



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

## MT9D115 Data Sheet

For the latest data sheet, refer to Aptina's Web site: [www.aplina.com](http://www.aplina.com)

### Features

- 2Mp resolution (1600H x 1200V)
- 1/5-inch optical format
- Same or better image quality compared to MT9D112
- Individual module ID support through one-time programmable (OTP) memory
- Surface fit lens correction (LC) to compensate for lens/small pixel vignetting and corner color variations
- Automatic functions: Exposure, white balance, black level offset correction, flicker detection and avoidance, color saturation control, defect identification and correction, aperture correction, and GPIO
- Programmable controls: Exposure, white balance, horizontal and vertical blanking, color, sharpness, gamma, lens shading correction, horizontal and vertical image flip, zoom, windowing, sampling rates, and GPIO
- 15 frames per second (fps) at 1600H x 1200V with moderate pixel clock frequency ( $\leq 64$  MHz) to minimize baseband reception interference and 30 fps at 800H x 600V
- 2 x 2 pixel binning to improve low-light image quality
- Support for external LED or xenon flash
- On-chip phase-locked loop (PLL) to minimize the number of system clocks
- Low power modes to prolong battery life of portable devices
- Fail-safe I/Os with programmable output slew rate
- Industry standard two-wire serial interface for controls
- 10-bit parallel or MIPI serial interfaces for image data

### Applications

- Cellular phones
- PC cameras
- PDAs

**Table 1: Key Performance Parameters**

Parameter	Value
Pixel size	1.75 $\mu$ m x 1.75 $\mu$ m
Optical format	1/5-inch
Array format (active)	1600H x 1200V = 1.92 Mp
Imaging area	2.8mm x 2.10mm: 3.50mm diagonal (4:3 aspect ratio)
CRA	25°
Color filter array	RGB Bayer
Scan mode	Progressive
Shutter	Electronic rolling shutter (ERS)
Input clock range	6 – 54 MHz
Output pixel clock maximum	85 MHz
Output MIPI data rate maximum	512 Mb/s
Max. Frame Rate	15 fps full res 30 fps 800x600
Responsivity	0.65 V/Lux-sec (550nm)
Signal-to-noise ratio	39 dB (MAX)
Dynamic range	63.9 dB (pixel)
Supply voltage	Digital 1.8V (nominal)
	Analog 2.8V (nominal)
	I/O 1.8V or 2.8V (nominal)
	MIPI 1.7-1.95
Power consumption	196mW <sup>1</sup>
Operating temperature range	–30°C to 70°C (at junction)
Package	Bare die, CSP

Note: 1. Power consumption for typical voltages at 800 x 600 video mode

**Table 2: Available Part Numbers**

Part Number	Description
MT9D115D00STC K25AC1	RGB color die
MT9D115EB3STC	CSP
MT9D115PACSTCH	RGB color demo headboard
MT9D115PACSTCD	RGB color demo kit



## Table of Contents

Features .....	1
Applications .....	1
Functional Description .....	6
Architecture Overview .....	6
Typical Connections .....	7
Decoupling Capacitor Recommendations .....	8
Signal Descriptions .....	9
Architecture .....	10
Firmware .....	11
External Host Interface (Two-Wire Slave-Only Interface) .....	12
Always-On Power Domain .....	14
Sensor Core .....	14
Pixel Array .....	15
Analog Processing .....	16
External Generated Master Clock .....	17
PLL-Generated Master Clock .....	18
PLL Setup .....	18
Digital Processing .....	18
Readout Options .....	18
Readout Modes .....	18
Input Interface to Image Flow Processor .....	26
Image Flow Processor .....	27
Pixel Reconstruction (Lens Shading Correction) .....	27
Color Rendering/Statistics Collection (Color Correction) .....	29
Digital Scaling/Output Format .....	30
YUV-to-RGB/YUV Conversion and Output Formatting .....	31
Output Interface from IFP .....	32
Parallel and MIPI Output .....	32
Output Format and Timing .....	33
Control Functions .....	35
Sequencer: Camera Operating System .....	35
Mode: Context Information .....	36
Camera Functions .....	37
Simple Rule-based Auto Exposure .....	37
AE Driver .....	37
Evaluative Algorithm .....	37
Accelerated Settling During Overexposure .....	38
Exposure Control .....	38
Flicker Detection and Avoidance .....	38
Auto White Balance .....	39
Flash .....	39
Histogram: Dark Level Adjustments, Low Light and Tonal Controls .....	39
Optics .....	40
Power Modes .....	41
Power Application Sequence .....	41
Power-On Reset .....	42
Hard Reset .....	44
Soft Reset .....	45
Power Removal Sequence .....	46
Standby Modes .....	47
Soft Standby .....	47



Low Leakage Hard Standby. . . . .	47
Hard Standby Mode. . . . .	48
Entering Standby Mode . . . . .	48
Exiting Standby Mode . . . . .	48
Soft Standby Mode. . . . .	49
Entering Standby Mode . . . . .	49
Exiting Standby Mode . . . . .	49
Other Features . . . . .	51
One-time Programmable Memory (OTPM) . . . . .	51
Programming the OTPM . . . . .	51
Sequence of Signals for OTPM Operation . . . . .	51
Reading the OTPM . . . . .	51
GPIO and Output Enable Controls . . . . .	52
General Purpose I/Os . . . . .	52
Output Enable Control. . . . .	52
GPIO Control . . . . .	53
Overview of GPIO Signals . . . . .	53
Electrical Specifications. . . . .	54
Absolute Maximum Rating. . . . .	54
I/O Parameters . . . . .	54
MIPI Interface AC and DC Electrical Specifications . . . . .	55
Introduction . . . . .	55
Electrical Specifications . . . . .	55
AC Electricals. . . . .	56
Appendix A- VDD_IO Current Addition . . . . .	59
Introduction . . . . .	59
VDD_IO Current . . . . .	59
IO Pin States . . . . .	60
RESET_BAR with Internal Pull-UP . . . . .	61
Recommended System and Test Setup with Multiple Serial Interface Slave Devices. . . . .	62
Test Sequence for Measuring VDD_IO Current at Hard Standby Mode . . . . .	65
Conclusion. . . . .	65
Revision History. . . . .	66



## List of Figures

Figure 1:	MT9D115 Block Diagram .....	6
Figure 2:	Typical Configuration (connection) .....	7
Figure 3:	SOC Block Diagram .....	10
Figure 4:	Firmware Architecture .....	11
Figure 5:	External Host Register Map .....	12
Figure 6:	Two-Wire Serial Control Bus Timing .....	13
Figure 7:	Sensor Core Block Diagram .....	15
Figure 8:	Pixel Color Pattern Detail (Top Right Corner) .....	16
Figure 9:	Imaging a Scene .....	16
Figure 10:	Six Pixels in Normal and Column Mirror Readout Modes .....	19
Figure 11:	Six Rows in Normal and Row Mirror Readout Modes .....	19
Figure 12:	Eight Pixels in Normal and Column Skip 2X Readout Modes .....	19
Figure 13:	Pixel Readout (no skipping) .....	20
Figure 14:	Pixel Readout (x_odd_inc = 3, y_odd_inc = 1) .....	21
Figure 15:	Pixel Readout (x_odd_inc = 1, y_odd_inc = 3) .....	21
Figure 16:	Pixel Readout (x_odd_inc = 3, y_odd_inc = 3) .....	22
Figure 17:	Pixel Readout (x_odd_inc = 3, y_odd_inc = 1, x_bin = 1) .....	23
Figure 18:	Pixel Readout (x_odd_inc = 3, y_odd_inc = 3, x_ybin = 1) .....	24
Figure 19:	Valid Image Data .....	25
Figure 20:	Pixel Data Timing Example .....	25
Figure 21:	Available Test Patterns .....	26
Figure 22:	IFP Block Diagram .....	27
Figure 23:	Gamma Correction Curve .....	30
Figure 24:	Timing of Full Frame Data or Scaled Data Passing Through the FIFO .....	33
Figure 25:	Sequencer Finite State Machine .....	36
Figure 26:	CRA vs. Image Height .....	40
Figure 27:	Power Application Sequence Timing .....	42
Figure 28:	Internal Power-On Reset .....	43
Figure 29:	Hard Reset Operation .....	44
Figure 30:	Soft Reset Operation .....	45
Figure 31:	Recommended Power Down Sequence .....	46
Figure 32:	Hard Standby Mode Operation .....	48
Figure 33:	Soft Standby Mode Operation .....	49
Figure 34:	Output Interface Timing Waveforms .....	56
Figure 35:	Recommended System and Test Setup for Minimum VDD_IO Power Consumption .....	59
Figure 36:	RESET_BAR Pad Architecture .....	61
Figure 37:	Recommended System and Test Setup for Minimum VDD_IO Power Consumption in the MT9D115 Sharing with Multiple Two-wire Serial Interface Devices .....	62
Figure 38:	Recommended System and Test Setup for Minimum VDD_IO Power Consumption in the MT9D115 Sharing with Multiple Two-wire Serial Interface Devices at Different IO Levels .....	63
Figure 39:	System Setup with Independent Voltage Sources for MT9D115 and Controller Chip .....	64
Figure 40:	Modifications to the Demo2 Sensor Head Board for VDD_IO Current Measurement .....	65



## List of Tables

Table 1:	Key Performance Parameters . . . . .	1
Table 2:	Available Part Numbers . . . . .	1
Table 3:	Signal Description and Direction . . . . .	9
Table 4:	List of Drivers . . . . .	11
Table 5:	Two-Wire Serial Interface Timing Data . . . . .	14
Table 6:	Row Address Sequencing (Sampling) . . . . .	23
Table 7:	Row Address Sequencing (Binning) . . . . .	24
Table 8:	Data Formats Supported by MIPI Interface . . . . .	32
Table 9:	YCbCr Output Data Ordering . . . . .	33
Table 10:	RGB Ordering in Default Mode . . . . .	33
Table 11:	2-Byte RGB Format . . . . .	34
Table 12:	Power Application Sequence Timing . . . . .	42
Table 13:	POR Parameters . . . . .	43
Table 14:	Hard Reset . . . . .	44
Table 15:	Soft Reset Signal Timing . . . . .	45
Table 16:	Power Down Signal Timing . . . . .	46
Table 17:	Hard Standby Signal Timing . . . . .	48
Table 18:	Soft Standby Signal Timing . . . . .	49
Table 19:	Status of Signals During Different States . . . . .	50
Table 20:	GPIO Related Registers and Variables . . . . .	53
Table 21:	Maximum Rating . . . . .	54
Table 22:	I/O Parameters . . . . .	54
Table 23:	MIPI High-Speed Transmitter DC Characteristics . . . . .	55
Table 24:	MIPI High-Speed Transmitter AC Characteristics . . . . .	55
Table 25:	MIPI Low-Power Transmitter DC Characteristics . . . . .	55
Table 26:	MIPI Low-Power Transmitter AC Characteristics . . . . .	55
Table 27:	AC Electricals . . . . .	56
Table 28:	DC Electricals . . . . .	57
Table 29:	Status of Signals During Standby State . . . . .	60
Table 30:	Typical Mismatch Current in VDD_IO Due to Mismatch in RESET_BAR level and VDD_IO Level . .	61

## Functional Description

Aptina's MT9D115 is a 1/5-inch 2 Mp CMOS digital image sensor with an integrated advanced camera system. This camera system features a microcontroller (MCU), a sophisticated image flow processor (IFP), MIPI and parallel output ports (only one output port can be used at a time). The microcontroller manages all functions of the camera system and sets key operation parameters for the sensor core to optimize the quality of raw image data entering the IFP. The IFP will be responsible for processing and enhancing the image.

The entire system-on-a-chip (SOC) has superior low-light performance that is particularly suitable for PC camera applications. The MT9D115 features Aptina's breakthrough low-noise CMOS imaging technology that achieves near-CCD image quality (based on signal-to-noise ratio and low-light sensitivity) while maintaining the inherent size, cost, and integration advantages of CMOS.

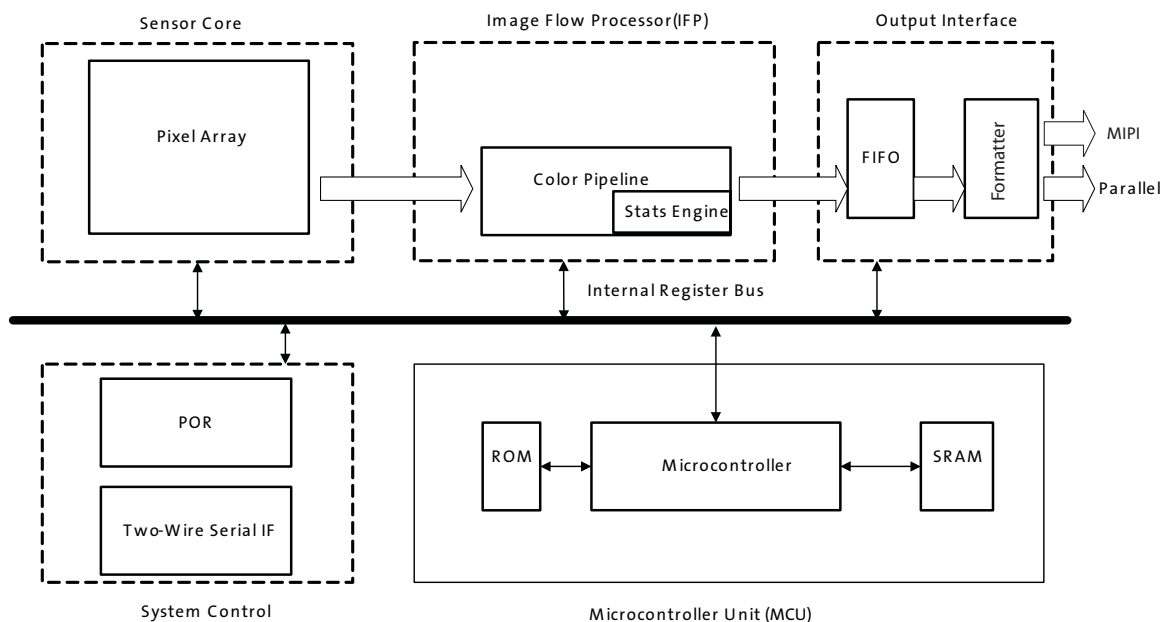
The Aptina MT9D115 can be operated in its default mode or programmed for frame size, exposure, gain, and other parameters. The default mode output is a 800x600 image size at 30 frames per second (fps), assuming a 24 MHz input clock. It outputs 8-bit data, using the parallel output port.

## Architecture Overview

The MT9D115 combines a 2 Mp sensor core with an IFP to form a stand-alone solution for 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, an integrated microcontroller autonomously controls most aspects of operation. The processed image data is transmitted to the host system either through the parallel or MIPI interface.

Figure 1 shows the major functional blocks of the MT9D115.

