# 1/4-Inch 5 Mp System-On-A-Chip (SOC) CMOS Digital Image Sensor

## MT9P111 Datasheet, Rev. G

For the latest datasheet, please visit www.onsemi.com

## Features

- Superior low-light performance
- Ultra-low-power, low-cost
- Anti-shake support
- One time programmable memory (OTPM) for automatic positional gain adjustments and other uses
- Parallel data output and serial mobile industry processor interface (MIPI) data output
- Integrated real-time JPEG encoder
- Flexible support for external auto focus
- Internal master clock generated by on-chip phase-locked loop (PLL) oscillator
- Electronic rolling shutter (ERS), progressive scan
- Integrated image flow processor (IFP) for single-die camera module
- Automatic image correction and enhancement
- Selectable output data format: YCbCr, 565RGB, 555RGB, 444RGB, JPEG 4:2:2, processed Bayer, RAW8- and RAW10-bit
- Output FIFO for data rate equalization
- Programmable I/O slew rate
- Xenon and LED flash support with fast exposure adaptation
- Configurable gamma correction based on scene brightness
- Arbitrary image scaling with anti-aliasing
- Two-wire serial interface providing access to registers and microcontroller memory, additional serial interface under user control
- Includes internal VCM driver and access to internal A/D converter

## Applications

- Cellular phones
- PC cameras
- PDAs

**Table 1: Key Performance Parameters**

| Parameter | | Value |
|---|---|---|
| Optical format | | 1/4-inch |
| Full resolution | | 2592 x 1944 pixels |
| Pixel size | | 1.4 µm x 1.4 µm |
| Dynamic range | | 62 dB |
| SNR MAX | | 35.2 dB |
| Responsivity | | 0.68 V/lux-sec |
| Chief ray angle | | 25.11° MAX at 80% image height |
| Color filter array | | RGB Bayer pattern |
| Active pixel array area | | 3.62 mm x 2.72 mm |
| Shutter type | | Electronic rolling shutter (ERS) and Global reset release (GRR) |
| Input clock frequency | | 10–48 MHz |
| Maximum frame rate | | 15 fps at full resolution (JPEG), 30 fps in preview mode (VGA bin2 skip2) |
| Maximum pixel data output | | MIPI: 768 Mb/s MAX<br>Parallel: 96 Mp/s |
| Maximum pixel clock frequency | | 96 MHz |
| Supply voltage | Analog | 2.5–3.1 V |
| | Digital | 1.7–1.95 V |
| | I/O | 1.7-1.9V or 2.5–3.1 V |
| | PLL | 2.5–3.1 V |
| | MIPI | 2.5–3.1 V |
| ADC resolution | | 12-bit, on-die |
| Power consumption | | 550 mW at 30 fps, 1280 x 720 video mode |
| | | 401 mW at 30 fps, HP preview mode |
| | | 230 mW at 23 fps, preview LP mode |
| Current consumption | | 10 µA, shutdown, at +70°C |
| Operating temperature (at junction) | | –30°C to +70°C |

# Ordering Information

**Table 2: Available Part Numbers**

| Part Number | Product Description | Orderable Product Attribute Description |
|---|---|---|
| MT9P111D00STCK28AC1-200 | 5 MP 1/4" SOC | Die Sales, 200μm Thickness |

## Table of Contents

## List of Figures

## List of Tables

## MT9P111 Overview

The MT9P111 has a color image sensor with a Bayer color filter arrangement and a 5Mp active-pixel array with electronic rolling shutter (ERS). The sensor core readout is 12-bit, supports skipping and binning, and can be flipped and/or mirrored. The sensor core also supports separate analog and digital gain for all four color channels (R, Gr, Gb, B).

The MT9P111 also has an embedded phase-locked loop oscillator (PLL) that can generate the internal sensor clock from the common clock signals available in typical mobile phone systems. When in use, the PLL adjusts the incoming clock frequency up, allowing the MT9P111 to run at almost any desired resolution and frame rate within the sensor's capabilities. The PLL can be bypassed and powered down to reduce power consumption.

The MT9P111 has numerous power-conserving features including a soft standby mode and a hard standby mode. In standby mode, the sensor can be configured to consume less power than normal operation, with the option of retaining a limited amount of the internal configuration settings. By default, entering standby disables the internal $V_{DD}$ power rail. In addition, there is a SHUTDOWN mode that will disable the power supplies in order to achieve the lowest power consumption possible.

The MT9P111 can be used with either a serial MIPI interface or the parallel data output interface, which has a programmable I/O slew rate to minimize EMI and an output FIFO to eliminate output data bursts. JPEG format can be output in both the MIPI and the parallel data output interfaces. EXIF, MIPI data type support is also included, along with Scalado support.

The advanced image flow processor (IFP) and flexible programmability of the MT9P111 provide a variety of ways to enhance and optimize the image sensor performance. Built-in optimization algorithms enable the MT9P111 to operate at factory settings as a fully automatic, highly adaptable camera; however, most of its settings are user-programmable.

These algorithms include black level conditioning, shading correction, defect correction, noise reduction, color interpolation, color correction, aperture correction, and image formatting such as cropping and scaling.

The MT9P111 also includes a sequencer that coordinates all events triggered by the user. The sequencer manages auto focus, auto white balance, flicker detection, anti-shake, and auto exposure for the different operating modes, which include preview, still capture, video, and snapshot with flash.

All modes of operation are individually configurable and are organized as two contexts. A context is defined by sensor image size, frame rate, resolution, and other associated parameters. The user can switch between the two contexts by sending a command through the two-wire serial interface.

A two-wire serial register interface bus enables read/write access to control registers, variables, and special function registers within the MT9P111. The hardware registers include sensor core controls, color pipeline controls, and output controls.

The general purpose VGPIO can be configured to allow the user extended platform functionality or achieve a 10-bit parallel Bayer output.

# Signal Description

Table 3 provides the signal descriptions for the MT9P111.

**Table 3: Signal Descriptions**

| Name | Type | Description | Notes |
|---|---|---|---|
| STANDBY | Input | Controls sensor's standby mode, active HIGH. | |
| RESET_BAR | Input | Master reset signal, active LOW (can be left floating if not used). | |
| SHUTDOWN | Input | Complete shutdown function for lowest power state (Must be tied to DGND if not used) | |
| EXTCLK | Input | Input clock signal 10–48 MHz. For low leakage, do not overdrive input signal. | |
| VPP | Input | High voltage programming pin for one-time programmable (OTP) memory (must be left floating for normal operation). | |
| SCLK | Input | Slave two-wire serial interface clock from the host processor. | |
| SADDR | Input | Selects device address for the slave two-wire serial interface. The address is 0x78 when SADDR is tied LOW, 0x7A if tied HIGH. | |
| SDATA | I/O | Slave two-wire serial interface data to and from the host processor. | |
| S_SCL | Output | Master two-wire serial interface clock to peripheral devices like AF mechanics. | |
| S_SDA | I/O | Master two-wire serial interface data to peripheral devices like AF mechanics. | |
| VGPIO[7:0] | I/O | General purpose digital I/O, used for auto focus function (can be left floating if not used). | |
| DOUT[7:0] | Output | 8-bit image data output or most significant bits (MSB) of 10-bit SOC bypass mode. If 10-bit Bayer is desired, VGIO[1:0] can be configured to output two least significant bits (LSB). | |
| PIXCLK | Output | Pixel clock. Used for sampling DOUT, FRAME_VALID, and LINE_VALID. | |
| LINE_VALID | Output | Identifies pixels in the active line. | |
| FRAME_VALID | Output | Identifies rows in the active image. | |
| DOUT_N | Output | Differential MIPI data (sub-LVDS, negative) (must be left floating if not used). | |
| DOUT_P | Output | Differential MIPI data (sub-LVDS, positive) (must be left floating if not used). | |
| CLK_N | Output | Differential MIPI clock (sub-LVDS, negative) (must be left floating if not used). | |
| CLK_P | Output | Differential MIPI clock (sub-LVDS, positive) (must be left floating if not used). | |
| VCM_OUT | I/O | VCM actuator driver pad. | |
| VCM_GND | I/O | Ground pad for VCM_OUT. | |
| ATEST0 | I/O | Internal ADC access (leave floating if not used). | |
| ATEST1 | I/O | Internal ADC access (leave floating if not used). | |
| TEST_EN | Input | Test enable (Must be tied to DGND). | |
| VDD | Supply | Digital power (1.8V typical). | |
| VAA_PIX | Supply | Pixel array power (2.8V typical). | |
| VAA | Supply | Analog power (2.8V typical). | |
| VDD_PLL | Supply | PLL power (2.8V typical). | |
| VDD_IO | Supply | I/O power supply (1.8V or 2.8V typical). | |
| GND_IO | Supply | I/O ground. | |
| DGND | Supply | Digital ground. | 1 |
| AGND | Supply | Analog ground. | 1 |
| VDDIO_TX | Supply | I/O power supply for the MIPI output interface, 2.8V typical, Can be disconnected if the interface is not used. | |
| GNDIO_TX | Supply | I/O ground supply for the MIPI output interface. Can be disconnected if the interface is not used). | |

**Table 3: Signal Descriptions**

| Name | Type | Description | Notes |
|---|---|---|---|
| VDD_VGPIO | Supply | I/O power supply for VGPIO[7:0] signals. Can be either 1.8 V or 2.8 V typical. Must be connected even if not used. | |
| GND_VGPIO | Supply | I/O ground for VGPIO[7:0]. Must be connected even if not used. | |

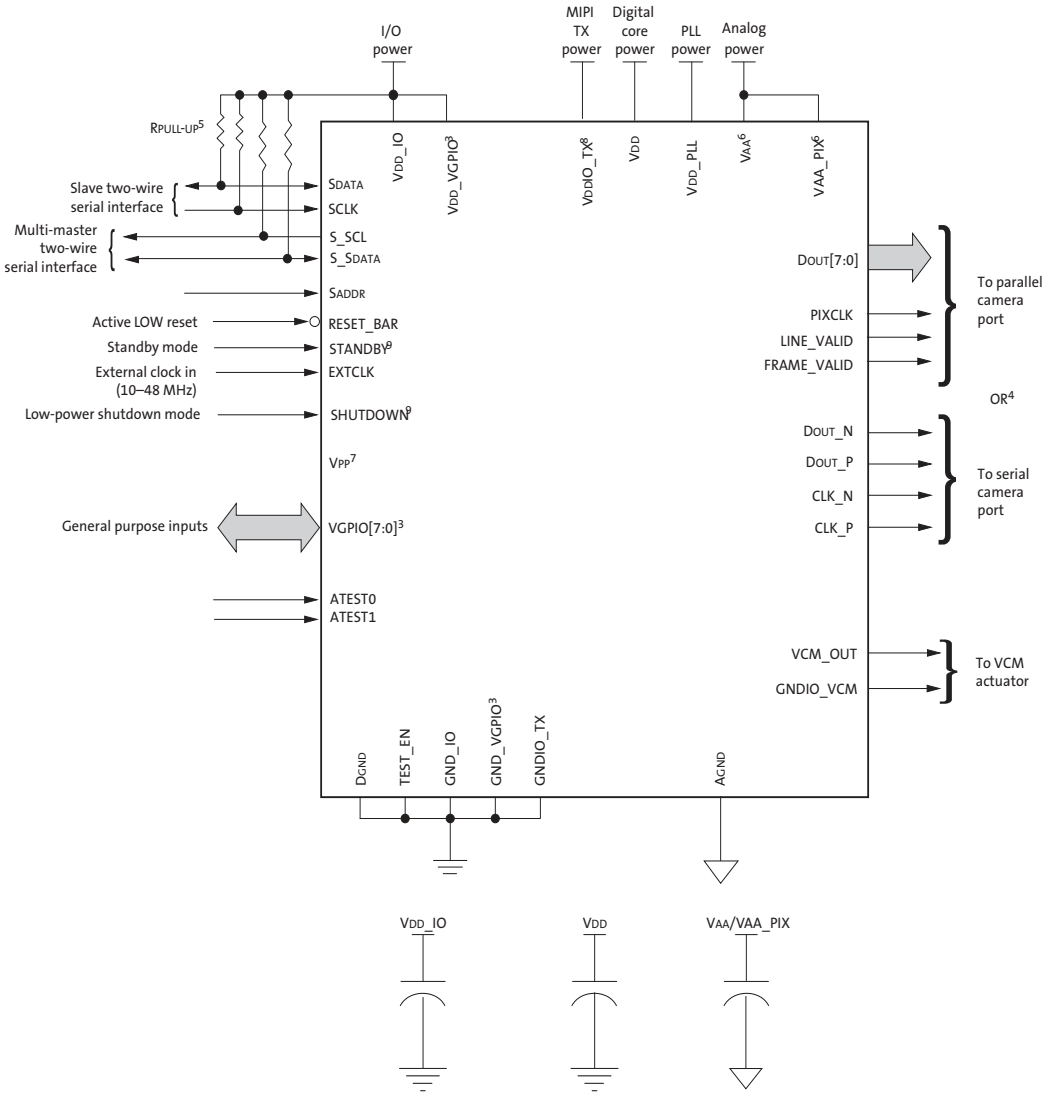
Note: AGND and DGND are not connected internally (inside the chip).

## Typical Connections

Figure 1 on page 9 shows typical MT9P111 device connections. For low-noise operation, the MT9P111 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. ON Semiconductor does not recommend the use of inductance filters on the power supplies or output signals.

The MT9P111 supports different digital core (VDD/DGND), MIPI output (VDDIO_TX/GNDIO_TX), and I/O (VDD_IO/GND_IO) power domains that can be at different voltages. The PLL requires a clean power source (VDD_PLL).

**Figure 1: Typical Configuration (connection)**




Notes:
1. This typical configuration shows only one scenario out of multiple possible variations for this sensor. The minimum recommended decoupling configuration is 0.1µF per supply on module and 10µF off module.
2. If a MIPI interface is not required, the following pads must be left floating: DOUT_P, DOUT_N, CLK_P, and CLK_N.
3. The VGPIO pads can serve multiple features that can be reconfigured. The function and direction will vary by applications. If VGPIO pads are not required, the VDD_VGPIO, GND_VGPIO, and VGPIO[7:0] pads can be left floating.
4. Only one of the output modes (serial or parallel) can be used at any time.
5. ON Semiconductor recommends a resistor value of 1.5KΩ to VDD_IO for the two-wire serial interface Rpull-up; however, greater values may be used for slower transmission speeds.
6. VAA and VAA_PIX must be tied together.
7. VPP is the one-time programmable (OTP) memory programming voltage and should be left floating during normal operation.
8. VDDIO_TX can be connected to VDD_IO if VDD_IO = 2.8V. If the MIPI output is not used, VDDIO_TX can be tied to the VAA supply if an ON Semiconductor-recommended decoupling capacitor is used. VDDIO_TX must be connected to a 2.8V supply.

9. If STANDBY and SHUTDOWN pins are not used, they must be connected to DGND.

## Decoupling Capacitor Recommendations

The minimum decoupling capacitor recommendation is 0.1 µF per supply in the module.

It is important to provide clean, well regulated power to each power supply. The ON Semiconductor recommendation for capacitor placement and values are based on our internal demo camera design and verified in hardware.

**Note:** Since hardware design is influenced by many factors, such as layout, operating conditions, and component selection, the customer is ultimately responsible to ensure that clean power is provided for their own designs.

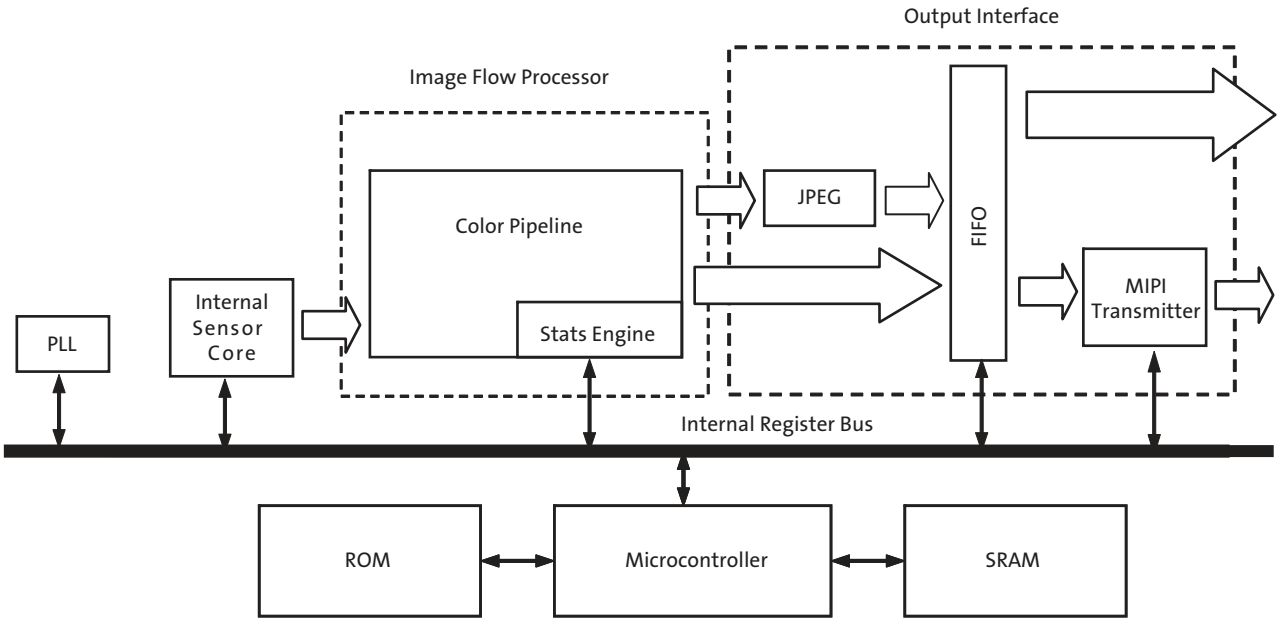
In order of preference, ON Semiconductor recommends:

1. Mount 0.1µF and 1µF decoupling capacitors for each power supply as close as possible to the pad and place a 10 µF capacitor nearby off-module.
2. If module limitations allow for only six decoupling capacitors for a three-regulator design (VDD_PLL tied to VAA), use a 0.1µF and 1µF capacitor for each of the three regulated supplies. ON Semiconductor also recommends placing a 10µF capacitor for each supply off-module, but close to each supply.
3. If module limitations allow for only three decoupling capacitors, a 1µF capacitor for each of the three regulated supplies is preferred. ON Semiconductor recommends placing a 10µF capacitor for each supply off-module but closed to each supply.
4. If module limitations allow for only three decoupling capacitors, a 0.1µF capacitor for each of the three regulated supplies is preferred. ON Semiconductor recommends placing a 10µF capacitor for each supply off-module but close to each supply.
5. Priority should be given to the VAA supply for additional decoupling capacitors.
6. Inductive filtering components are not recommended.
7. Follow best practices when performing physical layout. Refer to technical notes TN-09-131 and TN-09-214.

