	min_line_length_pck	max_line_length_pck
line_length_pck	((x_addr_end – x_addr_start + 1) / xskip) + min_line_blanking_pck	
frame_length_lines	((y_addr_end – y_addr_start + 1) / yskip) + min_frame_blanking_lines	
x_addr_start	x_addr_min	x_addr_max
x_addr_end	x_addr_start	x_addr_max
(x_addr_end – x_addr_start + 1)	must be positive	must be positive
x_addr_start[0]	0	0
x_addr_end[0]	1	1
y_addr_start	y_addr_min	y_addr_max
y_addr_end	y_addr_start	y_addr_max
(y_addr_end – y_addr_start + 1)	must be positive	must be positive
y_addr_start[0]	0	0
y_addr_end[0]	1	1
x_even_inc	min_even_inc	max_even_inc
x_even_inc[0]	1	1
y_even_inc	min_even_inc	max_even_inc
y_even_inc[0]	1	1
x_odd_inc	min_odd_inc	max_odd_inc
x_odd_inc[0]	1	1
y_odd_inc	min_odd_inc	max_odd_inc
y_odd_inc[0]	1	1
x_output_size	256 (this is enforced in hardware: values lower than this are treated as 256)	
x_output_size[0]	0 (this is enforced in hardware: bit[0] is read-only)	0

Table 15: Programming Rules (continued)

Parameter	Minimum Value	Maximum Value
y_output_size	2	frame_length_lines
y_output_size[0]	0 (this is enforced in hardware: bit[0] is read-only)	0
with subsampling, start and end pixels must be addressed (impact on x/y start/ end addresses, function of image orientation bits)		

Output Size Restrictions

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

- When 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 serial 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, 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).

Changing Registers While Streaming

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

- serial_channel_identifier
- pre_pll_clk_div
- pll_multiplier
- vt_sys_clk_div

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 on this signal.

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

One of the GPI can be used to perform the SADDR function. There is no dedicated pin.

This interface is described in detail in "Two-Wire Serial Register Interface" on page 11.

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 is designed to the MIPI 0.65 D-PHY requirements.

The DATA_P, DATA_N, CLK_P, and CLK_N signals go into ULP (Ultra Low Power) mode if the serial disable bit is asserted (R0x301A–B[12] = 1) or when the sensor is in the standby state. When the sensor is in the streaming state these signals will indicate LP-01 state between frames.

System States

The system states of the MT9V013 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 16.

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 16.