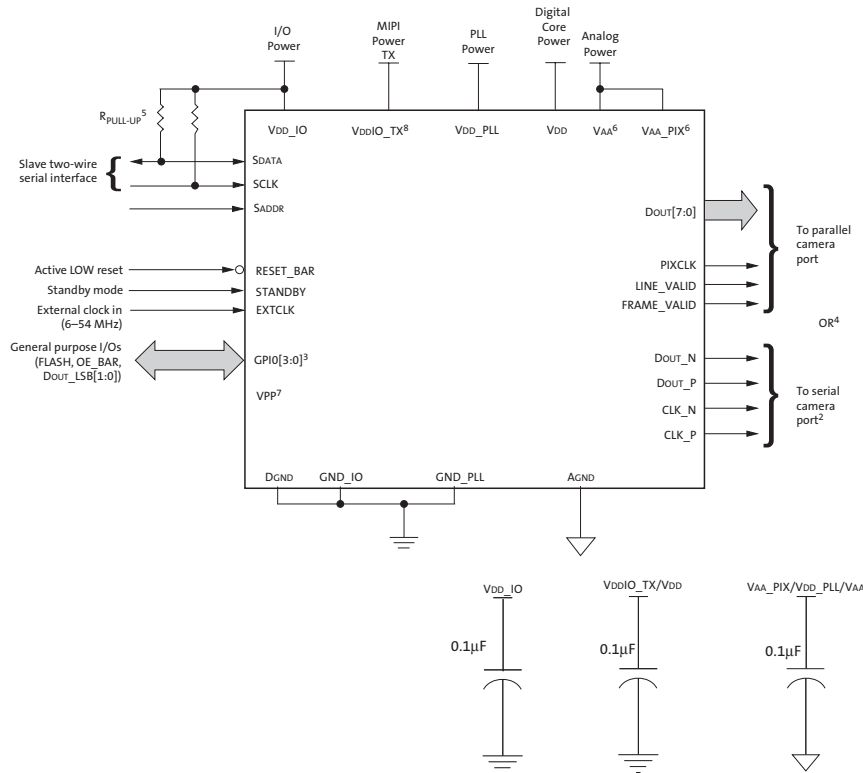
**Figure 1: MT9D115 Block Diagram**





## Typical Connections

Figure 2: Typical Configuration (connection)



- Notes:
1. This typical configuration shows only one scenario out of multiple possible variations for this sensor.
  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 GPIO pads can serve multiple features that can be reconfigured. The function and direction will vary by applications.
  4. Only one output mode (serial or parallel) can be used at any time.
  5. Aptina recommends a resistor value of 1.5KΩ to VDD\_IO for the two-wire serial interface R<sub>PULL-UP</sub>; Higher values can be used for slower transmission speed.
  6. VAA and VAA\_PIX can be tied together. Although separate decoupling capacitors are recommended for VAA and VAA\_PIX, decoupling capacitors can be shared if one would like to reduce module size.
  7. VPP is the OTP memory programming voltage and should be left floating during normal operation.
  8. 1.8V supply is shared by MIPI interface and VDD to reduce the number of decoupling caps, and, subsequently, the module size. VDDIO\_TX must be connected to a 1.8V power supply source, even though MIPI interface is not used.
  9. Aptina recommends that 0.1µF and 1µF decoupling capacitors for each power supply are mounted as close as possible to the pad and that a 10µF capacitor be placed nearby off-module. Actual values and results can vary depending on layout and design considerations. Please follow Aptina's recommended capacitor Recommendations.
  10. VDD\_PLL and VAA can share the same power source, in which case GND\_PLL must be connected to GND.
  11. Internal pull-up in RESET\_BAR pin and can be left floating when not connected.



## Decoupling Capacitor Recommendations

It is important to provide clean, well-regulated power to each power supply. The customer is ultimately responsible for ensuring that clean power is provided for their own designs because hardware design is influenced by many factors, including layout, operating conditions, and component selection.

The recommendations for capacitor placement and values listed below are based on the Aptina internal demo camera design and verified in hardware.

Aptina recommends the following, in order of preference:

1. Mount 0.1 $\mu$ F and 1 $\mu$ F decoupling capacitors for each power supply as close as possible to the pad and place a 10 $\mu$ F capacitor nearby off-module.
2. If module limitations allow for only six decoupling capacitors for a three-regulator design (VDD1V2 tied to external regulator), use a 0.1 $\mu$ F and 1 $\mu$ F capacitor for each of the three regulated supplies. Aptina also recommends placing a 10 $\mu$ F capacitor for each supply off-module, but close to each supply.
3. If module limitations allow for only three decoupling capacitors, use a 1 $\mu$ F capacitor (preferred) or a 0.1 $\mu$ F capacitor for each of the three regulated supplies. Aptina also recommends placing a 10 $\mu$ F capacitor for each supply off-module but close to each supply.
4. Give priority to the VAA supply for additional decoupling capacitors.

Aptina does not recommend inductive filtering components.

Follow best practices when performing physical layout. Refer to technical note TN-09-131.





## Signal Descriptions

**Table 3: Signal Description and Direction**

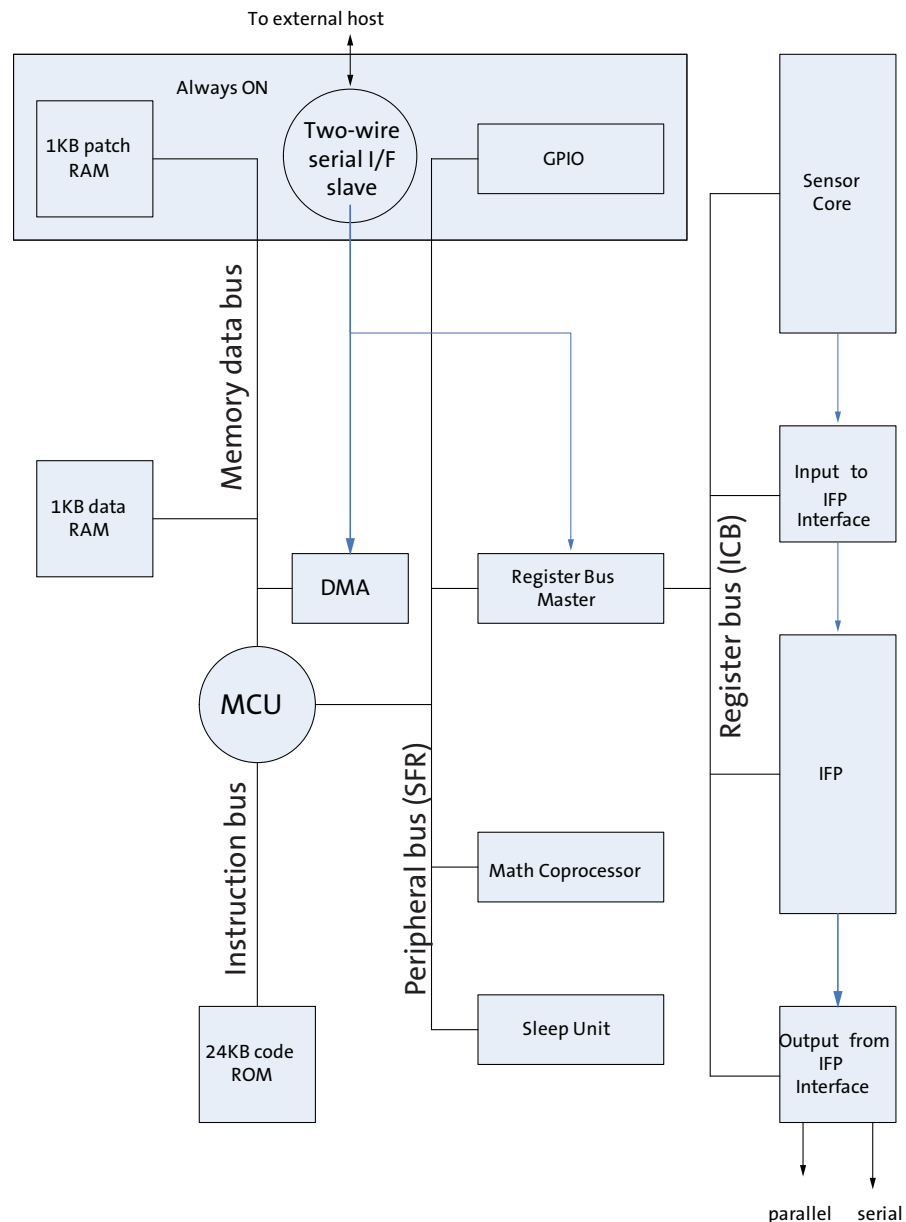
Name	Type	Description	Note
STANDBY	Input	Hardware standby	
EXTCLK	Input	External clock input	
SADDR	Input	Two-wire interface device select address	
SCLK	Input	Two-wire interface serial clock	
RESET_BAR	Input	Hardware reset	4
CLK_N	Output	MIPI differential clock N	5
CLK_P	Output	MIPI differential clock P	5
DOUT_N	Output	MIPI differential data N	5
DOUT_P	Output	MIPI differential data P	5
DOUT[7:0]	Output	Parallel image data	1
FRAME_VALID	Output	Parallel pixel bus frame valid	1
LINE_VALID	Output	Parallel pixel bus line valid	1
PIXCLK	Output	Parallel pixel bus pixel clock	1
SDATA	Bidirectional	Two-wire interface serial data	
GPIO_0/DOUT_LSB[0]	Bidirectional/Output	General-purpose I/O or LSB for Raw 10 data output during SOC Bypass	2
GPIO_1/DOUT_LSB[1]	Bidirectional/Output	General-purpose I/O or LSB for Raw 10 data output during SOC Bypass	2
GPIO_3/OE_BAR	Bidirectional/Output	General-purpose I/O or output enable	2
GPIO_2/FLASH	Bidirectional/Output	General-purpose I/O or flash control	2
VAA	Supply	Analog core power source 2.8V nominal	
VAA_PIX	Supply	Analog core power source 2.8V nominal	
VDD	Supply	Digital core power source 1.8V nominal	
VDD_IO	Supply	Digital IO power source 1.8V or 2.8V nominal	
VDD_PLL	Supply	Digital PLL power source 2.8V nominal	
VDDIO_TX	Supply	Digital MIPI IO power source 1.8V nominal.	6

- Notes:
1. In serial only mode, DOUT[7:0], PIXCLK, and GPIO[3:0] can be left floating by setting R0x0026[1] = 1. If GPIO signals are required, DOUT[7:0] and PIXCLK must be tied to DGND and OE\_BAR must be tied to VDD\_IO. GPIO\_3 should be configured as an input for OE\_BAR function and set R0x001A[8] = 1.
  2. GPIO can be left floating if not used and must be programmed as outputs.
  3. Must be connected to VDD\_IO, internal 100k ohms typical at 2.8V VDDIO used.
  4. Can be left floating if not used.
  5. Must be connected to VDD, even in designs where the MIPI interface is not used.

## Architecture

The MT9D115 from Aptina is the third-generation, two-megapixel camera SOC. It is a microprocessor-based camera system that combines a sensor core with an image flow processor (IFP) to form a standalone solution that includes image acquisition and processing. Both the sensor core and IFP have internal registers that can be accessed by an external host. In normal operation, the integrated system microprocessor autonomously controls most aspects of operation. The image data is transmitted to the external host system either through a parallel bus or a serial MIPI interface (see Figure 3).

**Figure 3: SOC Block Diagram**



The external host and the integrated microprocessor (MCU) can both access all the internal resources (RAM and registers). The external host always has higher priority. The following sections briefly describe the functionality of each key component of the system.

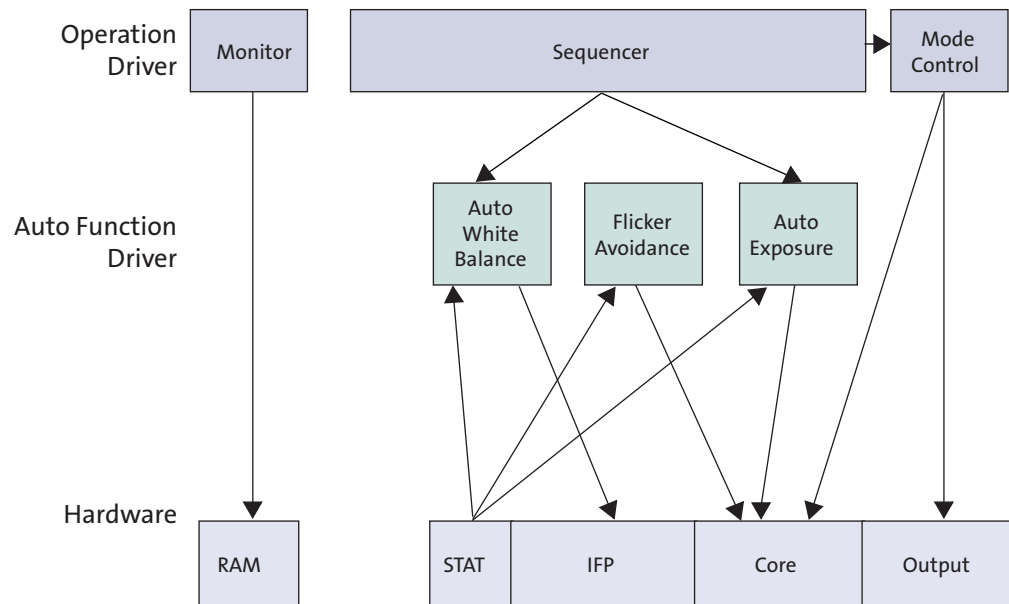
## Firmware

The firmware implements all automatic camera functions, such as auto exposure (AE), auto white balance (AWB), and flicker detection/avoidance (FD), as well as control functions such as sequencer, mode/context, and histogram (see Table 4). The firmware consists of drivers, generally one driver for each major automatic or control function (see Figure 4).

**Table 4: List of Drivers**

ID	Driver Name	Description
ID = 1	Sequencer	Controls of camera main function
ID = 2	AE	Auto exposure
ID = 3	AWB	Auto white balance
ID = 4	Flicker Detection	Flicker detection and avoidance
ID = 7	Mode/Context	Context variables
ID = 11	Histogram	Reduce image flare and analyze image histogram

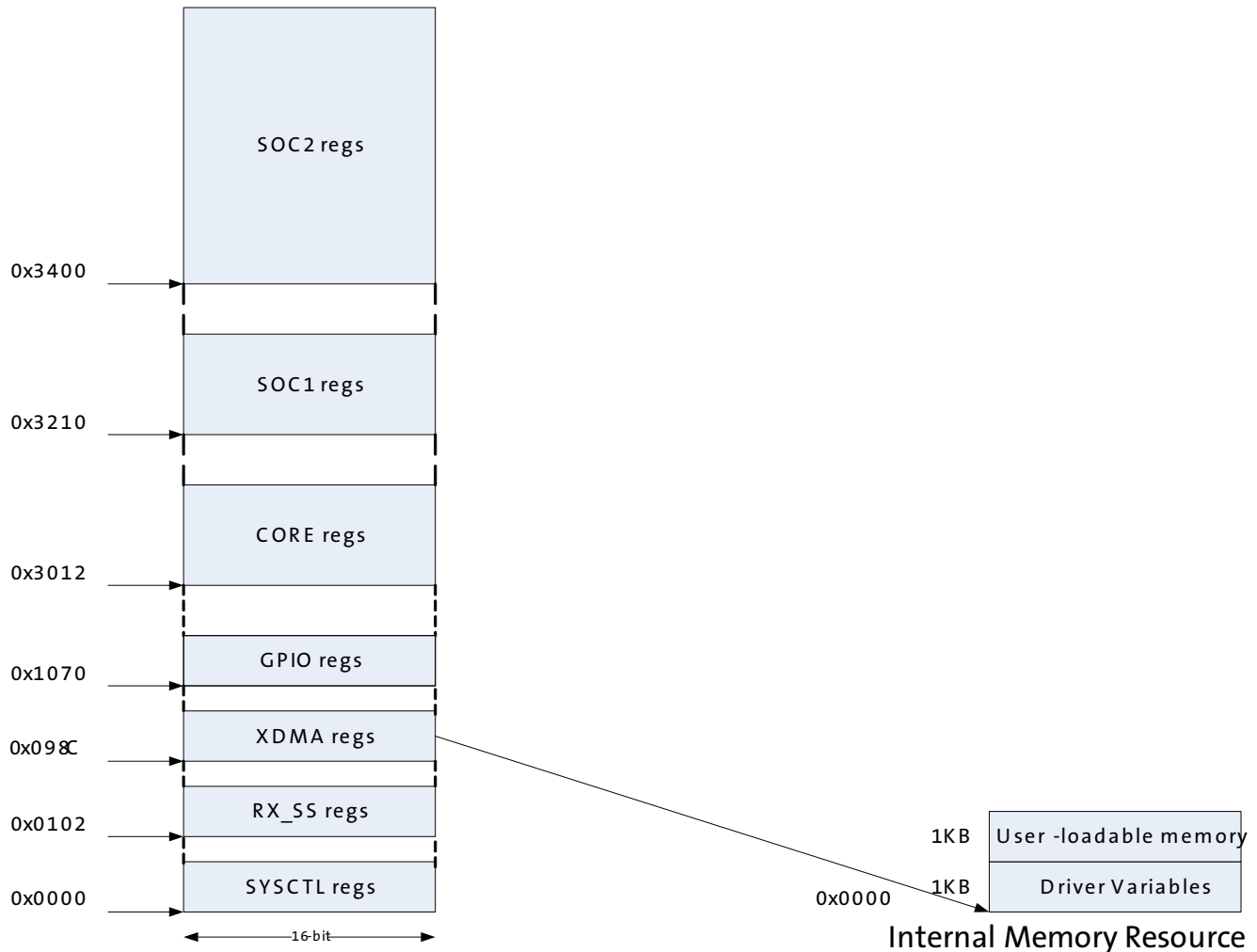
**Figure 4: Firmware Architecture**



## External Host Interface (Two-Wire Slave-Only Interface)

The MT9D115 will appear as a two-wire serial interface slave to the external host. Its base address is selectable by the external SADDR pin input (when SADDR = 0 then base address = 0x78; when SADDR = 1 then base address = 0x7A). There are 32K addressable 16-bit registers (that is, the starting address of each register always falls into even addresses) within the MT9D115 but not all of them are being used (see Figure 5).

