

1/4-Inch 2Mp CMOS Digital Image Sensor

MT9D014

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

Features

- DigitalClarity[®] CMOS imaging technology
- Low dark current
- Simple two-wire serial interface
- Auto black level calibration
- Programmable controls: gain, frame size/rate, exposure, left-right and top-bottom image reversal, window size and panning
- Data interface: CCP2 compliant sub-low-voltage differential signalling (sub-LVDS)
- SMIA-compatible
- On-chip phase-locked loop (PLL) oscillator
- Bayer-pattern down-size scaler
- Integrated lens shading correction
- Superior low-light performance
- Internal power switch for ultra-low standby current consumption
- 30 fps at full resolution

Applications

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

General Description

The Aptina Imaging MT9D014 is a 1/4-inch UXGA-format CMOS active-pixel digital image sensor with a pixel array of 1600H x 1200V (1608H x 1208V including border pixels). It incorporates sophisticated on-chip camera functions such as windowing, mirroring, column and subsampling modes. It is programmable through a simple two-wire serial interface and has very low power consumption.

Table 1: Key Performance Parameters

| Parameter | | Value |
|-----------------------|--------------------|--|
| Optical format | | 1/4-inch UXGA (4:3) |
| Active imager size | | 3.538mm(H) x 2.658mm(V) 4.425mm diagonal |
| Active pixels | | 1608H x 1208V |
| Pixel size | | 2.2 x 2.2µm |
| Color filter array | | RGB Bayer pattern |
| Shutter type | | Electronic rolling shutter (ERS) |
| Maximum data rate | | 640 Mb/s at CCP2 interface |
| Frame rate | UXGA (1600 x 1200) | Programmable up to 24.04 fps in profile 0 mode (RAW10) Programmable up to 30.05 fps in profile 1/2 mode (RAW10) |
| | XGA (1024 x 768) | Programmable up to 75.18 fps in profile 0 mode (RAW10) Programmable up to 93.98 fps in profile 1/2 mode (RAW10) |
| ADC resolution | | 10-bit, on-chip (61dB) |
| Responsivity | | 1.38 V/lux-sec |
| Dynamic range | | 68.3B |
| SNR _{MAX} | | 38.72dB |
| Supply voltage | Analog | 2.40–2.90V (2.80V nominal) |
| | Digital | 1.70–1.90V (1.80V nominal) |
| Power consumption | | 220mW at 30 fps (TYP) |
| Operating temperature | | –30°C to +70°C |
| Packaging | | Bare die |

Ordering Information

Table 2: Available Part Numbers

| Part Number | Description |
|----------------------|-------------|
| MT9D014D00STC C15BC1 | Bare die |

Table of Contents

| | |
|--|----|
| Features | 1 |
| Applications | 1 |
| General Description | 1 |
| Ordering Information | 1 |
| General Description | 6 |
| Functional Overview | 6 |
| Pixel Array | 7 |
| Operating Modes | 8 |
| Camera Module Integrator Identification Procedure | 9 |
| Signal Descriptions | 10 |
| Two-Wire Serial Register Interface | 11 |
| Protocol | 11 |
| Start Condition | 11 |
| Slave Address/Data Direction Byte | 11 |
| Acknowledge Bit | 11 |
| No-Acknowledge Bit | 11 |
| Stop Condition | 12 |
| Data Transfer | 12 |
| Typical Sequence | 12 |
| Single READ from Random Location | 13 |
| Single READ from Current Location | 13 |
| Sequential READ, Start from Random Location | 13 |
| Sequential READ, Start from Current Location | 14 |
| Single WRITE to Random Location | 14 |
| Sequential WRITE, Start at Random Location | 14 |
| Embedded Data Format and Control | 15 |
| Programming Restrictions | 21 |
| Output Size Restrictions | 22 |
| Effect of Scaler on Legal Range of Output Sizes | 22 |
| Effect of CCP2 Class on Legal Range of Output Sizes/Frame Rate | 24 |
| Output Data Timing | 25 |
| Changing Registers while Streaming | 25 |
| Control of the Signal Interface | 26 |
| Serial Register Interface | 26 |
| Default Power-Up State | 26 |
| Serial Pixel Data Interface | 27 |
| System States | 28 |
| Power-On Reset Sequence | 29 |
| Soft Reset Sequence | 29 |
| Signal State During Reset | 30 |
| General Purpose Inputs | 30 |
| Streaming/Standby Control | 31 |
| Clocking | 32 |
| Profile 0 Behavior | 32 |
| Programming the PLL Divisors | 35 |
| Influence of ccp_data_format | 35 |
| Influence of ccp2_signalling_mode | 36 |
| Features | 37 |
| Shading Correction (SC) | 37 |
| The Correction Function | 37 |
| Image Acquisition Modes | 37 |



| | |
|--|-----|
| Window Control | .38 |
| Pixel Border | .38 |
| Readout Modes | .38 |
| Horizontal Mirror | .38 |
| Vertical Flip | .38 |
| Subsampling | .39 |
| Frame Rate Control | .42 |
| Integration Time | .42 |
| Analog Gain | .43 |
| Using Per-color or Global Gain Control | .43 |
| SMIA Gain Model | .43 |
| Aptina Imaging Gain Model | .43 |
| Gain Code Mapping | .44 |
| Sensor Core Digital Data Path | .45 |
| Test Patterns | .45 |
| Solid Color Test Pattern | .45 |
| 100 Percent Color Bars Test Pattern | .45 |
| Fade-to-Gray Color Bars Test Pattern | .46 |
| PN9 Link Integrity Pattern | .47 |
| Test Cursors | .48 |
| Digital Gain | .49 |
| Pedestal | .49 |
| Digital Data Path | .50 |
| Timing Specifications | .51 |
| Power-Up Specifications | .51 |
| Power-Up Sequence | .51 |
| Power-Down Specification | .52 |
| Power-Down Sequence | .52 |
| Hard Standby and Hard Reset | .54 |
| Soft Standby and Soft Reset | .54 |
| Soft Standby | .54 |
| Soft Reset | .54 |
| Electrical Specifications | .55 |
| EXTCLK | .55 |
| Two-Wire Serial Register Interface | .55 |
| Serial Pixel Data Interface | .56 |
| Control Interface | .56 |
| Power-On Reset | .57 |
| Operating Voltages | .58 |
| Absolute Maximum Ratings | .59 |
| SMIA Specification Reference | .59 |
| Revision History | .60 |

List of Figures

| | | |
|------------|--|----|
| Figure 1: | Block Diagram | 6 |
| Figure 2: | Pixel Color Pattern Detail (Top Right Corner) | 7 |
| Figure 3: | Typical Configuration: Serial Pixel Data Interface | 8 |
| Figure 4: | Single READ from Random Location | 13 |
| Figure 5: | Single READ from Current Location | 13 |
| Figure 6: | Sequential READ, Start from Random Location | 13 |
| Figure 7: | Sequential READ, Start from Current Location | 14 |
| Figure 8: | Single WRITE to Random Location | 14 |
| Figure 9: | Sequential Write, Start at Random Location | 14 |
| Figure 10: | Effect of Limiter on the SMIA Data Path | 23 |
| Figure 11: | Timing of SMIA Data Path | 24 |
| Figure 12: | MT9D014 System States | 28 |
| Figure 13: | MT9D014 SMIA Profile 1/2 Clocking Structure | 33 |
| Figure 14: | MT9D014 SMIA Profile 0 Clocking Structure | 34 |
| Figure 15: | Effect of horizontal_mirror on Readout Order | 38 |
| Figure 16: | Effect of vertical_flip on Readout Order | 38 |
| Figure 17: | Effect of x_odd_inc = 3 on Readout Sequence | 39 |
| Figure 18: | Pixel Readout (No Subsampling) | 39 |
| Figure 19: | Pixel Readout (x_odd_inc = 3, y_odd_inc = 3) | 40 |
| Figure 20: | 100 Percent Color Bars Test Pattern | 46 |
| Figure 21: | Fade-to-Gray Color Bars Test Pattern | 47 |
| Figure 22: | Test Cursor Behavior when image_orientation | 49 |
| Figure 23: | Data Path | 50 |
| Figure 24: | Power-Up Sequence | 51 |
| Figure 25: | Power-Down Sequence | 52 |
| Figure 26: | Soft Standby and Soft Reset | 54 |
| Figure 27: | Internal Power-On Reset | 57 |



List of Tables

| | | |
|-----------|---|----|
| Table 1: | Key Performance Parameters | 1 |
| Table 2: | Available Part Numbers | 1 |
| Table 3: | Signal Descriptions | 10 |
| Table 4: | Embedded Data | 15 |
| Table 5: | Definitions for Programming Rules | 21 |
| Table 6: | Programming Rules | 21 |
| Table 7: | PLL in System States | 29 |
| Table 8: | Signal State During Reset | 30 |
| Table 9: | Streaming/STANDBY | 31 |
| Table 10: | Row Address Sequencing | 41 |
| Table 11: | Test Patterns | 45 |
| Table 12: | Power-Up Sequence | 52 |
| Table 13: | Power-Down Sequence | 53 |
| Table 14: | Electrical Characteristics (EXTCLK) | 55 |
| Table 15: | Two-Wire Serial Register Interface Electrical Characteristics | 55 |
| Table 16: | Electrical Characteristics (Serial Pixel Data Interface) | 56 |
| Table 17: | Electrical Characteristics (Control Interface) | 56 |
| Table 18: | Power-On Reset Characteristics | 57 |
| Table 19: | DC Electrical Definitions and Characteristics | 58 |
| Table 20: | Absolute Maximum Values | 59 |

General Description

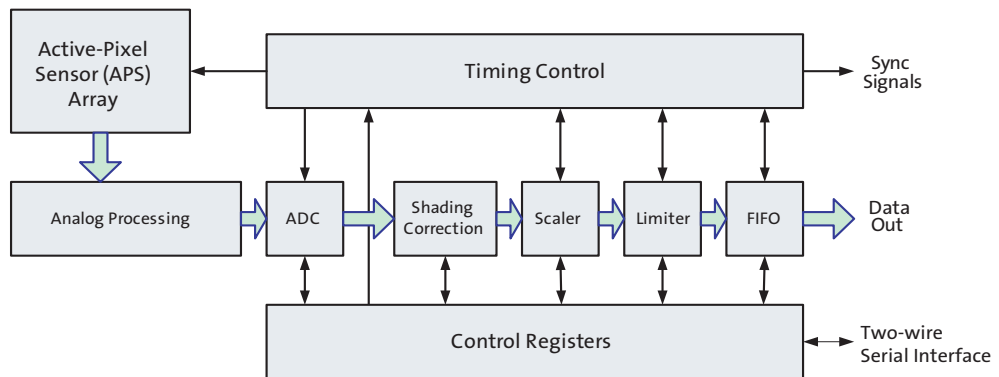
The MT9D014 digital image sensor features DigitalClarity—Aptina’s breakthrough low noise CMOS imaging technology that achieves near-CCD image quality (based on signal-to-noise ratio and low-light sensitivity) while maintaining the inherent size, cost, and integration advantages of CMOS.

When operated in its default mode, the sensor generates a UXGA image at 24.04 frames per second (fps) when `ext_clk_freq_mhz = 16 MHz`. An on-chip analog-to-digital converter (ADC) generates a 10-bit value for each pixel.

Functional Overview

The MT9D014 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 80 Mp/s, corresponding to a video timing pixel clock rate of 80 MHz. A block diagram of the sensor is shown in Figure 1.

Figure 1: Block Diagram



The core of the sensor is a 2Mp 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 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 on page 6 are partitioned into three logical parts:

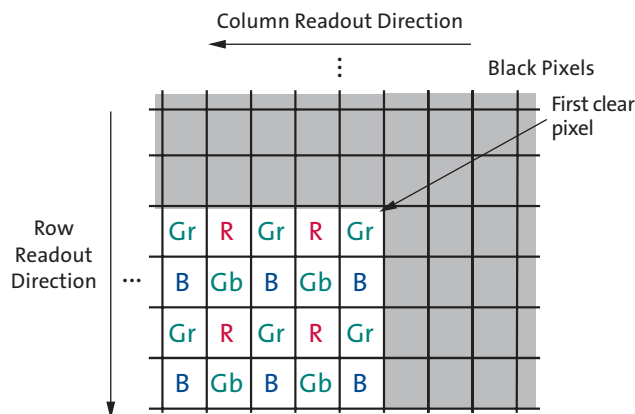
- A sensor core that provides array control and data path corrections.
- A digital shading correction block to compensate for color/brightness shading introduced by the lens or CRA curve mismatch.
- Functionality to support the SMIA standard. This includes a horizontal and vertical image scaler, a limiter, a data compressor, an output FIFO, and a serializer.