Figure 16: MT9V013 System States

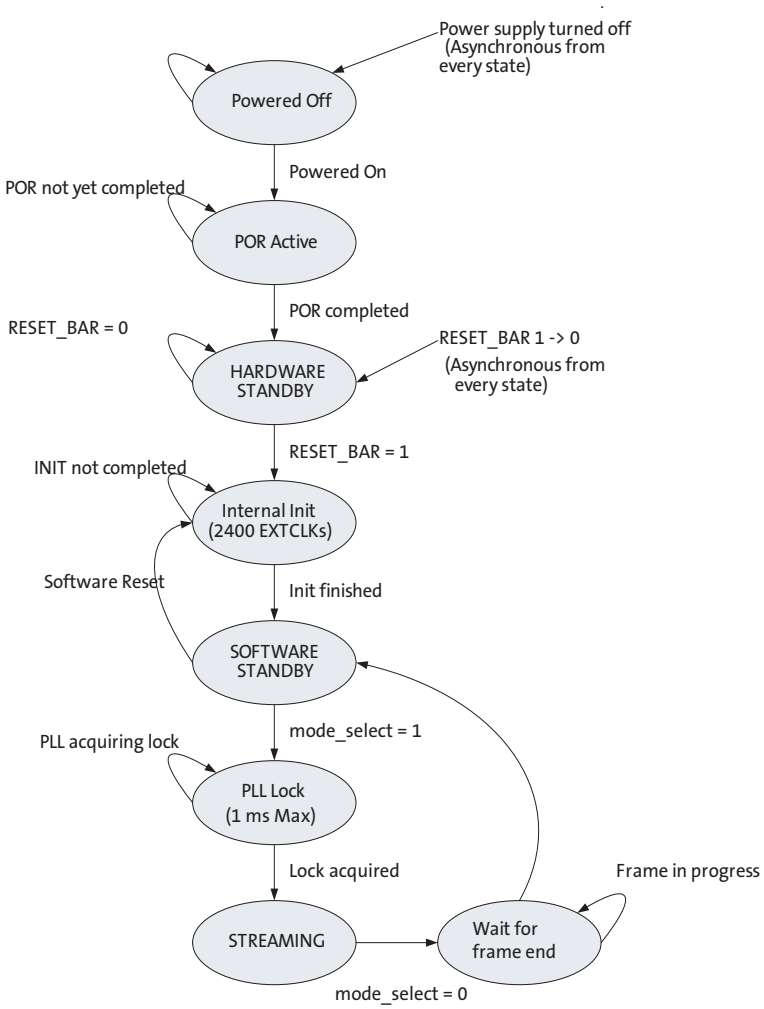


Table 16: PLL in System States

State	PLL
Powered Off	VCO powered down
POR Active	VCO powered down
Hardware Standby	VCO powered down
Internal Init	VCO powered down
Software Standby	VCO powered down
PLL Lock	VCO powering up and locking, PLL output bypassed
Streaming	VCO running, PLL output active

Power-On Reset Sequence

When power is applied to the MT9V013, it enters a low-power hardware standby state. Exit from this state is controlled by the later of two events:

- The negation of the RESET_BAR input.
- A time out 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.

While RESET_BAR is asserted (or the internal power-on reset circuit is active), the MT9V013 is in its lowest-powered, powered-up state; the internal PLL is disabled, the 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 2,400 EXTCLK cycles. After this time it enters a low-power software standby state. While the initialization sequence is in progress, the MT9V013 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, R0x0000–1. 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 (0x2200 if R0x0000–1 is read).

When the sensor leaves software standby mode and enables the VCO, an internal lock timer will delay going into streaming state, until the PLL has locked.

Software Reset Sequence

The MT9V013 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 17 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during software standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 17: Signal State During Reset

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Enabled, must be driven to a valid logic level.	
RESET_BAR (XSHUTDOWN)	Input	Enabled, must be driven to a valid logic level.	
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.
DATA_P	Output	Logic 0.	
DATA_N	Output		
CLK_P	Output		
CLK_N	Output		
GPI(1:0), SADDR	Input	Powered down. Can be left disconnected/floating.	
TEST	Input	Enabled, .must be driven to logic 1.	

General Purpose Inputs

The MT9V013 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
- Trigger (see the sections below)
- Standby functions (see the following sections)
- SADDR

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

Streaming/Standby Control

The MT9V013 can be switched between its software standby and streaming states under pin or register control, as shown in Table 18. Selection of a pin to use for the STANDBY function is described in "General Purpose Inputs" on page 35. The state diagram for transitions between software standby and streaming states is shown in Figure 16 on page 33.

Table 18: Streaming/STANDBY

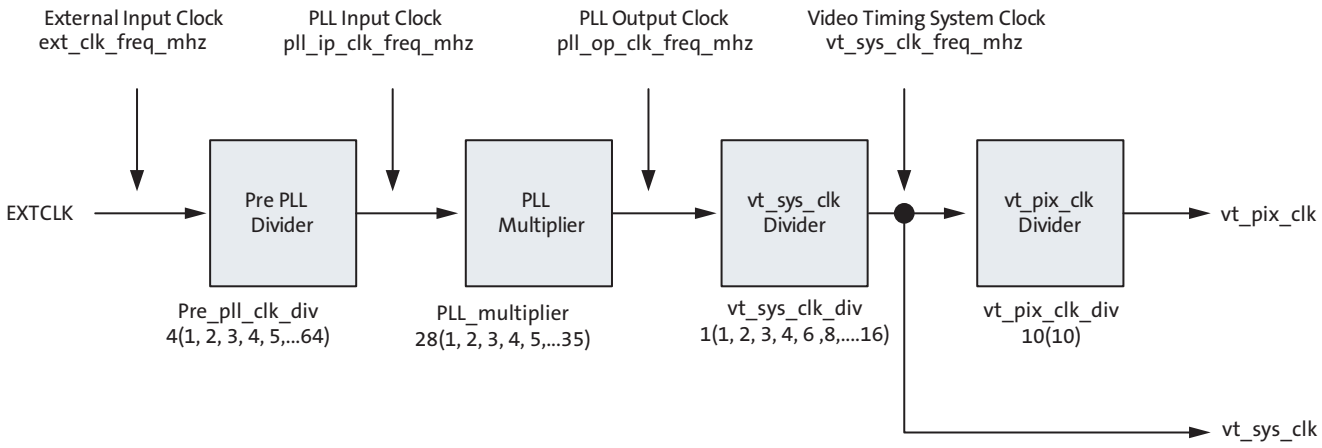
STANDBY	Streaming R0x301A–B[2]	Description
Disabled	0	Software standby
Disabled	1	Streaming
X	0	Software standby
0	1	Streaming
1	X	Software standby