**Figure 5: External Host Register Map**



The MT9D115 Register Reference provides detailed register explanations. Although most registers are self-explanatory, the next paragraphs contain enhanced information about XDMA registers are worth explaining here.

The XDMA registers allow the external host to indirectly access the internal memory resources of the MT9D115, which include the firmware driver variables. To access the variables, use logical accesses provided by the XDMA registers.

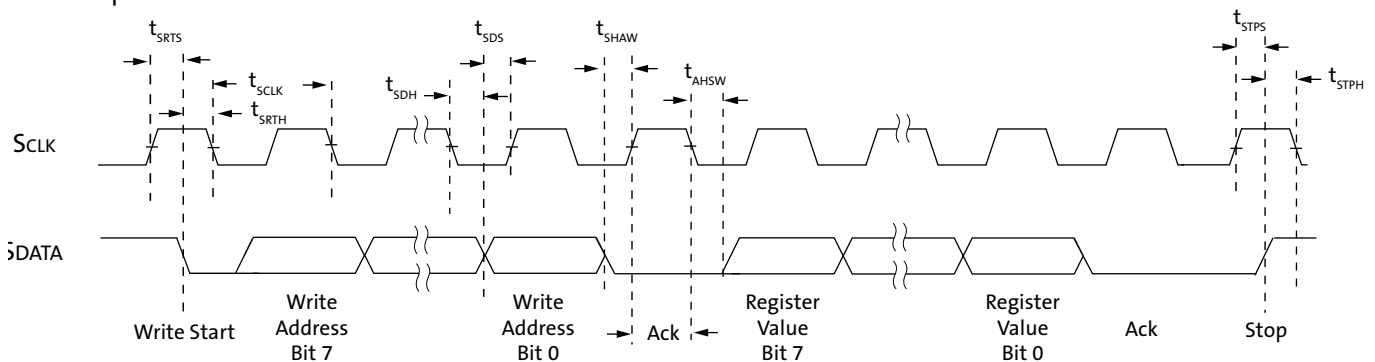
The external host interface is implemented through a two-wire interface that enables direct read/write access to hardware registers and indirect access to firmware variables within the MT9D115. The interface is designed to be compatible with the MIPI alliance standard for Camera Serial Interface 2 (CSI-2) 1.0, which uses the electrical characteristics and transfer protocols of the two-wire serial interface specifications.

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 to the external host which acts as a master device. The master generates a clock (SCLK) that is an input to the sensor and used to synchronize the transactions at the interface.

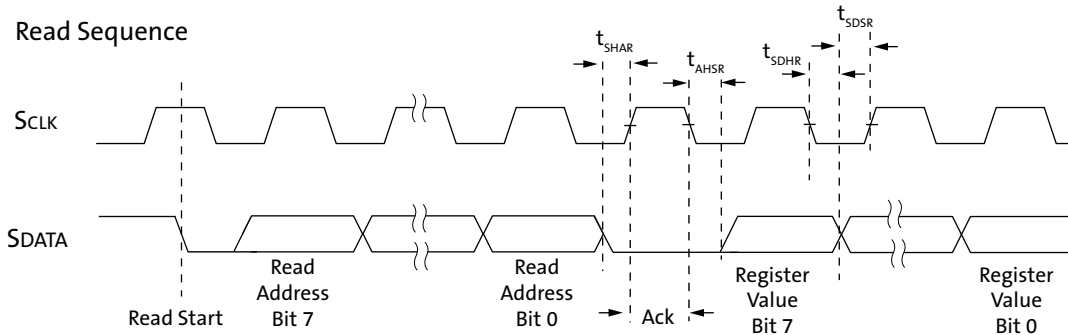
Data is transferred between the master and the slave on a bidirectional serial data bus (SDATA). Both SCLK and SDATA are pulled up to VDD\_IO off-chip by a 1.5K $\Omega$  resistor. Either the slave or master device can drive SDATA to LOW—the interface determines which device is allowed to drive SDATA at any given time.

**Figure 6: Two-Wire Serial Control Bus Timing**

#### Write Sequence



#### Read Sequence



**Table 5: Two-Wire Serial Interface Timing Data**

$f_{EXTCLK} = 14 \text{ MHz}$ ;  $VDD = 1.8\text{V}$ ;  $VDD_{IO} = 1.8\text{V}$ ;  $VAA = 2.8\text{V}$ ;  $VAA_{PIX} = 2.8\text{V}$ ;  
 $VDD_{PLL} = 2.8\text{V}$ ;  $VDD_{PHY} = \text{NA}$ ;  $TJ = 70^\circ\text{C}$ ;  $C_{LOAD} = 68.5\text{pF}$

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$f_{SCLK}$	Serial interface input clock frequency		100	—	400	kHz
$t_{SCLK}$	Serial interface input clock period		2.5	—	10	$\mu\text{s}$
	SCLK duty cycle		33	50	50	%
$t_r$	SCLK/SDATA rise time		—	—	300	ns
$t_{SRTS}$	Start setup time	Master write to slave	600	—	—	ns
$t_{SRTH}$	Start hold time	Master write to slave	300	—	—	ns
$t_{SDH}$	SDATA hold	Master write to slave	5	—	900	ns
$t_{SDS}$	SDATA setup	Master write to slave	100	—	—	ns
$t_{SHAW}$	SDATA hold to ack	Master write to slave	150	—	—	ns
$t_{AHSW}$	Ack hold to SDATA	Master write to slave	150	—	—	ns
$t_{STPS}$	Stop setup time	Master write to slave	300	—	—	ns
$t_{STPH}$	Stop hold time	Master write to slave	600	—	—	ns
$t_{SHAR}$	SDATA hold to ack	Master read from slave	300	—	—	ns
$t_{AHSR}$	Ack hold to SDATA	Master read from slave	300	—	—	ns
$t_{SDHR}$	SDATA hold	Master read from slave	300	—	650	ns
$t_{SDSR}$	SDATA setup	Master read from slave	300	—	—	ns

Note:  $t_R$  and  $t_F$  are dependent on system-level parameters such as the value of pull-up resistor used, how the two-wire serial bus is routed, whether there are other devices on the serial bus, and the strength of the supply used to pull-up the serial bus.

## Always-On Power Domain

The always-on power domain (AOPD) provides an area of functionality that will always be active while power is applied to the MT9D115. The external host interface is located in the AOPD. The domain also includes miscellaneous clock and reset controls as well as configuration registers for the sensor core, processor core, clock configuration, and clock reset control. The user-loadable patch memory is also included in this domain. This memory will remain powered when the main core power is shut down using the standby command.

## Sensor Core

The sensor core of the MT9D115 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 30 Mp/s, corresponding to a pixel clock rate of 63.25 MHz. See Figure 7 on page 15 for a block diagram of the sensor core. It includes a 2.0Mp 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. After a row is read, data from the columns are sequenced through an analog signal chain that provides offset correction and gain, and then through an ADC. The output from the ADC is a 10-bit value for each pixel in the array.

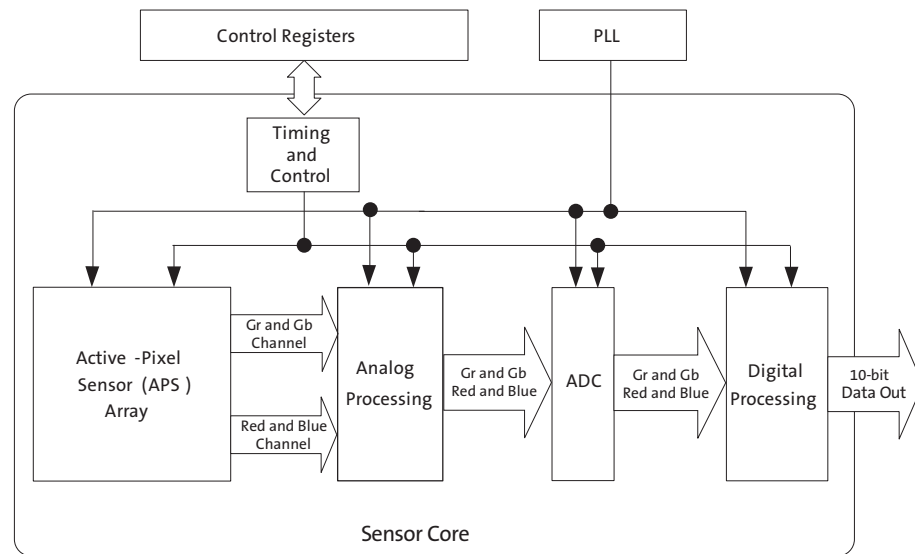
The pixel array contains optically active and light-shielded (dark) pixels. The dark pixels provide data for the offset-correction algorithms (black level control).

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 settings. These registers are controlled by the firmware and can be accessed through a two-wire serial interface. Register values written to the sensor core can be overwritten by firmware.

The output from the core is a Bayer pattern, where 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.

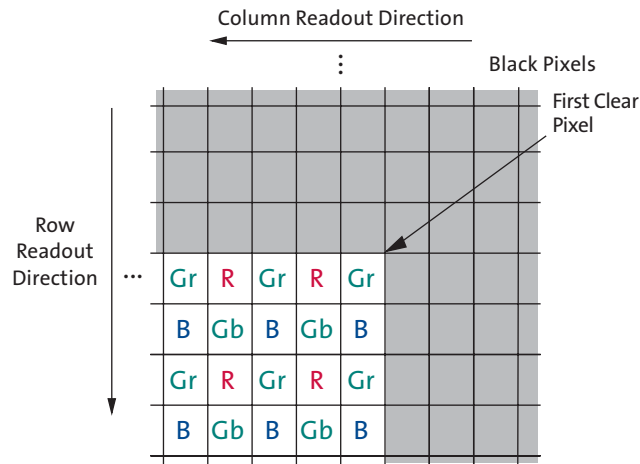
A flash strobe output signal is provided to allow an external xenon or LED light source to synchronize with the sensor exposure time.

**Figure 7: Sensor Core Block Diagram**



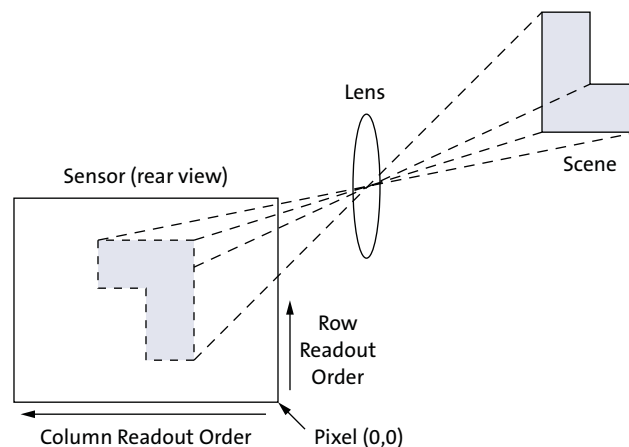
### Pixel Array

The sensor core uses a Bayer color pattern (see Figure 8). The even-numbered rows contain green and red pixels. The odd-numbered rows contain blue and green pixels. Even-numbered columns contain green and blue pixels. Odd-numbered columns contain red and green pixels.

**Figure 8: Pixel Color Pattern Detail (Top Right Corner)****Default Readout Order**

By convention, the sensor core pixel array is shown with pixel (0,0) in the top right corner. This reflects the actual layout of the array on the die. When the sensor is operating in a system, the active surface of the sensor faces the scene (see Figure 9).

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 9: Imaging a Scene****Analog Processing****Analog Readout Channel**

The sensor core features an analog readout channel, (see Figure 7 on page 15). The readout channel consists of a gain stage, a sample-and-hold stage with black level calibration capability, and a 10-bit ADC.





## Gain Options

The MT9D115 provides per-color gain control as well as the option of global gain control. The per-color and global gain control can be used interchangeably. A WRITE to a global gain register is aliased as a WRITE of the same data to the four associated color-dependent gain registers.

Integer digital gains in the range 0–7 can be programmed. A digital gain of 0 sets all pixel values to 0 (the pixel data will simply represent the value applied by the pedestal block). Gain settings are updated in every frame by the MCU auto functions such as AWB, AE, and FD. To make manual adjustments to gain settings, the MCU automatic exposure and automatic white balance adjustment features must be disabled.

## Integration Time

The integration time (exposure) of the MT9D115 is controlled by variables. While coarse integration time controls the integration duration in terms of row times, fine integration time allows for sub-row times accuracy in terms of pixel clocks. Integration time is updated in every frame by the MCU auto feature. Disable the MCU auto features to make manual adjustments to integration time.

Because of the basic operation of the Electronic Roller Shutter (ERS), it is not advisable to set an integration time that is greater than the frame time.

It is not necessary to reprogram the frame time on the MT9D115 to make longer integration times available because the frame time adjusts automatically. However, long integration times increase the likelihood of image degradation because of increased accumulation of dark current.

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

1. During frame  $n$ , the new integration time is held in the pending register.
2. At the start of frame  $(n + 1)$ , the new integration time is transferred to the live register. Integration for each row of frame  $(n + 1)$  has been completed using the old integration time.
3. The earliest time that a row can start integrating using the new integration time is immediately after that row has been read for frame  $(n + 1)$ .
4. When frame  $(n + 2)$  is read out, it will have been integrated using the new integration time.

If the integration time is changed on successive frames, each value written will be applied for a single frame; the latency between writing a value and it affecting the frame readout remains at two frames.

When the integration time and the gain are changed at the same time, the gain update is held off by one frame so that the first frame output with the new integration time also has the new gain applied.

## External Generated Master Clock

If application does not use PLL, then the clock bypass bit in R0x0014 must be set before exiting soft standby state as follows:

1. Write 0x25F9 to R0x0014 to set clock bypass bit
2. Delay min. of 100 ms
3. Write 0x4028 to R0x0018 to exit from soft standby state
4. After successful exit from soft standby state, disable the clock bypass bit by writing 0x21F9 to R0x0014

## PLL-Generated Master Clock

The PLL can generate a master clock signal whose frequency is up to 85 MHz (input clock from 6 MHz through 54 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. During this time, the behavior of its output clock signal is not guaranteed. The PLL output frequency is determined by two constants, M and N, and the input clock frequency.

$$VCO = \frac{Fin \times 2 \times M}{N + 1}$$

$$PLL \text{ output frequency} = \frac{VCO}{P1 + 1} \quad (EQ 1)$$

## 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 pixel averaging in x and y directions.

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

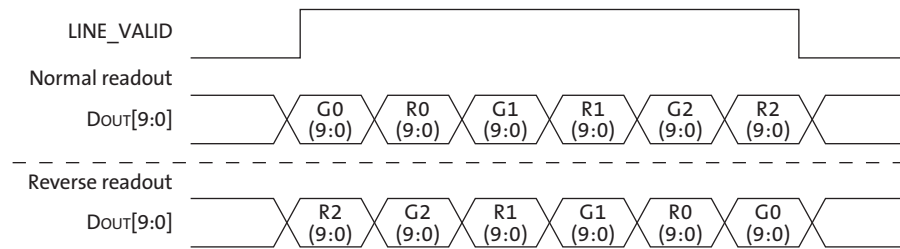
### Window Size

The image output size is set with registers x\_addr\_start, x\_addr\_end, y\_addr\_start, and y\_addr\_end. The edge pixels in the 1600 x 1200 array are present to avoid edge defects and should not be included in the visible window. Binning will change the image output size.