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

Pixel Array

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

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

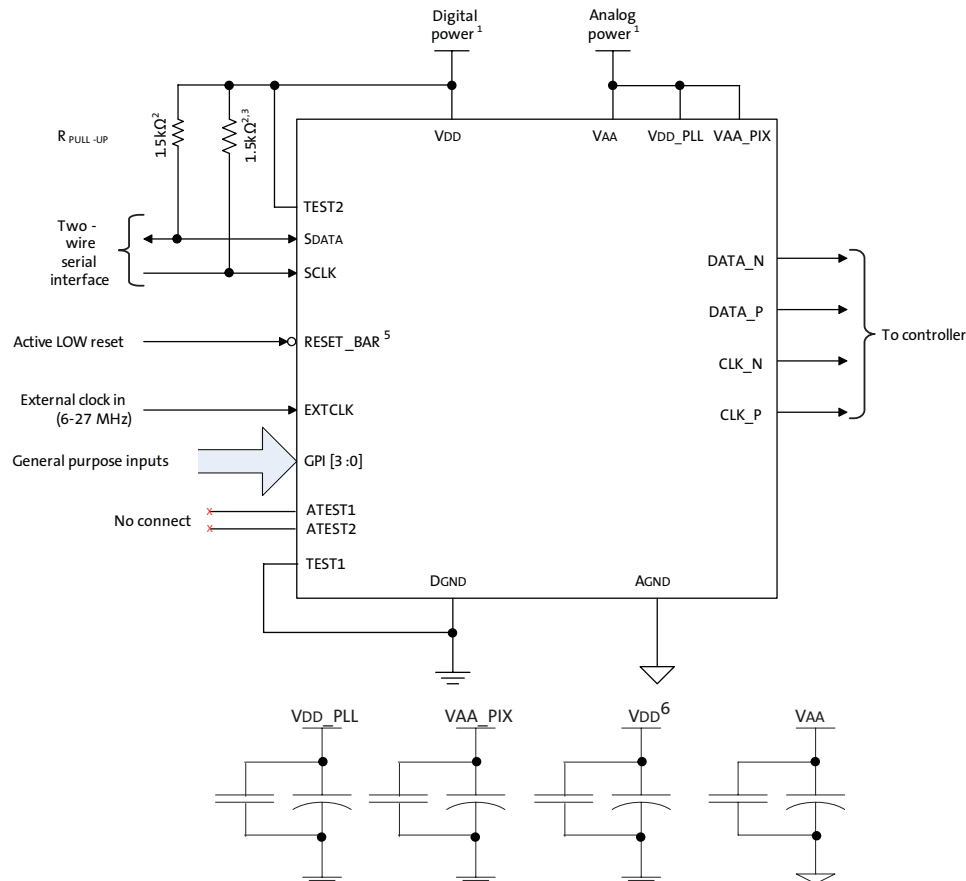


Operating Modes

The MT9D014 is a SMIA-compatible sensor with a CCP2 interface output data path as defined by SMIA.

For low-noise operation, the MT9D014 requires separate power supplies for analog and digital. Incoming digital and analog ground conductors can be tied together next to the die. Both power supply rails should be decoupled from ground using capacitors as close as possible to the die. The use of inductance filters is not recommended on the power supplies or output signals.

Figure 3: Typical Configuration: Serial Pixel Data Interface



- Notes:
1. All power supplies should be adequately decoupled.
 2. Aptina recommends a resistor value of 1.5k Ω , but a greater value may be used for slower two-wire speed.
 3. This pull-up resistor is not required if the controller drives a valid logic level at all times.
 4. TEST1 must be tied to GND. TEST2 must be tied to VDD.
 5. Also referred to as XSHUTDOWN.
 6. The GPI pins can be statically pulled HIGH or LOW and used as module IDs, or they can be programmed to perform special functions (SADDR, STANDBY) and be dynamically controlled. After reset, these pads are powered up (enabled). Failure to bond as required will result in excessive power consumption.
 7. Aptina recommends that 0.1 μ F and 1 μ F decoupling capacitors for each power supply are mounted as close as possible to the pad. Actual values and results may vary depending on layout and design considerations.



Camera Module Integrator Identification Procedure

See MT9D014 Developer Guide.

Signal Descriptions

Table 3 provides signal descriptions for MT9D014 die. For pad location and aperture information, refer to the MT9D014 die data sheet.

Table 3: Signal Descriptions

| Pad Name | Pad Type | Description |
|--------------------------|----------|---|
| EXTCLK | Input | Master clock input. PLL input clock. 6–27 MHz. |
| RESET_BAR (XSHUTDOWN) | Input | Asynchronous active LOW reset. When asserted, data output stops and all internal registers are restored to their factory default settings. |
| SCLK | Input | Serial clock for access to control and status registers. |
| GPI[3:0] | Input | General purpose inputs. After reset, these pads are powered up (enabled—see R0x301A) by default; these pads must be bonded to a HIGH or LOW state. Any of these pads can be configured to provide hardware control of the standby, output enable, and shutter trigger functions. Failure to bond as required will result in excessive power consumption. |
| TEST1 | Input | Enable manufacturing test modes. Wire to digital GND for functional operation. |
| TEST2 | Input | Tie HIGH for normal operation. |
| SDATA | I/O | Serial data for reads from and writes to control and status registers. |
| DATA_P | Output | Differential CCP2 (sub_lvds) serial 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). |
| VAA | Supply | Analog power supply. |
| VDD_PLL | Supply | PLL power supply. |
| VAA_PIX | Supply | Analog power supply. |
| AGND | Supply | Analog ground. |
| VDD | Supply | Digital power supply. |
| DGND | Supply | Digital ground. |

Two-Wire Serial Register Interface

The two-wire serial interface bus enables read/write access to control and status registers within the sensor. This interface is designed to be compatible with the SMIA 1.0 Part 2: CCP2 Specification camera control interface (CCI), which uses the electrical characteristics and transfer protocols of the I²C specification. The protocols described in the I²C specification allow the slave device to drive SCLK LOW; the sensor uses SCLK as an input only and therefore never drives it LOW.

Protocol

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

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

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

Start Condition

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

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 MT9D014 are 0x20 (write address) and 0x21 (read address) in accordance with the SMIA specification. Alternate slave addresses of 0x30 (write address) and 0x31 (read address) can be selected by enabling and asserting the SADDR signal through the GPI pad.

An alternate slave address can also be programmed through R0x31FC.

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.

Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data. The protocol used is outside the scope of the SMIA CCI.

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.

Typical Sequence

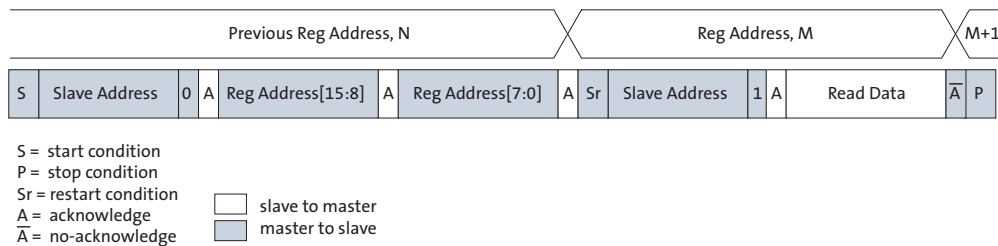
A typical READ or WRITE sequence begins by the master generating a start condition on the bus. After the start condition, the master sends the 8-bit slave address/data direction byte. The last bit indicates whether the request is for a READ or a WRITE, where a “0” indicates a WRITE and a “1” indicates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

If the request was a READ, the master sends the 8-bit write slave address/data direction byte and 16-bit register address, just as in the write request. The master then generates a (re)start condition and the 8-bit READ slave address/data direction byte, and clocks out the register data, 8 bits at a time. The master generates an acknowledge bit after each 8-bit transfer. The slave’s internal register address is automatically 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 4) 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 1 byte of register data. The master terminates the READ by generating a no-acknowledge bit followed by a stop condition. Figure 4 shows how the internal register address maintained by the MT9D014 is loaded and incremented as the sequence proceeds.

Figure 4: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 5) performs a read using the current value of the MT9D014 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.

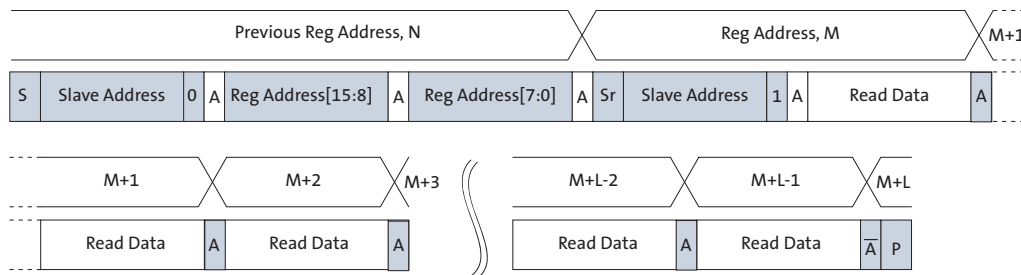
Figure 5: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 6) starts in the same way as the single READ from random location (Figure 4). 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 Random Location



Embedded Data Format and Control

When the serial pixel data path is selected, the first two rows of the output image contain register values that are appropriate for the image. In this mode, the first two lines and the last line of data are not equally spaced. The format of this data is shown in Table 4. In the table, 8-bit (RAW8) and 10-bit (RAW10) versions of the data are shown. The 10-bit format places the data byte in bits [9:2] and sets bits [1:0] to a constant value of 01. Register values that are shown as “?” are dynamic and may change from frame to frame.

When the parallel pixel data path is selected and R0x306E-F[2:0]=2 (parallel pixel data output MUX selects FIFO data). The output image contains two rows of embedded data.

Table 4: Embedded Data

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|---|--------|-------|--------------------------------|------------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 0 | 0x029 | 0x0A | | 2-byte tagged data format (embedded data) | 0x029 | 0x0A | | Start of embedded data line |
| 1 | 0x2a9 | 0xAA | | CCI register index MSB | 0x2a9 | 0xAA | | CCI register index MSB |
| 2 | 0x001 | 0x00 | | Address 00xx | 0x005 | 0x01 | | Address 01xx |
| 3 | 0x295 | 0xA5 | | CCI register index LSB | 0x295 | 0xA5 | | CCI register index LSB |
| 4 | 0x001 | 0x00 | | Address xx00 | 0x081 | 0x20 | | Address xx20 |
| 5 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | auto increment |
| 6 | 0x059 | 0x16 | 0000 | model_id hi | 0x001 | 0x00 | 120 | gain mode |
| 7 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 8 | 0x001 | 0x00 | 0001 | model_id lo | 0x009 | 0x02 | | Address 02xx |
| 9 | 0x169 | 0x5A | | | 0x295 | 0xA5 | | CCI register index LSB |
| 10 | 0x001 | 0x00 | 0002 | revision_number | 0x001 | 0x00 | | Address xx00 |
| 11 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 12 | 0x019 | 0x06 | 0003 | manufacturer_id | ?? | ?? | 0200 | fine_integration_time hi |
| 13 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 14 | 0x029 | 0x0A | 0004 | smia_version | ?? | ?? | 0201 | fine_integration_time lo |
| 15 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 16 | ?? | ?? | 0005 | frame_count | ?? | ?? | 0202 | coarse_integration_time hi |
| 17 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 18 | ?? | ?? | 0006 | pixel_order | ?? | ?? | 0203 | coarse_integration_time lo |
| 19 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 20 | 0x001 | 0x00 | | Address 00xx | ?? | ?? | 0204 | analogue_gain_code_global hi |
| 21 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 22 | 0x021 | 0x08 | | Address xx08 | ?? | ?? | 0205 | analogue_gain_code_global lo |
| 23 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 24 | ?? | ?? | 0008 | data_pedestal_hi | ?? | ?? | 0206 | analogue_gain_code_greenR hi |
| 25 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 26 | ?? | ?? | 0009 | data_pedestal lo | ?? | ?? | 0207 | analogue_gain_code_greenR lo |
| 27 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 28 | 0x001 | 0x00 | | Address 00xx | ?? | ?? | 0208 | analogue_gain_code_red hi |
| 29 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 30 | 0x101 | 0x40 | | Address xx40 | ?? | ?? | 0209 | analogue_gain_code_red lo |
| 31 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |



Table 4: Embedded Data (continued)

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|------------------------------|--------|-------|--------------------------------|------------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 32 | 0x005 | 0x01 | 0040 | frame_format_model_type | ?? | ?? | 020a | analogue_gain_code_blue hi |
| 33 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 34 | 0x049 | 0x12 | 0041 | frame_format_model_subtype | ?? | ?? | 020b | analogue_gain_code_blue lo |
| 35 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 36 | ?? | ?? | 0042 | frame_format_descriptor_0 hi | ?? | ?? | 020c | analogue_gain_code_greenB hi |
| 37 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 38 | ?? | ?? | 0043 | frame_format_descriptor_0 lo | ?? | ?? | 020d | analogue_gain_codegreenB lo |
| 39 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 40 | ?? | ?? | 0044 | frame_format_descriptor_1 hi | ?? | ?? | 020e | digital_gain_greenR hi |
| 41 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 42 | ?? | ?? | 0045 | frame_format_descriptor_1 lo | ?? | ?? | 020f | digital_gain_greenR lo |
| 43 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 44 | ?? | ?? | 0046 | frame_format_descriptor_2 hi | ?? | ?? | 0210 | digital_gain_red hi |
| 45 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 46 | ?? | ?? | 0047 | frame_format_descriptor_2 lo | ?? | ?? | 0211 | digital_gain_red lo |
| 47 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 48 | 0x001 | 0x00 | 0048 | frame_format_descriptor_3 hi | ?? | ?? | 0212 | digital_gain_blue hi |
| 49 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 50 | 0x001 | 0x00 | 0049 | frame_format_descriptor_3 lo | ?? | ?? | 0213 | digital_gain_blue lo |
| 51 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 52 | 0x001 | 0x00 | 004a | frame_format_descriptor_4 hi | ?? | ?? | 0214 | digital_gain_greenB hi |
| 53 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 54 | 0x001 | 0x00 | 004b | frame_format_descriptor_4 lo | ?? | ?? | 0215 | digital_gain_greenB lo |
| 55 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 56 | 0x001 | 0x00 | 004c | frame_format_descriptor_5 hi | 0x00d | 0x03 | | Address 03xx |
| 57 | 0x169 | 0x5A | | | 0x295 | 0xA5 | | CCI register index LSB |
| 58 | 0x001 | 0x00 | 004d | frame_format_descriptor_5 lo | 0x001 | 0x00 | | Address xx00 |
| 59 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 60 | 0x001 | 0x00 | 004e | frame_format_descriptor_6 hi | ?? | ?? | 0300 | vt_pix_clk_div hi |
| 61 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 62 | 0x001 | 0x00 | 004f | frame_format_descriptor_6 lo | ?? | ?? | 0301 | vt_pix_clk_div lo |
| 63 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 64 | 0x001 | 0x00 | 0050 | frame_format_descriptor_7 hi | ?? | ?? | 0302 | vt_sys_clk_div hi |
| 65 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 66 | 0x001 | 0x00 | 0051 | frame_format_descriptor_7 lo | ?? | ?? | 0303 | vt_sys_clk_div lo |
| 67 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 68 | 0x001 | 0x00 | 0052 | frame_format_descriptor_8 hi | ?? | ?? | 0304 | pre_pll_clk_div hi |
| 69 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 70 | 0x001 | 0x00 | 0053 | frame_format_descriptor_8 lo | ?? | ?? | 0305 | pre_pll_clk_div lo |
| 71 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 72 | 0x001 | 0x00 | 0054 | frame_format_descriptor_9 hi | ?? | ?? | 0306 | pll_multiplier_hi |
| 73 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 74 | 0x001 | 0x00 | 0055 | frame_format_descriptor_9 lo | ?? | ?? | 0307 | pll_multiplier_lo |