Clocking

The MT9V013 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.

Figure 17 shows the different clocks and the names of the registers that contain or are used to control their values. The figure shows the default setting for each divider/multiplier control register and the range of legal values for each divider/multiplier control register.

Figure 17: MT9V013 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
- 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 PLL input clock frequency range, after the pre-PLL divider stage, is 4.0–17.5 MHz. The op_clk is in the range of 70–140 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 serial pixel data output

The output clock frequencies can be calculated as:

$$vt\_pix\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz * pll\_multiplier}{pre\_pll\_clk\_div * vt\_sys\_clk\_div * 10} \quad \text{(EQ 3)}$$

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

Programming the PLL Divisors

The PLL divisors should be programmed while the MT9V013 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.

The internal lock timer will delay entering STREAMING state by a maximum of 1ms, this is to allow the PLL to lock.

The effect of programming the PLL divisors while the MT9V013 is in the STREAMING state is UNDEFINED.

Clock Control

The MT9V013 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 clocked only when required.

When the MT9V013 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

Image Acquisition Mode

The MT9V013 supports electronic rolling shutter (ERS) mode. When the MT9V013 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 MT9V013 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 serial 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 640H x 480V 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 MT9V013 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 every other two rows/columns during readout and is equivalent to the skip2 readout mode provided by earlier Aptina Imaging sensors. The effect of the different subsampling settings on the pixel array readout is shown in Figures 18 through 21.

Figure 18: Pixel Readout (no subsampling)

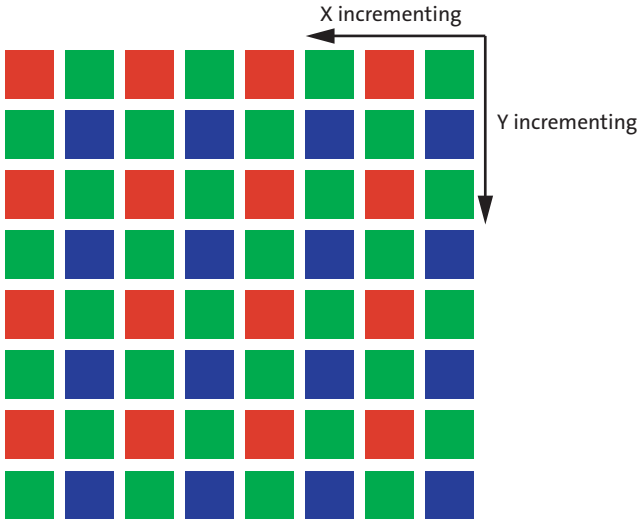


Figure 19: Pixel Readout (x_odd_inc = 3, y_odd_inc = 1)

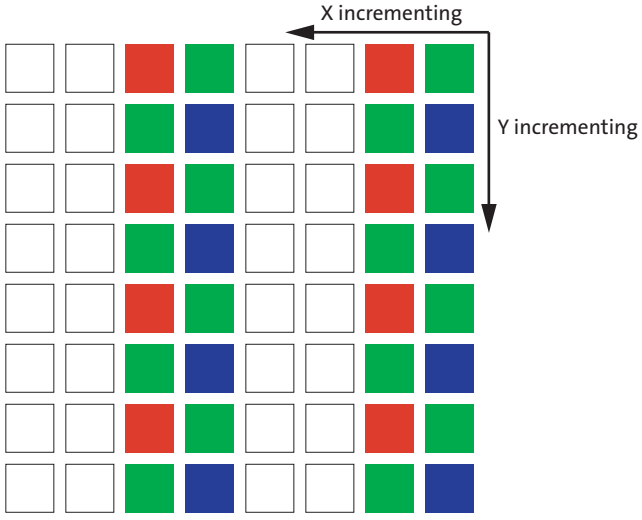


Figure 20: Pixel Readout (x_odd_inc =1, y_odd_inc = 3)