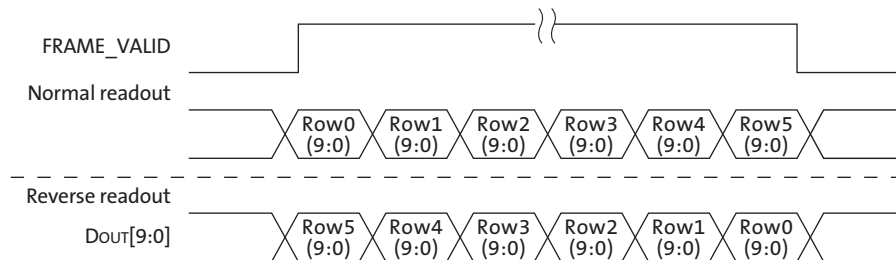
### Readout Modes

#### Horizontal Mirror

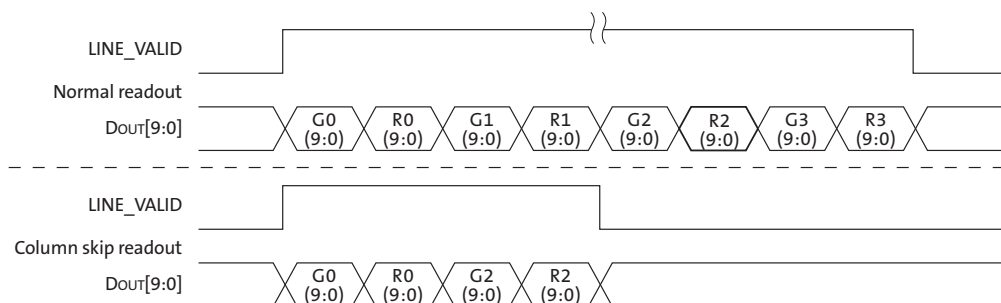
When the sensor is configured to mirror the image horizontally, the order of pixel readout within a row is reversed, so that readout starts from x\_addr\_end and ends at x\_addr\_start. Figure 10 shows a sequence of 6 pixels being read out with normal readout and reverse readout. The SOC corrects for this change in sensor core output.

**Figure 10: Six 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, so that row readout starts from **y\_addr\_end** and ends at **y\_addr\_start**. Figure 11 shows a sequence of six rows being read out with normal readout and reverse readout. The SOC corrects for this change in sensor core output.

**Figure 11: Six Rows in Normal and Row Mirror Readout Modes****Column and Row Skip**

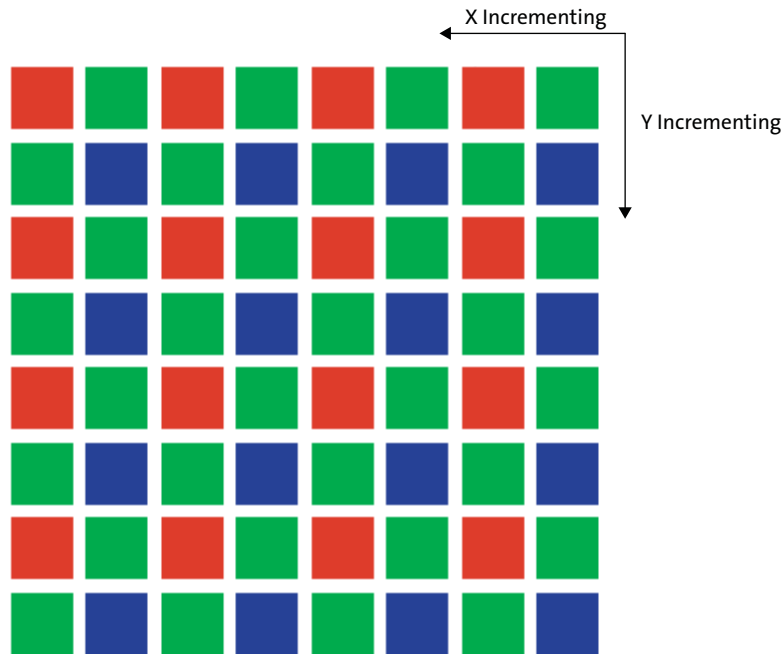
The sensor core supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the sensor and thereby allows the frame rate to be increased. This reduces the amount of row and column data processed and is equivalent to the skip2 readout mode provided by earlier Aptina image sensors. Set the proper image output and crop sizes before enabling subsampling.

**Figure 12: Eight Pixels in Normal and Column Skip 2X Readout Modes**

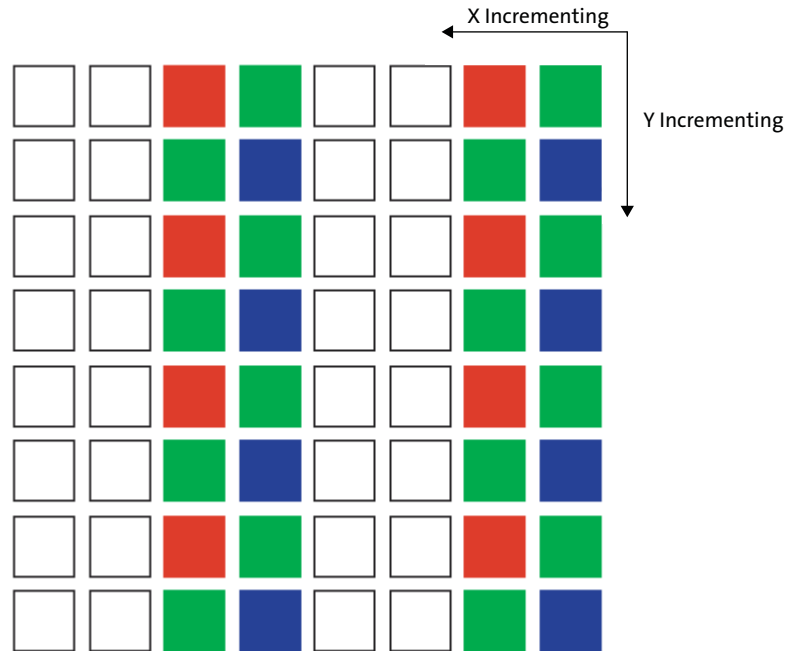
## Pixel Readouts

Figures 13 through Figure 16 on page 22 show a sequence of data being read out with no skipping, with  $x\_odd\_inc = 3$  and  $y\_odd\_inc = 1$ , with  $x\_odd\_inc = 1$  and  $y\_odd\_inc = 3$ , and with  $x\_odd\_inc = 3$  and  $y\_odd\_inc = 3$ .

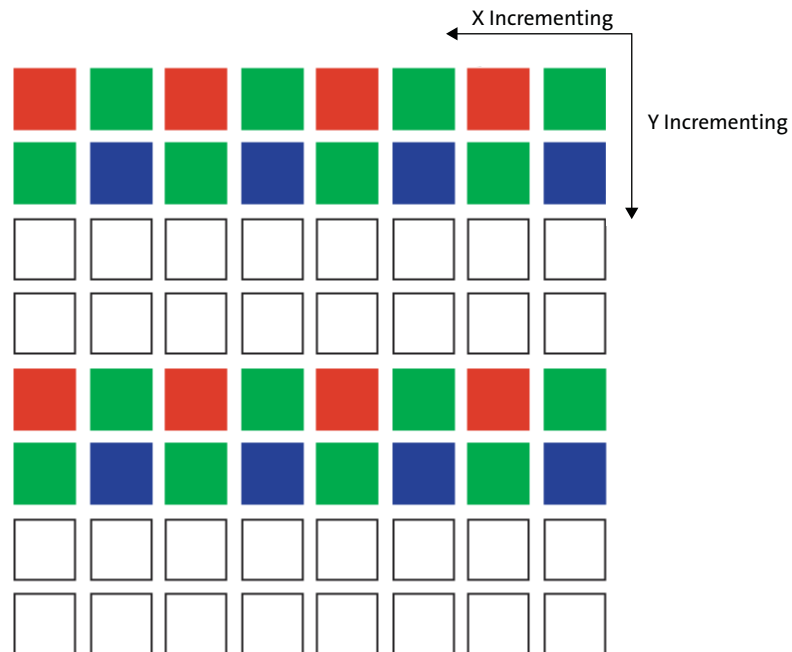
**Figure 13: Pixel Readout (no skipping)**

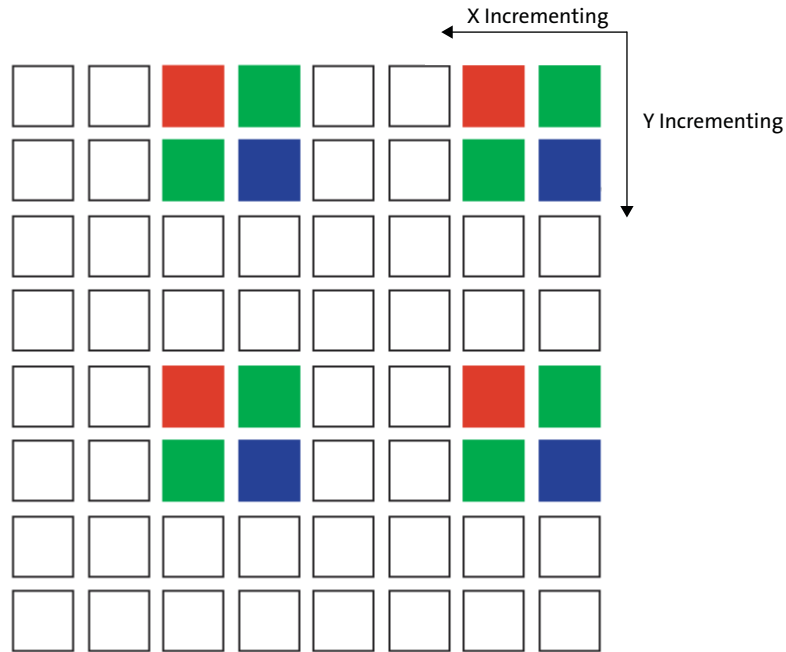


**Figure 14:** Pixel Readout ( $x\_odd\_inc = 3, y\_odd\_inc = 1$ )



**Figure 15:** Pixel Readout ( $x\_odd\_inc = 1, y\_odd\_inc = 3$ )



**Figure 16: Pixel Readout ( $x\_odd\_inc = 3, y\_odd\_inc = 3$ )**

### Programming Restrictions when Skipping

When skipping is enabled as a viewfinder mode, and the sensor is switched back and forth between full resolution and skipping, keep *line\_length\_pck* constant. This allows the same integration times to be used in each mode.

When subsampling is enabled, it might be necessary to adjust the *x\_addr\_end* and *y\_addr\_end* settings. The values for these registers must correspond with rows/ columns that form part of the subsampling sequence. Use the following rules to make the adjustment.

$$remainder = (addr\_end - addr\_start + 1) \text{ AND } 4 \quad (\text{EQ 2})$$

$$\text{if } (remainder == 0) \text{ } addr\_end = addr\_end - 2 \quad (\text{EQ 3})$$

Table 6 shows the row address sequencing for normal and subsampled (with  $y\_odd\_inc = 3$ ) readout. The same sequencing applies to column addresses for subsampled readout. Because the subsampling sequence only reads half of the rows and columns, there are two possible subsampling sequences, depending upon the alignment of the start address.

**Table 6: Row Address Sequencing (Sampling)**

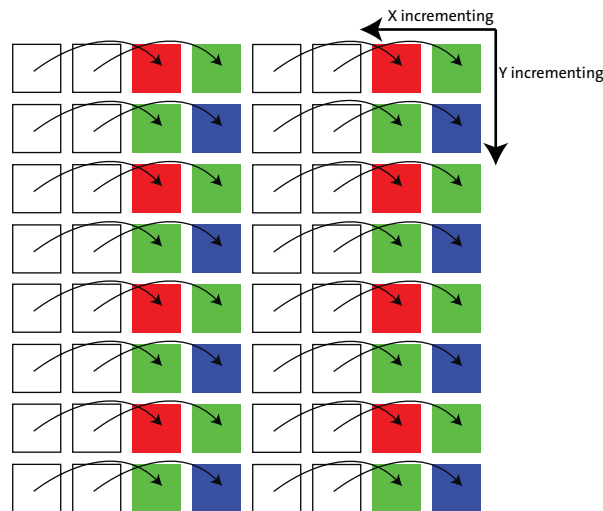
Normal	Subsampled Sequence 1	Subsampled Sequence 2
0	0	No data
1	1	No data
2	No data	2
3	No data	3
4	4	No data
5	5	No data
6	No data	6
7	No data	7

**Binning**

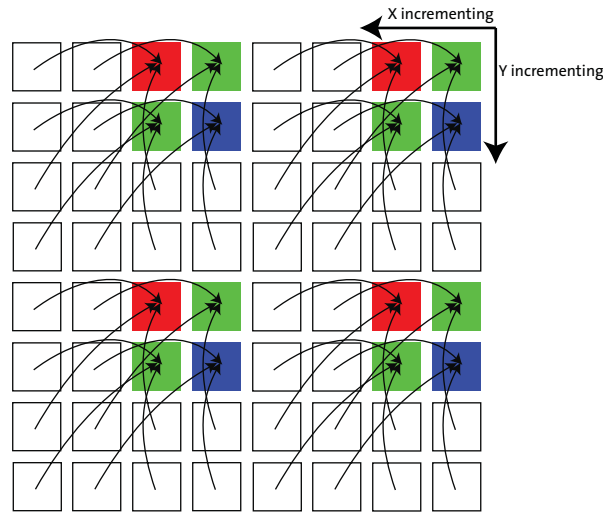
The MT9D115 sensor core supports 2 x 1 and 2 x 2 analog binning (column binning, also called x-binning and row/column binning, also called xy-binning). Binning has many of the same characteristics as subsampling. However, because binning gathers image data from all pixels in the active window, rather than from a subset of pixels, it achieves superior image quality and avoids the aliasing artifacts that can be a characteristic side effect of subsampling.

Enable binning by selecting the appropriate subsampling settings ( $x\_odd\_inc = 3$  and  $y\_odd\_inc = 1$  for x-binning,  $x\_odd\_inc = 3$  and  $y\_odd\_inc = 3$  for xy-binning) and setting the appropriate binning bit in read\_mode register. As for subsampling,  $x\_addr\_end$  and  $y\_addr\_end$  might require adjustment when binning is enabled.

The effect of the different subsampling settings is shown in Figure 17 and in Figure 18 on page 24.

**Figure 17: Pixel Readout ( $x\_odd\_inc = 3$ ,  $y\_odd\_inc = 1$ ,  $x\_bin = 1$ )**

**Figure 18: Pixel Readout (x\_odd\_inc = 3, y\_odd\_inc = 3, x\_ybin = 1)**



## Binning Limitations

Binning requires a different sequencing of the pixel array and imposes different timing limits on the operation of the sensor. In particular, xy-binning requires two READ operations from the pixel array for each line of output data, which has the effect of increasing the minimum line blanking time.

As a result, when xy-binning is enabled, some of the programming limits declared in the parameter limit registers are no longer valid. In addition, the default values for some of the manufacturer-specific registers need to be reprogrammed. None of these adjustments are required for x-binning. The sensor must be taken out of streaming mode before switching between binned and non-binned operation. The row addresses for various binning modes are shown in Table 7.

Table 7: 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, (see Figure 19). The amount of horizontal blanking and vertical blanking is programmable. LV is HIGH during the shaded region of the figure.

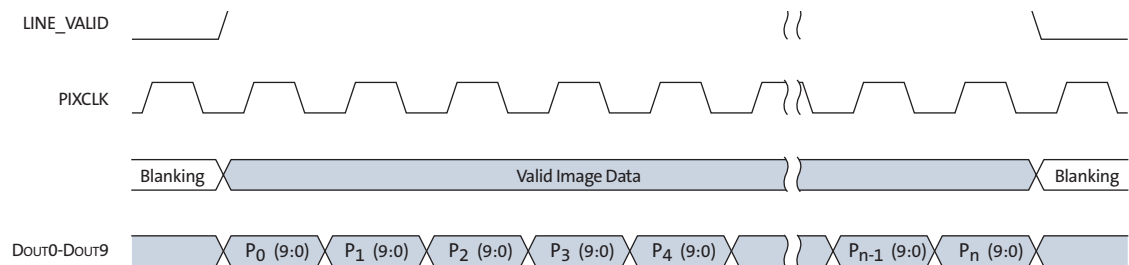


**Figure 19: Valid Image Data**

$P_{0,0} P_{0,1} P_{0,2} \dots P_{0,n-1} P_{0,n}$ $P_{1,0} P_{1,1} P_{1,2} \dots P_{1,n-1} P_{1,n}$  <div>VALID IMAGE</div>  $P_{m-1,0} P_{m-1,1} \dots P_{m-1,n-1} P_{m-1,n}$ $P_{m,0} P_{m,1} \dots P_{m,n-1} P_{m,n}$	$00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$  <div>HORIZONTAL BLANKING</div>  $00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$
$00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$  <div>VERTICAL BLANKING</div>  $00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$	$00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$  <div>VERTICAL/HORIZONTAL BLANKING</div>  $00\ 00\ 00 \dots 00\ 00\ 00$ $00\ 00\ 00 \dots 00\ 00\ 00$

**Raw Data Timing**

The sensor core output data 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 falling edges occur one-half of a master clock period after transitions on LV, FV, and DOUT[9:0] (see Figure 20). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking period.