# Architecture Overview

The MT9P111 combines a 5Mp sensor core with an image flow processor (IFP) to form a stand-alone solution that includes both image acquisition and processing. Both the sensor core and the IFP have internal registers that can be controlled by the user. In normal operation though, an integrated microcontroller autonomously controls most aspects of operation. The processed image data is transmitted to the host system either through a parallel bus or a serial data interface through the output interface.

**Figure 2: SOC Block Diagram**



## Sensor Core Description

The sensor core of the MT9P111 is a progressive-scan sensor that generates a stream of pixel data at a constant frame rate, qualified by LINE_VALID (LV) and FRAME_VALID (FV). The maximum pixel rate is 96 Mp/s, corresponding to a pixel clock rate of 96 MHz. Figure 3 shows a block diagram of the sensor core. It includes a 5Mp 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 12-bit value compressed to a 10-bit value for each pixel in the array.

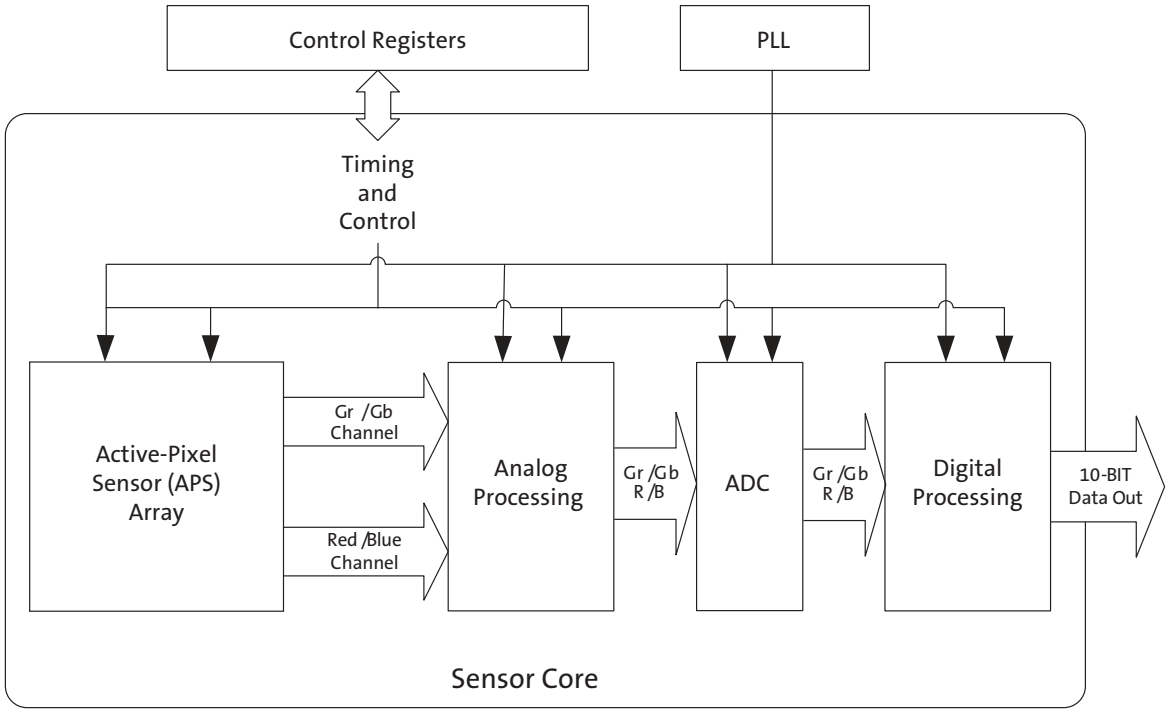
The pixel array contains optically active and light-shielded (dark) pixels. The dark pixels are used to provide data for the offset-correction algorithms (black level control).

The sensor core contains a set of control and status registers that can be used to control many aspects of the sensor behavior including the frame size, exposure, and gain setting. These registers are controlled by the SOC firmware and can be accessed through a two-wire serial interface. Register values written to the sensor core maybe overwritten by firmware.

The output from the core 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.

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

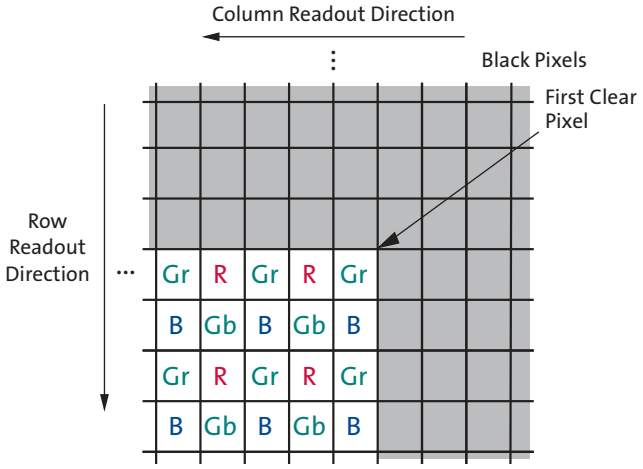
**Figure 3: Sensor Core Block Diagram**



## Pixel Array

The sensor core uses a Bayer color pattern, as shown in Figure 4.

**Figure 4: Pixel Color Pattern Detail (Top Right Corner)**

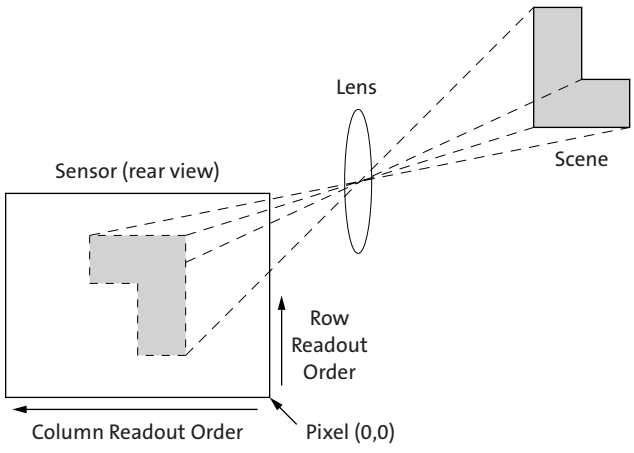



## Default Readout Order

When the sensor is operating in a system, the active surface of the sensor faces the scene as shown in Figure 5.

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

**Figure 5: Imaging a Scene**



## PLL

### PLL-Generated Clocks

The PLL can generate a pixel clock signal whose frequency is up to 96 MHz, using a EXTCLK input of 10 through 48 MHz.

### PLL Setup

Because the input clock frequency is unknown, the sensor starts up with the PLL disabled. The PLL takes time to power up. The behavior of its output clock signal during lock phase is not guaranteed. Another limitation is that the pll_bypass cannot be turned off until after the analog core is powered up. Failure to do so may make the clocking inoperable.

## Digital Processing

### Readout Options

The sensor core supports different readout options to modify the image before it is sent to the IFP. The readout can be limited to a specific window of the original pixel array.

For preview modes, the sensor core supports both skipping and binning in x and y directions.

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

## Window Size

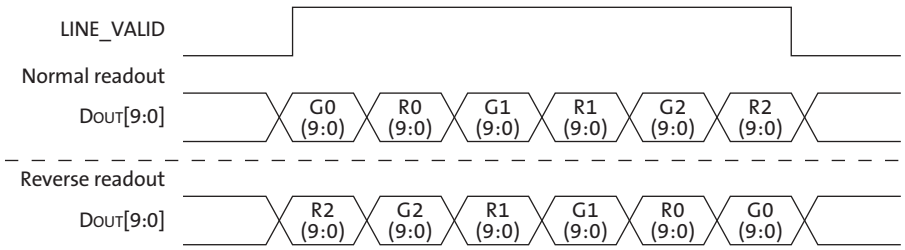
The image output size is set using firmware variables. The edge pixels in the array are present to avoid edge defects and should not be included in the visible window. Binning or skipping will change the image output size.

## Readout Modes

### Horizontal Mirror

When the sensor is configured to mirror the image horizontally, the order of pixel readout within a row is reversed. Figure 6 shows a sequence of 6 pixels being read out with normal readout and reverse readout.

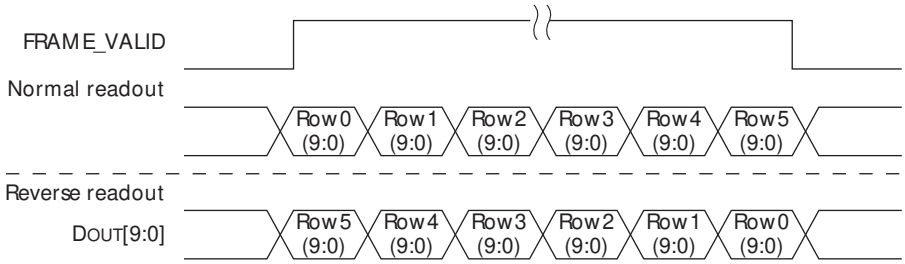
**Figure 6: 6 Pixels in Normal and Column Mirror Readout Modes**




### Vertical Flip

When the sensor is configured to flip the image vertically, the order in which pixel rows are read out is reversed. Figure 7 shows a sequence of 6 rows being read out with normal readout and reverse readout.

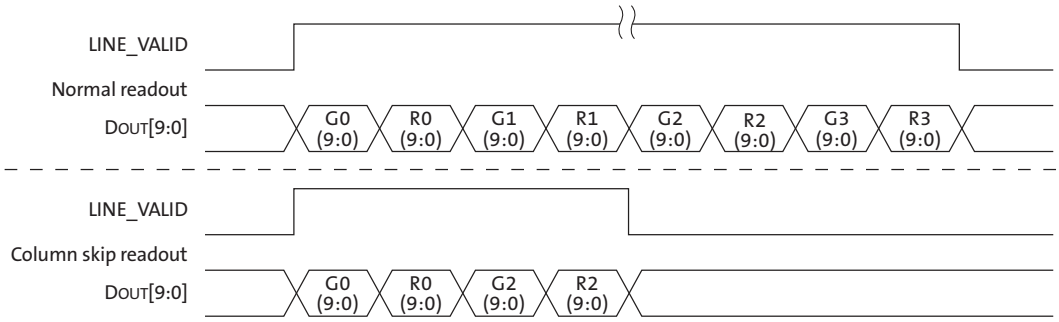
**Figure 7: Six Rows in Normal and Row Mirror Readout Modes**




### Column and Row Skip

The sensor core supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the sensor and thereby allows the frame rate to be increased. This reduces the amount of row and column data processed and is equivalent to the skip2 readout mode provided by earlier ON Semiconductor imaging sensors. When enabling subsampling, the proper image output and crop sizes must be updated beforehand.

**Figure 8: 8 Pixels in Normal and Column Skip 2X Readout Modes**




## Pixel Readouts

The following diagrams show a sequence of data being read out with no skipping. The effect of the different subsampling on the pixel array readout is shown in Figures 9 through 13.

**Figure 9: Pixel Readout (no skipping)**

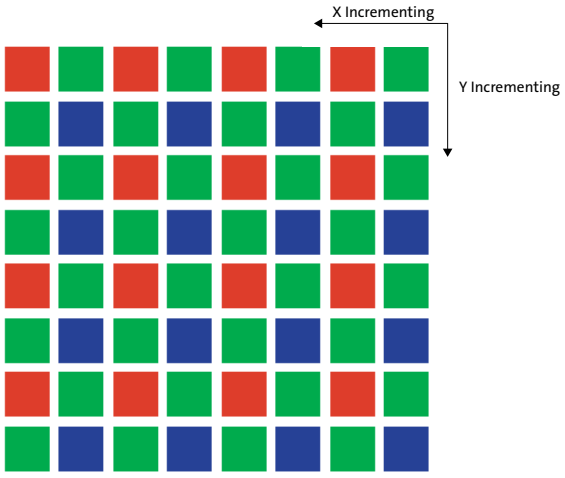


**Figure 10: Pixel Readout (column skipping)**

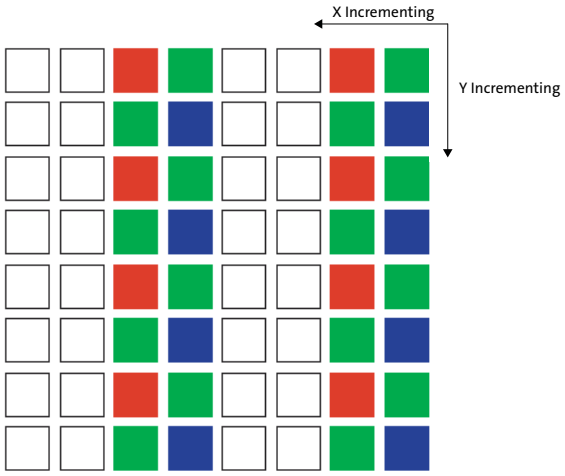



**Figure 11: Pixel Readout (row skipping)**

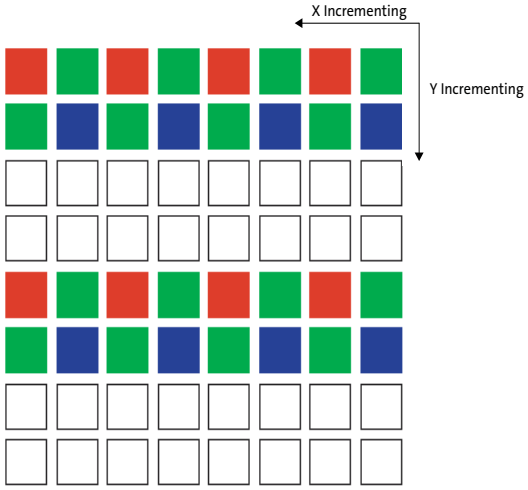


**Figure 12: Pixel Readout (column and row skipping)**

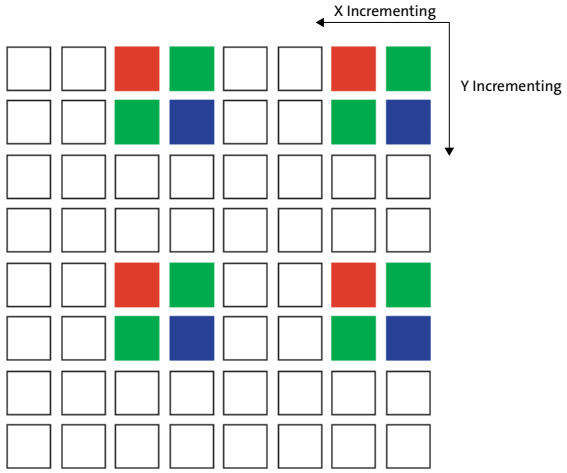



**Table 4: Row Address Sequencing (Sampling)**

| Normal | Subsampled Sequence 1 | Subsampled Sequence 2 |
|---|---|---|
| 0 | 0 | No data |
| 1 | 1 | No data |
| 2 | No data | 2 |
| 3 | No data | 3 |
| 4 | 4 | No data |
| 5 | 5 | No data |
| 6 | No data | 6 |
| 7 | No data | 7 |

## Binning

The MT9P111 sensor core supports 2 x 1, 2 x 2, and Bin2-Skip4 analog binning (column binning, also called x-binning and row/column binning, also called xy-binning). Binning has many of the same characteristics as subsampling but because it gathers image data from all pixels in the active window (rather than a subset of them), it achieves superior image quality and avoids the aliasing artifacts that can be a characteristic side effect of subsampling.

Binning is enabled by selecting the appropriate subsampling settings. Subsampling may require sensor window size adjustment when binning is enabled.

The effect of the different subsampling settings is shown in Figure 13 and Figure 14 on page 19.

**Figure 13: Pixel Readout (column binning)**

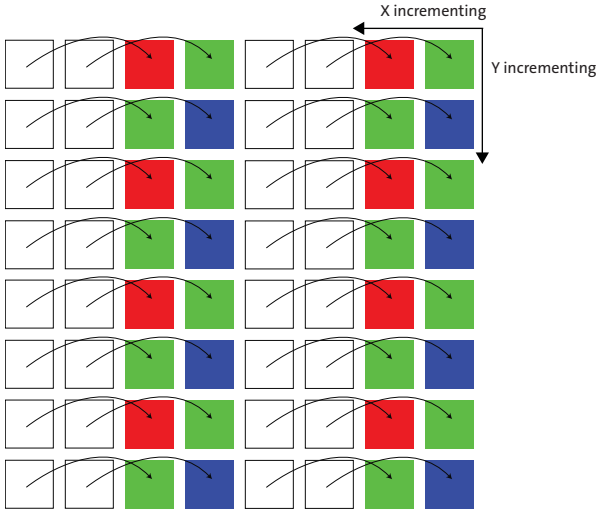
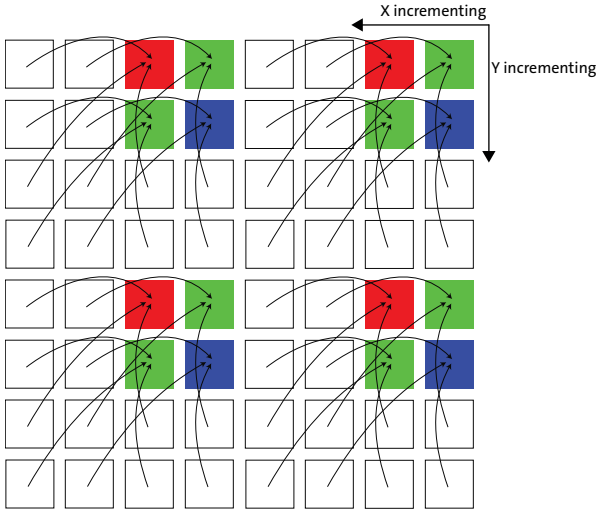



**Figure 14: Pixel Readout (column and row binning)**



### Binning Limitations

The sensor must be taken out of streaming mode before switching between binned and non-binned operation. Binning requires different sequencing of the pixel array and imposes different timing limits on the operation of the sensor. In particular, xy-binning requires two read operations from the pixel array for each line of output data, which has the effect of increasing the minimum line blanking time.