Table 4: Embedded Data (continued)

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|-------------------------------|--------|-------|--------------------------------|------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 75 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 76 | 0x001 | 0x00 | 0056 | frame_format_descriptor_10 hi | ?? | ?? | 0308 | op_pix_clk_div hi |
| 77 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 78 | 0x001 | 0x00 | 0057 | frame_format_descriptor_10 lo | ?? | ?? | 0309 | op_pix_clk_div lo |
| 79 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 80 | 0x001 | 0x00 | 0058 | frame_format_descriptor_11 hi | ?? | ?? | 030a | op_sys_clk_div hi |
| 81 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 82 | 0x001 | 0x00 | 0059 | frame_format_descriptor_11 lo | ?? | ?? | 030b | op_sys_clk_div lo |
| 83 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 84 | 0x001 | 0x00 | 005a | frame_format_descriptor_12 hi | 0x00d | 0x03 | | Address 03xx |
| 85 | 0x169 | 0x5A | | | 0x295 | 0x0A5 | | CCI register index LSB |
| 86 | 0x001 | 0x00 | 005b | frame_format_descriptor_12 lo | 0x101 | 0x40 | | Address xx40 |
| 87 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 88 | 0x001 | 0x00 | 005c | frame_format_descriptor_13 hi | ?? | ?? | 0340 | frame_length_lines hi |
| 89 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 90 | 0x001 | 0x00 | 005d | frame_format_descriptor_13 lo | ?? | ?? | 0341 | frame_length_lines lo |
| 91 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 92 | 0x001 | 0x00 | 005e | frame_format_descriptor_14 hi | ?? | ?? | 0342 | line_length_pck hi |
| 93 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 94 | 0x001 | 0x00 | 005f | frame_format_descriptor_14 lo | ?? | ?? | 0343 | line_length_pck lo |
| 95 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 96 | 0x001 | 0x00 | | Address 00xx | ?? | ?? | 0344 | x_addr_start hi |
| 97 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 98 | 0x201 | 0x80 | | Address xx80 | ?? | ?? | 0345 | x_addr_start lo |
| 99 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 100 | 0x001 | 0x00 | 0080 | analogue_gain_capability hi | ?? | ?? | 0346 | y_addr_start hi |
| 101 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 102 | 0x005 | 0x01 | 0081 | analogue_gain_capability lo | ?? | ?? | 0347 | y_addr_start lo |
| 103 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 104 | 0x001 | 0x00 | | Address 00xx | ?? | ?? | 0348 | x_addr_end hi |
| 105 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 106 | 0x211 | 0x84 | | Address xx84 | ?? | ?? | 0349 | x_addr_end lo |
| 107 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 108 | 0x001 | 0x00 | 0084 | analogue_gain_code_min hi | ?? | ?? | 034a | y_addr_end hi |
| 109 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 110 | 0x021 | 0x08 | 0085 | analogue_gain_code_min lo | ?? | ?? | 034b | y_addr_end lo |
| 111 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 112 | 0x001 | 0x00 | 0086 | analogue_gain_code_max hi | ?? | ?? | 034c | x_output_size hi |
| 113 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 114 | 0x1fd | 0x7F | 0087 | analogue_gain_code_max lo | ?? | ?? | 034d | x_output_size lo |
| 115 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 116 | 0x001 | 0x00 | 0088 | analogue_gain_code_step hi | ?? | ?? | 034e | y_output_size hi |
| 117 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |



Table 4: Embedded Data (continued)

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|-----------------------------|--------|-------|--------------------------------|------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 118 | 0x005 | 0x01 | 0089 | analogue_gain_code_step lo | ?? | ?? | 034f | y_output_size lo |
| 119 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 120 | 0x001 | 0x00 | 008a | analogue_gain_type hi | 0x00d | 0x03 | | Address 03xx |
| 121 | 0x169 | 0x5A | | | 0x295 | 0xA5 | | CCI register index LSB |
| 122 | 0x001 | 0x00 | 008b | analogue_gain_type lo | 0x201 | 0x80 | | Address xx80 |
| 123 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 124 | 0x001 | 0x00 | 008c | analogue_gain_m0 lo | ?? | ?? | 0380 | x_even_inc hi |
| 125 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 126 | 0x005 | 0x01 | 008d | analogue_gain_m0 lo | ?? | ?? | 0381 | x_even_inc lo |
| 127 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 128 | 0x001 | 0x00 | 008e | analogue_gain_c0 lo | ?? | ?? | 0382 | y_odd_inc hi |
| 129 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 130 | 0x001 | 0x00 | 008f | analogue_gain_c0 lo | ?? | ?? | 0383 | y_odd_inc lo |
| 131 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 132 | 0x001 | 0x00 | 0090 | analogue_gain_m1 lo | ?? | ?? | 0384 | y_even_inc hi |
| 133 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 134 | 0x001 | 0x00 | 0091 | analogue_gain_m1 lo | ?? | ?? | 0385 | y_even_inc lo |
| 135 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 136 | 0x001 | 0x00 | 0092 | analogue_gain_c1 lo | ?? | ?? | 0386 | x_odd_inc hi |
| 137 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 138 | 0x021 | 0x08 | 0093 | analogue_gain_c1 lo | ?? | ?? | 0387 | x_odd_inc lo |
| 139 | 0x2a9 | 0xAA | | CCI register index MSB | 0x2a9 | 0xAA | | CCI register index MSB |
| 140 | 0x001 | 0x00 | | Address 00xx | 0x011 | 0x04 | | Address 04xx |
| 141 | 0x295 | 0xA5 | | CCI register index LSB | 0x295 | 0xA5 | | CCI register index LSB |
| 142 | 0x301 | 0xC0 | | Address xxC0 | 0x001 | 0x00 | | Address xx00 |
| 143 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | auto increment |
| 144 | 0x005 | 0x01 | 00C0 | data_format_model_type | ?? | ?? | 0400 | scaling_mode hi |
| 145 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 146 | 0x00d | 0x03 | 00c1 | data_format_model_subtype | ?? | ?? | 0401 | scaling_mode lo |
| 147 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 148 | 0x029 | 0x0A | 00c2 | data_format_descriptor_0 hi | ?? | ?? | 0402 | spatial_sampling hi |
| 149 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 150 | 0x029 | 0x0A | 00c3 | data_format_descriptor_0 lo | ?? | ?? | 0403 | spatial_sampling lo |
| 151 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 152 | 0x021 | 0x08 | 00c4 | data_format_descriptor_1 hi | ?? | ?? | 0404 | scale_m hi |
| 153 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 154 | 0x021 | 0x08 | 00c5 | data_format_descriptor_1 lo | ?? | ?? | 0405 | scale_m lo |
| 155 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 156 | 0x029 | 0x0A | 00c6 | data_format_descriptor_2 hi | 0x001 | 0x00 | 0406 | scale_n hi |
| 157 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 158 | 0x021 | 0x08 | 00c7 | data_format_descriptor_2 lo | 0x041 | 0x10 | 0407 | scale_n lo |
| 159 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 160 | 0x001 | 0x00 | 00c8 | data_format_descriptor_3 hi | 0x015 | 5 | | Address 05xx |



Table 4: Embedded Data (continued)

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|-----------------------------|--------|-------|--------------------------------|-------------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 161 | 0x169 | 0x5A | | | 0x295 | 0xA5 | | CCI register index LSB |
| 162 | 0x001 | 0x00 | 00c9 | data_format_descriptor_3 lo | 0x001 | 0x00 | | Address xx00 |
| 163 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 164 | 0x001 | 0x00 | 00ca | data_format_descriptor_4 hi | 0x001 | 0x00 | 0500 | compression_mode hi |
| 165 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 166 | 0x001 | 0x00 | 00cb | data_format_descriptor_4 lo | 0x005 | 0x01 | 0501 | compression_mode lo |
| 167 | 0x169 | 0x5A | | | 0x2a9 | 0xAA | | CCI register index MSB |
| 168 | 0x001 | 0x00 | 00cc | data_format_descriptor_5 hi | 0x019 | 0x06 | | Address 06xx |
| 169 | 0x169 | 0x5A | | | 0x295 | 0xA5 | | CCI register index LSB |
| 170 | 0x001 | 0x00 | 00cd | data_format_descriptor_5 lo | 0x001 | 0x00 | | Address xx00 |
| 171 | 0x169 | 0x5A | | | 0x169 | 0x5A | | auto increment |
| 172 | 0x001 | 0x00 | 00ce | data_format_descriptor_6 hi | ?? | ?? | 0600 | test_pattern_mode hi |
| 173 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 174 | 0x001 | 0x00 | 00cf | data_format_descriptor_6 lo | ?? | ?? | 0601 | test_pattern_mode lo |
| 175 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 176 | 0x005 | 0x01 | | Address 01xx | ?? | ?? | 0602 | test_data_red hi |
| 177 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 178 | 0x001 | 0x00 | | Address xx00 | ?? | ?? | 0603 | test_data_red lo |
| 179 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 180 | ?? | ?? | 0100 | mode_select | ?? | ?? | 0604 | test_data_greenR hi |
| 181 | 0x169 | ?? | | | 0x169 | 0x5A | | |
| 182 | ?? | 0x00 | 0101 | image_orientation | ?? | ?? | 0605 | test_data_greenR lo |
| 183 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 184 | 0x005 | 0x01 | | Address 01xx | ?? | ?? | 0606 | test_data_blue hi |
| 185 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 186 | 0x00d | 0x03 | | Address xx03 | ?? | ?? | 0607 | test_data_blue lo |
| 187 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 188 | 0x001 | 0x00 | 0103 | software_reset | ?? | ?? | 0608 | test_data_greenB hi |
| 189 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 190 | ?? | ?? | 0104 | grouped_parameter_hold | ?? | ?? | 0609 | test_data_greenB lo |
| 191 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 192 | ?? | ?? | 0105 | mask_corrupted_frames | ?? | ?? | 060a | horizontal_cursor_width hi |
| 193 | 0x2a9 | 0xAA | | CCI register index MSB | 0x169 | 0x5A | | |
| 194 | 0x005 | 0x01 | | Address 01xx | ?? | ?? | 060b | horizontal_cursor_width lo |
| 195 | 0x295 | 0xA5 | | CCI register index LSB | 0x169 | 0x5A | | |
| 196 | 0x041 | 0x10 | | Address xx10 | ?? | ?? | 060c | horizontal_cursor_position hi |
| 197 | 0x169 | 0x5A | | auto increment | 0x169 | 0x5A | | |
| 198 | ?? | ?? | 0110 | CCP2_channel_identifier | ?? | ?? | 060d | horizontal_cursor_position lo |
| 199 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 200 | ?? | ?? | 0111 | CCP2_signalling_mode | ?? | ?? | 060e | vertical_cursor_width hi |
| 201 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |
| 202 | ?? | ?? | 0112 | CCP_data_format hi | ?? | ?? | 060f | vertical_cursor_width lo |
| 203 | 0x169 | 0x5A | | | 0x169 | 0x5A | | |


Table 4: Embedded Data (continued)

| Row 0 | | | | | Row 1 | | | |
|--------|--------|-------|--------------------------------|-------------------------------|--------|-------|--------------------------------|-------------------------------|
| Offset | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment | 10-Bit | 8-Bit | Two-wire Serial Interface Addr | Comment |
| 204 | ?? | ?? | 0113 | CCP_data_format lo | ?? | ?? | 0610 | vertical_cursor_position hi |
| 205 | 0x01d | 0x07 | | Null Data | 0x169 | 0x5A | | |
| 206 | 0x01d | 0x07 | | Null Data - up to end-of-line | ?? | ?? | 0611 | vertical_cursor_position lo |
| 207 | | | | | 0x01d | 0x07 | | Null Data |
| 208 | | | | | 0x01d | 0x07 | | Null Data - up to end-of-line |

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 MT9D014. Aptina recommends that these rules are encoded into the device driver stack—either implicitly or explicitly.