**Figure 20: Pixel Data Timing Example**

## Input Interface to Image Flow Processor

The input interface to the IFP is the front end of the IFP, in which it will choose between the sensor core output or a test pattern generator output. During normal operation, a stream of raw Bayer image data from the sensor is continuously fed into the IFP. For testing purposes, the test generator output is selected. The generator provides a selection of test patterns sufficient for basic testing of the IFP. Program variable (ID=7, Offset=0x66) followed by REFRESH command to the sequencer in order to access different test patterns (see Figure 21).

Depending on which test pattern has been selected, the user might need to program additional registers in order to see the intended effects.

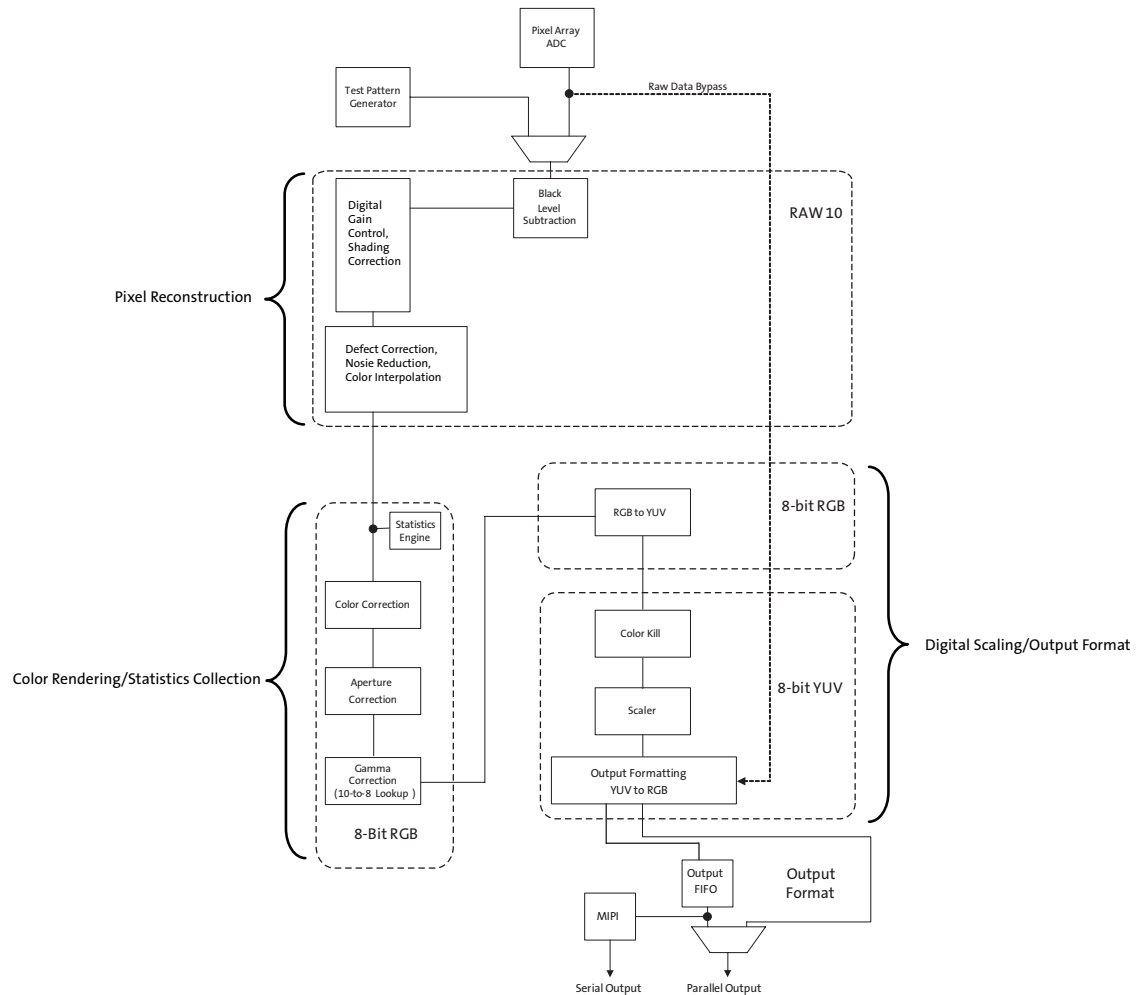
**Figure 21: Available Test Patterns**

Test Pattern	Registers/Variables	Example
Flat Field	<pre>mode_common settings_test_mode (ID=7, Offset=0x66)=1 test_pxl_red (R0x0102) = 0x1ff test_pxl_g1 (R0x0104) = 0x1ff test_pxl_g2 (R0x0106) = 0x1ff test_pxl_blue (R0x0108) = 0x1ff</pre>	
Vertical Ramp	<pre>mode_common_ mode_settings_ test_mode (ID=7, Offset=0x66)=2</pre>	
Color Bar	<pre>mode_common_ mode_settings_ test_mode (ID=7, Offset=0x66)=3</pre>	
Vertical Stripes	<pre>mode_common settings_test_mode (ID=7, Offset=0x66)=4 test_pxl_red (R0x0102) = 0x1ff test_pxl_g1 (R0x0104) = 0x17d test_pxl_g2 (R0x0106) = 0x000 test_pxl_blue (R0x0108) = 0x000</pre>	
Pseudo-Random	<pre>mode_common_mode settings_test_mode (ID=7, Offset=0x66)=5</pre>	
Horizontal Stripes	<pre>mode_common settings_test_mode (ID=7, Offset=0x66)=6 test_pxl_red (R0x0102) = 0x1ff test_pxl_g1 (R0x0104) = 0x17d test_pxl_g2 (R0x0106) = 0x000 test_pxl_blue (R0x0108) = 0x000</pre>	

## Image Flow Processor

Most IFP functions can be controlled directly by the MCU. The external host will control it indirectly through the driver variables. IFP processing can be broken into three different phases: pixel reconstruction, color rendering/statistics collection, and digital scaling/output format. Figure 22 shows a simplified IFP block diagram and the operating color space at each processing phase.

**Figure 22: IFP Block Diagram**



## Pixel Reconstruction (Lens Shading Correction)

### First 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."



## Shading Correction (SC)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. Other factors also cause fixed pattern signal gradients in images captured by image sensors. The cumulative result of all factors is known as image shading. The MT9D115 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.

## The Correction Function

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

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

$$P_{corrected}(row,col) = P_{sensor}(row,col) * f(row,col) \quad (EQ\ 4)$$

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

## Defect Correction

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

Noise reduction can be enabled or disabled. Thresholds can be set through register settings.

## Color Interpolation and Edge Detection

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. After the defect correction, the data stream 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.

## Aperture Correction

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 variable settings.



## Color Rendering/Statistics Collection (Color Correction)

### Color 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  $3 \times 3$  color correction matrix. The three components of the resulting color vector are all sums of three 10-bit numbers. Because 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 programmed by the user or automatically selected by the AWB algorithm implemented in the IFP. Ideally, color correction should produce output colors that are independent of 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.

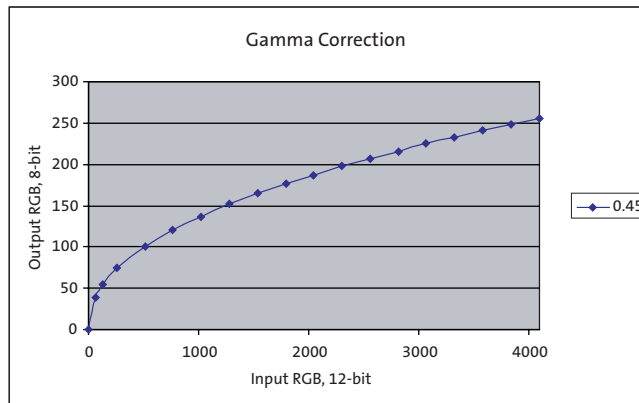
### Image Cropping

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 (see Figure 23 on page 30) 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 MT9D115 IFP includes a block for gamma correction that can adjust its shape based on brightness to enhance the performance under certain lighting conditions. Two custom gamma correction tables can be uploaded, one corresponding to a brighter lighting condition, the other corresponding to a darker lighting condition. At power-up, the IFP loads the two 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 the two tables. A single (non-adjusting) table for all conditions can also be used.

**Figure 23: Gamma Correction Curve****Color Kill**

A color kill circuit is included to remove high or low light color artifacts. 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.

**Digital Scaling/Output Format****Special Effects**

Special effects like negative image, sepia, or black and white 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.

**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**

The IFP includes a scaler module to ensure that the size of images output by the MT9D115 can be tailored to the needs of all users. 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.

Use the image cropping and scaler module 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 1600 x 1200 image rendered on a 250 x 150 display can be zoomed up to eight times, because  $1280/160 = 1024/128 = 8$ . A panning effect can be achieved by fixing the size of the cropping window and moving it around the pixel array.

### 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.

#### Color Conversion Formulas

##### Y'U'V'

This conversion is ITU-R BT.601 scaled to make YUV range from 0 through 255. This setting is recommended for JPEG encoding and is the most popular.

$$Y' = 0.299 \times R' + 0.587 \times G' + 0.114 \times B' \quad (\text{EQ 5})$$

$$U' = 0.564 \times (B' - Y') + 128 \quad (\text{EQ 6})$$

$$V' = 0.713 \times (R' - Y') + 128 \quad (\text{EQ 7})$$

There is an option where 128 is not added to U'V'.

##### Y'Cb'Cr' Using sRGB Formulas

The MT9D115 implements the sRGB standard. This option provides YCbCr coefficients for a correct 4:2:2 transmission.

**Note:**  $16 < Y' < 235; 16 < Cb < 240; 16 < Cr < 240; \text{ and } 0 \geq RGB \geq 255$

$$Y' = (0.2126 \times R' + 0.7152 \times G' + 0.0722 \times B') \times (219/256) + 16 \quad (\text{EQ 8})$$

$$Cb' = 0.5389 \times (B' - Y') \times (224/256) + 128 \quad (\text{EQ 9})$$

$$Cr' = 0.635 \times (R' - Y') \times (224/256) + 128 \quad (\text{EQ 10})$$

##### Y'U'V' Using sRGB Formulas

Similar to the previous set of formulas, but has YUV spanning a range of 0 through 255.

$$Y' = 0.2126 \times R' + 0.7152 \times G' + 0.0722 \times B' \quad (\text{EQ 11})$$

$$U' = 0.5389 \times (B' - Y') + 128 = -0.1146 \times R' - 0.3854 \times G' + 0.5 \times B' + 128 \quad (\text{EQ 12})$$

$$V' = 0.635 \times (R' - Y') + 128 = 0.5 \times R' - 0.4542 \times G' - 0.0458 \times B' + 128 \quad (\text{EQ 13})$$

There is an option to disable adding 128 to U'V'. The reverse transform is as follows:

$$R' = Y' + 1.5748 \times (V' - 128) \quad (\text{EQ 14})$$

$$G' = Y' - 0.1873 \times (U' - 128) - 0.4681 \times (V' - 128) \quad (\text{EQ 15})$$

$$B' = Y' + 1.8556 \times (U' - 128) \quad (\text{EQ 16})$$

## Output Interface from IFP

The output interface contains parallel image bus, MIPI serial bus interfaces, and walking 1s connectivity test generator.

### Parallel and MIPI Output

The user can select to use either the serial MIPI output or the 10-bit parallel output to transmit the data. Only one of the output modes can be used at any time.

The parallel output can be 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.

When scaling the image or skipping lines, the data would be generated in bursts and the pixel clock would turn on and off in intervals, which might lead to EMI problems. The output FIFO will group all active pixel data together so the pixel clock can be run at a constant speed.

The MIPI output transmitter implements a serial differential sub-LVDS transmitter capable of up to 512 Mb/s. It supports multiple formats, error checking, and custom short packets. The MIPI clock is defined as half the data rate frequency.

The virtual channel could be used by host MIPI RX circuits to differentiate between preview and capture image data if the FW changes the channel number when switching between contexts. The host cannot adequately time this switching of contexts and so cannot use channel number in this way without FW support.

The hardware supports a configurable MIPI virtual channel in the MIPI Control register (R0x3400). Firmware provides two variables that allow the MIPI virtual channel to be changed according to context (preview/A, capture/B). The host can specify what channel is used for which context through these variables.

**Table 8: 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

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



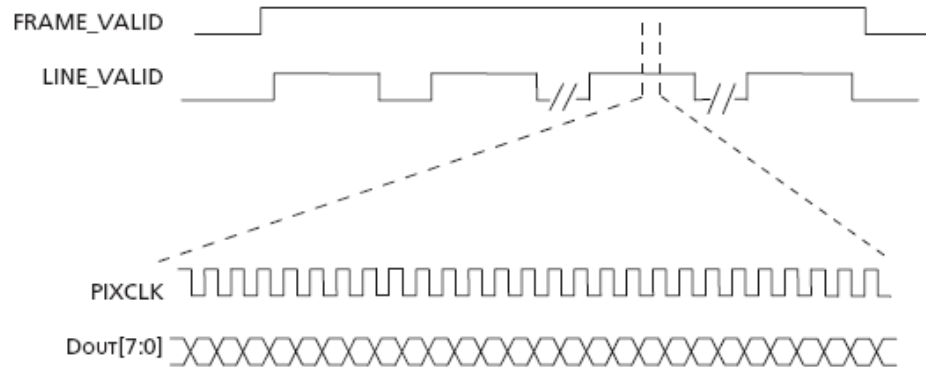
## Output Format and Timing

### YUV/RGB Output

YUV or RGB data can be output either directly from the output formatting block or through a FIFO buffer with a capacity of 1024 bytes. This size is large enough to hold one-fourth of a scan line at full resolution. Buffering of data is a way to equalize the data output rate when image scaling is used. Scaling produces an intermittent data stream consisting of short high-rate bursts separated by idle periods. High pixel clock frequency during bursts may be undesirable due to EMI concerns.

Figure 24 shows the output timing of a YUV/RGB scan line 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. Scaled data is output at a lower frequency than full size frames, which helps to reduce EMI.

**Figure 24: Timing of Full Frame Data or Scaled Data Passing Through the FIFO**



### YUV/RGB Data Ordering

The MT9D115 supports swapping YCbCr mode (see Table 9).

**Table 9: YCbCr Output Data Ordering**

Mode	Data Sequence			
Default (no swap)	Cb <sub>i</sub>	Y <sub>i</sub>	Cr <sub>i</sub>	Y <sub>i+1</sub>
Swapped CrCb	Cr <sub>i</sub>	Y <sub>i</sub>	Cb <sub>i</sub>	Y <sub>i+1</sub>
Swapped YC	Y <sub>i</sub>	Cb <sub>i</sub>	Y <sub>i+1</sub>	Cr <sub>i</sub>
Swapped CrCb, YC	Y <sub>i</sub>	Cr <sub>i</sub>	Y <sub>i+1</sub>	Cb <sub>i</sub>

The RGB output data ordering in default mode is shown in Table 10. The odd and even bytes are swapped when luma/chroma swap is enabled. R and B channels are bitwise swapped when chroma swap is enabled.

**Table 10: RGB Ordering in Default Mode**

Mode (Swap Disabled)	Byte	D <sub>7</sub> D <sub>6</sub> D <sub>5</sub> D <sub>4</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>
565RGB	Odd	R <sub>7</sub> R <sub>6</sub> R <sub>5</sub> R <sub>4</sub> R <sub>3</sub> G <sub>7</sub> G <sub>6</sub> G <sub>5</sub>
	Even	G <sub>4</sub> G <sub>3</sub> G <sub>2</sub> B <sub>7</sub> B <sub>6</sub> B <sub>5</sub> B <sub>4</sub> B <sub>3</sub>

**Table 10: RGB Ordering in Default Mode (continued)**

Mode (Swap Disabled)	Byte	D <sub>7</sub> D <sub>6</sub> D <sub>5</sub> D <sub>4</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>
555RGB	Odd	0 R <sub>7</sub> R <sub>6</sub> R <sub>5</sub> R <sub>4</sub> R <sub>3</sub> G <sub>7</sub> G <sub>6</sub>
	Even	G <sub>4</sub> G <sub>3</sub> G <sub>2</sub> B <sub>7</sub> B <sub>6</sub> B <sub>5</sub> B <sub>4</sub> B <sub>3</sub>
444xRGB	Odd	R <sub>7</sub> R <sub>6</sub> R <sub>5</sub> R <sub>4</sub> G <sub>7</sub> G <sub>6</sub> G <sub>5</sub> G <sub>4</sub>
	Even	B <sub>7</sub> B <sub>6</sub> B <sub>5</sub> B <sub>4</sub> 0 0 0 0
x444RGB	Odd	0 0 0 0 R <sub>7</sub> R <sub>6</sub> R <sub>5</sub> R <sub>4</sub>
	Even	G <sub>7</sub> G <sub>6</sub> G <sub>5</sub> G <sub>4</sub> B <sub>7</sub> B <sub>6</sub> B <sub>5</sub> B <sub>4</sub>

**10-Bit Bypass Output**

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

1. Using eight data output signals (DOUT[7:0]) and GPIO[1:0]. The GPIO signals are the lowest two bits of data.
2. Using only eight signals (DOUT[7:0]) and a special 8 + 2 data format, shown in Table 11.