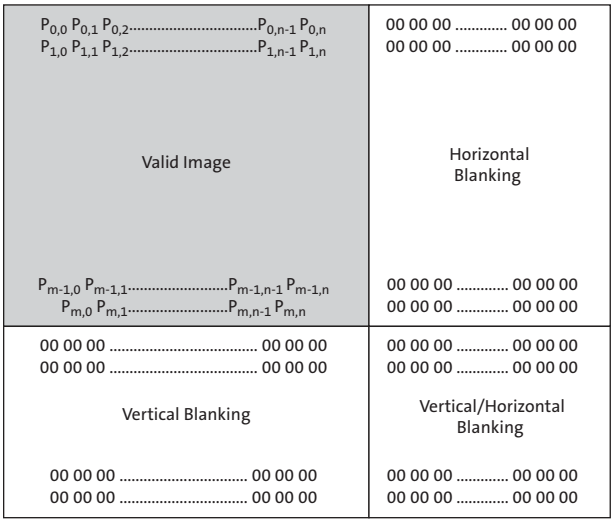
**Table 5: Row Address Sequencing (Binning)**

| Normal | Binning Sequence 1 | Binning Sequence 2 |
|---|---|---|
| 0 | 0,2 | No data |
| 1 | 1,3 | No data |
| 2 | No data | 2,4 |
| 3 | No data | 3,5 |
| 4 | 4,6 | No data |
| 5 | 5,7 | No data |
| 6 | No data | 6,8 |
| 7 | No data | 7,9 |

## Raw Data Format

The sensor core image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking, as shown in Figure 15 on page 20. The amount of horizontal blanking and vertical blanking is programmable. LV is HIGH during the shaded region of the figure.

**Figure 15: Valid Image Data**

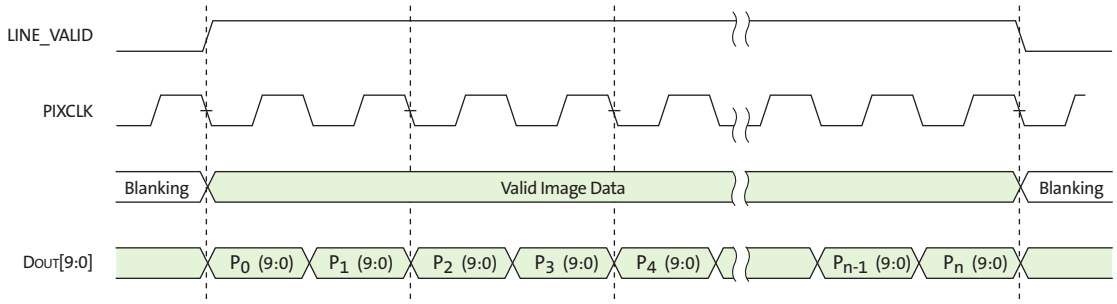



### Raw Data Timing

DOUT[9:0]is synchronized with the PIXCLK output. When LV is HIGH, one pixel's data is output on the 10-bit DOUT output bus every PIXCLK period. By default, the PIXCLK signal runs at the same frequency as the master clock, and its rising edges occur one-half

of a master clock period after transitions on LV, FV, and DOUT (see Figure 16). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled by default (but is configurable), even during the blanking period.

**Figure 16: Pixel Data Timing Example**




## Power Reduction Modes

### Low Power Mode

The MT9P111 supports low power operation during preview mode by reducing the pixel clock frequency. When the MT9P111 enters preview mode with low power mode enabled (Sensor core register 0x3040[9] = 1), the sensor clock will be reduced by half. Internal logic will disable the pixel clock and re-program the divider value. Internal delay will be applied during this change to avoid any clock glitches.

### Dynamic Power Mode

Dynamic Power Mode setting can also be used to significantly reduce the power levels in the sensor. Dynamic power mode will turn off power to the analog portions of the sensor that are not being utilized.

The following registers need to be asserted to enter dynamic power modes:

REG = 0x3170[11] = 1, Enable dynamic power modes

REG = 0x3EDA[6] = 1

REG = 0x3EDA[14] = 1

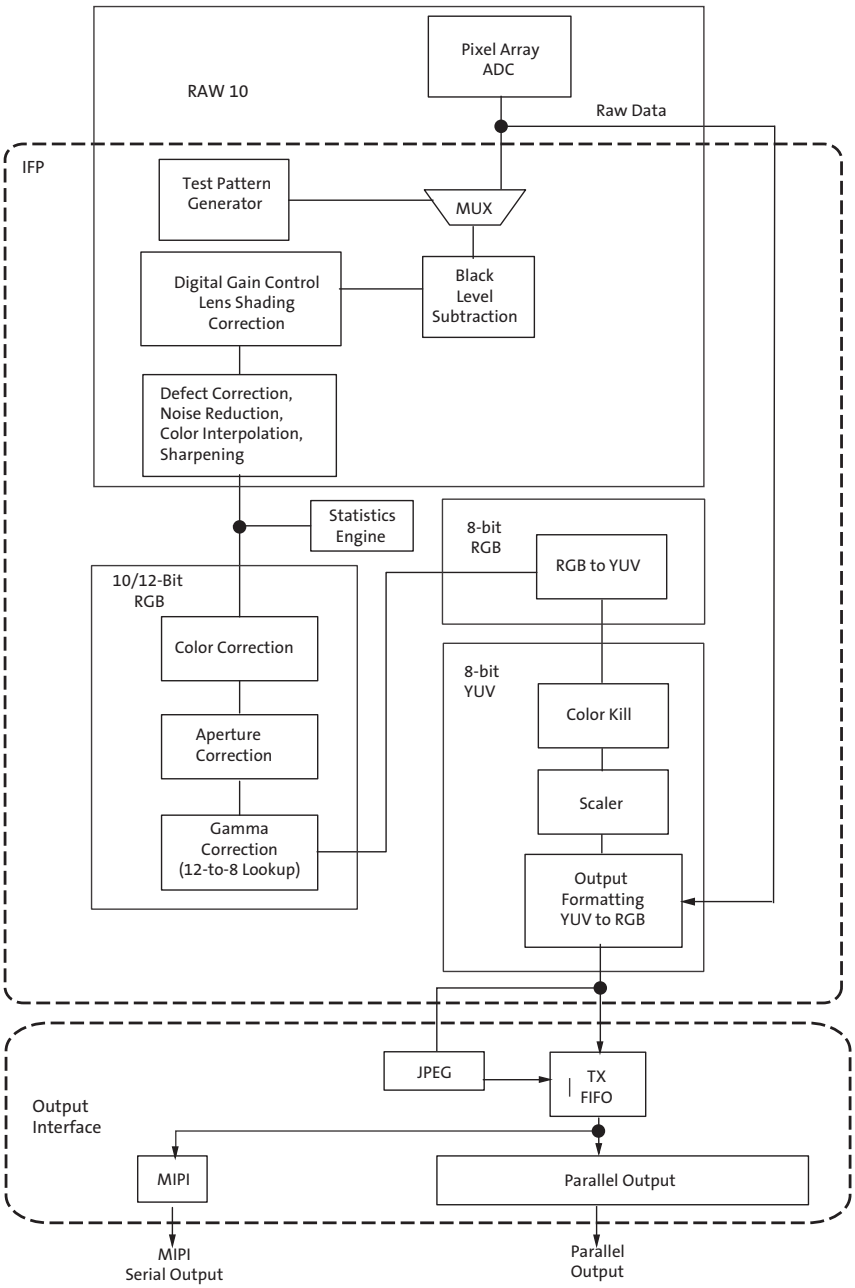
REG = 0x3EDA[15] = 1

# SOC Description

## Image Flow Processor

Image and color processing in the MT9P111 are implemented as an image flow processor (IFP) coded in hardware logic. During normal operation, the embedded microcontroller will automatically adjust the operation parameters. The IFP is broken down into different sections, as outlined in Figure 17.

**Figure 17: Color Pipeline**

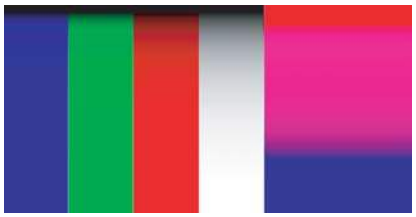


## Test Patterns

During normal operation of the MT9P111, a stream of raw image data from the sensor core is continuously fed into the color pipeline. For test purposes, this stream can be replaced with a fixed image generated by a special test module in the pipeline. The module provides a selection of test patterns sufficient for basic testing of the pipeline.

Test patterns are accessible by programming a register and are shown in Figure 18. Disabling the MCU is recommended before enabling test patterns.

**Figure 18: Color Bar Test Pattern**

| Test Pattern | Example |
|---|---|
| Flat Field |  |
| Vertical Ramp |  |
| Color Bar |  |
| Vertical Stripes |  |
| Pseudo-Random |  |

## Black Level Subtraction and Digital Gain

Image stream processing starts with black level subtraction and multiplication of all pixel values by a programmable digital gain. Both operations can be independently set to separate values for each color channel (R, Gr, Gb, B). Independent color channel digital gain can be adjusted with registers. Independent color channel black level adjustments can also be made. If the black level subtraction produces a negative result for a particular pixel, the value of this pixel is set to "0."

## Automatic Positional Gain Adjustments (APGA)

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 MT9P111 has an embedded shading correction module that can be programmed to counter the shading effects on each individual R, Gb, Gr, and B color signal.

In some cases, different lighting conditions can introduce different color shading response. To compensate for the dependency of the lens shading to the illuminant that can result, different settings of lens shading correction (LC) coefficients can be used. The MT9P111 provides up to three settings to be stored. Each PGA setting should be optimized at a particular color temperature. In the MT9P111, color temperature is detected, stored in the firmware variable ccmPosition, and an appropriate PGA setting is applied.

The variable (ccmPosition) has a range from 0 through 255 and reflects the current color temperature, 0 corresponding to lowest color temperature, 255 the highest. The host specifies a range of ccmPosition values for a particular PGA setting. The ranges should overlap to provide hysteresis and prevent thrashing between PGA settings.

### The Correction Function

For each illuminant, color-dependent solutions are calibrated using the sensor, lens system, and an image of an evenly illuminated, featureless gray 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:

$$P_{corrected}(row,col) = P_{sensor}(row,col) * f(row,col) \quad \text{(EQ 1)}$$

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

## One-Time Programmable Memory

The MT9P111 contains 5Kb of OTP memory, suitable for storing three separate lens shading correction settings, color calibration, external mechanisms, initialization settings, and module identification that can be programmed during the module manufacturing process. Programming the OTP memory requires the use of a high voltage at the VPP pin. During normal operation, the VPP pin should be left floating. The OTP memory can be accessed through the two-wire serial interface. Refer to the MT9P111 Developer Guide for programming procedures.

There is a one-time programmable memory timing calculator available for customer use. Please contact ON Semiconductor engineering support.

## Defect Correction and Noise Reduction

The IFP performs continuous defect correction that can mask pixel array defects such as high dark-current (hot) pixels and pixels that are darker or brighter than their neighbors due to photoresponse nonuniformity. The module is edge-aware with exposure that is based on configurable thresholds. The thresholds are changed continuously based on the brightness of the current scene. Noise reduction can be enabled and disabled and thresholds can be set through register settings.

## Color Interpolation

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

### Color Correction and Aperture Correction

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

To increase image sharpness, a programmable 2D aperture correction (sharpening filter) is applied to color-corrected image data. The gain and threshold for 2D correction can be defined through register settings.

## Image Cropping

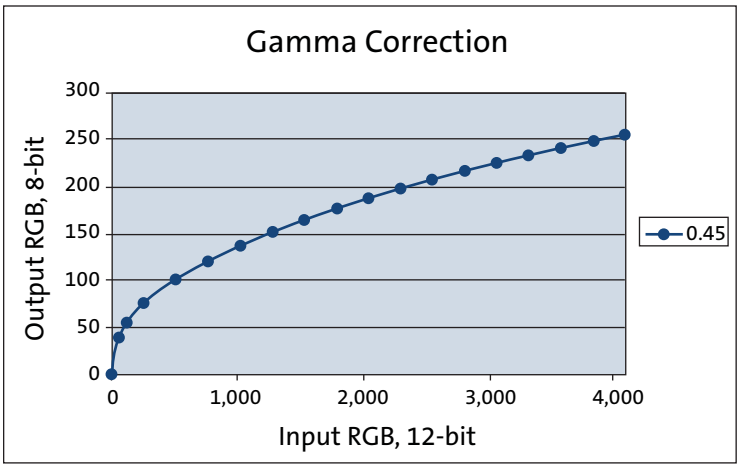
By configuring the cropped and output windows to various sizes, different zooming levels for example 4x, 2x, and 1x can be achieved. The location of the cropped window is also configurable so that panning is also supported. A separate cropped window is defined for context A and context B. In both contexts, the height and width definitions for the output window must be equal to or smaller than the cropped image.

## Gamma Correction

The gamma correction curve (as shown in Figure 19 on page 26) is implemented as a piecewise linear function with 19 knee points, taking 12-bit arguments and mapping them to 8-bit output. The abscissas of the knee points are fixed at 0, 64, 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2304, 2560, 2816, 3072, 3328, 3584, 3840, and 4096. The 8-bit ordinates are programmable through IFP registers.

The MT9P111 IFP includes a block for gamma correction that can adjust its shape based on brightness to enhance the performance under certain lighting conditions (see Figure 19 on page 26). Three custom gamma correction tables may be uploaded corresponding to a brighter lighting condition, a normal lighting condition, and a darker lighting condition. At power-up, the IFP loads the three tables with default values. The final gamma correction table used depends on the brightness of the scene and can take the form of either uploaded tables or an interpolated version of two of the three tables. A single (non-adjusting) table for all conditions can also be used.

**Figure 19: Gamma Correction Curve**




### Special Effects

Special effects like negative image, sepia, or B/W can be applied to the data stream at this point. These effects can be enabled and selected by registers.

### RGB to YUV Conversion

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

### Color Kill

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

### YUV Color Filter

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

### Image Scaling

To ensure that the size of images output by the MT9P111 can be tailored to the needs of all users, the IFP includes a scaler module. When enabled, this module performs rescaling of incoming images—shrinks them to arbitrarily selected width and height without reducing the field of view and without discarding any pixel values.

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

The image cropping and scaler module can be used together to implement a digital zoom and pan. If the scaler is programmed to output images smaller than images coming from the sensor core, zoom effect can be produced by cropping the latter from their maximum size down to the size of the output images. The ratio of these two sizes determines the maximum attainable zoom factor. For example, a 2560 x 1920 image rendered on a 256 x 192 display can be zoomed up to ten times, since 2560/256 = 1920/192 = 10. Panning effect can be achieved by fixing the size of the cropping window and moving it around the pixel array.

If downscaling by 3:1 or more, 2D aperture correction may be applied to increase image sharpness lost due to pixel binning during image scaling.

## YUV-to-RGB/YUV Conversion and Output Formatting

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

## Output Interface (Parallel and MIPI Output)

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

The parallel output is used with an output FIFO whose memory is shared with the MIPI output FIFO to retain a constant pixel output clock independent from the scaling factor.

The MIPI output transmitter implements a serial differential sub-LVDS transmitter capable of up to 768 Mb/s. It supports multiple formats, error checking, and custom short packets.

**Table 6: Data Formats Supported by MIPI Interface**

| Data Format | Data Type |
|---|---|
| YUV 422 8-bit | 0x1E |
| 565RGB | 0x22 |
| 555RGB | 0x21 |
| 444RGB | 0x20 |
| RAW8 | 0x2A |
| RAW10 | 0x2B |
| User-defined byte-based data (including compressed data) | 0x30<br>0x31<br>0x32<br>0x33 |

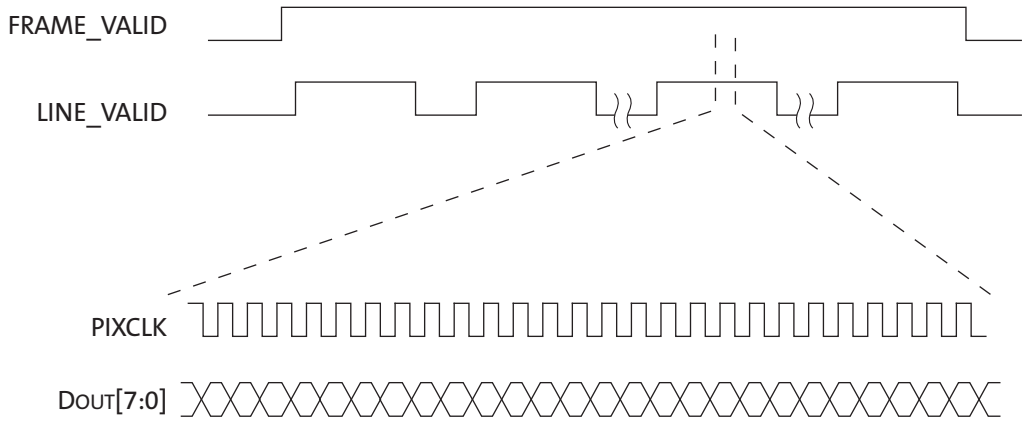
Notes: 1. Data will be packed as RAW8 if the data type specified does not match any of the above data types.

## Output Format and Timing

### YUV/RGB Output

Figure 20 depicts the output timing of YUV/RGB when a scaled data stream is equalized by buffering or when no scaling takes place. The pixel clock frequency remains constant during each LV high period.

**Figure 20: Timing of Full Frame Data or Scaled Data Passing Through the FIFO**




### YUV/RGB Data Ordering

The MT9P111 supports swapping YCbCr mode, as illustrated in Table 7.

**Table 7: YCbCr Output Data Ordering**

| Mode | Data Sequence | | | |
|---|---|---|---|---|
| Default (no swap) | $Cb_i$ | $Y_i$ | $Cr_i$ | $Y_{i+1}$ |
| Swapped CrCb | $Cr_i$ | $Y_i$ | $Cb_i$ | $Y_{i+1}$ |
| Swapped YC | $Y_i$ | $Cb_i$ | $Y_{i+1}$ | $Cr_i$ |
| Swapped CrCb, YC | $Y_i$ | $Cr_i$ | $Y_{i+1}$ | $Cb_i$ |

The RGB output data ordering in default mode is shown in Table 8. The odd and even bytes are swapped when luma/chroma swap is enabled. R and B channels are bit-wise swapped when chroma swap is enabled.