Table 5: Definitions for Programming Rules

| Name | Definition |
|-------|---|
| xskip | $xskip = 1$ if $x_odd_inc = 1$; $xskip = 2$ if $x_odd_inc = 3$ |
| yskip | $yskip = 1$ if $y_odd_inc = 1$; $yskip = 2$ if $y_odd_inc = 3$ |

Table 6: 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 line_length_pck | min_line_length_pck $((x_addr_end - x_addr_start + x_odd_inc) / xskip) + min_line_blanking_pck$ | max_line_length_pck | SMIA |
| frame_length_lines | $((y_addr_end - y_addr_start + y_odd_inc) / yskip) + min_frame_blanking_lines$ | | SMIA |
| x_addr_start | x_addr_min | x_addr_max | SMIA |
| x_addr_end | x_addr_start | x_addr_max | SMIA |
| $(x_addr_end - x_addr_start + x_odd_inc)$ | must be positive | must be positive | SMIA |
| x_addr_start[0] | 0 | 0 | SMIA |
| x_addr_end[0] | 1 | 1 | SMIA |
| y_addr_start | y_addr_min | y_addr_max | SMIA |
| y_addr_end | y_addr_start | y_addr_max | SMIA |
| $(y_addr_end - y_addr_start + y_odd_inc) /$ | must be positive | must be positive | SMIA |
| y_addr_start[0] | 0 | 0 | SMIA |
| y_addr_end[0] | 1 | 1 | SMIA |
| x_even_inc | min_even_inc | max_even_inc | SMIA |
| x_even_inc[0] | 1 | 1 | SMIA |
| y_even_inc | min_even_inc | max_even_inc | SMIA |
| y_even_inc[0] | 1 | 1 | SMIA |
| 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 |

Table 6: Programming Rules (continued)

| Parameter | Minimum Value | Maximum Value | Origin |
|------------------|--|--------------------|--|
| y_odd_inc[0] | 1 | 1 | SMIA |
| scale_m | scaler_m_min | scaler_m_max | SMIA |
| scale_n | scaler_n_min | scaler_n_max | SMIA |
| x_output_size | 256 | 1608 | Minimum from SMIA FS Section 5.2.2.5. Maximum is a consequence of the output FIFO size on this implementation. |
| x_output_size[0] | 0 (this is enforced in hardware: bit[0] is read-only) | 0 | SMIA FS Section 5.2.2.2. |
| y_output_size | 2 | frame_length_lines | Minimum ensures 1 Bayer row-pair. Maximum avoids output frame being longer than pixel array frame. |
| y_output_size[0] | 0 (this is enforced in hardware: bit[0] is read-only) | 0 | SMIA FS Section 5.2.2.2 |

- Notes: 1. With subsampling, start and end pixels must be addressed (impact on x/y start/end addresses, function of image orientation bits). SMIA FS Errata see “Subsampling” on page 39.

Output Size Restrictions

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

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

This restriction 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.

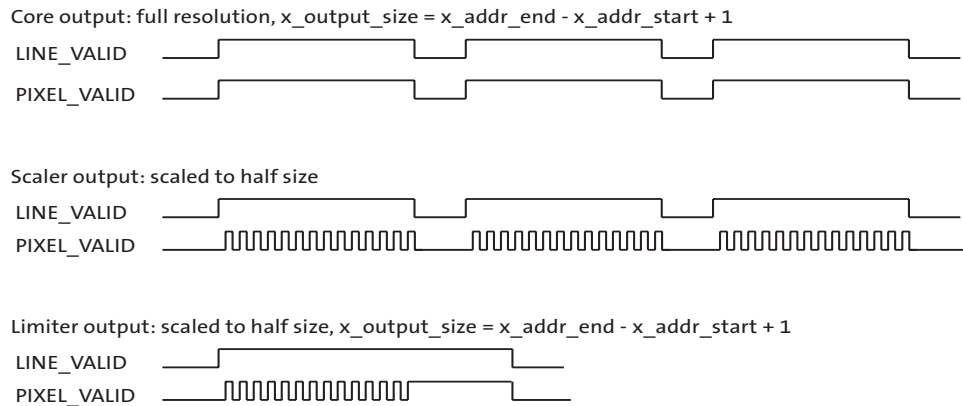
There is an additional restriction that `x_output_size` must be small enough such that the output row time (set by `x_output_size`, the framing and CRC overhead of 12 bytes, the `ccp_signalling_mode` and the output clock rate) must be less than the row time of the video array (set by `line_length_pck` 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 MT9D014 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 10.

Figure 10: Effect of Limiter on the SMIA Data Path



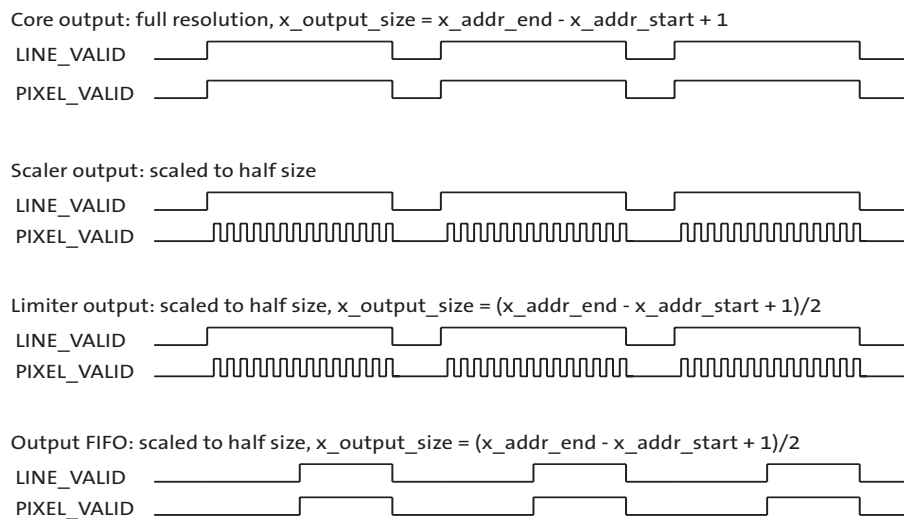
In Figure 10, three different stages in the SMIA data path (see “Digital Data Path” on page 50) are shown. The first stage is the output of the sensor core. The core is running at full resolution and x_output_size is set to match the active array size. The $LINE_VALID$ (LV) signal is asserted once per row and remains asserted for N pixel times. The $PIXEL_VALID$ signal toggles with the same timing as LV, 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 LV across the whole of the horizontal blanking time, the MT9D014 will cease to generate output frames.

A correct configuration is shown in Figure 11 on page 24, in addition to showing the x_output_size reduced to match the output size of the scaler. In this configuration, the output of the limiter does not extend LV.

Figure 11 on page 24 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 11: 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 $2184 * 1219 = 2662296$ pixel periods. This value includes vertical and horizontal blanking times so that the full-size image 1600 x 1202 (1200 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 $2662296 / 64e6 = 41.60ms$, giving rise to a frame rate of 24.04 fps.

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

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

The only way to get a full image out is to reduce the pixel clock rate until it is appropriate for the maximum CCP2 class 0 data rate. This requires the pixel rate to be reduced to 20.8 MHz. This has the side effect of reducing the frame rate. Repeating the calculation above, at 20.8 MHz internal clock, this corresponds to $2662296 / 20.8e6 = 128ms$, giving rise to a frame rate of 7.81 fps.

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

Output Data Timing

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

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

Maximum frame rate is achieved by setting the video timing clock (`vt_clk_freq_mhz`) to 80 MHz and using the FIFO to reduce horizontal blanking data rate to 640 Mb/s. At this setting, a maximum frame rate of 30.05 fps can be achieved.

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 (R0x306A).

Changing Registers while Streaming

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

- `ccp2_channel_identifier`
- `ccp2_signalling_mode`
- `ccp_data_format`
- `scale_m`
- `vt_pix_clk_div`
- `vt_sys_clk_div`
- `pre_pll_clk_div`
- `pll_multiplier`
- `op_pix_clk_div`
- `op_sys_clk_div`

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 (through a GPI pad)

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

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

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

This interface is described in detail in “EXTCLK” on page 55.

Default Power-Up State

The MT9D014 provides interfaces for pixel data through the CCP2 high-speed serial interface described by the SMIA specification.

At power up and after a hard or soft reset, the reset state of the MT9D014 is to enable the SMIA CCP2 high-speed serial interface.

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

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

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



Serial Pixel Data Interface

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

- DATA_P
- DATA_N
- CLK_P
- CLK_N

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

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

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

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

System States

The system states of the MT9D014 are represented as a state diagram in Figure 12 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 7 on page 29.

The sensor's operation is broken down into three separate states: hardware standby, soft standby, and streaming. The transition between these states might take a certain amount of clock cycles as outlined in Table 7.

Figure 12: MT9D014 System States

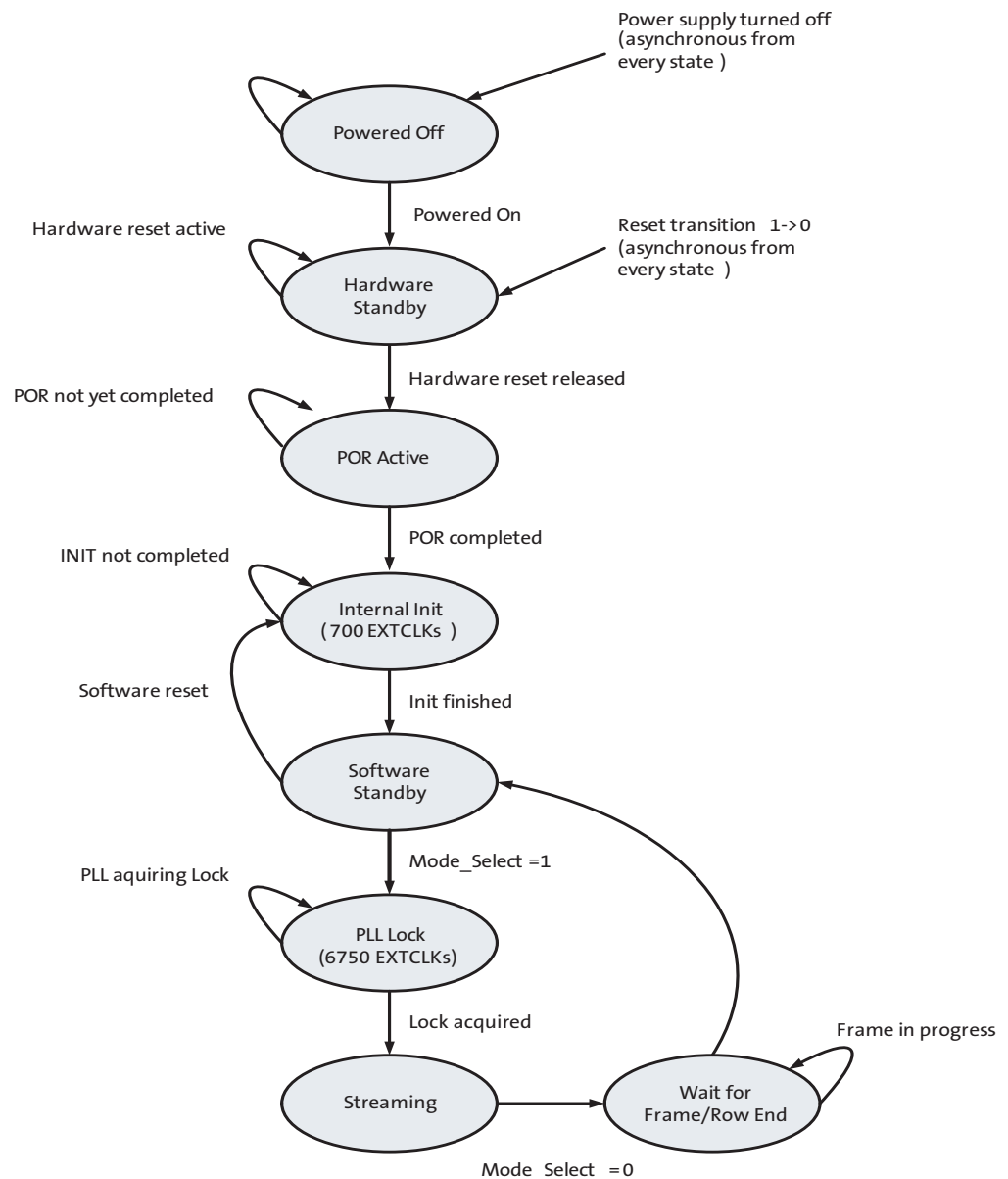


Table 7: PLL in System States

| State | EXTCLKs | PLL |
|-------------------------|---------|--|
| Powered off | | |
| Hardware standby | | |
| POR active | | |
| Internal initialization | 700 | VCO powered down |
| Software standby | | |
| PLL lock | 6750 | VCO powering up and locking, PLL output bypassed |
| Streaming | | VCO running, PLL output active |
| Wait for frame end | | |

Notes: 1. VCO = voltage-controlled oscillator.

Power-On Reset Sequence

When power is applied to the MT9D014, it enters a low-power hardware standby state. Exit from this state is controlled by the later of two events:

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

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

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

When the sensor leaves the hardware standby state, it waits for power-on reset and performs an internal initialization sequence that takes 700 EXTCLK cycles. After this time, it enters a low-power soft standby state. While the initialization sequence is in progress, the MT9D014 will not respond to READ transactions on its two-wire serial interface. Therefore, a method to determine when the initialization sequence has completed is to poll a sensor register; for example, R0x0000. While the initialization sequence is in progress, the sensor will not respond to its device address and READs from the sensor will result in a NACK on the two-wire serial interface bus. When the sequence has completed, READs will return the operational value for the register (0x1501 if R0x0000 is read).

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

Soft Reset Sequence

The MT9D014 can be reset under software control by writing “1” to software_reset (R0x0103). A software reset asynchronously resets the sensor, truncating any frame that is in progress. The sensor starts the internal initialization sequence, while the PLL and analog blocks are turned off. At this point, the behavior is exactly the same as for the power-on reset sequence.

Signal State During Reset

Table 8 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during soft standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 8: Signal State During Reset

| Pad Name | Pad Type | Hardware Standby | Software Standby |
|--------------------------|----------|--|------------------|
| EXTCLK | Input | Self biased. Can be left disconnected/floating. | |
| RESET_BAR (XSHUTDOWN) | Input | Enabled. Must be driven to a valid logic level. | |
| 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. | |
| DATA_P | Output | High-Z. | |
| DATA_N | Output | | |
| CLK_P | Output | | |
| CLK_N | Output | | |
| GPI[3:0] | Input | Powered up. Must be connected to VDD or DGND. | |
| TEST1 | Input | Must be driven to a logic 0. | |
| TEST2 | Input | Must be driven to a logic 1. | |

General Purpose Inputs

The MT9D014 provides four general purpose inputs. After reset, the input pads associated with these signals are powered on by default, requiring the pads to be tied to a defined logic level.