**Table 11: 2-Byte RGB Format**

Byte	Bits Used	Bit Sequence
Odd bytes	8 data bits	D <sub>9</sub> D <sub>8</sub> D <sub>7</sub> D <sub>6</sub> D <sub>5</sub> D <sub>4</sub> D <sub>3</sub> D <sub>2</sub>
Even bytes	2 data bits + 6 unused bits	0 0 0 0 0 0 D <sub>1</sub> D <sub>0</sub>

**FIFO**

During normal pipeline operation, the output data rate is determined by a number of factors: input image size, degree of scaling, and sensor operation mode. As these parameters change during normal sensor operation, output frequency changes. This output frequency may generate RF noise, interfering with the mobile device. By using an output FIFO to maintain a constant output clock frequency, noise is easily filtered out.

The FIFO accumulates data and after a certain number of bytes are stored, it will output them in a single burst, making sure that the data rate within the burst remains constant. This approach utilizes a free-running clock, thus making possible minimal RF interference.



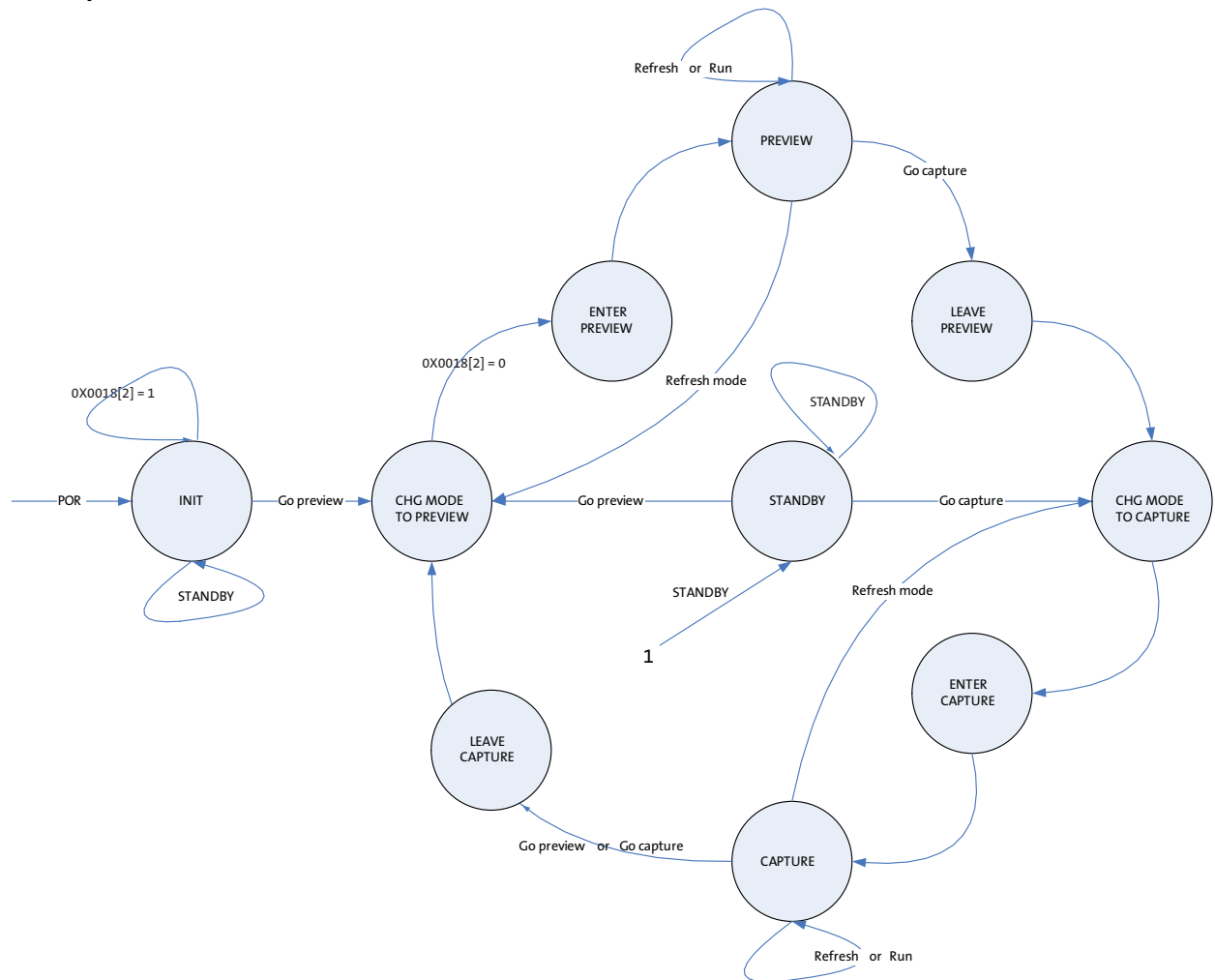
## Control Functions

### Sequencer: Camera Operating System

The sequencer is a finite state machine that controls the general operations of the camera and switching between operating modes. Camera operation is organized in states such as enter preview, preview, leave preview, enter capture, and capture. The sequencer carries out a number of commands such as run, go preview, go capture, refresh, and refresh mode. The current state of the sequencer is indicated in a variable *seqr.state*. To execute a command, the user must set a particular command number in the variable *seqr.cmd*. Aptina recommends that the external host monitor *seqr.state* to know when to change resolution or capture frames.

Each state has its configuration; therefore, the user should set up the state configuration to customize the camera configuration before executing the corresponding program such as GO TO CAPTURE. A typical camera operating scenario is:

1. Configure mode variables after hardware reset.
2. Configure preview mode.
3. Execute sensor REFRESH program.
4. Run in preview until shutter button is pressed.
5. Capture a frame.

**Figure 25: Sequencer Finite State Machine**

Note: Any state except INIT can transition to Standby state by either hard standby or soft standby.

## Mode: Context Information

The mode driver reduces integration efforts by managing most aspects of switching the two contexts. It remembers vital register values for each image acquisition context and loads these values to the appropriate registers in the IFP upon context switching.

For the mode driver variables to take effect, the user changes the variable values in the mode driver (ID = 7). Upon the next mode change or sequencer's REFRESH command, these driver variable values will be loaded to the appropriate physical sensor core and IFP registers. The image processing will take the new values at the beginning of the next frame acquired.

To control the output image size, the user can modify the mode driver variables such as *output\_width\_A*, *output\_height\_A*, *output\_width\_B*, and *output\_height\_B*. The mode driver will automatically apply any appropriate downscaling filter to achieve this output image size as well as update the watermark of the output FIFO. It is important to set up the sensor core to output an image equal to or larger than the crop window size, which in turn is equal to or larger than the desired output image size.



## Camera Functions

### Simple Rule-based Auto Exposure

The AE 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 (FW) 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 16 windows organized as a 4 x 4 grid.

An AE algorithm mode is available: average brightness tracking (ABT).

The ABT 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.

### AE Driver

DRT exposure mode is activated during preview. This mode can also be enabled during video capture mode. The DRT exposure mode relies on the statistics engine to track 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.

The driver calculates image brightness based on average luma values received from 16 programmable equal-size rectangular windows forming a 4 x 4 grid. In preview mode, 16 windows are combined in two segments: central and peripheral. The central segment includes four central windows. All remaining windows belong to the peripheral segment. Scene brightness is calculated as average luma in each segment taken with certain weights.

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 filtered luma is larger than the AE target step and pushes the luma beyond the threshold.

### Evaluative Algorithm

A scene-evaluative AE algorithm is available for use in snapshot mode. The algorithm performs scene analysis and classification with respect to its brightness, contrast, and composure and then decides to increase, decrease, or keep the original exposure target. It makes the most difference for backlight and bright outdoor conditions.

### Accelerated Settling During Overexposure

The AE speed is direction-dependent. Transitioning from overexposure to target can take more time than transitioning from underexposure. 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 that has saturated regions, the average luma is underestimated because of 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, `ae_base_target` (ID = 2, offset = 0x4F) is available for the user to adjust the overall brightness of the scene.

### Flicker Detection and Avoidance

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.

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 to an integral factor of the flicker period.

Flicker cannot be avoided for integration times below the light intensity period (10ms for 50Hz environment).

Flicker shows up in the image as horizontal bars that roll up or down. The MCU looks for these rolling bars using a thin horizontal window, which outputs luma average and is applied to 48 points in the upper half of the image. The MCU repeats the same sampling on the next frame; 48 samples from the previous frame are subtracted from corresponding samples from the current frame. Skipping more frames between subtraction can be set by a variable. The MCU then smooths the 48 sampling points, applies an amplitude threshold to avoid false detection, and looks at the resulting waveform. If flicker is present, the waveform should have a frequency within the search range. Assuming the flicker power is a sine wave, subtracting two frames results in:

$$\sin(wt) - \sin(wt + a) = 2 \times \sin(a/2) \times \cos(wt + (a/2)) \quad (\text{EQ 17})$$

which is a cosine wave of the original frequency.



## Auto White Balance

The MT9D115 has a built-in AWB 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 gains, and sensor core analog gains. While default settings of these algorithms are adequate in most situations, the user can reprogram the base color correction matrices, place limits on color channel gains, and control the speed of both matrix and gain adjustments.

## Flash

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

1. The camera might 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 the images are captured, the camera automatically returns to context A and resumes running in preview mode.

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

## Histogram: Dark Level Adjustments, Low Light and Tonal Controls

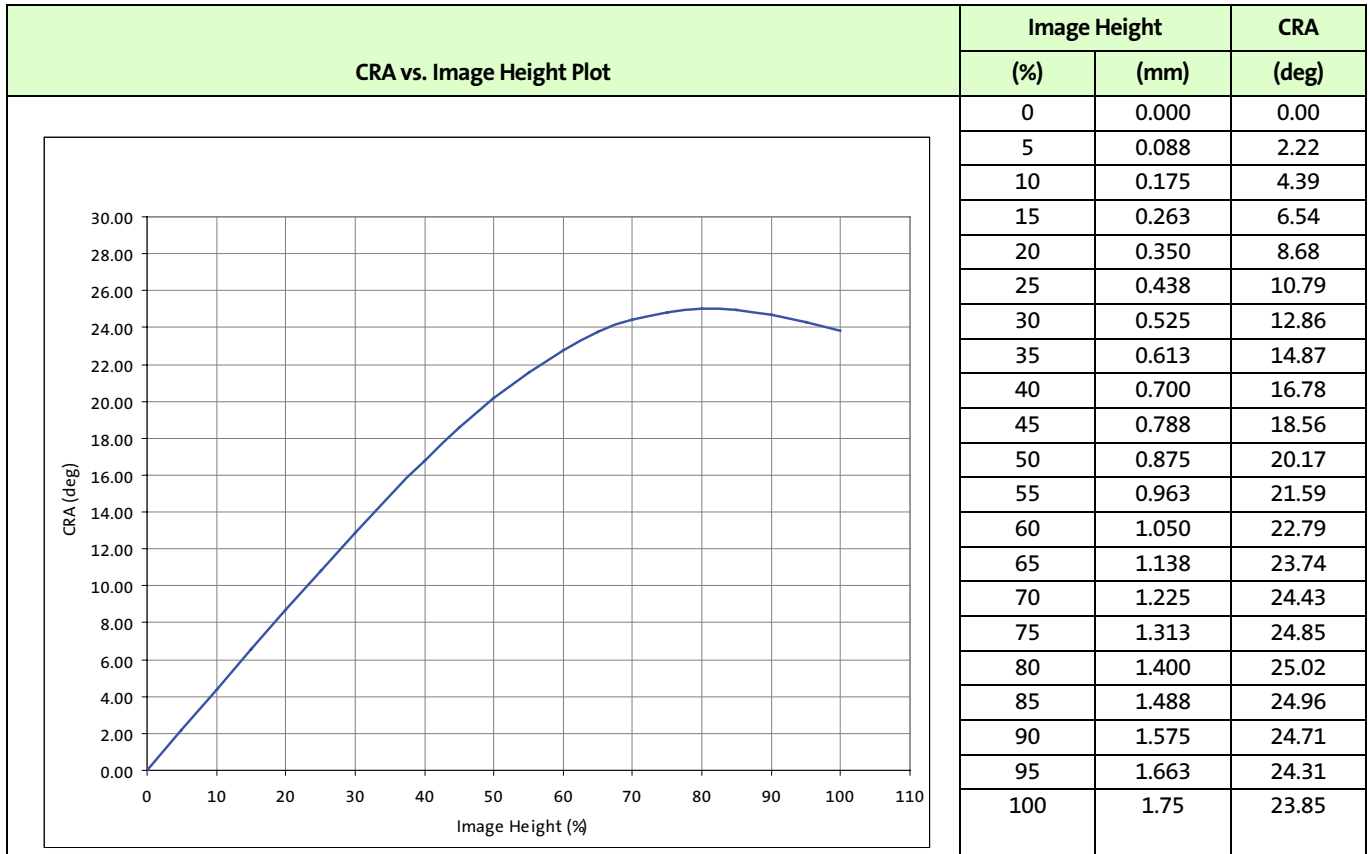
The histogram driver continually works to reduce image flare and continually analyzes input image histogram and dynamically adjusts the black level. When flare is present (hence, the histogram does not contain dark tones), it causes the driver to subtract a higher black level, thus regaining the lost contrast. In certain situations, the scene may contain no dark tones without flare. The histogram driver cannot distinguish this situation and alters the black level just the same, causing the image to have more contrast, which looks acceptable in many situations.

Besides black level adjustments and low light scene parameters, this driver also contains variables for the preloaded and custom gamma tables.

## Optics

Figure 26 shows the CRA versus image height.

**Figure 26: CRA vs. Image Height**



**Note:** Aptina recommends a 670nm IR cut filter to achieve the best image quality; however, a 650nm IR cut filter is acceptable.





## Power Modes

### Power Application Sequence

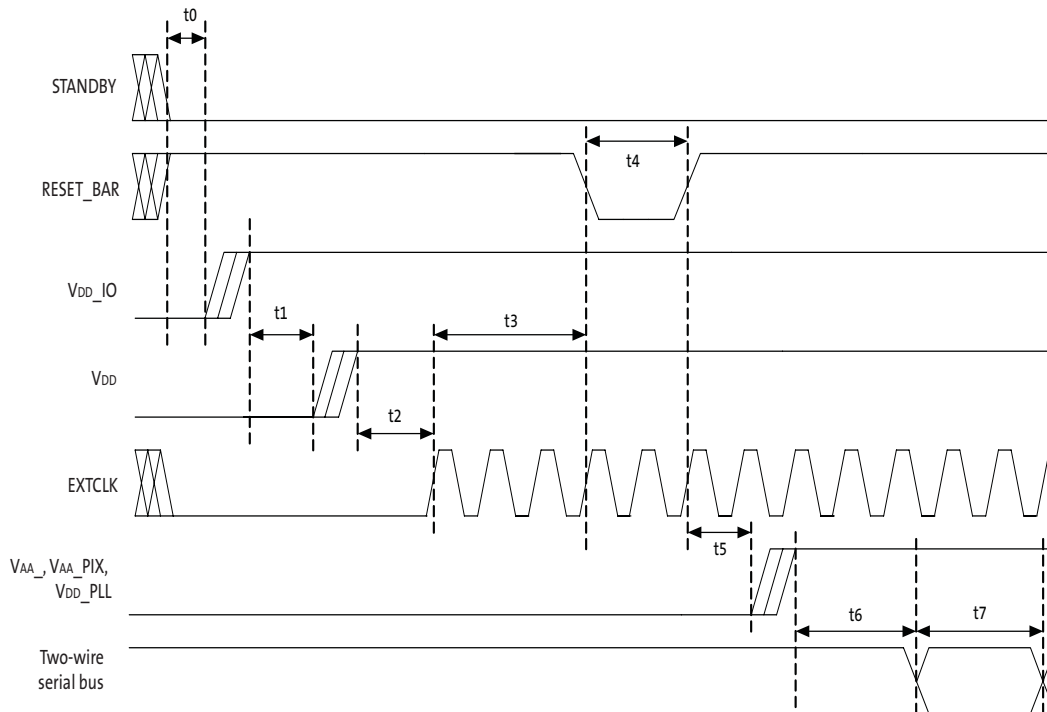
Aptina recommends the following sequence to maintain low power consumption during this process:

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

1. Ensure that STANDBY is de-asserted and RESET\_BAR is asserted.
2. Apply IO supply (VDD\_IO) to the image sensor and wait for the IO supply to be stable.
3. Minimum of 1 ms after IO supply is stable, apply digital supply to the image sensor and wait.
4. Enable EXTCLK and wait for the EXTCLK signal to stabilize.
5. De-assert RESET\_BAR for a minimum of 10 EXTCLK cycles.
6. After asserting the RESET\_BAR, apply analog supplies (VAA, VAA\_PIX, and VDD\_PLL) to the image sensor.
7. After 6000 EXTCLK cycles from the end of step 6, the image sensor will be in soft standby state.
8. Communication with the sensor thorough two-wire serial interface can start 1 EXT-CLK after step 7.