**Table 8: RGB Ordering in Default Mode**

| Mode (Swap Disabled) | Byte | $D_7D_6D_5D_4D_3D_2D_1D_0$ |
|---|---|---|
| 565RGB | Odd | $R_7R_6R_5R_4R_3G_7G_6G_5$ |
| | Even | $G_4G_3G_2B_7B_6B_5B_4B_3$ |
| 555RGB | Odd | $0\ R_7R_6R_5R_4R_3G_7G_6$ |
| | Even | $G_5G_4G_3B_7B_6B_5B_4B_3$ |
| 444xRGB | Odd | $R_7R_6R_5R_4G_7G_6G_5G_4$ |
| | Even | $B_7B_6B_5B_4\ 0\ 0\ 0\ 0$ |
| x444RGB | Odd | $0\ 0\ 0\ 0\ R_7R_6R_5R_4$ |
| | Even | $G_7G_6G_5G_4B_7B_6B_5B_4$ |

## Uncompressed 10-Bit Bypass Output

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

- Using 8 data output signals (DOUT[7:0]) and VGPIO[1:0]. The VGPIO signals are the least significant 2 bits of data.
- Using only 8 signals (DOUT[7:0]) and a special 8 + 2 data format, shown in Table 9.

**Table 9: 2-Byte RGB Format**

| Byte | Bits Used | Bit Sequence |
|---|---|---|
| Odd bytes | 8 data bits | $D_9D_8D_7D_6D_5D_4D_3D_2$ |
| Even bytes | 2 data bits + 6 unused bits | $0\ 0\ 0\ 0\ 0\ 0\ D_1D_0$ |

# JPEG Encoder

The JPEG compression engine in the MT9P111 is a highly integrated, high-performance solution that provides for low power consumption and full programmability of JPEG compression parameters for image quality control.

The JPEG encoding block is designed for continuous image flow and is ideal for low power applications. After initial configuration for a target application, it can be controlled easily for instantaneous stop or restart. A flexible configuration and control interface allows for full programmability of various JPEG-specific parameters and tables.

## JPEG Encoding Highlights

- Sequential DCT (baseline) ISO/IEC 10918-1 JPEG-compliant
- YCbCr 4:2:2 format compression, but does not support YUV 4:2:0 output format
- Support for JPEG 4:2:0 output for image widths that are less than 1280 pixels
- Support for two pairs of programmable quantization tables
- Quality/compression ratio control capability
- 15 fps JPEG capability at full resolution with or without JFIF- or EXIF-compliant header
- Support for interleaved RGB or YUV thumbnail up to 640 x 480
- Capture color pipe bypass stream (8- or 10-bit), JPEG bypass stream (16-bit), or JPEG encoded stream (8-bit), as programmed by host or microcontroller
- JPEG encoded stream can work in continuous mode or spoof mode
- JPEG encoded stream working in continuous mode can only transmit on the parallel output port
- Thumbnail can be enabled for the JPEG encoded stream in both continuous and spoof mode
- In spoof mode, data is output with programmed spoof frame sizes; dummy pixels may be padded as necessary
- Support for Scalado SpeedTags™
- MIPI data types
- Spoof-frame height can be ignored in spoof mode
- Optional JFIF or EXIF header generation

## JPEG Output Interface

### JPEG Data

JPEG data can be output in both the parallel and the serial MIPI streams. In the parallel output interface, JPEG data is output on the 8-bit parallel bus DOUT[7:0], with FV, LV, and PIXCLK. JPEG output data is valid when both FV and LV are asserted. When the JPEG data output for the frame completes, or buffer overflow occurs, LV and FV are de-asserted.

The MT9P111 can transmit JPEG data using two different formats: JPEG continuous stream and JPEG spoof stream. In both formats, JPEG status segments containing information (resolution, file size, and status) about the image and the offsets of thumbnail data can be inserted into the output streams. The following sections describe the two streaming methods.

### RGB or YCbCr Thumbnail

To support display of captured images without decoding a JPEG file, the MT9P111 can output a resized version of the captured JPEG data as an RGB or YCbCr thumbnail image embedded in the JPEG stream.

This thumbnail image is computed from the same image that is input to the JPEG compressor, and is scaled to a user-programmable size, from 160 x 120 to 640 x 480. The thumbnail size must be configured to be at least two times smaller than the JPEG image size.

This image can be separated by parsing the stream for tags surrounding the embedded image. Alternatively, the embedded image can be extracted without parsing by reading thumbnail data offsets from the thumbnail pointer table. This thumbnail pointer table is optionally output in the image status segment, and contains one entry for each line of thumbnail data.

**Note:** There is a specific usage case that may produce skipped frames when used in JPEG Full Resolution with thumbnail mode. If Spoof full height and No header, SOI/EOI only with status is implemented.

### JPEG Continuous Stream

JPEG continuous stream goes out only through the parallel output interface, and supports the following features:

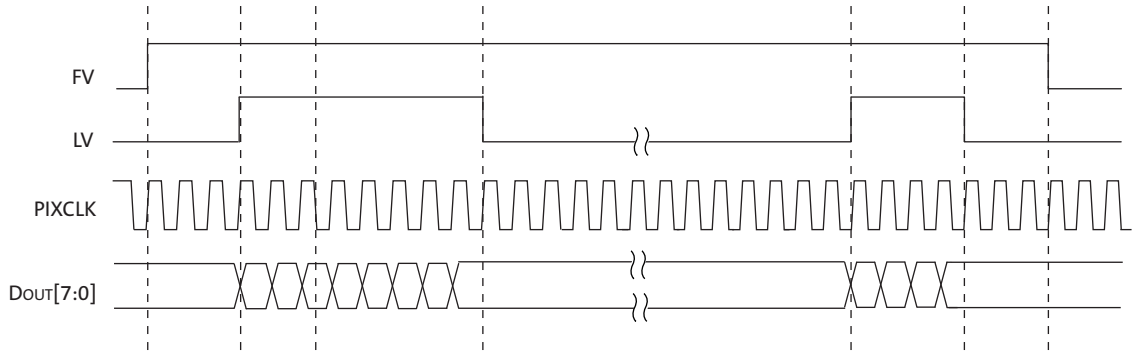
- Adaptive clock switching
- Duplicate FV on LV
- Append JPEG status segment at the end of the data stream

When enabled, the pixel clock output can be generated continuously during invalid data periods (between FV and between LV). In this streaming mode, the amount of valid data within each line (LV = 1) is variable. When adaptive clock mode is enabled, the pixel clock is adjusted to lower clock rates, based on the fullness of the output FIFO. Figure 21 through Figure 24 on page 33 are examples of the JPEG stream through the parallel output interface.

Figure 21 illustrates data output when the pixel clock output is generated continuously during invalid data periods. LV is of variable length based on data output rate.

In default mode, data transitions on the falling of PIXCLK and the host must capture data on the rising edge of PIXCLK. The PIXCLK is also configurable and its polarity can be reversed through the use of register settings.

**Figure 21: JPEG Continuous Data Output**



Notes: 1. Under default conditions FV and LV are asserted on the falling edge of PIXCLK.
2. Data must be captured by the host on the rising edge of PIXCLK.

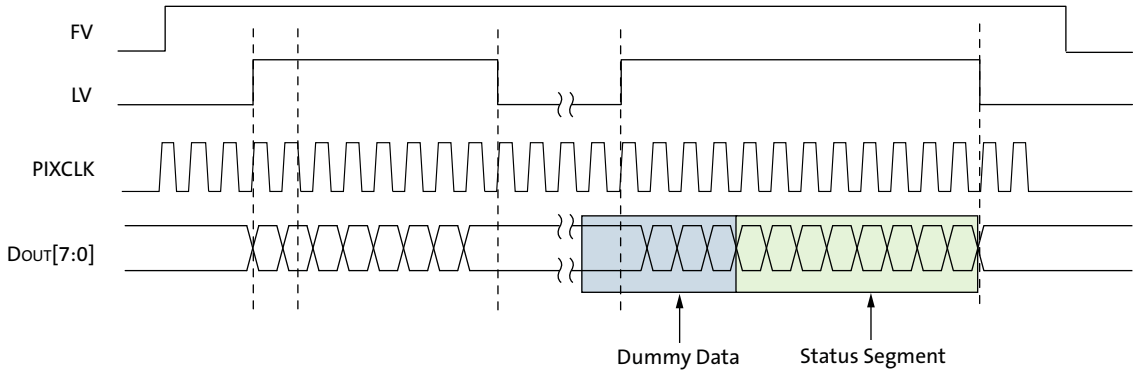
## JPEG Spoof Stream

The JPEG compressed data can be output in spoof mode. The amount of expected pixel data is defined by the width and height registers in spoof mode. If the valid JPEG data is less than expected size defined, a register-programmable dummy data pattern with a default value of 0xFF will be padded.

When enabled, the pixel clock output can be generated continuously during invalid data periods (between FV and between LV). In this streaming mode, the amount of valid data within each line (LV = 1) is constant. When adaptive clock mode is enabled, the pixel clock is readjusted to lower clock rates, based on the fullness of the output FIFO. Below are some examples of the JPEG spoof stream.

Figure 22 illustrates the JPEG spoof output when pixel clock is generated continuously during invalid data periods between LV. The status segment is inserted at the end of the stream.

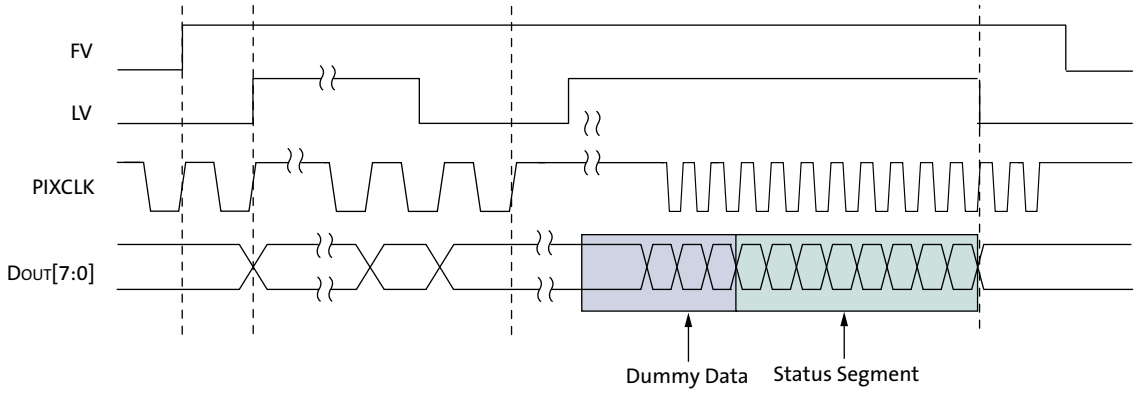
**Figure 22: JPEG Spoof Mode Timing with Continuous Clock**




Notes: 1. PIXCLK is reversed in this example with data output on the rising edge of PIXCLK and data captured by the host on the falling edge of PIXCLK.

Figure 23 illustrates the JPEG spoof output when the adaptive clock mode is enabled. With continuous PIXCLK, the switching of the PIXCLK frequency can happen at any time.

**Figure 23: JPEG Spoof Mode Timing with Adaptive Clock**



Notes: 1. PIXCLK is reversed in this example with data output on the rising edge of PIXCLK and data captured by the host on the falling edge of PIXCLK.

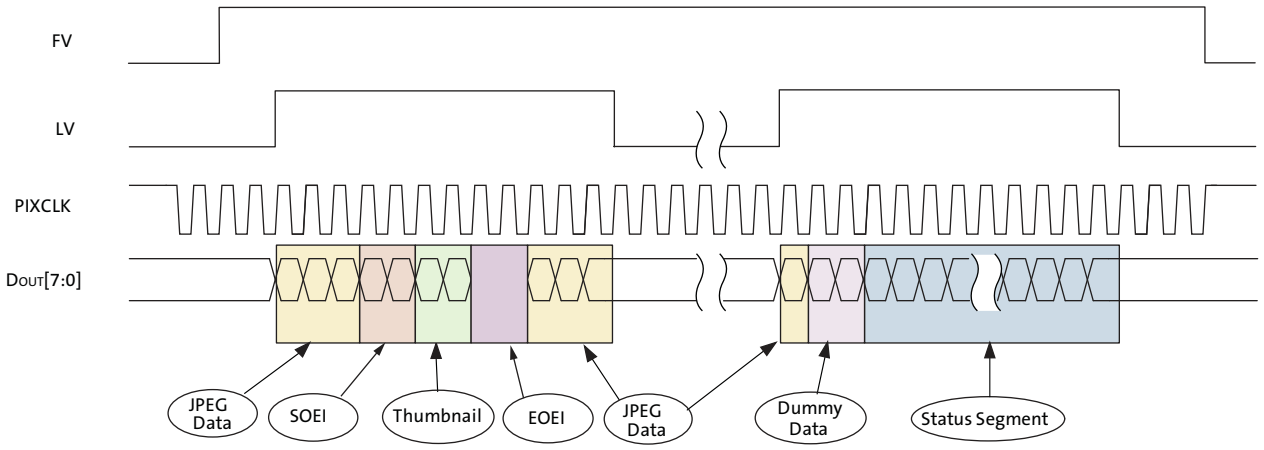
### JPEG Spoof Stream in MIPI Output Mode

In MIPI output mode, only the JPEG spoof stream can be output. Similar to the parallel output interface, the amount of expected pixel data is defined by the width and height registers in spoof mode. If the valid JPEG data is less than expected size defined, register-specified dummy data will be padded.

## JPEG Stream with Embedded Thumbnail Image

In JPEG mode, it is possible to embed a scaled uncompressed image to the compressed data stream. This image is interleaved within the data (as shown in Figure 24), and must be separated before saving the compressed image. The embedded image is separated from the main image by optional Start of Embedded Image (SOEI) and End of Embedded Image (EOEI) tags. These tags are register-programmable codes that enable a host to parse the thumbnail data from the compressed image stream.

**Figure 24: JPEG Spoof Mode Timing with Thumbnail**




Notes: 1. PIXCLK is inverted in this example.
2. Thumbnail start and end codes are programmable by register setting.
3. Status segment includes JPEG pointer table.

In addition, the output formatter can append a table of thumbnail data offsets to the status segment of the image. This thumbnail index pointer shall have one entry for each line of thumbnail data. Each entry is a 4-byte pointer containing the offset of the valid thumbnail data.

## JPEG Status Segment

To provide the user quick knowledge of the status when the JPEG plus thumbnail is enabled, a JPEG status segment is appended at the end of frame. This segment is optional in continuous mode, while it is mandatory for spoof mode.The status segment is enclosed by SOSI/EOSI codes, as shown in Figure 25.

**Figure 25: JPEG Status Segment Structure**

| SOSI on next line (optional) | Thumbnail Index Table Height * 4 bytes (optional) | Thumbnail Size (optional) | Original JPEG Size 4 bytes (optional) | Frame Length (4 bytes) | TXF Status (2 bytes) | TN 2 bytes (optional) | EOSI (0xFFBD) |
|---|---|---|---|---|---|---|---|

The contents of the status segment are summarized as follows:

- SOSI, start of status information, which is coded as 0xFFBC
- Thumbnail index table (every entry has 4 bytes) is asserted and thumbnail is enabled
- The width of thumbnail in pixels (2 bytes)
- The height of thumbnail (2 bytes)
- The width of uncompressed full image
- The height of uncompressed full image
- 4-byte JPEG plus thumbnail length
- 2-byte status
- EOSI, end of status information, which is coded as 0xFFBD
- Options to use to match legacy parts

Either thumbnail data or JPEG data starts first, depending on the time of their availability.

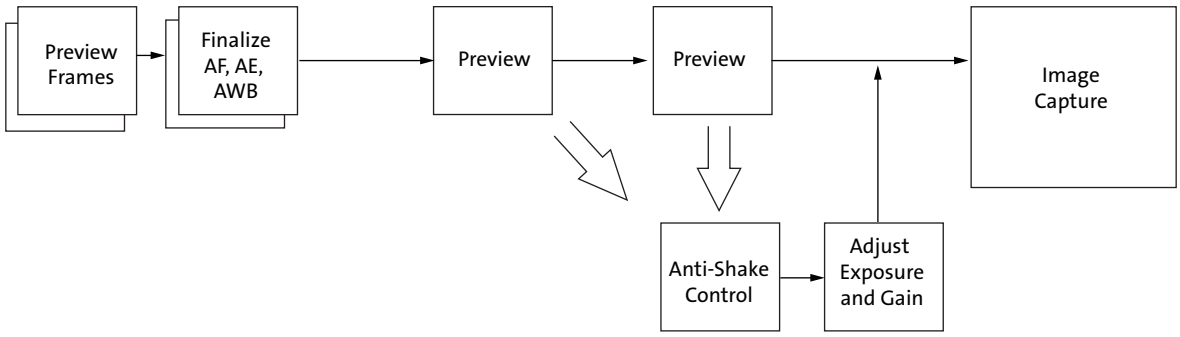
## Scalado SpeedTags™ Support

The MT9P111 supports Scalado SpeedTags™ by inserting markers into the JPEG stream. This is enabled by the register bit TX_SS.jpeg_ctrl.jpeg_insert_rajpeg_markers (R0x3C40[7].

## Anti-Shake (AS)

As mobile devices become smaller, unavoidable handshaking make it difficult for a user to hold a slim mobile camera steady enough to get a flawless shot, especially when exposure time increases due to low light conditions. To reduce motion blur, the MT9P111 includes an anti-shake mode. When motion is detected, it will increase the sensor sensitivity and reduce the exposure time correspondingly. The anti-shake mode will reduce the motion blur caused by camera handshaking. Figure 26 shows the block diagram for the anti-shake algorithm.

**Figure 26: Anti-Shake Algorithm**



# Camera Control

## General Purpose I/Os

The eight general purpose I/Os of the MT9P111 can be configured in multiple ways. Each of the I/Os can be used for multiple purposes and can be programmed from the host. The VGPIOs are powered by their own power supply domain. The VGPIO configurations are shown in Table 10.

If the auto-focus mechanisms are controlled by the serial master, all eight VGPIOs will be available for advanced flash and mechanical shutter operations.