The general purpose inputs are disabled by setting `reset_register[8]` (R0x301A–B). Once disabled, the inputs can be left floating. The state of the general purpose inputs can be read through `gpi_status[3:0]` (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:

- Trigger (see the following sections)
- Standby functions (see the following sections)
- SADDR selection (see “Serial Register Interface” on page 26).

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



Streaming/Standby Control

The MT9D014 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 9. Selection of a pin to use for the STANDBY function is described in “General Purpose Inputs” on page 30. The state diagram for transitions between soft standby and streaming states is shown in Figure 12 on page 28.

Table 9: 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 MT9D014 contains a PLL for timing generation and control. The PLL contains a prescaler to divide the input clock applied on EXTCLK, a VCO to multiply the prescaler output, and a set of dividers to generate the output clocks.

Profile 0 Behavior

Aptina SMIA sensors are profile 2 sensors and have separate video timing and output clock domains.

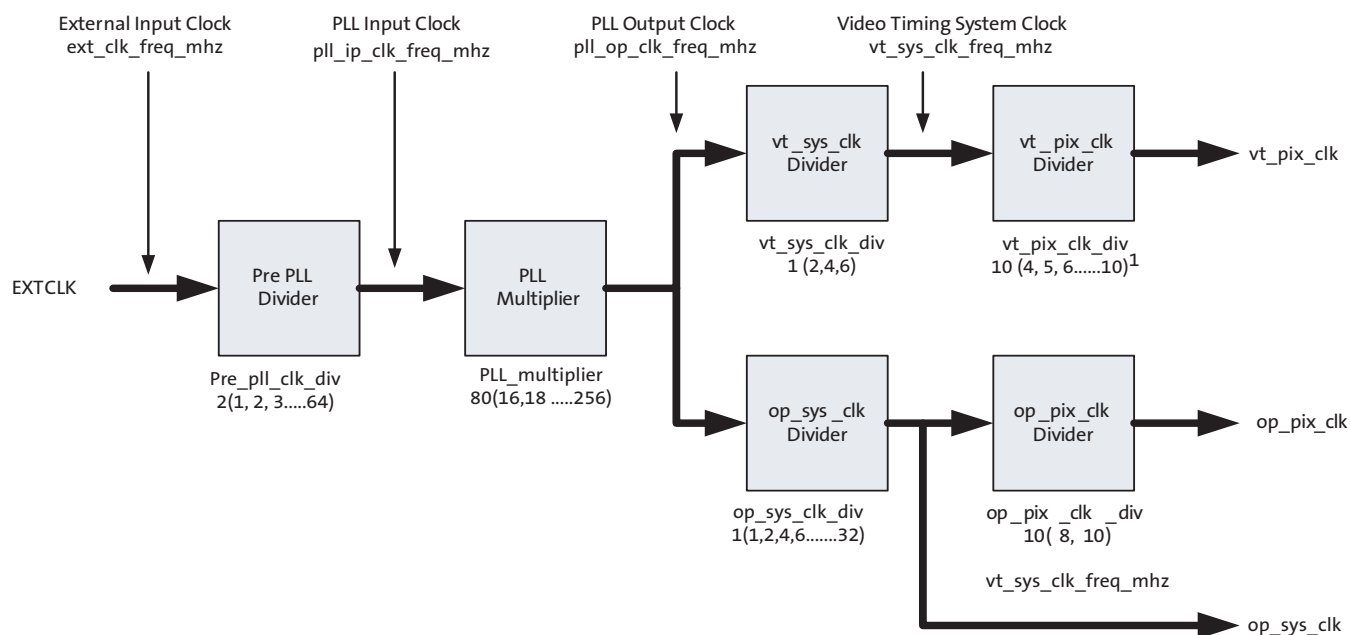
If the video timing and output clock domains are programmed with the same dividers, the part will operate in profile 0 mode as indicated by R0x30E-F[7]. For example, if Equation 1 is true, then the PLL will have profile 0 behavior:

$$\begin{aligned} \text{Profile0_behavior} = & (vt_sys_clk_div == op_sys_clk_div) & \text{(EQ 1)} \\ & \& (vt_pix_clk_div == op_pix_clk_div) \end{aligned}$$

When the PLL is programmed to be in profile 0 behavior then the output clock domain is connected internally to the video timing domain thus ensuring that the sensor behave as an profile 0 sensor with respect to the PLL.

In profile 0 mode the number of bits between one sync code and the subsequent one are guaranteed to be equal.

Note that legacy sensors used the profile bit in the datapath_select register R0x306E[7] to set this behavior. The new behavior of profile 0 mode is equivalent with the old one once it is set by the host system.

Figure 13: MT9D014 SMIA Profile 1/2 Clocking Structure


- Notes:
1. The combinations $vt_sys_clk_div = 1$ and $vt_pix_clk_div = (4,6,8,10)$ are also supported even though the capability register does not advertise this.
 2. The $pll_multiplier$ only accepts even values when $ccp2_class$ is set to data/clock signalling. Odd values will be rounded down to the first even number by setting LSB to "0."
 3. The default value for $vt_sys_clk_div$ is outside the range of legal values defined by the capability registers. This results in correct behavior for the cases listed in Note 1. The default setting is selected to ensure profile 0 behavior as default with the highest possible frame rate.

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

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

The usage of the output clocks is:

- vt_pix_clk is used by the sensor core to control the timing of the pixel array. The sensor core produces one 10-bit pixel each vt_pix_clk period. The line length

(line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of the vt_pix_clk period.

- op_pix_clk is used to load parallel pixel data from the output FIFO (see Figure 23 on page 50) to the CCP2 serializer. The output FIFO generates one pixel each op_pix_clk period. The pixel is either 8-bit or 10-bit depending upon the output data format, controlled by R0x0112-3 (ccp_data_format).
- op_sys_clk is used to generate the serial data stream on the CCP2 output. The relationship between this clock frequency and the op_pix_clk frequency is dependent upon the output data format.

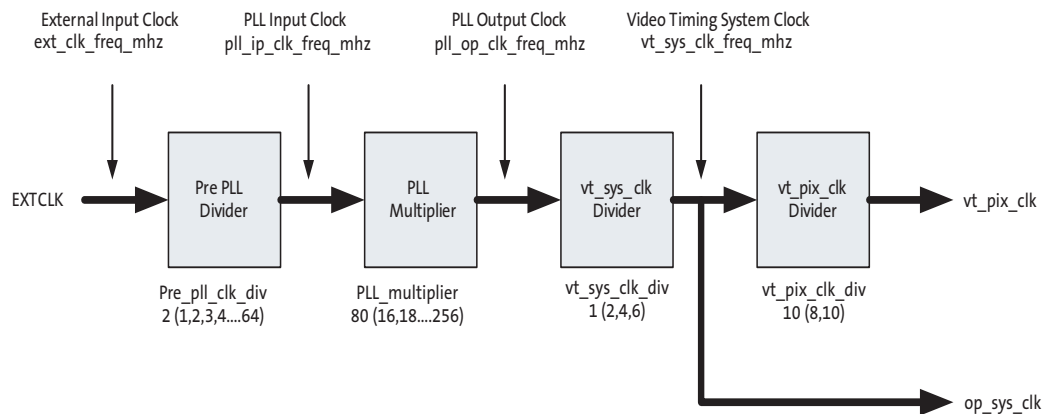
In profile 1/2, the output clock frequencies can be calculated as:

$$vt_pix_clk_freq_mhz = \frac{ext_clk_freq_mhz * pll_multiplier}{pre_pll_clk_div * vt_sys_clk_div * vt_pix_clk_div} \quad (EQ 2)$$

$$op_pix_clk_freq_mhz = \frac{ext_clk_freq_mhz * pll_multiplier}{pre_pll_clk_div * op_sys_clk_div * op_pix_clk_div} \quad (EQ 3)$$

$$op_sys_clk_freq_mhz = \frac{ext_clk_freq_mhz * pll_multiplier}{pre_pll_clk_div * op_sys_clk_div} \quad (EQ 4)$$

Figure 14: MT9D014 SMIA Profile 0 Clocking Structure



- Notes:
1. The legal range yielding profile 0 behavior is limited to the PLL values where the “vt domain” equals the “op domain”. The vt_sys_clk_div values in the parentheses are therefore the legal values for both vt_sys_clk_div and op_sys_clk_div, and the vt_pix_clk_div values in the parentheses are legal values for both vt_pix_clk_div and op_pix_clk_div.
 2. The default value for vt_sys_clk_div is outside the range of legal values defined by the capability registers. This will result in correct behavior for the cases listed in Note 1. The default setting is selected to ensure profile 0 behavior as default with the highest possible frame rate.

When the video timing domain and the output timing domain have the same divider values, the PLL is equivalent to the SMIA profile 0 clocking structure. This is achieved by driving the op_sys_clk domain from the vt_sys_clk output and by driving the op_pix_clk domain from the vt_pix_clk output.



Programming the PLL Divisors

The PLL divisors should be programmed while the MT9D014 is in the soft standby state. After programming the divisors, it is necessary to wait for the VCO lock time before enabling the PLL. The PLL is enabled by entering the streaming state.

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

The effect of programming the PLL divisors while the MT9D014 is in the streaming state is undefined.

Influence of `ccp_data_format`

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

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

Influence of `ccp2_signalling_mode`

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

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

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

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

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

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

Features

Shading Correction (SC)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing fixed pattern signal gradients in images captured by image sensors. The cumulative result of all these factors is known as image shading. The MT9D014 has an embedded shading correction module that can be programmed to counter the shading effects on each individual Red, GreenB, GreenR, and Blue color signal.

The Correction Function

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

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

$$P_{corrected}(row, col) = P_{sensor}(row, col) * f(row, col) \quad (EQ 5)$$

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

Each function includes a set of color-dependent coefficients defined by registers R0x3600–3726. The function's origin is the center point of the function used in the calculation of the coefficients. Using an origin near the central point of symmetry of the sensor response provides the best results.

The correct sequence to write to the SC registers is as follows:

1. Set R0x3780–1 = 0x0000.
2. Load SC coefficients.
3. Set R0x3780–1 = 0x8000.

To read the SC coefficients, SC should be disabled (set R0x3780–1 = 0x0000) before reading the register values.

Image Acquisition Modes

The MT9D014 supports ERS mode. When the MT9D014 is streaming, it generates frames at a fixed rate, and each frame is integrated (exposed) using the ERS. 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 MT9D014 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`, and `y_addr_end` registers. The output image size is controlled by the `x_output_size` and `y_output_size` registers.

Pixel Border

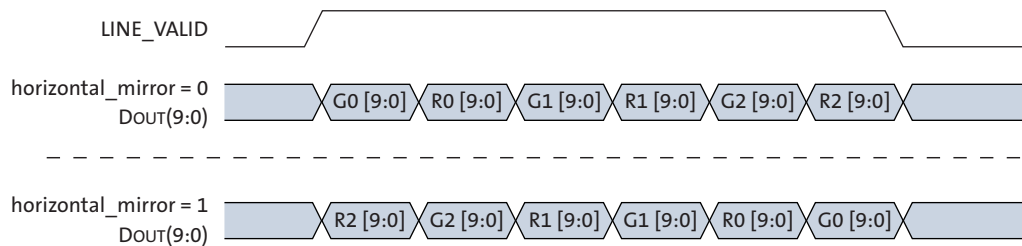
The default settings of the sensor provide a 1600H x 1200V image. A border of up to 4 pixels on each edge can be enabled by reprogramming the `x_addr_start`, `y_addr_start`, `x_addr_end`, `y_addr_end`, and `x_output_size` and `y_output_size` registers accordingly.

Readout Modes

Horizontal Mirror

When the `horizontal_mirror` bit is set in the `image_orientation` register, the order of pixel readout within a row is reversed, so that readout starts from `x_addr_end` and ends at `x_addr_start`. Figure 15 shows a sequence of 6 pixels being read out with `horizontal_mirror = 0` and `horizontal_mirror = 1`. Changing `horizontal_mirror` causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the `pixel_order` register.

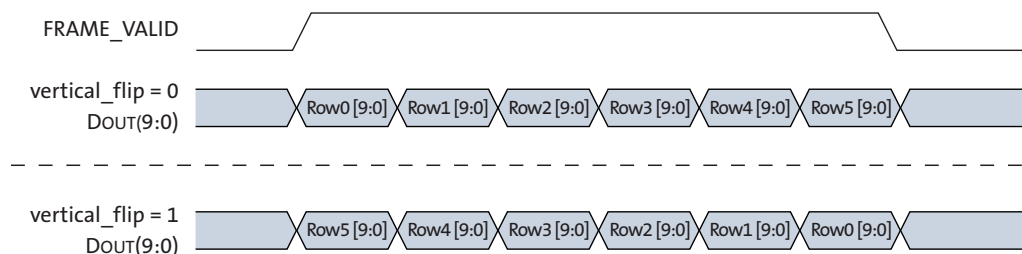
Figure 15: Effect of `horizontal_mirror` on Readout Order



Vertical Flip

When the `vertical_flip` bit is set in the `image_orientation` register, the order in which pixel rows are read out is reversed, so that row readout starts from `y_addr_end` and ends at `y_addr_start`. Figure 16 shows a sequence of 6 rows being read out with `vertical_flip = 0` and `vertical_flip = 1`. Changing `vertical_flip` causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the `pixel_order` register.

Figure 16: Effect of `vertical_flip` on Readout Order



Subsampling

The MT9D014 supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the MT9D014 thereby allowing the frame rate to be increased. Subsampling is enabled by setting `x_odd_inc` and/or `y_odd_inc`. Values of 1 and 3 can be supported. Setting both of these variables to 3 reduces the amount of row and column data processed. Figure 17 shows a sequence of 8 columns being read out with `x_odd_inc = 3` and `y_odd_inc = 1`.