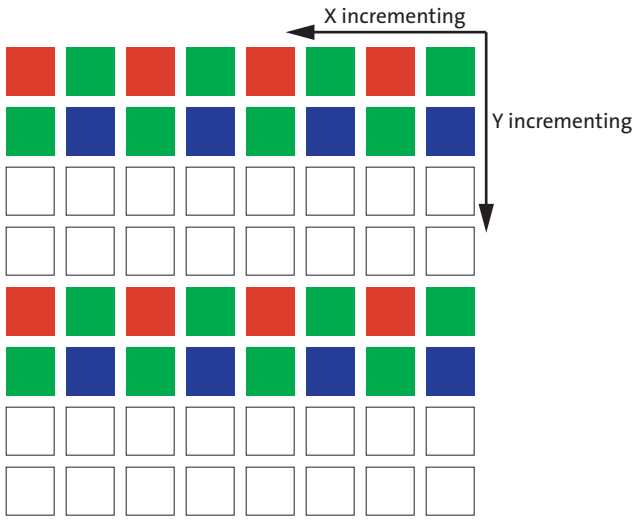
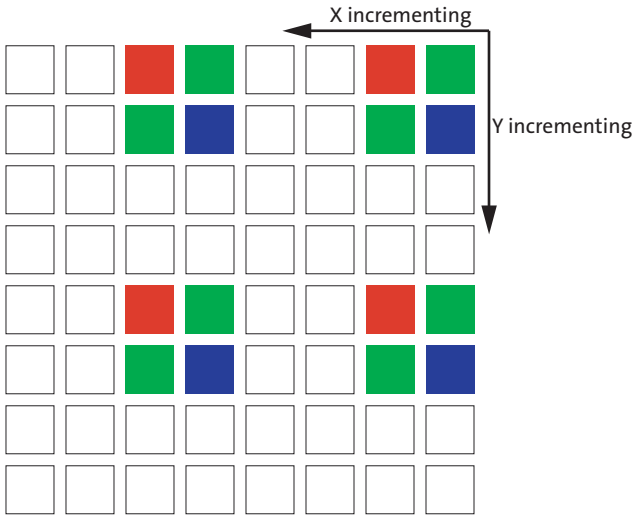


Figure 21: Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)



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 using the following rule:

$$remainder = (addr\_end - addr\_start + 1) \text{ AND } 0x0002 \quad \text{(EQ 5)}$$

$$if\ (remainder == 0)\ addr\_end = addr\_end - 2 \quad \text{(EQ 6)}$$

Table 19 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 19: 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 formulas for calculating the frame rate of the MT9V013 are shown below:

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

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

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

Frame Rates

Table 20 shows the maximum frame rates that can be achieved. The frame rates are shown both with subsampling disabled (full resolution) and with subsampling enabled (*x_odd_inc = 3, y_odd_inc = 3*). The frame rates assume a pixel rate of 14 Mp/s (*vt_pix_clk = 14 MHz*) and the minimum line blanking of 194 pixels/minimum frame blanking of 19 lines.

Table 20: Frame Rates

Image Size	Subsampling Disabled 10-Bit	Subsampling Enabled 10-Bit
VGA (640 x 480)	32.795 fps (including 4-pixel border) line blank = 194 frame blank = 19 1 frame = (842 * 507) / 14e6 = 30.49ms fps = 1 / 29.88ms = 32.79	102.17 fps (including border)

Valid Data Signal Options

Integration Time

The integration (exposure) time of the MT9V013 is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

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

The limits for the coarse integration time are defined by:

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

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:

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

With default settings and a vt_pix_clk of 14 MHz, the maximum integration time that can be achieved without reducing the frame rate is given by:

$$Maximum\ integration\ time\ [sec] = \frac{((0x1FB\text{-}1) * 0x34A) + 0x6B)}{(14\ MHz * 1*10^{6})} = 30.43ms \quad \text{(EQ 13)}$$

It is fundamental to the operation of an ERS that an integration time should not be programmed to be greater than the frame time. However, setting an integration time greater than the frame time increases the frame time beyond frame_length_lines, which will to make longer exposure times available.