**Table 10: VGPIO Configurations**

| VGPIO[7:0] | Standard Configuration w/ VGPIO as Inputs | Standard Configuration w/ VGPIO as Outputs | Optional Configuration w/ VGPIO as Inputs and Sensor Core Not Needed | Optional Configuration w/ VGPIO as Outputs and Sensor Core Not Needed | Default |
|---|---|---|---|---|---|
| VGPIO[0] | SHUTTER_SEL | Dout_LSB[0] | GPI | GPO_PWM | GPI |
| VGPIO[1] | FLASH_SEL | Dout_LSB[1] | GPI | GPO_PWM | GPI |
| VGPIO[2] | OE_BAR | SHUTTER | GPI | GPO_PWM | GPI |
| VGPIO[3] | GPI | FLASH | GPI | GPO_PWM | GPI |
| VGPIO[4] | GPI | GPO_PWM | GPI | GPO_PWM | GPI |
| VGPIO[5] | GPI | GPO_PWM | GPI | GPO_PWM | GPI |
| VGPIO[6] | GPI | GPO_PWM | GPI | GPO_PWM | GPI |
| VGPIO[7] | GPI | GPO_PWM | GPI | GPO_PWM | GPI |

The general purpose inputs are enabled or disabled through register settings. The state of the general purpose inputs can be read from a register.

## Output Enable Control

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

## Trigger Control

When the global reset feature is in use, the trigger for the sequence can be initiated either under pin or register control.

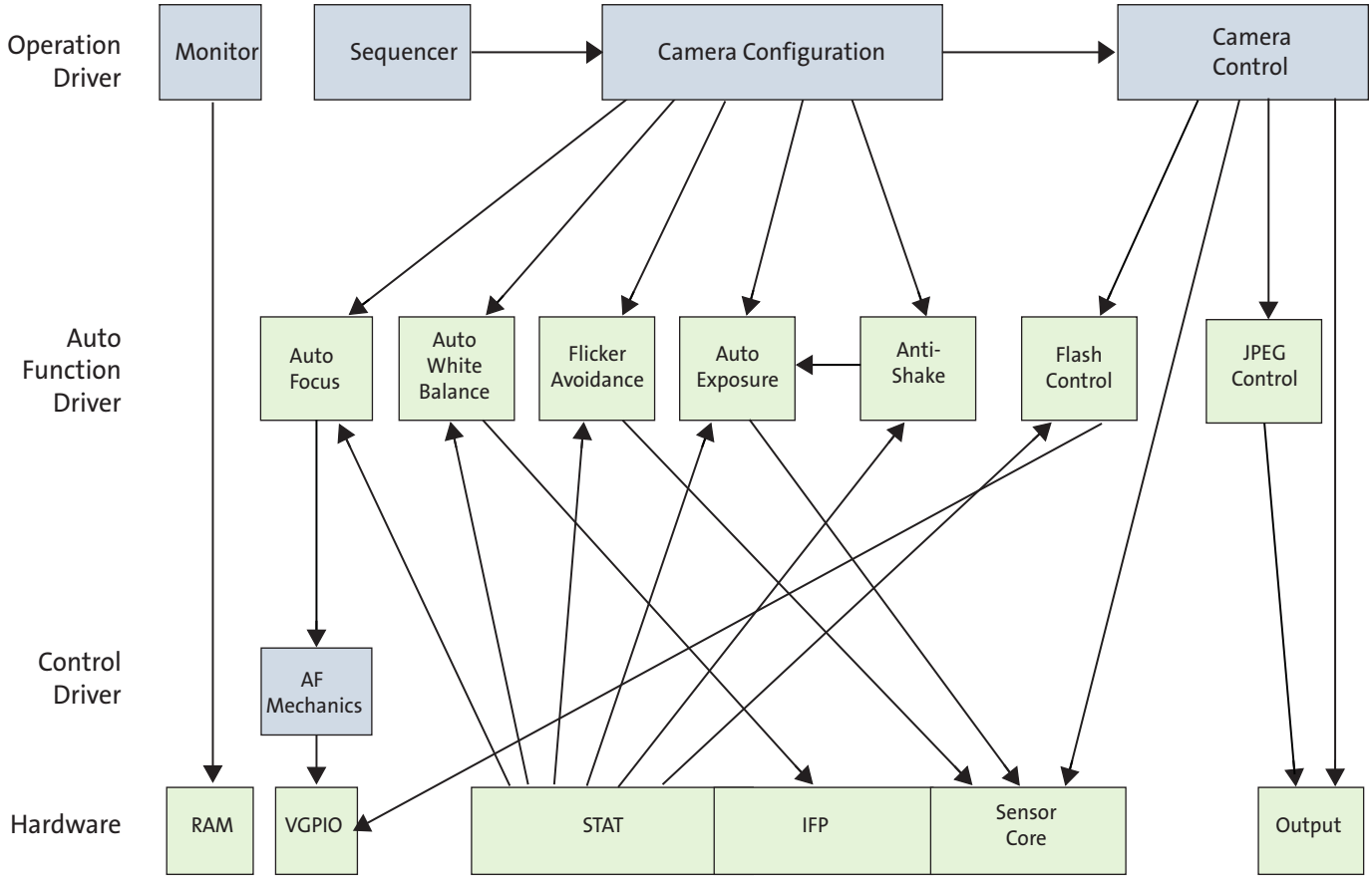
**Table 11: Trigger Control**

| GPI Configured TRIGGER Pin | Global Trigger | Description |
|---|---|---|
| Disabled | 0 | Idle |
| Disabled | 1 | Trigger |
| 0 | 0 | Idle |
| X | 1 | Trigger |
| 1 | X | Trigger |

## Firmware Architecture

The firmware for the MT9P111 is implemented in multiple drivers that are responsible for different parts of operation.

**Figure 27: Firmware Architecture Block Diagram**




## Sequencer

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

## Context and Operational Modes

The MT9P111 can operate in several modes including preview, still capture (snapshot), and full resolution video. All modes of operation are individually configurable and are organized as two contexts—context A and context B. Context switching can be accomplished by sending a command through the two-wire serial interface.

### Preview Mode

Context A is primarily intended for use in the preview mode. During preview, the sensor usually outputs low resolution images at a relatively high frame rate, and its power consumption is kept to a minimum. All automatic functions are enabled in this mode to adjust to the best image possible.

### Still Capture and Video Modes

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

### Snapshot and Flash

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

1. The camera may turn on its LED flash, if it has one and is required to use it. With the flash on, the camera exposure and white balance are automatically adjusted to the changed illumination of the scene.
2. The camera captures one or more frames of desired size. A camera equipped with a xenon flash strobes while capturing images. When capturing images is completed, the camera automatically returns to context A and resumes running in preview mode.

**Note:** This sequence of events can take up to 10 frames.

### Video

To start video capture, the user must change relevant context B settings, such as capture mode, image size and frame rate, and again send a context change command. Upon receiving it, the MT9P111 switches to the modified context B settings, while continuing to output YUV-encoded image data. AE adjusts automatically and provides a smooth continuous operation. To exit the video capture mode, the user must send another context change command, causing the sensor to switch back to context A.

### Multi-Shot Image Capture Mode

The MT9P111 can support Multi-Shot Image Capture Mode (Batch Capture) This mode allows the user to monitor full resolution images, then select a series of images with short intervals to be captured by continually storing full-scale images in a ring buffer (in the customer system) to allow the user to select the optimal image that occurred, before, or after the capture moment.

## Auto Exposure

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

Auto exposure is implemented by a firmware driver that analyzes image statistics collected by the exposure measurement engine, makes a decision, and programs the sensor core and color pipeline to achieve the desired exposure. The measurement engine subdivides the image into a 5 x 5 grid.

The available AE options are described in the AE_rule variables (variable page 9). The average brightness tracking AE uses a constant average tracking algorithm where a target brightness value is compared to a current brightness value, and the gain and integration time are adjusted accordingly to meet the target requirement.

Continuous AE can add weighting to the AE zones such that the center of edge (backlight compensation)-based options are available. In addition, the statistics window can be adjusted to focus on an area of interest.

### AE Driver

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

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

The driver changes AE parameters (integration time, gains, and so on) to drive brightness to the programmable target. The value of the single step approach to the target value can be controlled.

To avoid unwanted reaction of AE on small fluctuations of scene brightness or momentary scene changes, the AE driver uses a temporal filter for luma and a threshold around the AE luma target. The driver changes AE parameters only if the buffered luma is larger than the AE target step and pushes the luma beyond the threshold.

### Accelerated Settling During Overexposure

The AE speed is direction-dependent. Transitioning from oversaturation to target can take more time than transitioning from undersaturation. The AE driver has a mode that speeds up AE for overexposed scenes.

The AE driver counts the number of AE windows whose average brightness is equal to or greater than some value, 250 by default. For a scene having saturated regions, the average luma is underestimated due to signal clipping. The driver compensates underestimation by a factor that can be defined.

### Exposure Control

To achieve the required amount of exposure, the AE driver adjusts the sensor integration time, gains, ADC reference, and IFP digital gains. In addition, a variable is available for the user to adjust the overall brightness of the scene. To reject flicker, integration time is

typically adjusted in increments of steps. The incremental step specifies the duration in row times equal to one flicker period. Thus, flicker is rejected if integration time is kept a natural factor of the flicker period.

## Auto White Balance

The MT9P111 has a built-in auto white balance algorithm designed to compensate for the effects of changing spectra of the scene illumination on the quality of the color rendition. The algorithm consists of two major parts: a measurement engine performing statistical analysis of the image and a driver performing the selection of the optimal color correction matrix, digital, and sensor core analog gains. While default settings of these algorithms are adequate in most situations, the user can reprogram base color correction matrices, place limits on color channel gains, and control the speed of both matrix and gain adjustments. For optimal image quality, ON Semiconductor recommends keeping the analog values less than 2.

## Flicker Detection

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

## Auto Focus

### Overview

The auto focus (AF) algorithm implemented in the MT9P111 firmware seeks to maximize sharpness of vertical lines in images output by the sensor by guiding an external lens actuator to the position of best lens focus. The algorithm is actuator-independent; it provides guidance by means of an abstract one-dimensional position variable, leaving the translation of its changes into physical lens movements to a separate AF mechanics (AFM) driver. The AF algorithm relies on the AFM driver to generate digital output signals needed to move different lens actuators and to correctly indicate at all times if the lens is stationary or moving. The latter is required to prevent the AF algorithm from using line sharpness measurements distorted by concurrent lens motion.

For measuring line sharpness, the AF algorithm relies on the focus measurement engine in the color pipeline, which is a programmable vertical-edge-filtering module. In every interpolated image, statistics are collected in 16 equal-sized rectangular sub-blocks, referred to as AF windows or zones.

There are several motion sequences through which the MT9P111 AF algorithm can bring a lens to best focus position. All these sequences begin with a jump to a preselected start position, for example, the infinity focus position. This jump is referred to as the first flyback. It is followed by a unidirectional series of steps that puts the lens at up to 19 preselected positions different from the start position. This series of steps is called the first scan.

Before and during this scan, the AF algorithm stops the lens at each preselected position long enough to obtain valid sharpness scores. The first normalized score from each AF window is stored as both the worst (minimum) and best (maximum) score for that window. These two extreme scores are then updated as the lens moves from one position to the next and a new maximum position is memorized at every update of the maximum

score. In effect, the preselected set of lens positions is scanned for maxima of the normalized sharpness scores, while at the same time information needed to validate each maximum is being collected.

### Modes

There are two AF camera modes that the MT9P111 can fully support if it controls the position of the camera lens.

#### Snapshot mode

In this mode, a camera performs auto focusing upon a user command to do so. When the auto focusing is finished, a snapshot is normally taken and there is no further AF activity until the next appropriate user command. The MT9P111 can do the auto focusing using its built-in AF algorithm or a substitute algorithm loaded into RAM. It can then wait or automatically proceed with other operations required to take a snapshot.

#### Manual mode

In this mode there is no AF activity—focusing the camera is left to the user. The user typically can move the camera lens in steps, by manually issuing commands to the lens actuator, and observing the effect of his actions on a preview display. The MT9P111 can provide 30 fps image input for the display and simultaneously translate user commands received through the two-wire serial interface into digital waveforms driving the lens actuator.

## Lens Actuator Interface

Actuators used to move lenses in AF cameras can be classified into several broad categories that differ significantly in their requirements for driving signals. These requirements also vary from one device to another within each category. To ensure its compatibility with many different actuators, the MT9P111 includes a general purpose input/output auto focus module.

The VGPIO is a programmable rectangular waveform generator, with eight individually controllable output signals (VGPIO0 through VGPIO7), a separate power supply pad ($V_{DD}$_VGPIO), and a separate clock domain that can be disconnected from the master clock to save power when the VGPIO is not in use. The VGPIO can toggle its output signals as fast as half the master clock frequency.

An external host processor or the embedded microcontroller of the MT9P111 has two ways to control the voltages on the VGPIO output signals:

- Setting or clearing bits in a control register
  The state of the VGPIO signals is updated immediately after writing to the register. Because writing through the two-wire serial interface takes some time, this way does not give the host processor a very precise control over VGPIO output timing.
- Waveform programming
  The second way to obtain a desired output from the VGPIO is to program a set of periodic waveforms to the control registers and initialize their generation. The VGPIO then generates the programmed waveforms on its own, without waiting for any further input, and therefore with the best attainable timing precision. If necessary, the VGPIO can notify the MCU and the host processor about reaching certain points in the waveforms generation, for example, the end of a particular waveform.

The MT9P111 can be set up not only to output digital signals to a lens actuator and/or other similar devices, but also to receive their digital feedback. All VGPIO output signals are reconfigurable as high-impedance digital inputs. The logical state of each VGPIO pad

is mirrored by the state of a bit in a dedicated register, which allows the MCU and host processor to sample digital input signals at intervals equal to their respective register read times.

In addition, the MT9P111 has an additional serial master available for use (S_CLK, S_DAT).

It may be implemented in such a way that if the auto-focus mechanisms are controlled by the serial master, all eight VGPIOs would be available for advanced flash and mechanical shutter operations.

### Internal VCM Driver

The MT9P111 utilizes an internal Voice Coil Motor (VCM) driver. The VCM functions are register-controlled through the serial interface.

There are two output ports, VCM_OUT and GNDIO_VCM, which would connect directly to the AF actuator.

Take precautions in the design of the power supply routing to provide a low impedance path for the ground return. Appropriate filtering would also be required on the actuator supply. Typical values would be a 0.1µF and 10µF in parallel.

**Figure 28: VCM Driver Typical Diagram**

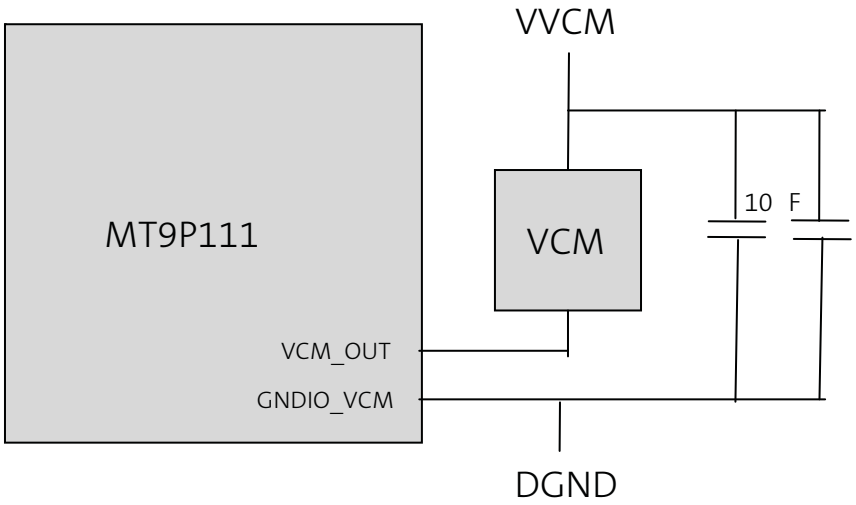



**Table 12: VCM Driver Typical**

| Characteristic | Parameter | Minimum | Typical | Maximum | Units |
|---|---|---|---|---|---|
| VCM_OUT | Voltage at VCM current sink | 2.5 | 2.8 | 3.3 | V |
| WVCM | Voltage at VCM actuator | 2.5 | 2.8 | 3.3 | V |
| INL | Relative accuracy | | ±1.5 | ± 4 | LSB |
| RES | Resolution | | 8 | | bits |
| DNL | Differential nonlinearity | –1 | | +1 | LSB |
| IVCM | Output current | 5 | | 100 | mA |
| | Slew rate | | .3 | | mA/µs |

### User -Accessible Internal ADC

The MT9P111 provides access to an internal 12-bit ADC for customer use. The ADC provides sampling, correction, and filtering.

One application for the internal ADC is to use as an interface to components that require analog feedback support. The access to the internal ADC is through the ATEST0/1 pins and the data is accessible through the serial interface. Refer to the Developer Guide for details.

### Multimaster Serial Interface

The MT9P111 provides, in addition to the standard serial interface (SDATA, SCLK), a secondary master to receive read and write commands from an external host and execute the commands to the attached slave devices through the S_SCLK and S_DATA pins. These could be used for any number of uses to control external slave components such as EEPROM, autofocus, mechanical shutter, and flash drivers.

# Two-Wire Serial Interface

The two-wire serial interface bus enables read/write access to control and status registers within the MT9P111. This interface is designed to be compatible with the MIPI Alliance Standard for Camera Serial Interface 2 (CSI-2) 1.0, which uses the electrical characteristics and transfer protocols of the two-wire serial interface specification.

The interface protocol uses a master/slave model in which a master controls one or more slave devices. The sensor acts as a slave device. The master generates a clock (SCLK) that is an input to the sensor and used to synchronize transfers.

Data is transferred between the master and the slave on a bidirectional signal (SDATA). SDATA is pulled up to VDD_IO off-chip by a 1.5KΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The MT9P111 is a multi-master device. A separate serial master is provided for the control of external components. These are the S_CLK and S_DAT pins.

## Protocol

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

- a start or restart condition
- a slave address/data direction byte
- a 16-bit register address
- an acknowledge or a no-acknowledge bit
- data bytes
- a stop condition

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

### Start Condition

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

### Data Transfer

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

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

### Slave Address/Data Direction Byte

Bits [7:1] of this byte represent the device slave address and bit [0] indicates the data transfer direction. A “0” in bit [0] indicates a write, and a “1” indicates a read. The default slave addresses used by the MT9P111 are 0x78 (write address) and 0x79 (read address). Alternate slave addresses of 0x7A (write address) and 0x7B (read address) can be selected by asserting the SADDR input signal.

### Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data. The protocol used is outside the scope of the two-wire serial interface specification.