Figure 17: Effect of `x_odd_inc = 3` on Readout Sequence

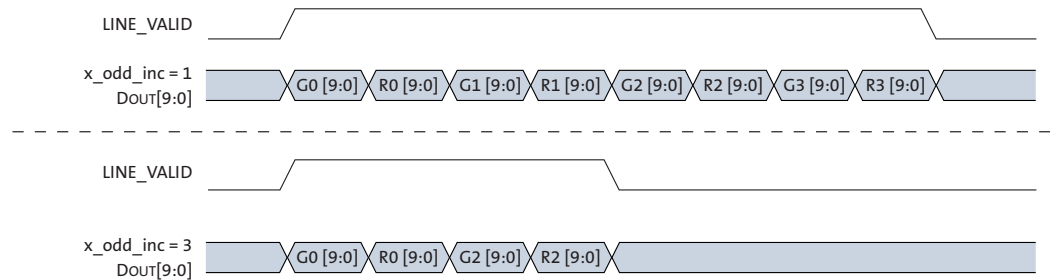


Figure 18: Pixel Readout (No Subsampling)

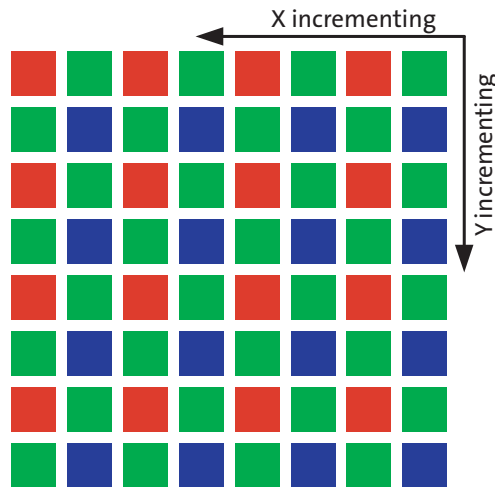
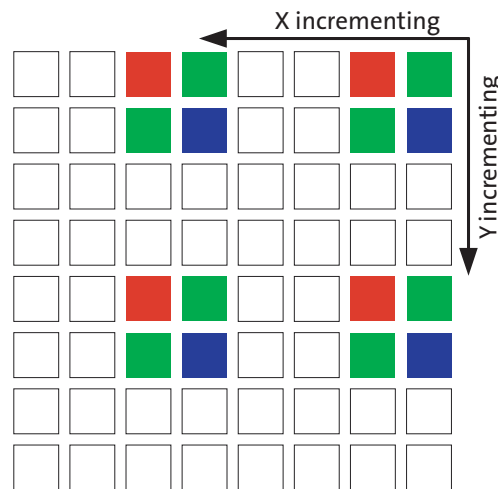


Figure 19: Pixel Readout ($x_odd_inc = 3, y_odd_inc = 3$)

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, Aptina recommends that `line_length_pck` be kept constant between the two modes. This allows the same integration times to be used in each mode.

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

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

Example:

To achieve 1600 x 1200 full resolution without subsampling, the recommended register settings are:

```
[full resolution starting address with (4, 4)]
REG=0x0104, 1 //GROUPED_PARAMETER_HOLD
REG=0x0382, 1 //X_ODD_INC
REG=0x0386, 1 //Y_ODD_INC
REG=0x0344, 4 //X_ADDR_START
REG=0x0346, 4 //Y_ADDR_START
REG=0x0348, 1603 //X_ADDR_END
REG=0x034A, 1203 //Y_ADDR_END
REG=0x034C, 1600 //X_OUTPUT_SIZE
REG=0x034E, 1200 //Y_OUTPUT_SIZE
REG=0x0104, 0 //GROUPED_PARAMETER_HOLD
```




To achieve a 800 x 600 resolution with 1/2 subsampling, the recommended register settings are:

```
[1/2 subsampling starting address with (8, 8)]
REG=0x0104, 1 //GROUPED_PARAMETER_HOLD
REG=0x0382, 3 //X_ODD_INC
REG=0x0386, 3 //Y_ODD_INC
REG=0x0344, 8 //X_ADDR_START
REG=0x0346, 8 //Y_ADDR_START
REG=0x0348, 1605 //X_ADDR_END
REG=0x034A, 1205 //Y_ADDR_END
REG=0x034C, 800 //X_OUTPUT_SIZE
REG=0x034E, 600 //Y_OUTPUT_SIZE
REG=0x0104, 0 //GROUPED_PARAMETER_HOLD
```

Table 10 shows the row address sequencing for normal and subsampled readout. The same sequencing applies to column addresses for subsampled readout. There are two possible subsampling sequences for the rows (because the subsampling sequence only read half of the rows) depending upon the alignment of the start address.

Table 10: Row Address Sequencing

| odd_inc = 1 | odd_inc = 3 | |
|-------------|-------------|-----------|
| Normal | Normal | |
| start = 0 | start = 0 | start = 2 |
| 0 | 0 | |
| 1 | 1 | |
| 2 | | 2 |
| 3 | | 3 |
| 4 | 4 | |
| 5 | 5 | |
| 6 | | 6 |
| 7 | | 7 |
| 8 | 8 | |
| 9 | 9 | |
| 10 | | 10 |
| 11 | | 11 |
| 12 | 12 | |
| 13 | 13 | |
| 14 | | 14 |
| 15 | | 15 |

Frame Rate Control

The formula for calculating the frame rate of the MT9D014 are shown below:

$$line_length_pck = \left(\frac{x_addr_end - x_addr_start + x_odd_inc}{subsampling_factor} + min_line_blinking_pck \right) \quad (EQ\ 6)$$

$$frame_length_lines = \left(\frac{y_addr_end - y_addr_start + y_odd_inc}{subsampling_factor} + min_frame_blinking_lines \right) \quad (EQ\ 7)$$

$$frame\ rate\ [FPS] = \frac{(vt_pixel_clock_mhz * 1 \times 10^6)}{(line_length_pck * frame_length_lines)} \quad (EQ\ 8)$$

Integration Time

The integration (exposure) time of the MT9D014 is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

$$fine_integration_time_min < = fine_integration_time < = (line_length_pck - fine_integration_time_max_margin) \quad (EQ\ 9)$$

The limits for the coarse integration time are defined by:

$$coarse_integration_time_min < = coarse_integration_time \quad (EQ\ 10)$$

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:

$$integration_time\ [sec] = \frac{((coarse_integration_time * line_length_pck) + fine_integration_time)}{(vt_pix_clk_freq_mhz * 1 * 10^6)} \quad (EQ\ 11)$$

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

$$Maximum\ integration\ time\ [sec] = \quad (EQ\ 12)$$

$$\frac{(((frame_length_lines - 1) * line_length_pck) + (line_length_pck - fine_integration_time_max_margin))}{(vt_pix_clk_freq_mhz * 1 * 10^6)}$$

$$= \frac{(((0x04C3 - 1) * 0x0888) + 0x797)}{(64\ MHz * 1 * 10^6)} = 41.59ms$$

Setting an integration time that is greater than the frame time increases the frame time beyond frame_length_lines to make longer exposure times available.

Analog Gain

The MT9D014 provides two mechanisms for setting the analog gain. The first uses the SMIA gain model; the second uses the traditional Aptina Imaging gain model. The following sections describe both models, the mapping between the models, and the operation of the per-color and global gain control.

Using Per-color or Global Gain Control

The read-only analogue_gain_capability register returns a value of “1,” indicating that the MT9D014 provides per-color gain control. However, the MT9D014 also provides the option of global gain control. Per-color and global gain control can be used interchangeably. A write to a global gain register is aliased as a write of the same data to the four associated color-dependent gain registers. A read from a global gain register is aliased to a read of the associated greenB/greenR gain register.

The read/write gain mode register required by SMIA has no defined function in the SMIA specification. In the MT9D014 this register has no side effects on the operation of the gain; per-color and global gain control can be used interchangeably regardless of the state of the gain_mode register.

SMIA Gain Model

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

- analogue_gain_code_global
- analogue_gain_code_greenR
- analogue_gain_code_red
- analogue_gain_code_blue
- analogue_gain_code_greenB

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

$$gain = \frac{analogue_gain_m0 \times analogue_gain_code}{analogue_gain_c1} = \frac{analogue_gain_code_<color>}{8} \quad (EQ\ 13)$$

Aptina Imaging Gain Model

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

- global_gain
- greenr_gain
- red_gain
- blue_gain
- greenb_gain

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

$$gain = (<color>_gain[7] + 1) \times \frac{<color>_gain[6:0]}{16} \quad (EQ\ 14)$$

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, because this will result in lower noise.

Gain Code Mapping

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

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

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

The result 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 MT9D014 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 11.

Table 11: 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 MT9D014 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`

The MT9D014 will disable digital corrections automatically when test patterns are activated. The test cursor is now added to the end of the data path.

Solid Color Test Pattern

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

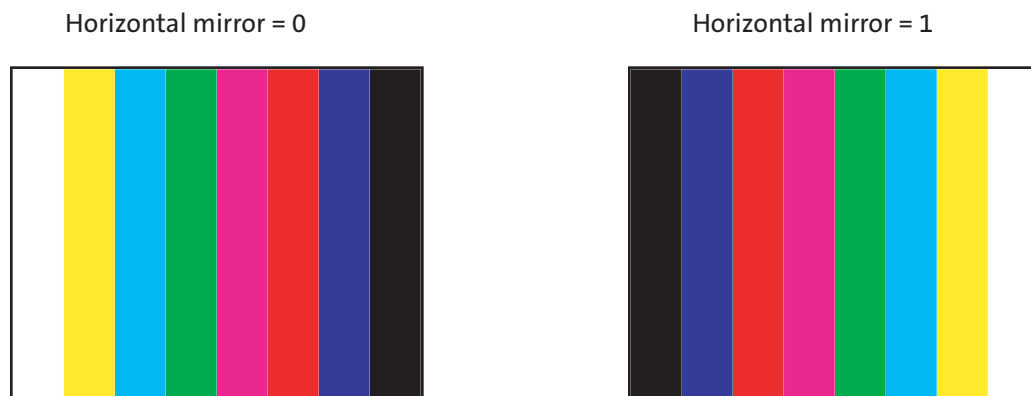
100 Percent Color Bars Test Pattern

In this test pattern, shown in Figure 20 on page 46, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, and black). Each bar is 200 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 $8 * 200 = 1600$ 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 200 pixels.

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

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

Figure 20: 100 Percent Color Bars Test Pattern

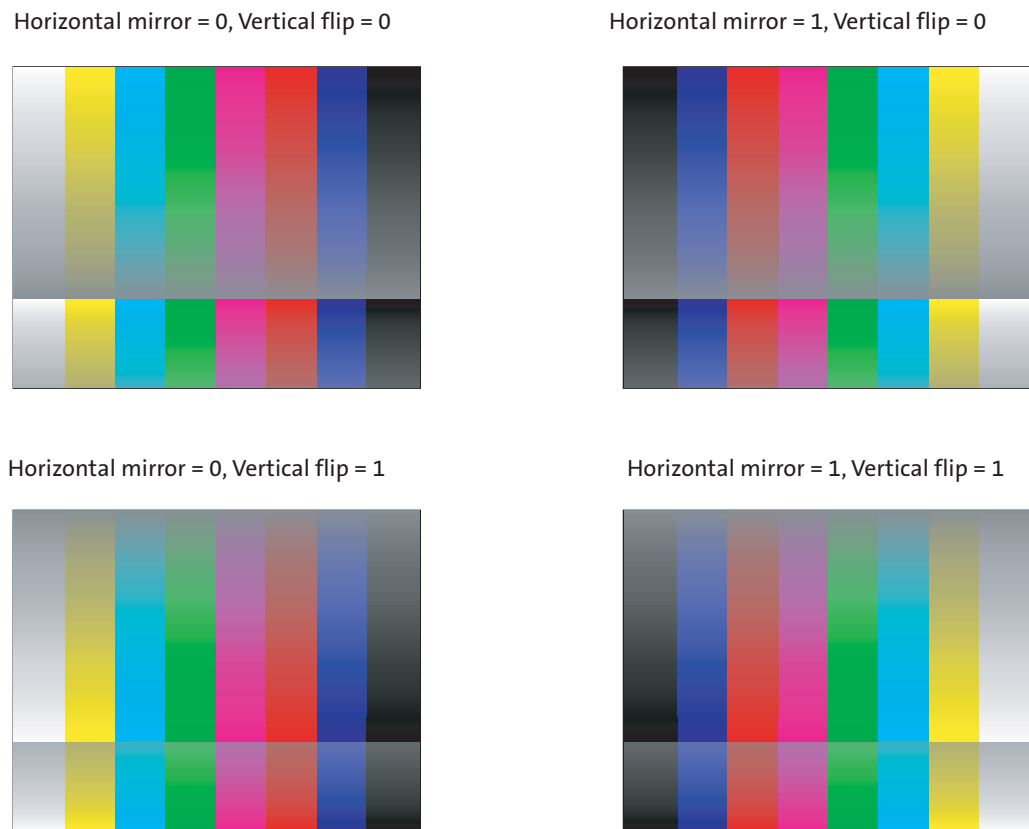


Fade-to-Gray Color Bars Test Pattern

In this test pattern, shown in Figure 21 on page 47, all pixel data is replaced by a Bayer version of an 8-color, color-bar chart (white, yellow, cyan, green, magenta, red, blue, and black). Each bar is 200 pixels wide and occupies 1024 rows of the output image. Each color bar fades vertically from full intensity at the top of the image to 50 percent intensity (mid-gray) on the 1024th row. Each color bar is divided into a left and a right half, in which the left half fades smoothly and the right half fades in quantized steps every 8 pixels for a given color. Due to the Bayer pattern of the colors this means that the level changes every 16 rows. The pattern repeats horizontally after $8 * 200 = 1600$ pixels and vertically after 1024 rows (using 10-bit data, the fade-to-gray pattern goes from 100 to 50 percent or from 0 to 50 percent for each color component, so only half of the 2^{10} states of the 10-bit data are used. However, to get all of the gray levels, each state must be held for two rows, hence the vertical size of $2^{10} / 2 * 2 = 1024$). The image size is set by `x_addr_start`, `x_addr_end`, `y_addr_start`, and `y_addr_end` and may be affected by the setting of `x_output_size` and `y_output_size`. The color-bar pattern starts at the column identified by `x_addr_start`. The number of colors that are visible in the output is dependent upon `x_addr_end - x_addr_start` and the setting of `x_output_size`. The width of each color-bar is fixed at 200 pixels.