Flash Control

The MT9V013 supports both xenon and LED flash through the FLASH output signal. The timing of the FLASH signal with the default settings is shown in Figures 22, 23, and 24 on page 43. 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.

The flash output can be inverted by asserting R0x3046–7[7] as shown in Figure 17. The default setting is R0x3046–7[7] = 0.

Figure 22: Xenon Flash Enabled

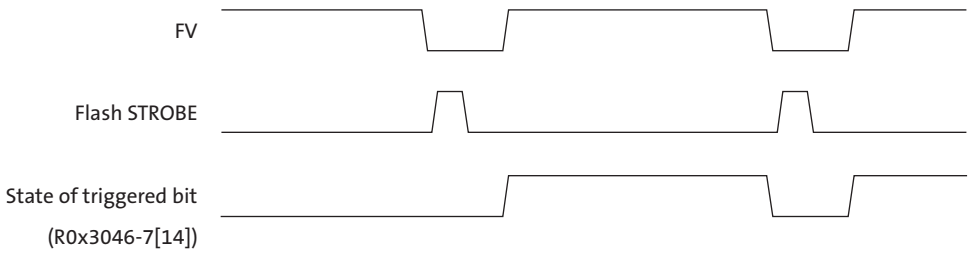
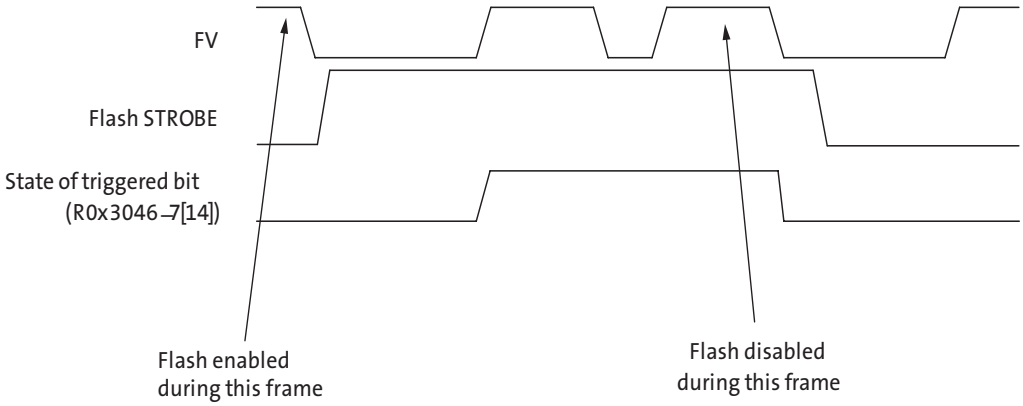
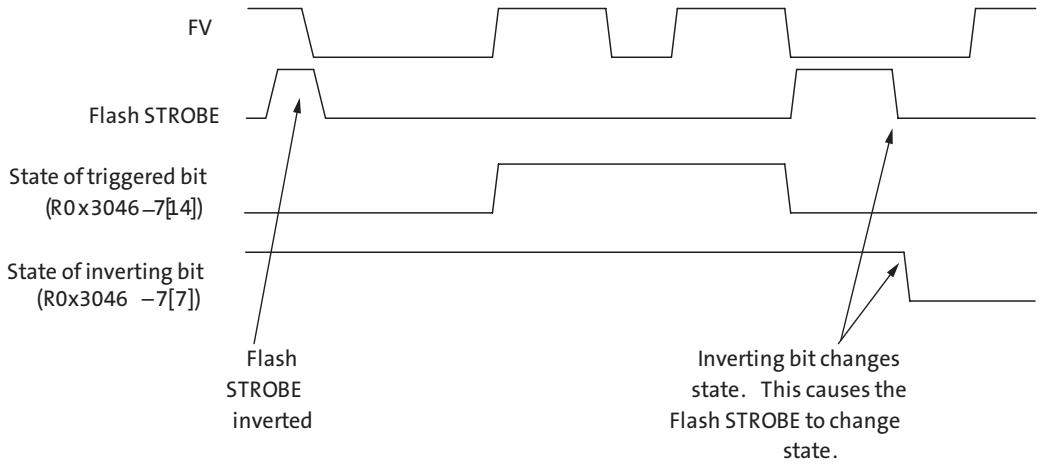


Figure 23: LED Flash Enabled



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

Figure 24: LED Flash Using Inverted Strobe



Analog Gain

The MT9V013 provides two mechanisms for setting the analog gain. The following sections describe both models, describe the mapping between the models and describe the operation of the per-color and global gain control.