### Acknowledge Bit

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

### No-Acknowledge Bit

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

### Stop Condition

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

## Typical Serial Transfer

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

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

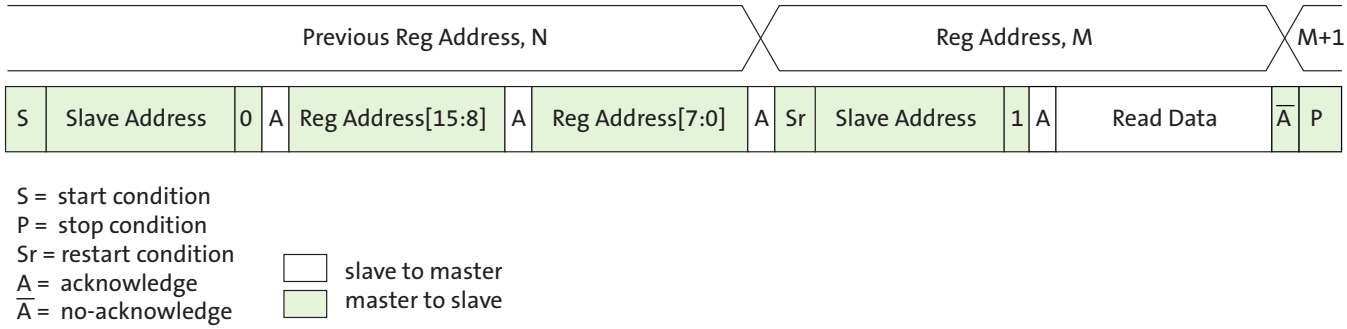
If the request was a read, the master sends the 8-bit write slave address/data direction byte and 16-bit register address, just as in the write request. The master then generates a (re)start condition and the 8-bit read slave address/data direction byte, and clocks out the register data, 8 bits at a 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.

**Note:** If a customer is using direct memory writes (XDMA), AND the first write ends on an odd address boundary AND the second write starts on an even address boundary AND the first write is not terminated by a STOP, the write data can become corrupted. To avoid this, ensure that a serial write is terminated by a STOP.

## Single Read from Random Location

This sequence (see Figure 29) starts with a dummy write to the 16-bit address that is to be used for the read. The master terminates the write by generating a restart condition. The master then sends the 8-bit read slave address/data direction byte and clocks out one byte of register data. The master terminates the read by generating a no-acknowledge bit followed by a stop condition. Figure 29 shows how the internal register address maintained by the MT9P111 is loaded and incremented as the sequence proceeds.

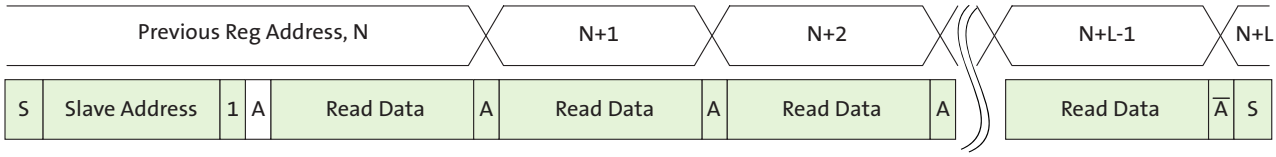
**Figure 29: Single Read from Random Location**




## Single Read from Current Location

This sequence (Figure 30) performs a read using the current value of the MT9P111 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.

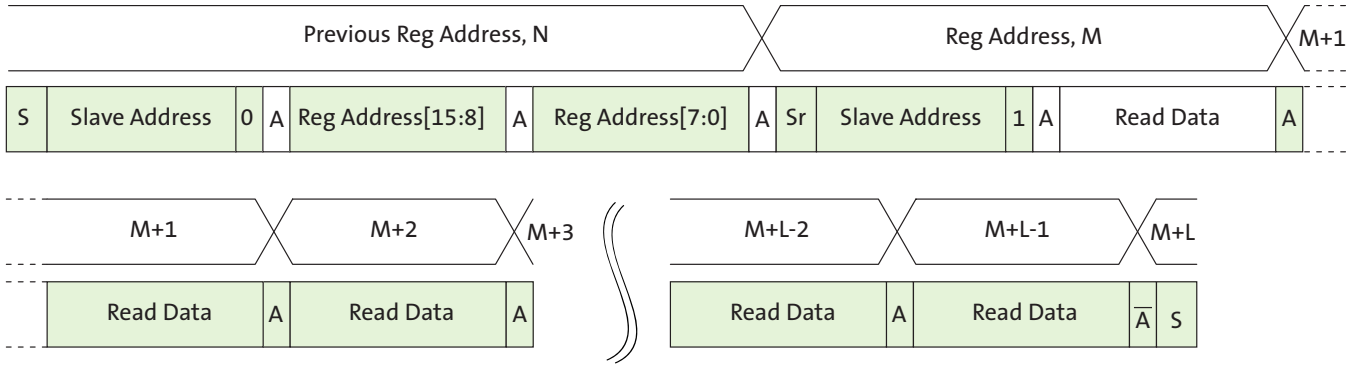
**Figure 30: Single Read from Current Location**



## Sequential Read, Start from Random Location

This sequence (Figure 31) starts in the same way as the single read from random location (Figure 29). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit, and continues to perform byte reads until L bytes have been read.

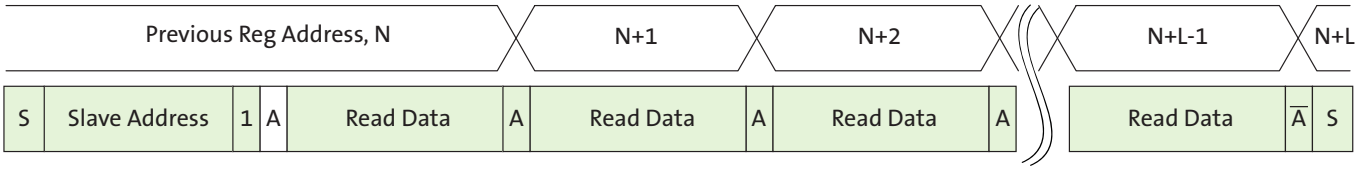
**Figure 31: Sequential Read, Start from Random Location**




## Sequential Read, Start from Current Location

This sequence (Figure 32) starts in the same way as the single read from current location (Figure 30). 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.

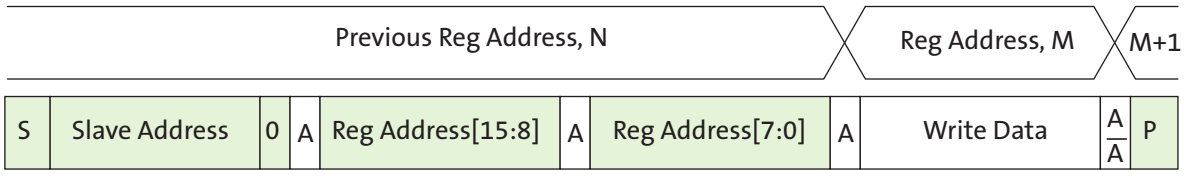
**Figure 32: Sequential Read, Start from Current Location**




## Single Write to Random Location

This sequence (Figure 33) begins with the master generating a start condition. The slave address/data direction byte signals a write and is followed by the high then low bytes of the register address that is to be written. The master follows this with the byte of write data. The write is terminated by the master generating a stop condition.

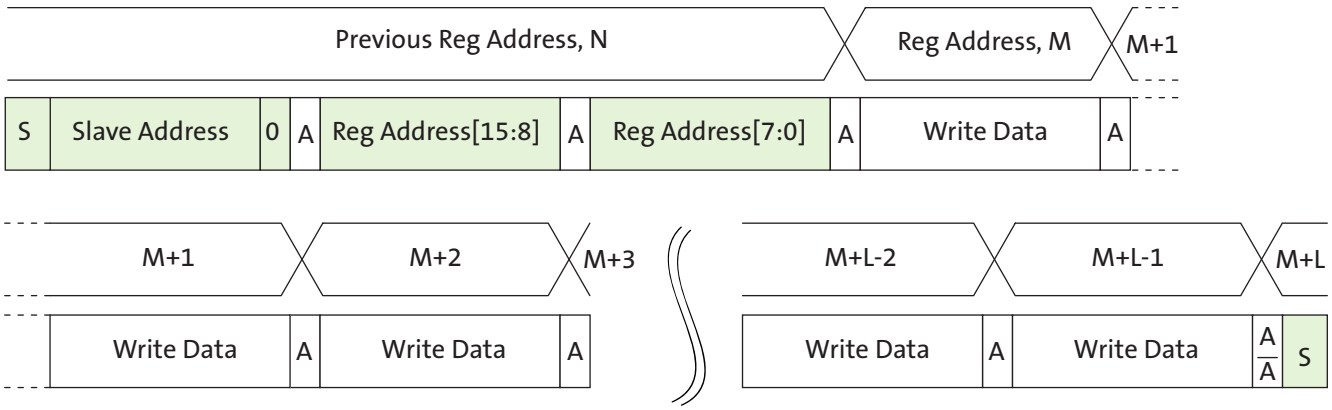
**Figure 33: Single Write to Random Location**



## Sequential Write, Start at Random Location

This sequence (Figure 34) starts in the same way as the single write to random location (Figure 33). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit, and continues to perform byte writes until L bytes have been written. The write is terminated by the master generating a stop condition.

**Figure 34: Sequential Write, Start at Random Location**



# Timing Specifications

## Power-Up Sequence

Powering up the sensor is independent of voltages applied in a particular order, as shown in Figure 35. The timing requirements for other signals are shown in Table 13. It is advised that the user manually assert a hard reset upon power up.

**Caution** **Applying power to analog supplies prior to applying digital and IO supplies follow the correct power-up sequence may result in high current consumption This can potentially result in performance and reliability issues.**

**Figure 35: Power-Up Sequence**

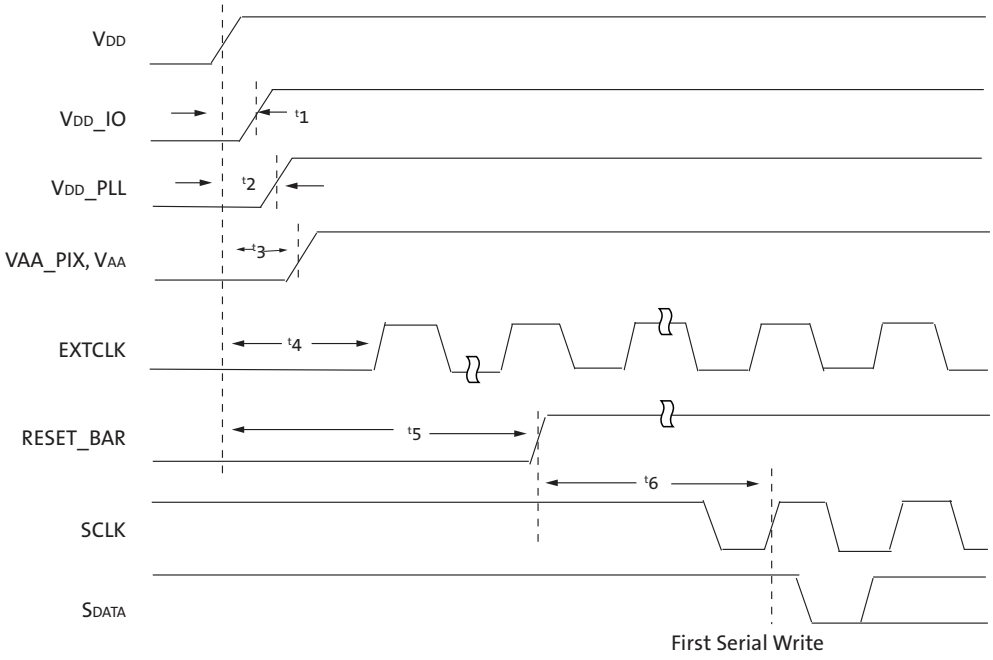



**Table 13: Power-Up Signal Timing**

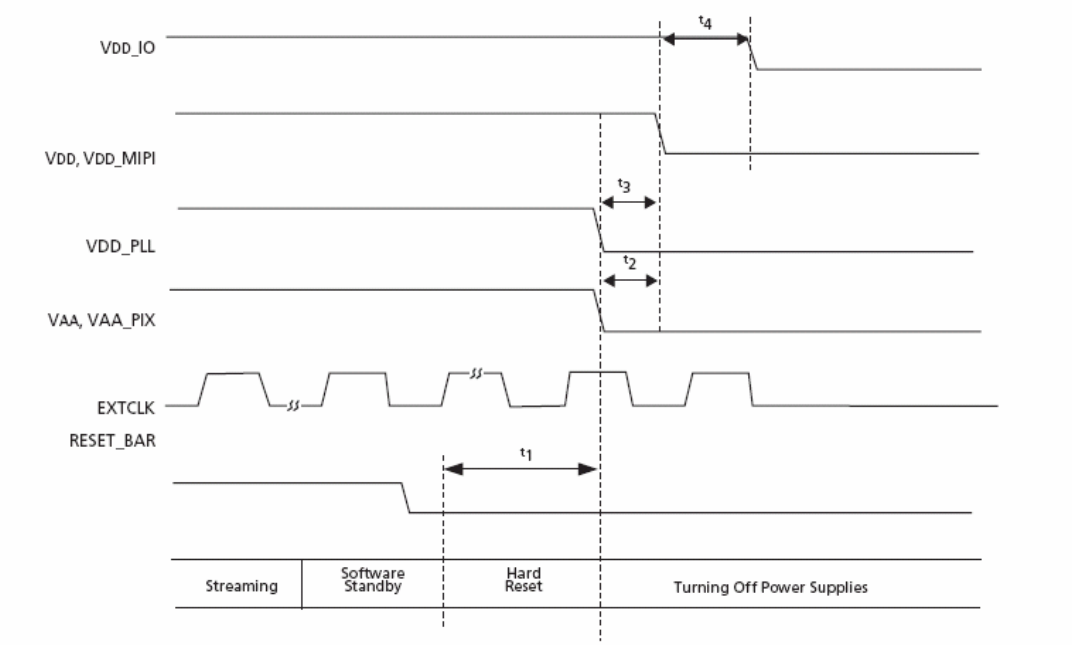
| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| VDD to VDD_IO | $^{t}1$ | 0 | – | 500 | ms |
| VDD to VDD_PLL | $^{t}2$ | 0 | – | 500 | |
| VDD to VAA_PIX | $^{t}3$ | 0 | – | 500 | |
| VDD to EXTCLK Activation | $^{t}4$ | 1 | 500 | – | |
| RESET_BAR activation time | $^{t}5$ | 70 | – | – | EXTCLKs |
| First serial write | $^{t}6$ | 100 | – | – | EXTCLKs |

Notes: 1. All supplies are referenced to VDD.
2. Outputs will be in High Z state while RESET_BAR is asserted.

## Power-Down Sequence

Figure 36 shows the recommended power-down sequence for the MT9P111.

**Figure 36: Power-Down Sequence**




Note: Outputs will be in High Z state while RESET_BAR is asserted.

The best condition for the power down would be turning all the power supplies down at the same time.

Table 14 shows the minimum conditions for power-down sequence.

**Table 14: Power-Down Sequence**

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Hard reset | $t_1$ | 1 | – | – | ms |
| VAA, VAA_PIX to VDD, VDD_MIPI time | $t_2$ | 0 | – | – | ms |
| VDD_PLL TO VDD, VDD_MIPI time | $t_3$ | 0 | – | – | ms |
| VDD, VDD_MIPI to VDD_IO time, | $t_4$ | 0 | – | – | ms |

## Reset

Two types of reset are available:

- A hard reset is issued by toggling RESET_BAR.
- A soft reset is issued by writing commands through the two-wire serial interface.

### Hard Reset

After hard reset, the output FIFO is configured for operation but disabled and all outputs are tri-stated. These outputs can be enabled through the two-wire serial interface. After hard reset, the output FIFO is configured for operation but disabled and all outputs are tri-stated. These outputs can be enabled through the two-wire serial interface. The hard reset signal sequence is shown in Figure 37 on page 51. Hard reset timing is shown in Table 15 on page 51.

**Figure 37: Hard Reset Signal Sequence**

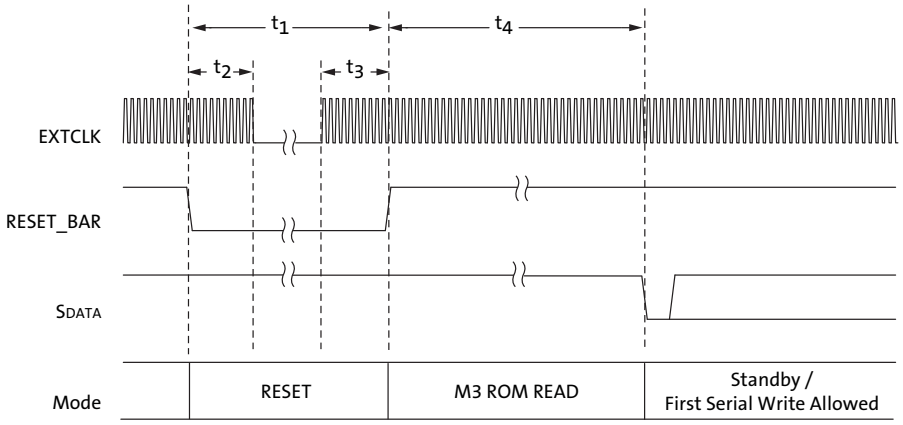



**Table 15: Hard Reset Signal Timing**

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| RESET_BAR pulse width | $t_1$ | 70 | – | – | EXTCLKs |
| Active ECXTCLK after RESET_BAR is asserted | $t_2$ | 10 | – | – | |
| Active EXTCLK before RESET_BAR is de-asserted | $t_3$ | 10 | – | – | |
| First two-wire serial interface communication after RESET is HIGH | $t_4$ | – | 100 | – | |

Note: The MT9P111 does not support the special usage case where $V_{AA}$ = 0 and $V_{DDIO\_TX}$ = 2.8V during reset. Higher leakage currents will occur while RESET_BAR = 0.

### Soft Reset

A soft reset sequence to the sensor has the same affect as the hard reset and can be activated writing to a register through the two-wire serial interface. The soft reset signal sequence is shown in Figure 38. Soft reset timing is shown in Table 16 on page 52. Standard start-up procedures will need to be followed after a soft reset.