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

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

Figure 21: Fade-to-Gray Color Bars Test Pattern


PN9 Link Integrity Pattern

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

When this test pattern is enabled:

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

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

Test Cursors

The MT9D014 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 in R0x31E8–0x31EE. Each cursor can be inhibited by setting its width to “0.”

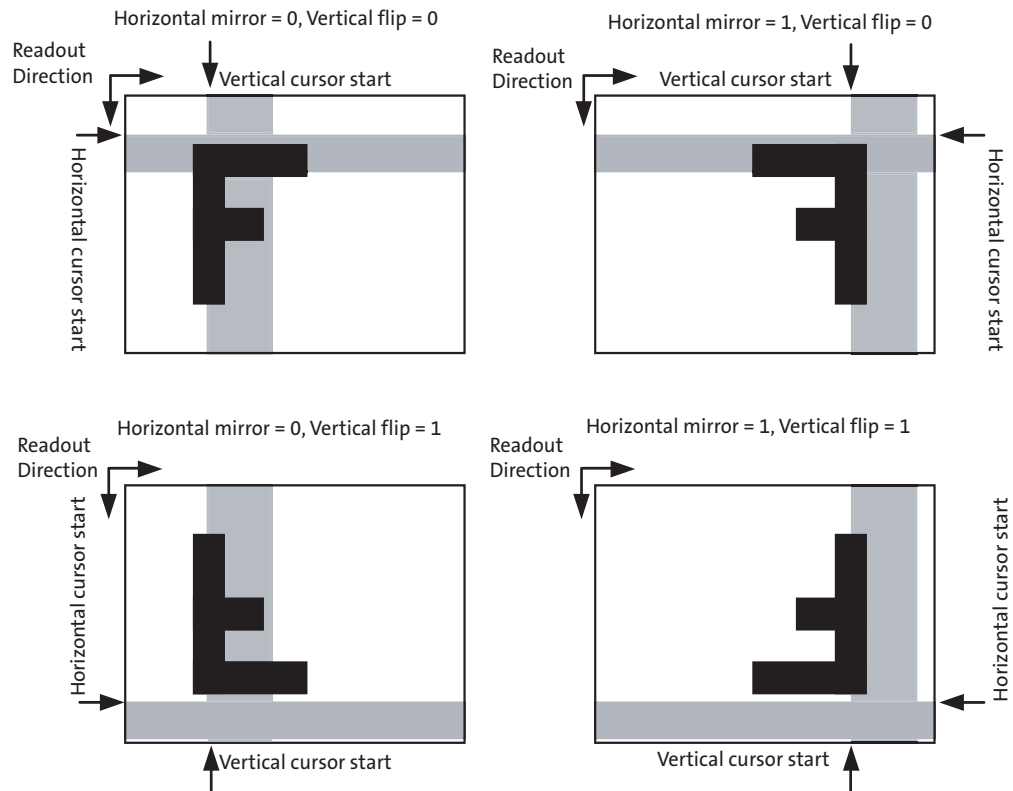
The programmed cursor position corresponds to an absolute row or column in the pixel array. For example, setting `horizontal_cursor_position` to the same value as `y_addr_start` would result in a horizontal cursor being drawn starting on the first row of the image.

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

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

The effect of enabling the test cursors when the `image_orientation` register is non-zero is not defined by the SMIA specification. The behavior of the MT9D014 is shown in Figure 22 on page 49. In this 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 22: Test Cursor Behavior when image_orientation


Digital Gain

Integer digital gains in the range 1–7 can be programmed. A digital gain of “0” sets all pixel values to “0” (the pixel data will simply represent the value applied by the pedestal block).

Pedestal

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

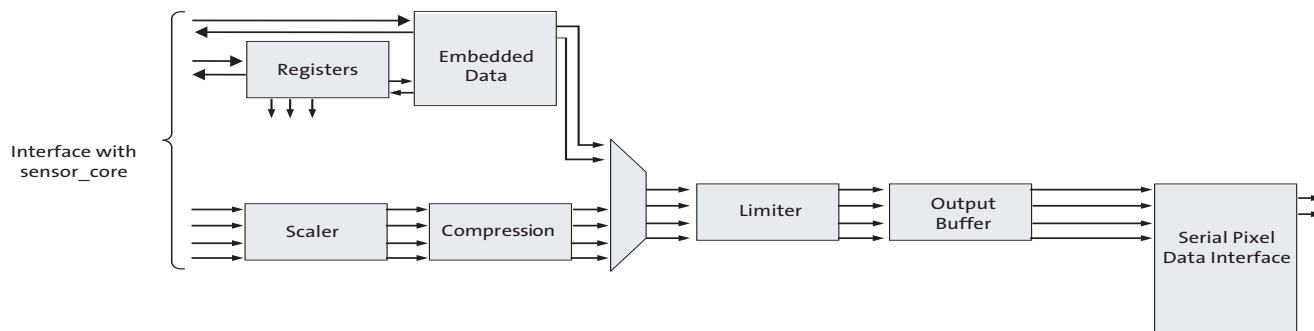
The data_pedestal register is read-only by default but can be made read/write by clearing the lock_reg bit in R0x301A–B.

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

Digital Data Path

The digital data path after the sensor core is shown in Figure 23.

Figure 23: Data Path



Timing Specifications

Power-Up Specifications

The digital and analog supply voltages can be powered up in any order. However, Aptina recommends the following power-up sequence to minimize current consumption. The power-up sequence corresponds to the requirements described in section 3.2 of the SMIA functional specification.

Power-Up Sequence

The recommended power-up sequence for the MT9D014 is shown in Figure 24 and Table 12 on page 52. The available power supplies—VDD, VDD_PLL, VAA, VAA_PIX—can be turned on at any point or have the separation specified below for reducing current consumption during power-up sequence.

1. Turn on the VDD power supply.
2. After 1–500ms, turn on VDD_PLL and VAA/VAA_PIX power supplies.
3. After the last power supply is stable, enable EXTCLK.
4. Assert RESET_BAR for at least 1ms.
5. Wait 2400 EXTCLKs for internal initialization into soft standby.
6. Configure PLL, output, and image settings to desired values.
7. Set mode_select = 1 (R0x0100).
8. Wait 6750 EXTCLKs for the PLL to lock before streaming state is reached (enforced in hardware).

Figure 24: Power-Up Sequence

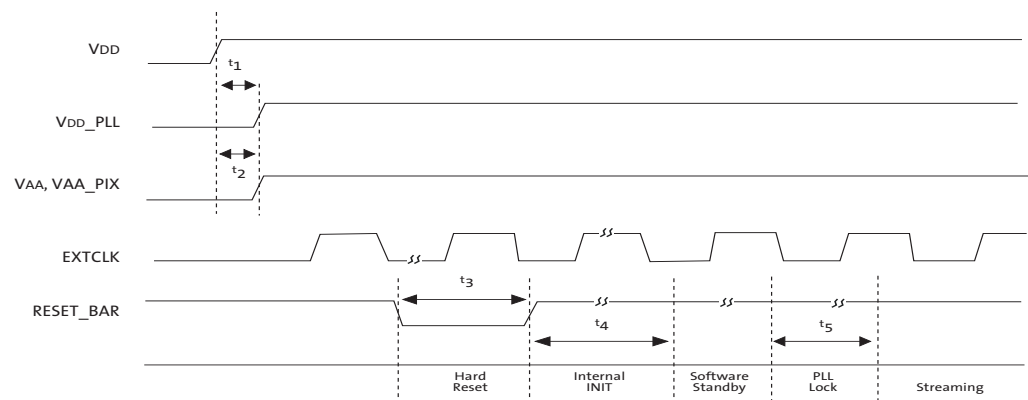


Table 12: Power-Up Sequence

| Definition | Symbol | Min | Typ | Max | Unit |
|-------------------------|--------|------|-----|-----|---------|
| VDD to VDD_PLL time | t_1 | 0 | – | 500 | ms |
| VDD to VAA/VAA_PIX time | t_2 | 0 | – | 500 | ms |
| Active hard reset | t_3 | 1 | – | – | ms |
| Internal initialization | t_4 | 2400 | – | – | EXTCLKs |
| PLL lock time | t_5 | 6750 | – | – | EXTCLKs |

Power-Down Specification

The digital and analog supply voltages can be powered down in any order. However, Aptina recommends the following power-down sequence to minimize current consumption. The power-down sequence corresponds to the requirements described in section 3.2 of the SMIA functional specification.

Power-Down Sequence

The recommended power-down sequence for the MT9D014 is shown in Figure 25 and in Table 13 on page 53. The available power supplies—VDD, VDD_PLL, VAA, VAA_PIX—can be turned off at any point or have the separation specified below for reducing current consumption during power-down sequence.

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

Figure 25: Power-Down Sequence

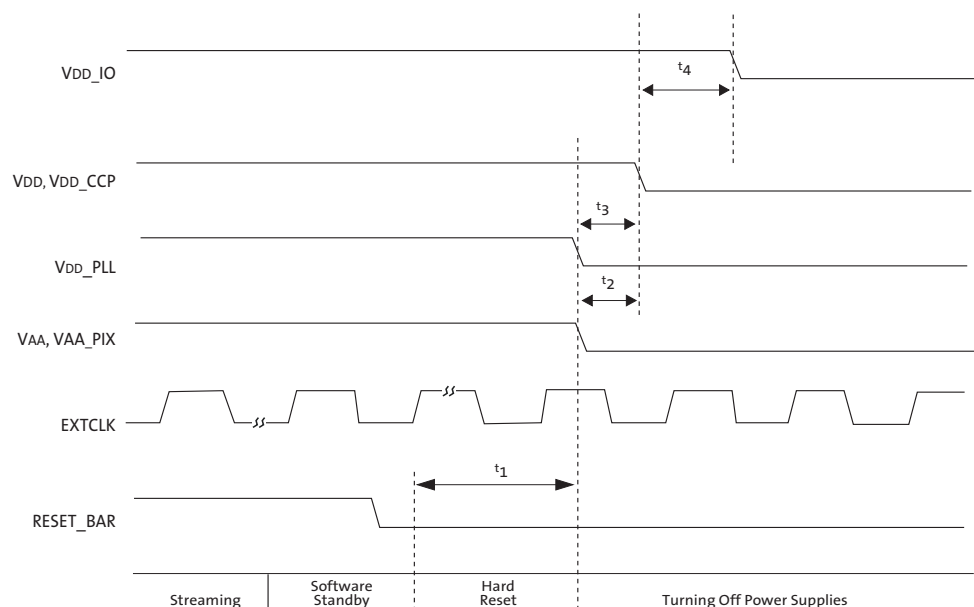



Table 13: Power-Down Sequence

| Definition | Symbol | Min | Typ | Max | Unit |
|-----------------------------|--------|-----|-----|-----|------|
| Hard reset | t_1 | 1 | – | – | ms |
| VDD/VAA/VAA_PIX to VDD time | t_2 | 0 | – | 500 | ms |
| VDD_PLL to VDD time | t_3 | 0 | – | 500 | ms |

Hard Standby and Hard Reset

The hard standby state is reached by the assertion of the RESET_BAR pad (hard reset). Register values are not retained by this action, and will be returned to their default values once hard reset is completed. The minimum power consumption is achieved by the hard standby state. This operating mode complies with section 3.1 of the SMIA Functional Specification.

Soft Standby and Soft Reset

The MT9D014 can reduce power consumption by switching to the soft standby state when the output is not needed. Register values are retained in the soft standby state. Once this state is reached, soft reset can be optionally enabled to return all register values back to the default. The details of the sequence are shown in Figure 26.

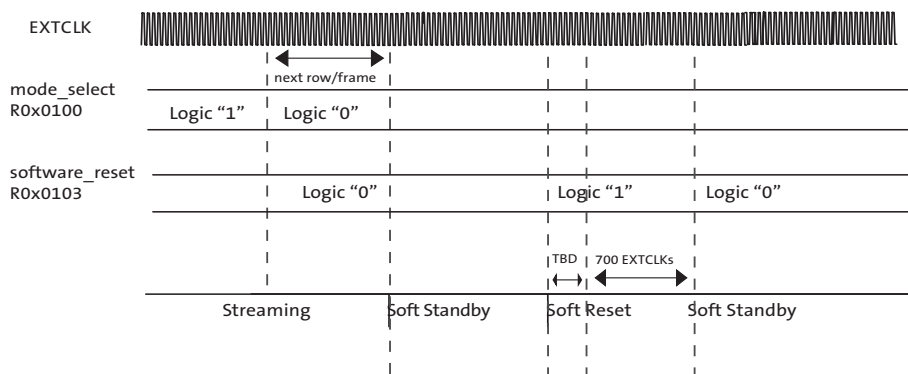
Soft Standby

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

Soft Reset

1. Follow the soft standby sequence list.
2. Set software_reset = 1 (R0x0103) to start the internal initialization sequence.
3. After 700 EXTCLKs, the internal initialization sequence is completed and the current state returns to soft standby automatically. All registers, including software_reset, returns to their default values.

Figure 26: Soft Standby and Soft Reset



Electrical Specifications

EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 14. The EXTCLK input supports an AC-coupled sine-wave input clock or a DC-coupled square-wave input clock.