Using Per-Color or Global Gain Control

The read-only analog_gain_capability register returns a value of "1," indicating that the MT9V013 provides per-color gain control. However, the MT9V013 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.

First Gain Model

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

- analog_gain_code_global
- analog_gain_code_greenR
- analog_gain_code_red
- analog_gain_code_blue
- analog_gain_code_greenB

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

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

Second Gain Model

The second 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 6-bit fine gain stage [5:0] and a gain doubling stage [7:6]. As a result, the step size varies depending upon whether the gain doubling bits [7:6] are enabled.

The analog gain is given by:

$$gain = \left(2 \wedge (<color>\_gain[7:6]) \times \frac{<color>\_gain[5:0]}{32}\right) \quad \text{(EQ 15)}$$

where
[7:6] = 00 -> 1x
[7:6] = 01 -> 2x
[7:6] = 10 -> 4x
[7:6] = 11 -> 8x
[5:0] = at least 0x0020 -> 1x gain

Gain Code Mapping

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

When the first gain model is in use and values have been written to the analog_gain_code_<color> registers, the associated value in the second gain model can be read from the associated <color>_gain register.

When the second gain model is in use and values have been written to the <color>_gain registers, data read from the associated analog_gain_code_<color> register is UNDEFINED. The reason for this is that many of the gain codes available in the second gain model have no corresponding value in the first 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 MT9V013 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 21.

Table 21: 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 1–3 replace pixel data in the output image . Test pattern 4 replaces all data in the output image .

For all of the test patterns, the MT9V013 registers must be set appropriately to control the frame rate and output timing. These include:

- 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 (the effect of sub-sampling is undefined for test patterns). These include:

- Noise cancellation
- Black pedestal adjustment
- Dark current compensation
- Black level

These effects can be eliminated by the following register settings:

- R0x3044–5[10]=0
- R0x30C0–1[0] = 0
- R0x301A–B[3]= 0 (enable writes to data pedestal)
- R0x301E–F = 0x0000 (set data pedestal to “0”)
- R0x3180–81[15] = 0 (disable dark current tracking)
- R0x30C2–30C9 = 0x0000

Solid Color Test Pattern

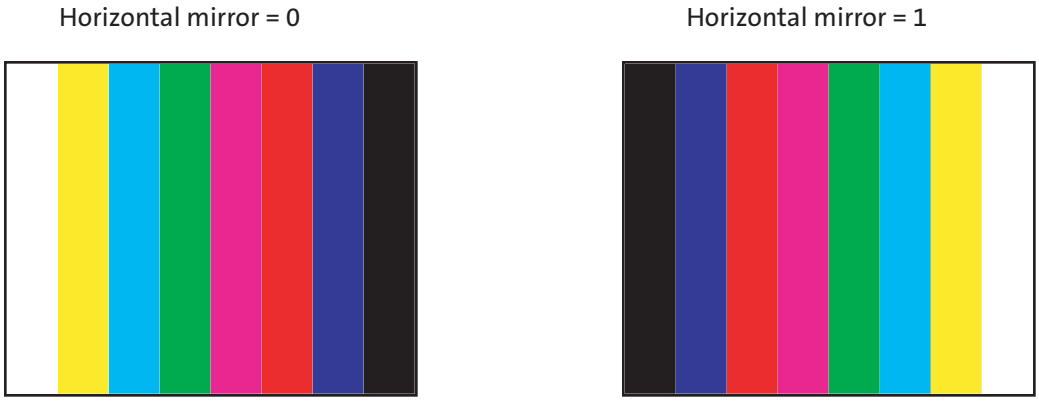
In this mode, all pixel data is replaced by fixed Bayer pattern test data. The intensity of each pixel is set by its associated test data register (test_data_red, test_data_greenR, test_data_blue, test_data_greenB).