In cases where the recommended procedure cannot be followed, the following condition would affect the sensor's power consumption during the power application sequence:

- When analog supplies are applied prior to the digital and IO supplies, high current consumption on the analog supplies may be present.

**Figure 27: Power Application Sequence Timing****Table 12: Power Application Sequence Timing**

Parameters	Symbol	Min	Typ	Max	Units
Delay from stable RESET_BAR, STANDBY signals to VDDIO power start	t0	0	—	—	ns
Delay from stable VDD_IO to VDD start	t1	1	—	—	ms
Delay from stable VDD power to EXTCLK start	t2	0	—	—	ns
Delay from EXTCLK start to stable EXTCLK	t3	1	—	—	EXTCLK
RESET_BAR pulse width	t4	10	—	—	EXTCLK
Delay from RESET_BAR de-asserting to analog power supplies start	t5	1	—	—	EXTCLK
Delay from analog power stable to soft standby mode	t6	6000	—	—	EXTCLK
Delay from soft standby mode to first two-wire bus transaction	t7	1	—	—	EXTCLK

## Power-On Reset

The sensor includes a power-on reset feature that initiates a reset upon power-up. Even though this feature is included on the device, it is advised that the user still manually assert a hard reset upon power-up.

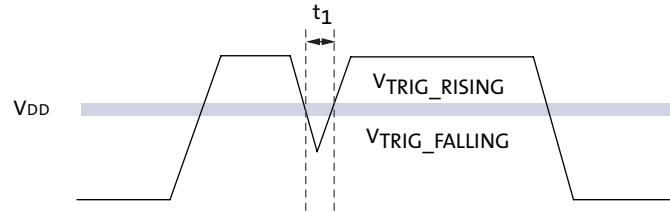
The MT9D115's POR circuit generates internal reset only and it will not generate the external reset signal through RESET\_BAR.

The POR circuit requires VDD ramp time to be less than 10 $\mu$ s. If the ramp time is longer than 10 $\mu$ s, the POR operation is not guaranteed and external reset must be used.

The POR circuit will generate an internal reset when VDD falls below 1.25V (typical) for 1 $\mu$ s (typical) period, as shown in Figure 28 and described in Table 13.

The POR circuit reset and external RESET\_BAR signals are gated together to generate an internal reset for MT9D115.

**Figure 28: Internal Power-On Reset**



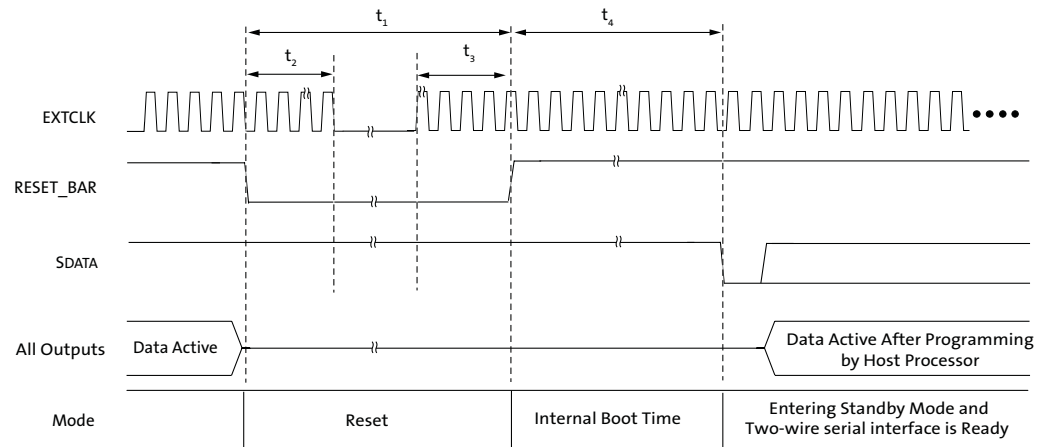
**Table 13: POR Parameters**

Symbol	Definition	Min	Typ	Max	Unit
$t_1$	Minimum VDD spike width below VTRIG_FALLING considered to be a reset	—	1	—	$\mu$ s
VTRIG_RISING	VDD rising trigger voltage	1.15	1.4	1.55	V
VTRIG_FALLING	VDD falling trigger voltage	1	1.25	1.45	V

## Hard Reset

The MT9D115 enters the reset state when the external RESET\_BAR signal is asserted LOW, as shown in Figure 29 and described in Table 14. All of the output signals will be in High-Z state except the MIPI outputs, which will be driven LOW.

**Figure 29: Hard Reset Operation**



**Table 14: Hard Reset**

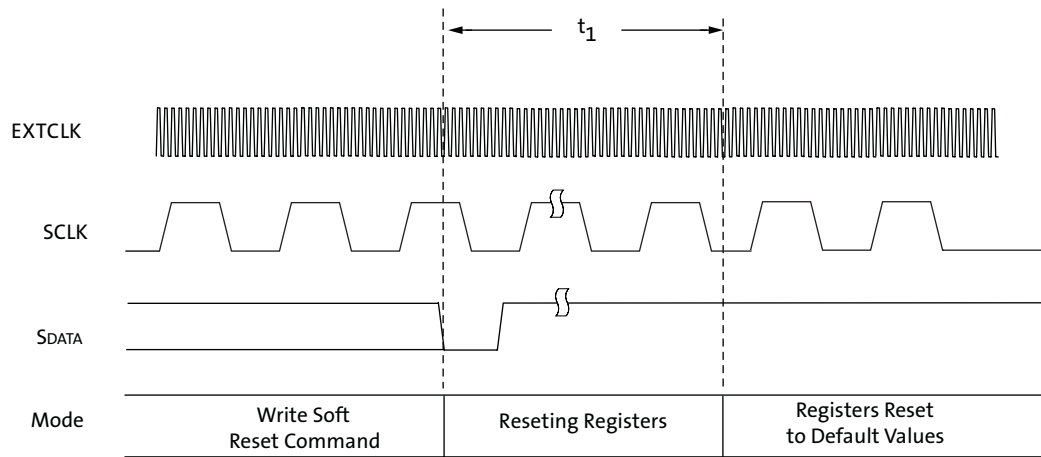
Symbol	Definition	Min	Typ	Max	Unit
$t_1$	RESET_BAR pulse width	100	—	—	EXTCLK cycles
$t_2$	Active EXTCLK required after RESET_BAR asserted	10	—	—	
$t_3$	Active EXTCLK required before RESET_BAR de-asserted	10	—	—	
$t_4$	Maximum internal boot time	—	—	6000	

## Soft Reset

The host processor can reset the MT9D115 using the two-wire serial interface by writing to SYSCCTL 0x001A. Two types of soft reset are available. SYSCCTL 0x001A[0] is used to reset the MT9D115, which is similar to external RESET\_BAR signal.

1. Set SYSCCTL 0x001A[0] to 0x3 to initiate internal reset cycle.
2. Wait 6000 EXTCLK cycles.
3. Reset SYSCCTL 0x001A[0] to 0x0 for normal operation.

**Figure 30: Soft Reset Operation**



**Table 15: Soft Reset Signal Timing**

Symbol	Parameter	Min	Typ	Max	Unit
$t_1$	Maximum soft reset time	–	–	6000	EXTCLKs

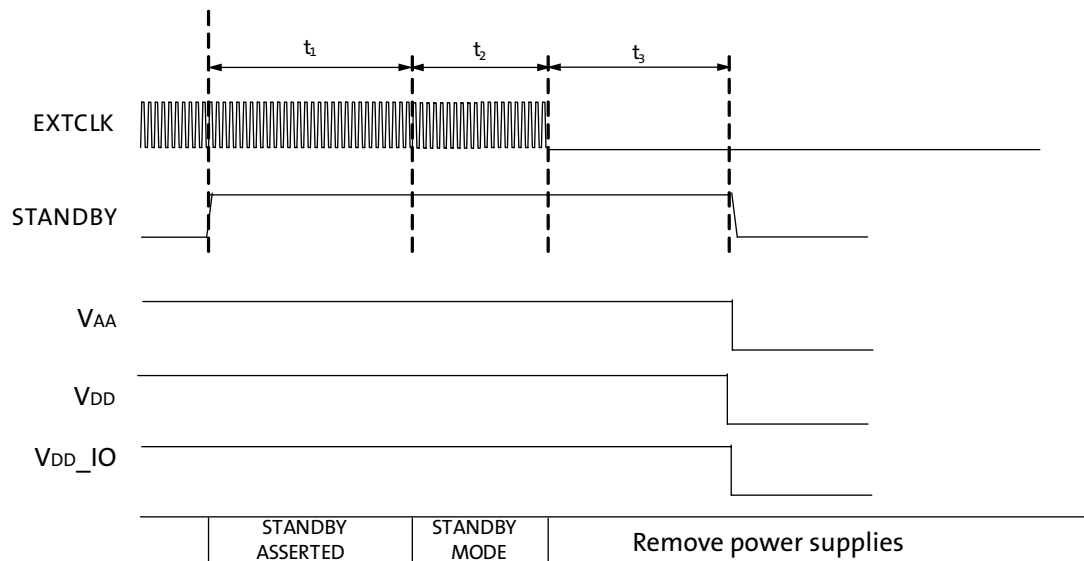
## Power Removal Sequence

Aptina recommends using the following method to remove the power supplies from the image sensor.

1. Put the image sensor into either hard standby or soft standby mode as described in “Standby Modes” on page 47.
2. Remove all digital and analog supplies.

During step 2, the digital and analog supplies can be removed safely either one by one or at the same time, provided that step 1 is executed previously. To achieve a known power down state in the sensor, execute step 1 before removing any power supplies during the power removal process.

**Figure 31: Recommended Power Down Sequence**



**Table 16: Power Down Signal Timing**

Symbol	Parameter	Min	Typ	Max	Unit
$t_1$	Standby entry complete	1 frame + 40	–	–	EXTCLKs
$t_2$	Active EXTCLK required after STANDBY asserted	10	–	–	EXTCLKs
$t_3$	Power Supply Removal can be in any order and anytime after completion of $t_2$	0	–	–	EXTCLKs



## Standby Modes

The MT9D115 supports three standby modes:

- Soft standby
- Low leakage hard standby
- High leakage hard standby

For each mode, entry can be overridden by programming the *standby\_control* register. Patch codes, PLL configurations, parallel and MIPI IO selections are retained in all standby modes.

### Soft Standby

Soft standby can be enabled by register access; it disables the sensor core and most of the digital logic. The two-wire serial interface is still active and the MT9D115 can be programmed through register commands. All register settings and RAM content will be preserved. Soft standby can be performed in any sequencer state.

### Low Leakage Hard Standby

Hard standby mode uses the STANDBY signal to shut down digital power (VDD) and ensure the lowest power consumption. All the two-wire serial interface settings and firmware variables except patch codes, PLL configuration, parallel/MIPI IO and context selections will be lost in this mode. Starting up from this mode is equivalent to power up and current context selections. The two-wire serial interface will be inactive and the sensor must be started up by de-asserting the STANDBY signal.

### High Leakage Hard Standby (Without Loss of Variable Data)

High leakage hard standby mode (without the loss of variable data) can also be achieved. This mode stores the variables and state of the sensor before entering standby (similar to soft standby). The power consumption is lower than that of soft standby, with EXTCLK enabled, as internal clocks are turned off, and the two-wire serial interface will be inactive.

Because the high leakage hard standby mode (without the loss of variable data) is also activated by STANDBY, the *en\_vdd\_dis\_soft* register needs to be programmed to indicate the selection of this mode before STANDBY is asserted. De-asserting STANDBY will cause the sensor to come out of the standby mode. This also causes the sensor to resume operation from the state before the STANDBY signal was asserted. By default, asserting the STANDBY signal causes the hard standby mode described above.

## Hard Standby Mode

The MT9D115 can enter hard standby mode by using external STANDBY signal, as shown in Figure 32. EXTCLK can be stopped to reduce the power consumption during hard standby mode. Two-wire serial interface and IFP block shut down even when EXTCLK is running during hard standby mode.

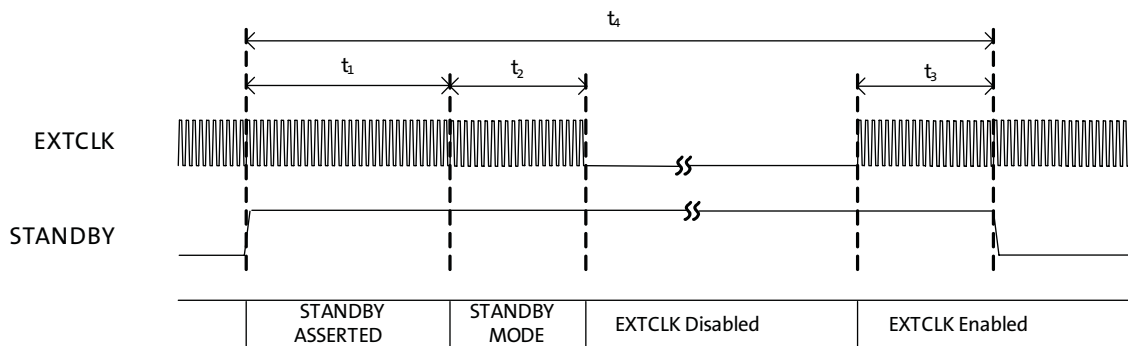
### Entering Standby Mode

1. Assert STANDBY signal (HIGH).

### Exiting Standby Mode

1. De-assert STANDBY signal (LOW).
2. Follow “Hard Reset” on page 44.

**Figure 32: Hard Standby Mode Operation**



Note: In hard standby mode, EXTCLK is automatically gated off, and the two-wire serial interface is not active.

**Table 17: Hard Standby Signal Timing**

Symbol	Parameter	Min	Typ	Max	Unit
$t_1$	Standby entry complete	1 Frame + 40	—	—	EXTCLKs
$t_2$	Active EXTCLK required after STANDBY asserted	10	—	—	EXTCLKs
$t_3$	Active EXTCLK required before STANDBY de-asserted	10	—	—	EXTCLKs
$t_4$	STANDBY time	1 Frame + 120	—	—	EXTCLKs
$t_5$	Active EXTCLK required before RESET_BAR asserted	10	—	—	EXTCLKs



## Soft Standby Mode

The MT9D115 can enter soft standby mode by writing to a SYSTL register through the two-wire serial interface, as shown in Figure 33. EXTCLK can be stopped to reduce the power consumption during soft standby mode. However, since two-wire serial interface requires EXTCLK to operate, Aptina recommends that EXTCLK run continuously. If EXTCLK needs to be stopped, Aptina recommends using hardware standby mode.