**Figure 38: Soft Reset Signal Sequence**

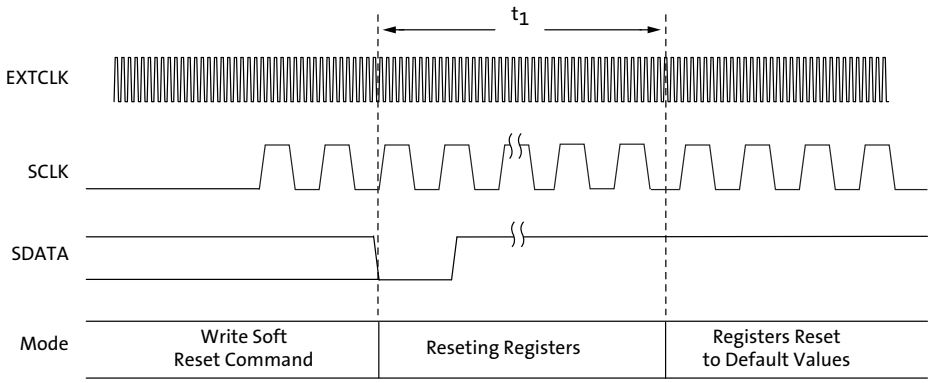



**Table 16: Soft Reset Signal Timing**

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Active EXTCLK after soft reset command is asserted | $t_1$ | – | 120 | – | EXTCLKs |

## Standby Modes

The MT9P111 supports the following standby modes:

- Hard standby
- Soft standby with state retention

For hard and soft standby modes, entry can be inhibited by programming the standby_-control register. To optimize low leakage in standby, a controlled EXTCLK signal mst be used without overshoot.

### Hard Standby Mode

The hard standby mode uses STANDBY to shut down digital power (VDD) and enter low power standby mode. The host will not have to reload the PLL, clock divider settings, and patches but other sensor settings such as context settings, LSC, CCM, and so forth will have to be reloaded. The two-wire serial interface will be inactive and the sensor must be started up by de-asserting STANDBY. During STANDBY, only the sysctrl registers are safely accessible.

The signal sequence is shown in Figure 39. The timing is shown in Table 17.

**Figure 39: Hard Standby Signal Sequence Mode**

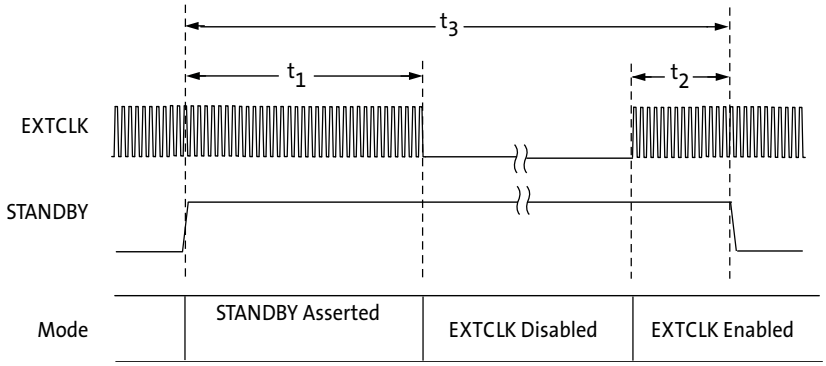



**Table 17: Hard Standby Signal Timing**

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Standby entry complete | $t_1$ | 20 | – | – | µs |
| Active EXTCLK before STANDBY de-asserted | $t_2$ | 10 | – | – | |
| STANDBY pulse width | $t_3$ | 100 | – | – | |
| STANDBY de-assertion to First Allowable Serial Write | $t_4$ | 20 | – | – | |

### Soft Standby with State Retention

Soft standby with state retention can be enabled by register access and disables the sensor core and most of the digital logic. The two-wire serial interface is still active and the sensor can be programmed through register commands. All register settings and RAM content will be preserved. Soft standby can be performed in any sequencer state after all AE, AWB, histogram, and flicker calculations are finished and the sensor core has been disabled.

The execution of standby will take place after the completion of the current line by default. It is possible to synchronize the execution of standby with the end of frame through the standby_control register. The soft standby signal sequence is shown in Figure 40. The timing for the signals is shown in Table 18.

**Figure 40: Soft Standby Signal Sequence**

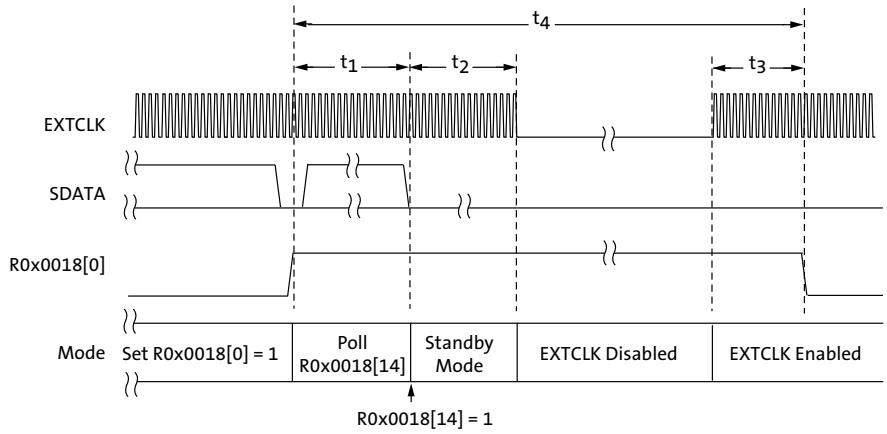



**Table 18: Soft Standby Signal Timing**

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Standby entry complete | $t_1$ | 20 | – | – | µs |
| Active EXTCLK before soft standby de-activates | $t_2$ | 10 | – | – | |
| Minimum standby time | $t_3$ | 100 | – | – | |

## Shutdown Mode

The shutdown mode is entered when the SHUTDOWN pin is asserted. All power to the MT9P111 is disabled and no state, register or patch information is retained. De-assertion of the SHUTDOWN pin will cause a full POR.

**Note:** The MT9P111 does not support the special usage case where $V_{AA}$ = 0 and $V_{DD}$IO_TX = 2.8V during SHUTDOWN. This will cause higher leakage currents to occur.

Refer to the MT9P111 Errata document for special usage notes on Hard Standby and Shutdown modes.

**Table 19: DC Electrical Definitions and Characteristics—Parallel Mode**

$^{f}$EXTCLK = 24 MHz; $^{f}$PIXCLK = 96 MHz; $V_{DD}$ = 1.8V; $V_{DD}$_IO = 1.8V or 2.8V; $V_{AA}$ = 2.8V; VAA_PIX = 2.8V; $V_{DD}$_PLL = 2.8V; $V_{DD}$IO_TX = 2.8V; $T_J$ = 70°C; Refer to "Power Reduction Modes" for descriptions of "Low Power" and "Dynamic Power" modes.

| Symbol | Parameter | Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DD}$ | Core digital voltage | | 1.7 | 1.8 | 1.95 | V |
| $V_{DD}$_IO1 | I/O digital voltage at 1.8V option | | 1.7 | 1.8 | 1.95 | V |
| $V_{DD}$_IO2 | I/O digital voltage at 2.8V option | | 2.5 | 2.8 | 3.1 | V |
| $V_{AA}$ | Analog voltage[3] | | 2.5 | 2.8 | 3.1 | V |
| VAA_PIX | Pixel supply voltage | | 2.5 | 2.8 | 3.1 | V |
| $V_{DD}$_PLL | PLL supply voltage | | 2.5 | 2.8 | 3.1 | V |
| $I_{DD}$ | Digital operating current | Context A (VGA, YCbCr, 10-bit) Low Power mode (R0x3040[9] = 1) | – | 56 | – | mA |
| $I_{AA}$ | Analog operating current | Context A | – | 76 | – | mA |
| $I_{AA}$_PIX | Pixel supply current | Context A | – | 5 | – | mA |
| $I_{DD}$_PLL | PLL supply current | Context A | – | 22 | – | mA |
| $I_{DD}$IO_MIPI | MIPI supply current | Context A | – | 4 | – | mA |
| | Total supply current | Context A | – | 163 | – | mA |
| | Total power consumption | Context A | – | 401 | – | mW |
| $I_{DD}$ | Digital operating current | Context A (VGA,YCbCr,10-bit) High Power mode (R0x3040[9] = 0) with Dynamic Power Mode asserted | – | 74 | – | mA |
| $I_{AA}$ | Analog operating current | Context A | – | 64 | – | mA |
| $I_{AA}$_PIX | Pixel supply current | Context A | – | 7 | – | mA |
| $I_{DD}$_PLL | PLL supply current | Context A | – | 17 | – | mA |
| $I_{DD}$IO_MIPI | MIPI supply current | Context A | – | 4 | – | mA |
| | Total supply current | Context A | – | 166 | – | mA |
| | Total power consumption | Context A | – | 389 | – | mW |
| $I_{DD}$ | Digital operating current | Context A (VGA,YCbCr,10-bit) Low Power mode (R0x3040[9] = 1) with Dynamic Power Mode asserted | – | 54 | – | mA |
| $I_{AA}$ | Analog operating current | Context A | – | 26 | – | mA |
| $I_{AA}$_PIX | Pixel supply current | Context A | – | 5 | – | mA |
| $I_{DD}$_PLL | PLL supply current | Context A | – | 21 | – | mA |
| $I_{DD}$IO_MIPI | MIPI supply current | Context A | – | 4 | – | mA |
| | Total supply current | Context A | – | 110 | – | mA |
| | Total power consumption | Context A | – | 253 | – | mW |
| $I_{DD}$ | Digital operating current | Context B (Full Resolution JPEG, 10 bit) High Power mode (Reg 0x3040[9] = 0) with Dynamic Power Mode asserted | – | 120 | – | mA |

**Table 19: DC Electrical Definitions and Characteristics—Parallel Mode (continued)**

$^{f}$EXTCLK = 24 MHz; $^{f}$PIXCLK = 96 MHz; $V_{DD}$ = 1.8V; $V_{DD}$_IO = 1.8V or 2.8V; $V_{AA}$ = 2.8V; VAA_PIX = 2.8V; $V_{DD}$_PLL = 2.8V; $V_{DD}$IO_TX = 2.8V; $T_J$ = 70°C; Refer to "Power Reduction Modes" for descriptions of "Low Power" and "Dynamic Power" modes.

| Symbol | Parameter | Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{AA}$ | Analog operating current | Context B | – | 98 | – | mA |
| IAA_PIX | Pixel supply current | Context B | – | 10 | – | mA |
| $I_{DD}$_PLL | PLL supply current | Context B | – | 20 | – | mA |
| $I_{DD}$IO_TX | MIPI supply current | Context B | – | 4 | – | mA |
| | Total supply current | Context B | – | 252 | – | mA |
| | Total power consumption | Context B | – | 586 | – | mW |
| Hard standby | Total standby current when asserting the STANDBY signal | $V_{DD}$ Disable ON R0x0028[0] = 1 ($T_J$ = 70°C) | | 10 | | µA |
| Hard standby | Total standby current when asserting the STANDBY signal | $V_{DD}$ Disable OFF R0x0028[0] = 0 ($T_J$ = 70°C) | | 300 | | µA |
| Soft standby (clock on at 24 MHz) | Total standby current when asserting R0x0018[0] = 1 | $V_{DD}$ Disable ON R0x0028[0] = 1 | | 550 | | µA |
| Soft standby (clock on at 24 MHz) | Total standby current when asserting R0x0018[0] = 1 | $V_{DD}$ Disable OFF R0x0028[0] = 0 | | 8500 | | µA |
| Soft standby (clock OFF) | Total Standby Current when asserting R0x0018[0] = 1 | $V_{DD}$ Disable ON R0x0028[0] = 1 | | 10 | | µA |
| Soft standby (clock OFF) | Total Standby Current when asserting R0x0018[0] = 1 | $V_{DD}$ Disable OFF R0x0028[0] = 0 | | 300 | | µA |
| SHUTDOWN (clock ON) | Total standby current when asserting the SHUTDOWN signal | At maximum voltage and clock running | | | 10 | µA |

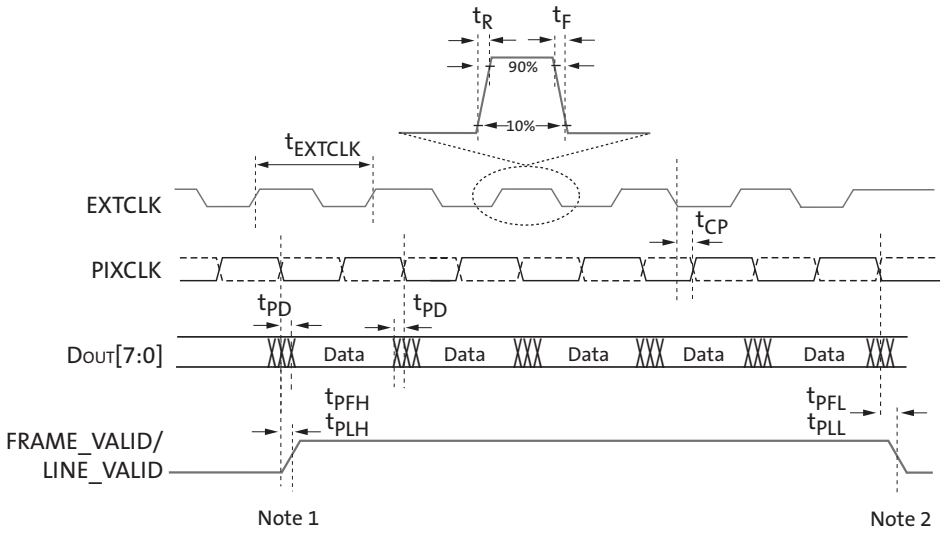
Notes:
1. Context A: 30 fps preview YUV mode.
2. Context B: 15 fps full resolution JPEG mode.
3. In standby mode, VAA should not be grounded
4. To optimize low leakage state in standby, a controlled EXTCLK signal must be used without overshoot

**Table 20: I/O Parameters**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{IH}$ | Input HIGH voltage | At specified $I_{IN}$ | $V_{DD}$_IO – 0.4 | | $V_{DD}$_IO + 0.3 | V |
| $V_{IL}$ | Input LOW voltage | $V_{DD}$_IO = 2.8V at specified $I_{IN}$ | –0.3 | | 0.6 | V |
| | | $V_{DD}$_IO = 1.8V at specified $I_{IN}$ | –0.3 | | 0.4 | V |
| $I_{IN}$ | Input leakage current | No pull-up resistor; $V_{IN}$ = $V_{DD}$ or $D_{GND}$ | –10 | | 10 | µA |
| $V_{OH}$ | Output HIGH voltage | At specified $I_{OH}$ | $V_{DD}$_IO – 0.4 | | – | V |
| $V_{OL}$ | Output LOW voltage | At specified $I_{OL}$ | – | | 0.4 | V |
| $I_{OH}$ | Output HIGH current | At minimum of 1.4V $V_{OH}$ | – | | –13 | mA |
| | | At minimum of 2.4V $V_{OH}$ | – | | –20 | mA |
| $I_{OL}$ | Output LOW current | At maximum of 0.4V $V_{OL}$ | – | | 12 | mA |
| | | At specified $V_{OL}$ | – | | 15 | mA |
| $I_{OZ}$ | Tri-state output leakage current | | | | 5 | µA |

Note: High speed parameters are not included.

**Figure 41: I/O Timing Diagram**




Notes:
1. FV leads LV by 6 PIXCLKs.
2. FV trails LV by 6 PIXCLKs.
3. PLL disabled for $t_{CP}$

**Table 21: I/O Timing Specifications**

$^{f}$EXTCLK = 24 MHz; $^{f}$PIXCLK = 96 MHz; $V_{DD}$ = 1.8V; $V_{DD}$_IO = 1.8V or 2.8V; $V_{AA}$ = 2.8V; VAA_PIX = 2.8V; $V_{DD}$_PLL = 2.8V; $V_{DD}$IO_TX = 2.8V; $T_J$ = 70°C

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit | Note |
|---|---|---|---|---|---|---|---|
| $^{f}$EXTCLK | External clock frequency | | 10 | – | 48 | MHz | |
| $^{t}$EXTCLK | External clock period | | 20.8 | | 100 | ns | |
| | EXTCLK duty cycle | | 45 | 50 | 55 | % | |
| $^{t}$R | EXTCLK rise time (for PLL Bypass mode) | | | | 0.05 * $^{t}$EXTCLK | ns | |
| $^{t}$F | EXTCLK fall time (for PLL Bypass mode) | | | | 0.05 * $^{t}$EXTCLK | ns | |
| $^{t}$JITTER | EXTCLK jitter (peak-peak cycle jitter) | | – | 0.5 | 1 | ns | 1 |
| $^{t}$CP | EXTCLK to PIXCLK propagation delay | | | | | ns | 3 |
| $^{f}$PIXCLK | PIXCLK frequency | Default | 6 | – | 96 | MHz | |
| $^{t}$RPIXCLK | PIXCLK rise time | $C_{load}$ = 15pF | – | 2 | 5 | ns | |
| $^{t}$FPIXCLK | PIXCLK fall time | $C_{load}$ = 15pF | – | 2 | 5 | ns | |
| $^{t}$PD | PIXCLK to data valid | Default | 0.4 * $^{t}$PIXCLK | 0.5 * $^{t}$PIXCLK | 0.6 * $^{t}$PIXCLK | ns | 2 |
| $^{t}$PFH | PIXCLK to FV HIGH | Default | 0.4 * $^{t}$PIXCLK | 0.5 * $^{t}$PIXCLK | 0.6 * $^{t}$PIXCLK | ns | 2 |
| $^{t}$PLH | PIXCLK to LV HIGH | Default | 0.4 * $^{t}$PIXCLK | 0.5 * $^{t}$PIXCLK | 0.6 * $^{t}$PIXCLK | ns | 2 |
| $^{t}$PFL | PIXCLK to FV LOW | Default | 0.4 * $^{t}$PIXCLK | 0.5 * $^{t}$PIXCLK | 0.6 * $^{t}$PIXCLK | ns | 2 |
| $^{t}$PLL | PIXCLK to LV LOW | Default | 0.4 * $^{t}$PIXCLK | 0.5 * $^{t}$PIXCLK | 0.6 * $^{t}$PIXCLK | ns | 2 |
| PIXCLK signal slew | R0x001E[10:8] = 000 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 0.21 | | V/ns | 4 |
| | R0x001E[10:8] = 100 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 0.66 | | V/ns | 4 |
| | R0x001E[10:8] = 111 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 1.2 | | V/ns | 4 |
| DOUT[7:0] signal slew | R0x001E[2:0] = 000 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 0.21 | | V/ns | 4 |
| | R0x001E[2;0] = 100 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 0.66 | | V/ns | 4 |
| | R0x001E[2;0] = 111 | $V_{DD}$_IO: Typ $C_{load}$ = 45pF | | 1.2 | | V/ns | 4 |
| CIN | Input pin capacitance | | – | 2.5 | – | pF | |
| RIN | Input pin impedance | | – | 400 | – | KΩ | |

Notes:
1. Measured in terms of standard deviation.
2. From falling edge of PIXCLK.
3. PLL disabled for $t_{CP}$.
4. PIXCLK = 24 MHz.