100 Percent Color Bars Test Pattern

In this test pattern, shown in Figure 25, 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 80 pixels wide, if the same 8-color bar chart is displayed across VGA 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*80 = 640 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 80 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-hand side of the output image. Any pattern repeat occurs at the right-hand side of the output image regardless of the setting of horizontal_mirror. The state of vertical_flip has no effect on this test pattern.

Figure 25: 100 Percent Color Bars Test Pattern



Fade-to-Gray Color Bars Test Pattern

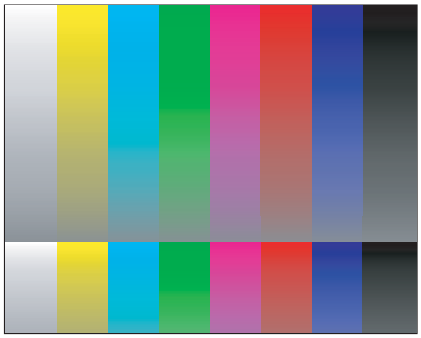
In this test pattern, shown in Figure 26 on page 48, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, black). Each bar is 80 pixels wide and occupies 480 rows of the output image. Each color bar fades vertically from full intensity at the top of the image to 50 percent intensity (mid-gray) on the 256th 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*80 = 640 pixels and vertically after 256 rows. 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 80 pixels, if all 8 color bars are to be displayed.

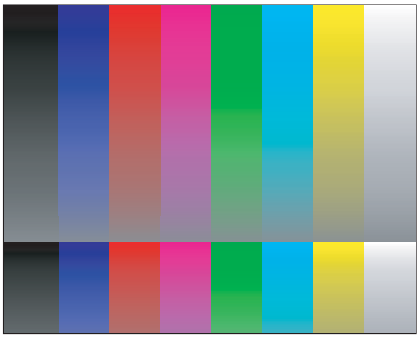
The effect of setting horizontal_mirror or vertical_flip in conjunction with this test pattern is that the order in which the colors are generated is reversed. The black bar appears at the left-hand side of the output image. Any pattern repeat occurs at the right-hand side of the output image regardless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

Figure 26: Fade-to-Gray Color Bars Test Pattern

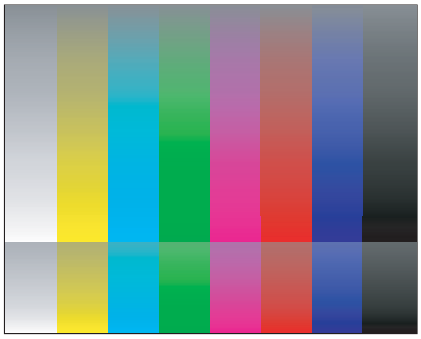
Horizontal mirror = 0, Vertical flip = 0



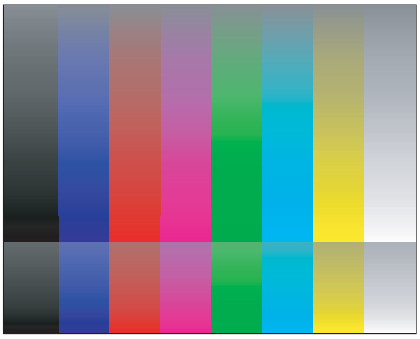
Horizontal mirror = 1, Vertical flip = 0



Horizontal mirror = 0, Vertical flip = 1



Horizontal mirror = 1, Vertical flip = 1



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 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 MT9V013 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.