Table 14: Electrical Characteristics (EXTCLK)

| Definition | Condition | Symbol | Min | Typ | Max | Unit |
|--|--------------------------------------|--------------|-------|------|-----------------------|-----------------|
| Input clock frequency | | f_{EXTCLK} | 6 | 16 | 27 | MHz |
| Input clock period | | t_{EXTCLK} | 166.7 | 62.5 | 37 | ns |
| Input clock amplitude (AC coupled sine wave) | External coupling cap value 50–100pF | V_{IN_AC} | 0.5 | 1.0 | 1.2 | V _{PP} |
| Input clock duty cycle | | | 45 | 50 | 55 | % |
| Input clock jitter | | t_{JITTER} | | | 300 | ps |
| PLL VCO lock time | | t_{LOCK} | | | 6750 ¹ | EXTCLK cycles |
| Input pad capacitance | | | | 2.5 | | pF |
| Input HIGH leakage current | $V_{IN} = V_{DD}$ | I_{IH} | –10 | – | 10 | µA |
| Input LOW leakage current | $V_{IN} = DGND$ | I_{IL} | –10 | – | 10 | µA |
| Input HIGH voltage (DC coupled) | | V_{IH} | 1.0 | – | 2.9 | V |
| Input LOW voltage (DC coupled) | At specified I_{IL} | V_{IL} | –0.5 | – | 0.3 x V _{DD} | V |

Notes: 1. 6750 EXTCLK cycles or 1ms, whichever is smaller.

Two-Wire Serial Register Interface

The electrical characteristics of the two-wire serial register interface (SCLK, SDATA) are shown in Table 15. The SCLK and SDATA signals feature fail-safe input protection, Schmitt trigger input, and suppression of input pulses of less than 50ns.

Table 15: Two-Wire Serial Register Interface Electrical Characteristics

| Definition | Condition | Symbol | Min | Typ | Max | Unit |
|----------------------------------|--|------------|-----------------------|-----|-----------------------|------|
| Input HIGH voltage | | V_{IH} | 0.7 x V _{DD} | – | V _{DD} + 0.5 | V |
| Input LOW voltage | | V_{IL} | –0.5 | – | 0.3 x V _{DD} | V |
| Input leakage current | No pull-up resistor; $V_{IN} = V_{DD}$ or DGND | I_{IN} | – | – | 10 | µA |
| Output LOW voltage | At specified I_{OL} | V_{OL} | – | – | 0.4 | V |
| Output LOW current | At specified V_{OL} | I_{OL} | 8.9 | – | 18.5 | mA |
| Tri-state output leakage current | | I_{OZ} | – | – | 1 | µA |
| Input pad capacitance | | C_{IN} | – | – | 6 | pF |
| Load capacitance | | C_{LOAD} | – | – | 15 | pF |

Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLK_P, CLK_N, DATA_P, and DATA_N) are shown in Table 16

Table 16: Electrical Characteristics (Serial Pixel Data Interface)
VDD = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Ambient Temperature

| Definition | Symbol | Min | Typ | Max | Unit |
|--|--------|------|-----|-----|----------|
| Operating frequency | | 180 | – | 360 | MHz |
| Fixed common mode voltage | VCMF | 0.8 | 0.9 | 1 | V |
| Differential voltage swing | VOD | 100 | 150 | 200 | mV |
| Drive current range | | 0.83 | 1.5 | 2 | mA |
| Drive current variation | | – | – | 15 | % |
| Output impedance | | 40 | – | 140 | Ω |
| Output impedance mismatch | | – | – | 10 | % |
| Clock duty cycle at 416 MHz | | 45 | 50 | 55 | % |
| Rise time (20–80%) | VOD | 300 | – | 400 | ps |
| Fall time (20–80%) | VOD | 300 | – | 400 | ps |
| Differential skew | | – | – | 500 | ps |
| Channel-to-channel slew | | – | – | 200 | ps |
| Maximum data rate | | – | – | | |
| Data/strobe mode | | | | 640 | Mb/s |
| Data/clock mode | | | | 208 | |
| Power supply rejection ratio (PSRR) 0–100 MHz | | 30 | – | – | dB |
| Power supply rejection ratio (PSRR) 100–1000 MHz | | 10 | – | – | dB |

Control Interface

The electrical characteristics of the control interface (RESET_BAR, TEST, GPIO, GPI1, GPI2, and GPI3) are shown in Table 17.

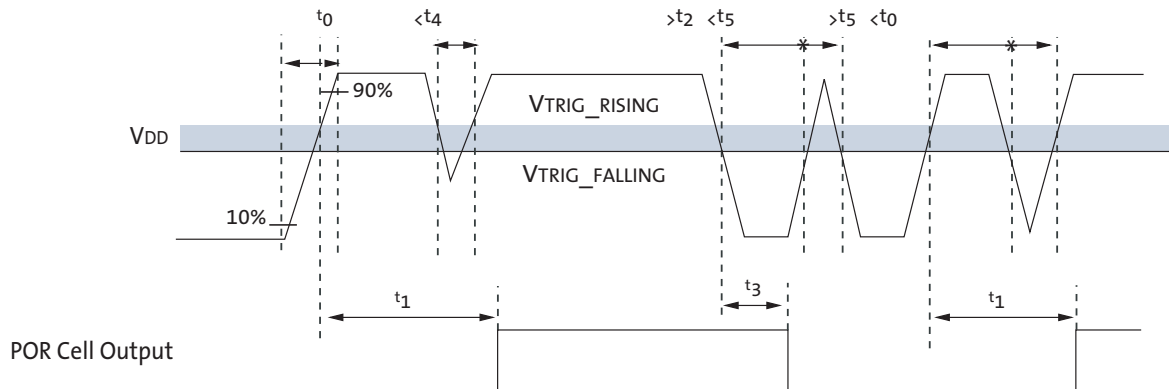
Table 17: Electrical Characteristics (Control Interface)

| Definition | Condition | Symbol | Min | Typ | Max | Unit |
|-----------------------|--|-----------------|-----------------------|-----|-----------------------|------|
| Input HIGH voltage | | V _{IH} | 0.7 x V _{DD} | – | V _{AA} | V |
| Input LOW voltage | | V _{IL} | –0.3 | – | 0.3 x V _{DD} | V |
| Input leakage current | No pull-up resistor; V _{IN} = V _{DD} or DGND | I _{IN} | – | – | 10 | µA |
| Input pad capacitance | | C _{IN} | – | 6.5 | – | pF |

Power-On Reset

Table 18: Power-On Reset Characteristics

| Definition | Condition | Symbol | Min | Typ | Max | Unit |
|--|---|---------------|------|-----|------|---------|
| VDD rising, crossing VTRIG_RISING; Internal reset being released | | t_1 | 7 | 10 | 15 | μ s |
| VDD falling, crossing VTRIG_FALLING; Internal reset active | | t_2 | – | 0.5 | 1 | μ s |
| Minimum VDD spike width below VTRIG_FALLING; considered to be a reset when POR cell output is HIGH | | t_3 | – | 0.5 | – | μ s |
| Minimum VDD spike width below VTRIG_FALLING; considered to be a reset when POR cell output is LOW | | t_4 | – | 0.5 | – | μ s |
| Minimum VDD spike width above VTRIG_RISING; considered to be a stable supply when POR cell output is LOW | While the POR cell output is LOW, all VDD spikes above VTRIG_RISING less than t_5 must be ignored | t_5 | – | 1 | – | ns |
| VDD rising trigger voltage | | VTRIG_RISING | 1.15 | – | 1.55 | V |
| VDD falling trigger voltage | | VTRIG_FALLING | 1.0 | – | 1.45 | V |

Figure 27: Internal Power-On Reset




Operating Voltages

VAA and VAA_PIX must be at the same potential for correct operation of the MT9D014.

Table 19: DC Electrical Definitions and Characteristics

VDD = 1.8V; VAA = 2.8V; VAA_PIX = 2.8V; VDD_PLL = 2.8V; Junction Temperature = 70°C, 30 fps;
 Dark lighting conditions

| Definition | Condition | Symbol | Min | Typ | Max | Unit |
|--|----------------------------|---------|-----|-----|-----|------|
| VDD | | VDD | 1.7 | 1.8 | 1.9 | V |
| VAA | | VAA | 2.4 | 2.8 | 2.9 | V |
| VAA_PIX | | VAA_PIX | 2.4 | 2.8 | 2.9 | V |
| VDD_PLL | | VDD_PLL | 2.4 | 2.8 | 2.9 | V |
| Digital operating current (IDD) | Streaming, full resolution | IDIG | | 42 | 65 | mA |
| Analog operating current (IAA + IAA_PIX + IDD_PLL) | Streaming, full resolution | IANA | | 51 | 80 | mA |
| Hard standby | Analog | | | 1 | 5 | ?A |
| | Digital | | | 1 | 10 | ?A |
| Soft standby (clock off) | Analog | | | 2 | 10 | ?A |
| | Digital | | | 80 | 150 | ?A |
| Soft standby (clock on (6 MHz)) | Analog | | | 4 | 10 | ?A |
| | Digital | | | 270 | 400 | ?A |

Absolute Maximum Ratings

Caution Stresses greater than those listed in Table 20 may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect reliability.

Table 20: Absolute Maximum Values

| Definition | Condition | Symbol | Min | Max | Unit |
|-----------------------|---------------------|-------------|-----------|-----------|------|
| Core digital voltage | | VDD_MAX | -0.3 | 2.2 | V |
| Analog voltage | | VAA_MAX | -0.3 | 3.2 | V |
| Pixel supply voltage | | VAA_PIX_MAX | -0.3 | 3.2 | V |
| PLL supply voltage | | VDD_PLL_MAX | -0.3 | 3.2 | V |
| Input HIGH voltage | | VIH_MAX | 0.7 x VDD | VAA + 0.3 | V |
| Input LOW voltage | | VIH_MAX | -0.3 | 0.3 x VDD | V |
| Operating temperature | Measure at junction | TOP | -30 | 70 | °C |
| Storage temperature | | TSTG | -40 | 125 | °C |

Notes: 1. This is a stress rating only, and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

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. J | 5/25/10 |
| • Updated to non-confidential | |
| Rev. H | 2/4/10 |
| • Update to Aptina template | |
| Rev. G | 6/20/08 |
| • Added "Embedded Data Format and Control" on page 15 | |
| Rev. F | 4/10/08 |
| • Update active imager size in Table 1, "Key Performance Parameters," on page 1 | |
| Rev. E | 3/27/08 |
| • Added "Power-Up Sequence" on page 51 | |
| • Added Figure 24: "Power-Up Sequence," on page 51 | |
| • Added Table 12, "Power-Up Sequence," on page 52 | |
| • Added "Power-Down Sequence" on page 52 | |
| • Added Figure 25: "Power-Down Sequence," on page 52 | |
| • Added Table 13, "Power-Down Sequence," on page 53 | |
| Rev. D | 1/17/08 |
| • Update Table 14, "Electrical Characteristics (EXTCLK)," on page 55 | |
| • Update Table 15, "Two-Wire Serial Register Interface Electrical Characteristics," on page 55 | |
| • Update Table 16, "Electrical Characteristics (Serial Pixel Data Interface)," on page 56 | |
| • Update Table 17, "Electrical Characteristics (Control Interface)," on page 56 | |
| • Update Table 19, "DC Electrical Definitions and Characteristics," on page 58 | |
| Rev. C | 11/20/07 |
| • Update "Shading Correction (SC)" on page 37 | |
| Rev. B | 10/30/07 |
| • Update Table 1, "Key Performance Parameters," on page 1 | |
| • Update Figure 2: "Pixel Color Pattern Detail (Top Right Corner)," on page 7 | |
| • Update Table 6, "Register List and Default Values—SMIA Configuration," on page 1 | |
| • Update Table 7, "Register List and Default Values—SMIA Parameter Limits," on page 3 | |
| • Update Table 8, "Register List and Default Values—Manufacturer-Specific," on page 6 | |
| • Update Table 9, "Register Descriptions—SMIA Configuration," on page 8 | |
| • Update Table 10, "Register Descriptions—SMIA Parameter Limits," on page 13 | |
| • Update Table 11, "Register Descriptions—Manufacturer-Specific," on page 17 | |
| • Update Table 5, "Definitions for Programming Rules," on page 21 | |
| • Update "Power-On Reset Sequence" on page 29 | |
| • Update "Profile 0 Behavior" on page 32 | |
| • Update note in Figure 13: "MT9D014 SMIA Profile 1/2 Clocking Structure," on page 33 | |
| • Update note in Figure 14: "MT9D014 SMIA Profile 0 Clocking Structure," on page 34 | |
| • Update "Programming Restrictions when Subsampling" on page 40 | |
| • Update "Hard Standby and Hard Reset" on page 54 | |
| • Remove "Hard Standby and Hard Reset" figure | |
| • Add "EXTCLK" on page 55 | |



- Add Table 14, “Electrical Characteristics (EXTCLK),” on page 55
- Update Table 15, “Two-Wire Serial Register Interface Electrical Characteristics,” on page 55
- Update Table , “Vdd = 1.8V; Vaa = 2.8V; VAA_PIX = 2.8V; Vdd_PLL = 2.8V; Ambient Temperature,” on page 56
- Update Table 17, “Electrical Characteristics (Control Interface),” on page 56
- Update Table 18, “Power-On Reset Characteristics,” on page 57
- Update Table 19, “DC Electrical Definitions and Characteristics,” on page 58
- Update Table 20, “Absolute Maximum Values,” on page 59

Rev. A9/19/07

- Initial release

10 Eunos Road 8 13-40, Singapore Post Center, Singapore 408600 prodmktg@aptina.com www.apina.com
 Aptina, Aptina Imaging, DigitalClarity, Osmium, and the Aptina logo are the property of Aptina Imaging Corporation
 All other trademarks are the property of their respective owners.
 Preliminary: This data sheet contains initial characterization limits that are subject to change upon full characterization of production devices.