## Two-Wire Serial Bus Timing

Figure 42 and Table 22 on page 60 describe the timing for the two-wire serial interface.

**Figure 42: Two-Wire Serial Bus Timing Parameters**

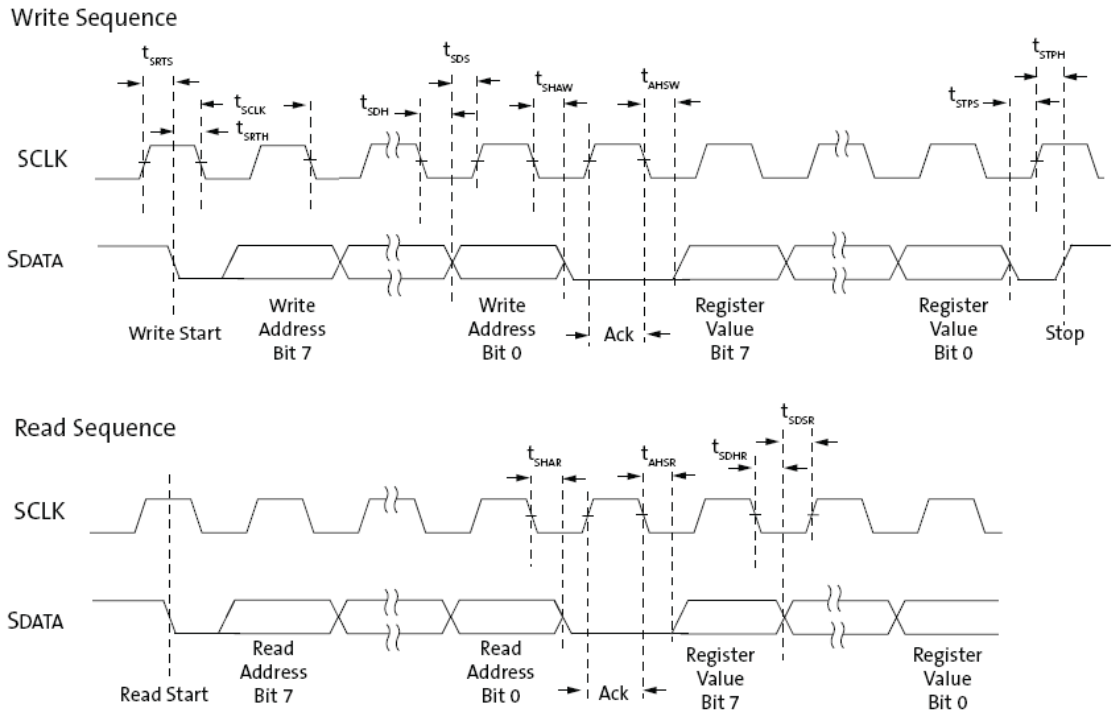


**Table 22: Two-Wire Serial Bus Characteristics**

[f]EXTCLK = 10–48 MHz; $V_{DD}$ = 1.8V; $V_{DD\_IO}$ = 1.8V or 2.8V; $V_{AA}$ = 2.8V; $V_{AA\_PIX}$ = 2.8V; $V_{DD\_PLL}$ = 2.8V; $V_{DD\_PHY}$ = 2.8V; $T_J$ = 70°C

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit | Note |
|---|---|---|---|---|---|---|---|
| [f]SCLK | Serial interface input clock frequency | | 100 | – | 400 | kHz | |
| [t]SCLK | Serial interface input clock period | | 2.5 | – | 10 | ps | Master clock cycle units or PLL cycles if enabled |
| | SCLK duty cycle | | 40 | 50 | 60 | % | |
| [t]SRTH | Start hold time | Write/Read | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SDH | SDATA hold | Write | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SDS | SDATA setup | Write | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SHAW | SDATA hold to ack | Write | 0.15 | – | 0.75 | µs | Master clock cycle units or PLL cycles if enabled |
| [t]AHSW | Ack hold to SDATA | Write | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]STPS | Stop setup time | Write/Read | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]STPH | Stop hold time | Write/Read | 0.6 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SHAR | SDATA hold to ack | Read | 0.15 | – | 0.75 | µs | Master clock cycle units or PLL cycles if enabled |
| [t]AHSR | Ack hold to SDATA | Read | 0.15 | – | 1 | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SDHR | SDATA hold | Read | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| [t]SDSR | SDATA setup | Read | 0.3 | – | – | µs | Master clock cycle units or PLL cycles if enabled |
| $C_{IN\_SI}$ | Serial interface input pin capacitance | | – | – | 3.3 | pF | |
| $C_{LOAD\_SD}$ | SDATA max load capacitance | | – | – | 30 | pF | |
| $R_{SD}$ | SDATA pull-up resistor | | – | 1.5 | – | KΩ | |

**Caution** **Table 23 shows stress ratings only, and functional operation of the device at these or any other conditions above those indicated in the product specification is not implied. Stresses above those listed may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.**

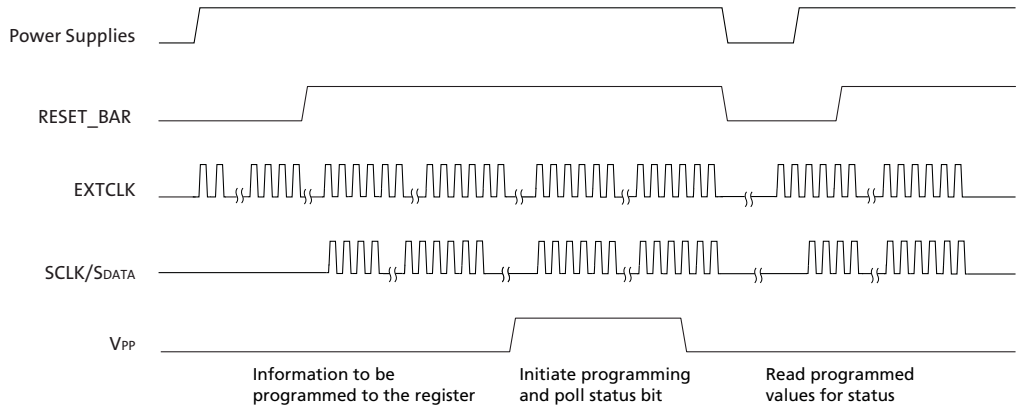
**Table 23: Absolute Maximum Ratings**

| Symbol | Parameter | Rating | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| $V_{DD}$_MAX | Core digital voltage | –0.3 | 2.4 | V |
| $V_{DD}$_IO_MAX | I/O digital voltage | –0.3 | 4.0 | V |
| $V_{AA}$_MAX | Analog voltage | –0.3 | 4.0 | V |
| VAA_PIX_MAX | Pixel supply voltage | –0.3 | 4.0 | V |
| $V_{DD}$_PLL_MAX | PLL supply voltage | –0.3 | 4.0 | V |
| $V_{IH}$_MAX | Input HIGH voltage | | $V_{DD}$_IO + 0.3 | V |
| $V_{IL}$_MAX | Input LOW voltage | –0.3 | | V |
| T_OP | Operating temperature (measured at junction) | –30 | 75 | °C |
| T_ST | Storage temperature | –40 | 85 | °C |

## One-Time Programmable Memory Programming Sequence

Figure 43 shows the sequence of signals to be used for OTP memory programming sequence. The supply voltages and EXTCLK to be used are shown in Table 24 on page 63.

**Figure 43: Sequence of Signals for OTP Memory Operation**




Note: There is a one-time programmable memory timing calculator available for customer use. Please contact ON Semiconductor engineering support.

**Table 24: Supplies Voltages and Clock Frequency for OTP Memory Programming**

| Symbol | Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $^{f}$CLKIN | Input clock frequency | 10 | 12 | 48 | MHz |
| VDD | Core digital voltage | | 1.8 | | V |
| VDD_IO | I/O digital voltage | | 1.8 or 2.8 | | V |
| VAA | Analog voltage | 2.6 | 2.8 | 3.1 | V |
| VAA_PIX | Pixel supply voltage | | 2.8 | | V |
| VDD_PLL | PLL supply voltage | | 2.8 | | V |
| VPP | Programming voltage | | 8.5 | | V |
| VDDIO_TX | MIPI supply voltage | | 2.8 | | V |

## Signal States

**Table 25: Status of Signals During Different States**

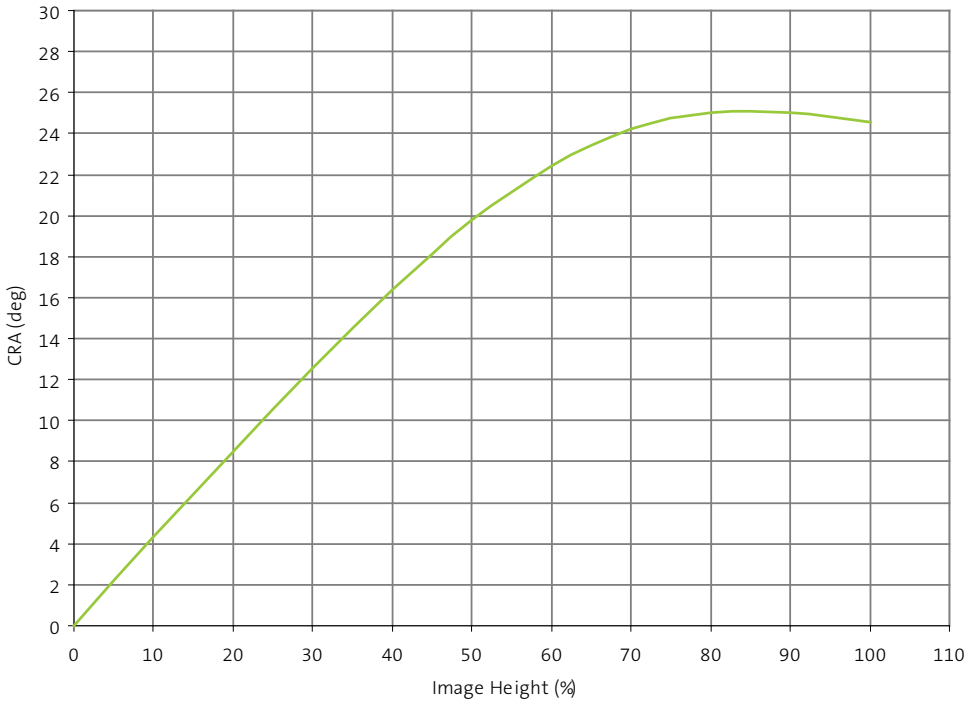
| Signal | Reset | Post-Reset | Standby | Standby with SHUTDOWN | Power Down |
|---|---|---|---|---|---|
| DOUT[7:0] | High-Z | High-Z | High-Z by default (configurable via OE_BAR or two-wire serial interface reg) | High-Z by default | X |
| PIXCLK | High-Z | High-Z | High-Z by default (configurable) | High-Z by default | X |
| LV | High-Z | High-Z | High-Z by default (configurable) | High-Z by default | X |
| FV | High-Z | High-Z | High-Z by default (configurable) | High-Z by default | X |
| DOUT_N | 0 | 0 | 0 | 0 | X |
| DOUT_P | 0 | 0 | 0 | 0 | X |
| CLK_N | 0 | 0 | 0 | 0 | X |
| CLK_P | 0 | 0 | 0 | 0 | X |
| VGPIO[7:0] | High-Z | High-Z | Depending on how system uses them | Depending on how system uses them | X |
| SADDR | Input | Input | Input | Input | |
| SDATA | Input | I/O | Input | Input | |
| SCLK | Input | Input | Input | Input | |
| S_SCL | High-Z | High-Z | High-Z by default (configurable) | High-Z by default (configurable) | X |
| S_SDA | Input | I/O | Input | Input | |

Note: X = “Don’t Care.”

# Spectral Characteristics

**Figure 44: Chief Ray Angle (CRA) vs. Image Height**

CRA vs. Image Height Plot




| Image Height | | CRA |
|---|---|---|
| (%) | (mm) | (deg) |
| 0 | 0 | 0 |
| 5 | 0.113 | 2.19 |
| 10 | 0.227 | 4.33 |
| 15 | 0.340 | 6.43 |
| 20 | 0.454 | 8.50 |
| 25 | 0.567 | 10.55 |
| 30 | 0.680 | 12.57 |
| 35 | 0.794 | 14.52 |
| 40 | 0.907 | 16.39 |
| 45 | 1.021 | 18.15 |
| 50 | 1.134 | 19.76 |
| 55 | 1.247 | 21.20 |
| 60 | 1.361 | 22.43 |
| 65 | 1.474 | 23.44 |
| 70 | 1.588 | 24.21 |
| 75 | 1.701 | 24.74 |
| 80 | 1.814 | 25.03 |
| 85 | 1.928 | 25.11 |
| 90 | 2.041 | 25.01 |
| 95 | 2.155 | 24.80 |
| 100 | 2.268 | 24.55 |

**Note:** ON Semiconductor recommends the use of a 670 nm IR cut filter for the MT9P111 for improved performance.

## Revision History

**Rev. G . . . . . 5/5/15**

- Updated "Ordering Information" on page 2

**Rev. F . . . . . 3/27/15**

- Converted to ON Semiconductor template
- Removed Confidential marking

**Rev E. . . . . 8/11/10**

- Updated Figure 31: "Sequential Read, Start from Random Location," on page 47
- Updated Figure 32: "Sequential Read, Start from Current Location," on page 47
- Updated Figure 34: "Sequential Write, Start at Random Location," on page 48
- Updated "Standby Modes" on page 53
- Updated Table 19, "DC Electrical Definitions and Characteristics—Parallel Mode," on page 55
- Updated Table 21, "I/O Timing Specifications," on page 58
- Updated Figure 42: "Two-Wire Serial Bus Timing Parameters," on page 59

**Rev. D . . . . . 2/11/10**

- Updated to Production
- Updated the following parameters in Table 1, "Key Performance Parameters," on
- page 1:
  - $SNR_{MAX}$
  - Responsivity
  - Power consumption
- Updated descriptions for STANDBY, SHUTDOWN, VGPIO[7:0], $V_{DD}$_VGPIO and GND_VGPIO in Table 3, "Signal Descriptions," on page 8
- Updated Note 8 for Figure 1: "Typical Configuration (connection)," on page 9
- Added first paragraph and updated Note 7 in "Decoupling Capacitor Recommendations" on page 10
- Updated recommended maximum analog value in "Auto White Balance" on page 40
- Added Caution to "Power-Up Sequence" on page 49
- Added last paragraph to "Shutdown Mode" on page 55
- Updated Parameter column for $t_R$ and $t_F$ in Table 21, "I/O Timing Specifications," on page 59

**Rev. C . . . . . 5/11/09**

- Updated to Aptina Imaging Corporation template
- Deleted "with dithering" from "Features" on page 1
- Changed RAJPEG to SpeedTags™ throughout document
- Updated Table 1, "Key Performance Parameters," on page 1
  - Changed dynamic range to 62dB
  - Changed responsivity to 0.64 V/lux-sec
  - Changed I/O supply voltage to "1.7-1.9V or 2.5–3.1V"
- Deleted last sentence of 2nd paragraph in "MT9P111 Overview" on page 6
- Updated Table 3, "Signal Descriptions," on page 7
  - Updated descriptions of EXTCLK and $D_{OUT}$[7:0]
- Updated item 2 in "Decoupling Capacitor Recommendations" on page 10
- Deleted 2nd sentence in "PLL-Generated Clocks" on page 14

- Added 3rd sentence in “PLL Setup” on page 14
- Deleted PMB3 in “Binning” on page 19
- Added “Power Reduction Modes” on page 21
- Added last two sentences in“One-Time Programmable Memory” on page 25
- Updated “JPEG Encoding Highlights” on page 30
- Added note to “RGB or YCbCr Thumbnail” on page 31
- Added “Scalado SpeedTags™ Support” on page 34
- Deleted bullets in “Multi-Shot Image Capture Mode” on page 38
- Updated “Auto Exposure” on page 39
- Added last sentence in “Auto White Balance” on page 40
- Updated “User -Accessible Internal ADC” on page 43
- Updated “Multimaster Serial Interface” on page 43
- Updated heading of “Two-Wire Serial Interface” on page 44
- Added note to “Typical Serial Transfer” on page 45
- Removed register tables (moved to separate document)
- Updated Table 13, “Power-Up Signal Timing,” on page 49
- Added “Power-Down Sequence” on page 50
- Updated “Soft Reset” on page 51
- Added note to Table 15, “Hard Reset Signal Timing,” on page 51
- Updated Table 16, “Soft Reset Signal Timing,” on page 52
- Added 2nd sentence in 2nd paragraph of “Standby Modes” on page 53
- Added note to “Shutdown Mode” on page 54
- Updated Table 19, “DC Electrical Definitions and Characteristics—Parallel Mode,” on page 55
- Updated Table 21, “I/O Timing Specifications,” on page 58
- Updated Table 22, “Two-Wire Serial Bus Characteristics,” on page 60
- Added note to Figure 43: “Sequence of Signals for OTP Memory Operation,” on page 62
- Added note to Figure 44: “Chief Ray Angle (CRA) vs. Image Height,” on page 64

**Rev. B . . . . . 10/1/08**

- In Table 1, “Key Performance Parameters,” on page 1:
  - Updated CRA from 25.03° to 25.11° MAX at 80% image height
  - Updated Input clock frequency from 8-54 MHz to 10-48 MHz
- Updated Figure 1: “Typical Configuration (connection),” on page 9, including notes.
- In Table 3, “Signal Descriptions,” on page 7:
  - Updated SHUTDOWN description
  - Updated EXTCLK description
  - Added TEST_EN
- In first paragraph of “PLL-Generated Clocks” on page 14, updated EXTCLK input from “8 through 54” to “10 through 48” MHz.
- Updated description in Name column of R0x00003330, bit 8 in Table 44, “1: SOC1 Registers,” on page 174
- In Table 19, “DC Electrical Definitions and Characteristics—Parallel Mode,” on page 55:
  - Updated $^{f}$EXTCLK from 54 to 48 MHz
  - Changed VDD_MIPI to VDDIO_TX

– Fixed wrong symbol font in Table 22, “Two-Wire Serial Bus Characteristics,” on page 60
- Fixed numbering sequence in various notes

**Rev. A, Advance ..... 7/11/2008**
- Initial release