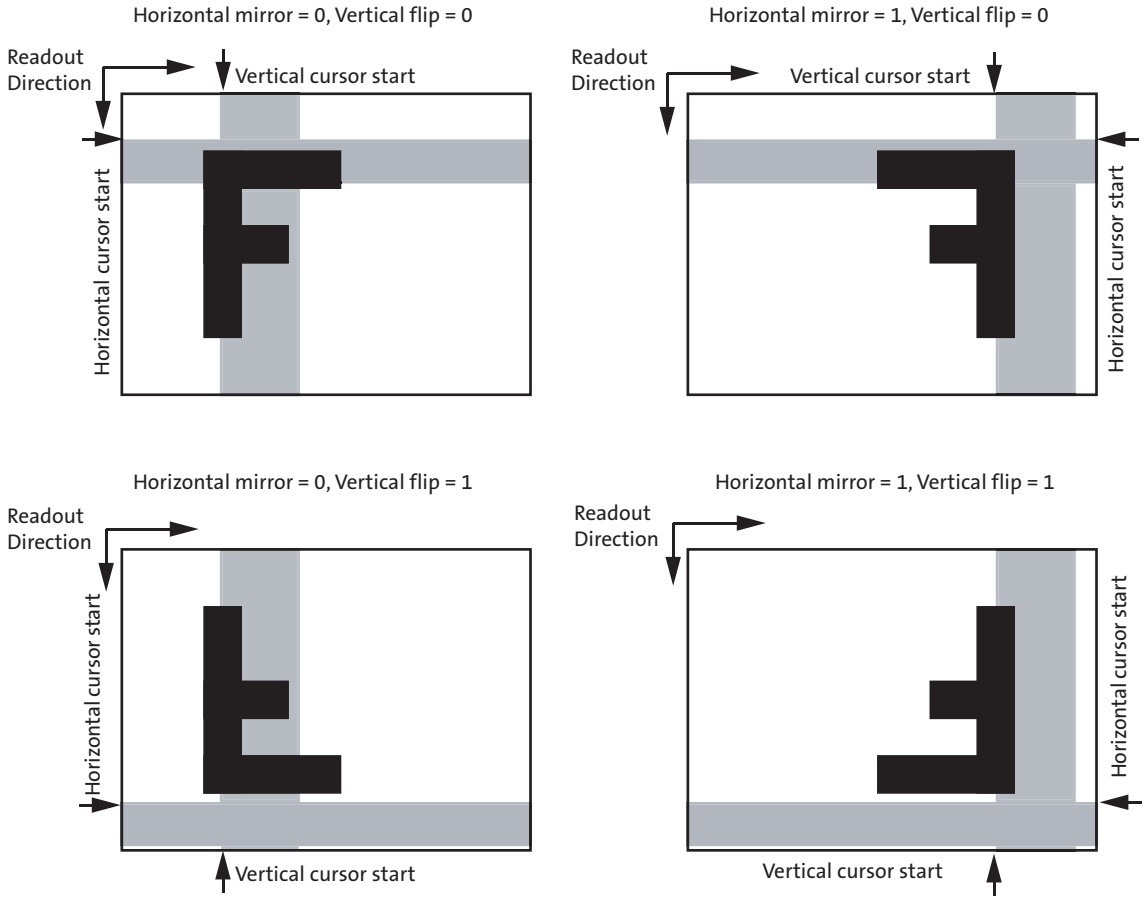
The cursors are opaque (they replace data from the imaged scene or test pattern). The color of each cursor is set by the values of the Bayer components in the test_data_red, test_data_greenR, test_data_blue, and test_data_greenB registers. As a consequence, the cursors are the same color as test pattern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

When vertical_cursor_position = 0x03FF, the vertical cursor operates in an automatic mode in which its position advances every frame. In this mode, the cursor starts at the column. 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 design specification. The behavior of the MT9V013 is shown in Figure 27 on page 50. 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 27: 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 the value from R0x301E–F[10:0] (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."

2.

3.

Revision History

Rev. D 6/17/10

- Updated to Aptina template

Rev. C 1/23/08

- Updated document from Preliminary to Production
- Added Figure 12: “Chief Ray Angle (CRA) vs. Image Height,” on page 17
- Updated Table 10, “Power-On-Reset Characteristics,” on page 23
- Updated Figure 14: “Two-Wire Serial Bus Timing Parameters,” on page 26

Rev. B 07/17/07

- Updated Table 8, “Electrical Characteristics,” on page 21
- Updated Table 13, “Two-Wire Serial Bus Characteristics,” on page 27
- Updated Table 12, “Absolute Maximum Values,” on page 25
- Updated Table 11, “DC Electrical Definitions and Characteristics,” on page 24
- Updated Table 9, “AC Electrical Characteristics (Control Interface),” on page 22
- Updated Figure 14: “Two-Wire Serial Bus Timing Parameters,” on page 26
- Updated Figure 11: “Quantum Efficiency,” on page 16
- Upgrade document from Advance to Preliminary

Rev. A 11/20/06

- Initial release

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