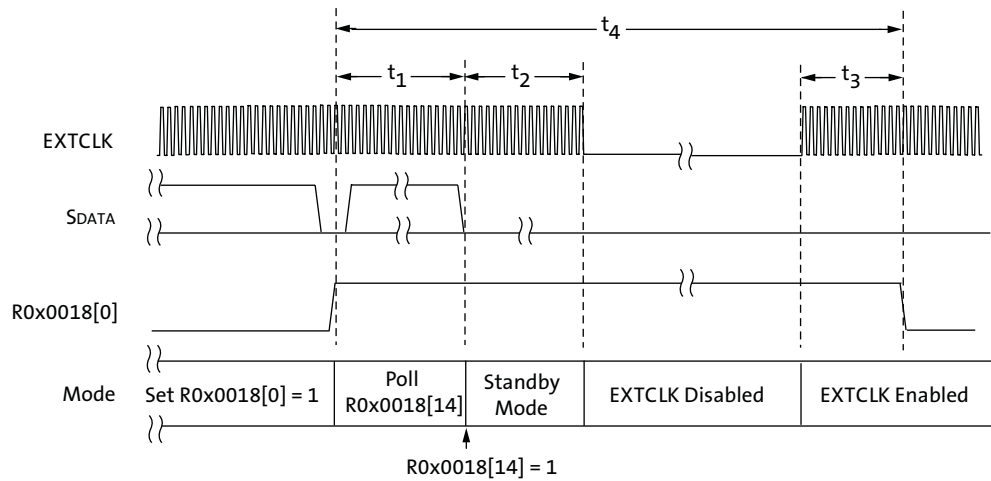
### Entering Standby Mode

1. SYSTL 0x0018[3] must be set to “1” to activate standby mode. This will generate an interrupt to the MCU when SYSTL 0x0018[0] is set to “1.”
2. Set SYSTL 0x0018[0] to “1” to initiate standby mode.
3. Check until SYSTL 0x0018[14] changes to “1” to indicate MT9D115 is in standby mode.

### Exiting Standby Mode

1. Reset SYSTL 0x0018[0] to “0.”
2. Check until SYSTL 0x0018[14] changes to “0” to indicate the MT9D115 is out of standby mode.

**Figure 33: Soft Standby Mode Operation**



**Table 18: Soft Standby Signal Timing**

Symbol	Parameter	Min	Typ	Max	Unit
$t_1$	Standby entry complete (R0x18[14] = 1)	1 Frame + 40	—	—	EXTCLKs
$t_2$	Active EXTCLK required after soft standby activates	10	—	—	EXTCLKs
$t_3$	Active EXTCLK required before soft standby de-activates	10	—	—	EXTCLKs
$t_4$	Minimum standby time	1 Frame + 120	—	—	EXTCLKs

**Table 19: Status of Signals During Different States**

Signal	Reset	Post-Reset	Software Standby	Low Power Hard Standby	Power Down
DOUT[7:0]	High-Z	High-Z	High-Z by default (configurable through OE_BAR or two-wire serial interface register)	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
GPIO[3:0]	High-Z	High-Z	Depending on how the system uses them as DOUT_LSB1/DOUT_LSB0/FLASH/OE_BAR	Depending on how the system uses them as DOUT_LSBs/FLASH/OE_BAR	X
SADDR	Input	Input	Input	Input	
SDATA	Input	I/O	Input	Input	
SCLK	Input	Input	Input	Input	

Note: X = "Don't Care."

## Other Features

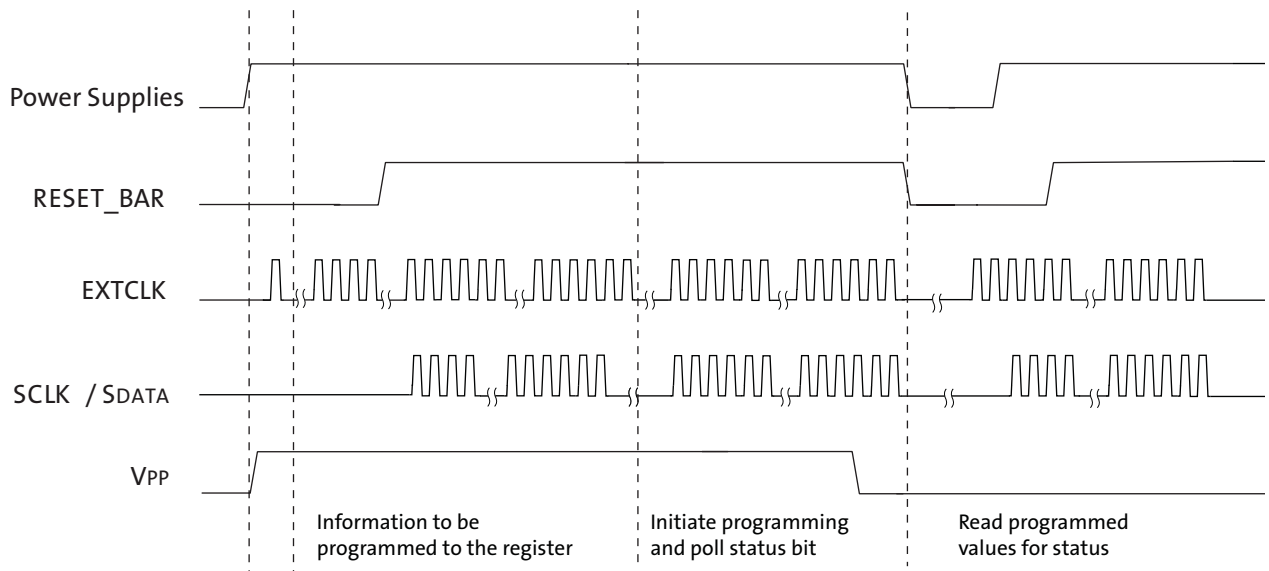
### One-time Programmable Memory (OTPM)

The MT9D115 contains two bytes of OTPM, suitable for storing 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 OTPM can be accessed through the two-wire serial interface.

#### Programming the OTPM

Refer to TN-09-189: Programming OTP Memory to program the OTP Memory.

#### Sequence of Signals for OTPM Operation



#### Reading the OTPM

Reading the OTPM data requires the sensor to be fully powered and operational with its clock input applied. The data can be read through a register from the two-wire serial interface.



## GPIO and Output Enable Controls

### General Purpose I/Os

The four GPIOs of the MT9D115 can be configured in multiple ways. Each of the I/Os can be used as a simple input/output that can be programmed from the host. The status of the GPIO is read at power up and can be used as a module ID to identify different module suppliers. In addition, module ID can be stored in the OTP memory of the sensor. Information on the OTP memory can be found in “One-time Programmable Memory (OTPM)” on page 51.

If 10-bit RAW output is required, GPIO[1:0] can be configured as bit 1 and bit 0 (the LSBs) of a 10-bit data bus.

GPIO[2] can be configured to output a flash pulse to trigger an external xenon or LED flash or a shutter pulse to control an external shutter.

GPIO[3] can also be configured as an input to be used as an OE\_BAR signal for the data bus.

The general purpose inputs are enabled or disabled through register settings. Once enabled, all four inputs must be driven to valid logic levels by external signals. 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 being driven and High-Z under signal or register control.



## GPIO Control

The MT9D115 has four general purpose I/O (GPIO) signals and that can be individually programmed to perform various functions. Table 20 describes the GPIO registers and variables.

### Overview of GPIO Signals

- Extension of two lower data bits during 10-bit output mode
- External flash control output
- External OE\_BAR control input for parallel port signals

**Table 20: GPIO Related Registers and Variables**

Map	Address	Bits	Descriptions
SYSCTL	0x001A	[8]	GPIO[3] Enable. GPIO[3] can be configured as an Output Enable pad. 0: GPIO[3] does not function as OE. 1: GPIO[3] functions as OE.
SYSCTL	0x001E	[6:4]	Controls the output slew rate of GPIO signals. “111” programs the fastest slew rate. Refer to AC/DC specification in the data sheet for the numbers. Default value is “000” for slowest slew rate.
SYSCTL	0x0024	[3:0]	The state of GPIO signals during power on. Read-only.
GPIO	0x1070	[12:9]	GPIO data port. State of current GPIO signals can be read through this register.
GPIO	0x1074	[12:9]	GPIO Set command. When GPIO is configured as output, setting this register to “1” will write “1” to GPIO port. Use GPIO Clear command to clear.
GPIO	0x1076	[12:9]	GPIO Clear command. When GPIO is configured as output, setting this register to “1” will write “0” to GPIO port. Use GPIO Set command to set.
GPIO	0x1078	[12:9]	GPIO direction control for each GPIO signals. 0 = Output 1 = Input After power-on, GPIO ports are in input mode.



## Electrical Specifications

### Absolute Maximum Rating

**Caution** Table 21 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 21: Maximum Rating**

Symbol	Parameter	Rating		Unit
		Min	Max	
VDD_MAX	Core digital voltage	−0.3	2.4	V
VDD_IO_MAX	I/O digital voltage	−0.3	4.0	V
VAA_MAX	Analog voltage	−0.3	4.0	V
VAA_PIX_MAX	Pixel supply voltage	−0.3	4.0	V
VDD_PLL_MAX	PLL supply voltage	−0.3	4.0	V
VIH_MAX	Input HIGH voltage	−0.3	VDD_IO + 0.3	V
VIL_MAX	Input LOW voltage	−0.3	−	V
T_OP	Operating temperature (measured at junction)	−30	75	°C
T_ST	Storage temperature	−40	85	°C

### I/O Parameters

**Table 22: I/O Parameters**

$f_{EXTCLK} = 6\text{--}54\text{ MHz}$ ,  $V_{DD} = V_{DDIO\_TX} = V_{DD\_IO} = 1.8\text{V}$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8\text{V}$ ;  $T_j = 25^\circ\text{C}$

Symbol	Parameter	Min	Typ	Max	Unit	Note
VIH	Input logic HIGH threshold	$V_{DD\_IO} - 0.3$	−	$V_{DD\_IO} + 0.3$	V	
VIL	Input logic LOW threshold	−0.1	−	0.6	V	$V_{DD\_IO} = 2.8\text{V}$
		−0.1	−	0.3	V	$V_{DD\_IO} = 1.8\text{V}$
VOH	Output logic HIGH threshold	$V_{DD\_IO} - 0.3$	−	−	V	At specified at IOH
VOL	Output logic LOW threshold	−	−	0.3	V	−
CIN	Input pins - (SCLK, SDATA) capacitance	−	3.5	3.9	pF	−
CLOAD	Output pins - (SDATA) load capacitance	−	−	30	pF	−
PUP_SDATA	SDATA pull-up resistor	−	1.5	−	k $\Omega$	−

Note: VIH and VIL specifications apply to the over- and undershoot (ringing) present in the MCLK.



## MIPI Interface AC and DC Electrical Specifications

### Introduction

The MIPI interface for the MT9D115 image sensor was designed to meet the MIPI Alliance Specification for D-PHY Version 1.0. Please refer to this document for details on the MIPI specification and usage.

### Electrical Specifications

**Table 23: MIPI High-Speed Transmitter DC Characteristics**

f<sub>EXTCLK</sub> = 48 MHz; V<sub>DD</sub> = V<sub>DD\_TX</sub> = V<sub>DD\_IO</sub> = 1.8V; V<sub>AA</sub> = V<sub>AA\_PIX</sub> = V<sub>DD\_PLL</sub> = 2.8V; Ambient temperature

Parameter	Description	Min	Nom	Max	Units
V <sub>CMTX</sub>	HS transmit static common mode voltage	150	200	250	mV
ΔV <sub>CMTX</sub> (1,0)	V <sub>CMTX</sub> mismatch when output is Differential-1 or Differential-0			16	mV
V <sub>OD</sub>	HS transmit differential voltage	140	200	270	mV
ΔV <sub>OD</sub>	V <sub>OD</sub> mismatch when output is Differential-1 or Differential-0		14	mV	
V <sub>OHHS</sub>	HS output high voltage			360	mV
Z <sub>OS</sub>	Single-ended output impedance	39	50	64	Ohm
ΔZ <sub>OS</sub>	Single-ended output impedance mismatch			10	%

**Table 24: MIPI High-Speed Transmitter AC Characteristics**

f<sub>EXTCLK</sub> = 48 MHz; V<sub>DD</sub> = V<sub>DD\_TX</sub> = V<sub>DD\_IO</sub> = 1.8V; V<sub>AA</sub> = V<sub>AA\_PIX</sub> = V<sub>DD\_PLL</sub> = 2.8V; Ambient temperature

Parameter	Description	Min	Nom	Max	Units
t <sub>R</sub> and t <sub>F</sub>	20%-80% rise time and fall time	150			pS

**Table 25: MIPI Low-Power Transmitter DC Characteristics**

f<sub>EXTCLK</sub> = 48 MHz; V<sub>DD</sub> = V<sub>DD\_TX</sub> = V<sub>DD\_IO</sub> = 1.8V; V<sub>AA</sub> = V<sub>AA\_PIX</sub> = V<sub>DD\_PLL</sub> = 2.8V; Ambient temperature

Parameter	Description	Min	Nom	Max	Units
V <sub>OH</sub>	Thevenin output high level	1.1	1.2	1.4	V
V <sub>OL</sub>	Thevenin output low level	-50		50	mV
Z <sub>OLP</sub>	Output impedance of LP transmitter	110			Ohm

**Table 26: MIPI Low-Power Transmitter AC Characteristics**

f<sub>EXTCLK</sub> = 48 MHz; V<sub>DD</sub> = V<sub>DD\_TX</sub> = V<sub>DD\_IO</sub> = 1.8V; V<sub>AA</sub> = V<sub>AA\_PIX</sub> = V<sub>DD\_PLL</sub> = 2.8V; Ambient temperature

Parameter	Description	Min	Nom	Max	Units
TRLP/TFLP	15%-85% rise time and fall time			25	nS
TREOT	30%-85% rise time and fall time			35	nS
ΔV/ΔtSR	Slew rate @ C <sub>LOAD</sub> = 70pF	83		122	mV/nS
C <sub>LOAD</sub>	Load capacitance	0		70	pF



## AC Electricals

Figure 34: Output Interface Timing Waveforms

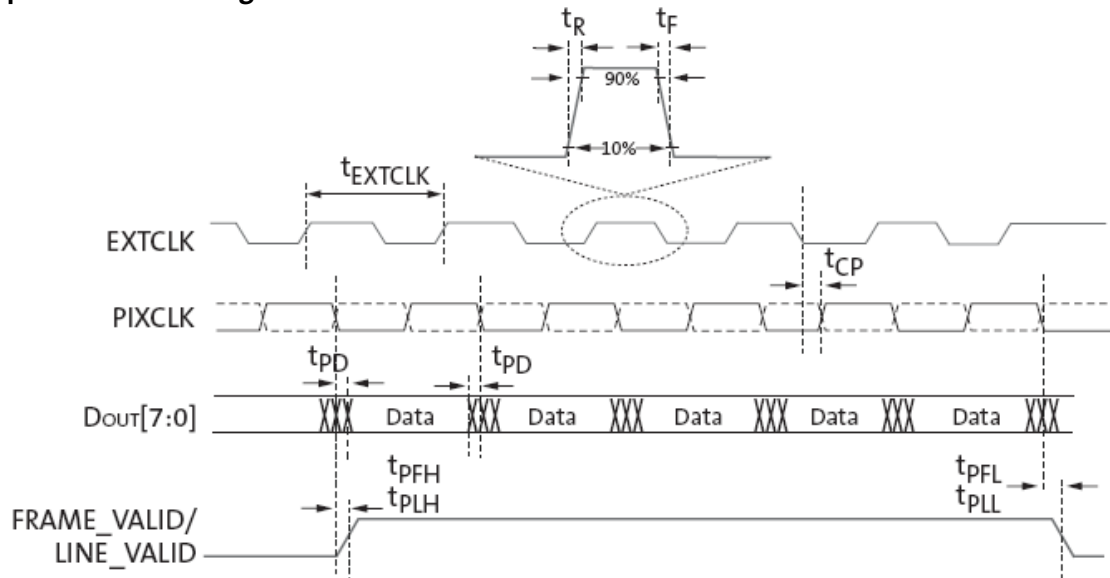


Table 27: AC Electricals

 $f_{EXTCLK} = 6\text{--}54\text{ MHz}$ ,  $V_{DD} = V_{DDIO\_TX} = V_{DD\_IO} = 1.8\text{V}$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8\text{V}$ ;  $T_j = 25^\circ$ 

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	Note
$f_{EXTCLK}$	External input clock frequency		6	-	54	MHz	2
$t_r$	External input clock rise time	From 10% to 90% of $V_{p-p}$	-	2	5	ns	1
$t_f$	External input clock fall time	From 10% to 90% of $V_{p-p}$	-	2	5	ns	1
$DCEXTCLK$	External input clock duty cycle		40	50	60	%	
$t_{JITTER}$	External input clock jitter	Peak-to-peak	-	-	500	ps	
$t_{CP}$	EXTCLK to PIXCLK propagation delay		5	-	45	ns	3
$f_{PIXCLK}$	Pixel clock frequency		6	-	85	MHz	
$t_{RPIXCLK}$	Pixel clock rise time	Load = 15pF	-	2	5	ns	
$t_{FPIXCLK}$	Pixel clock fall time	Load = 15pF	-	2	5	ns	
$t_{PD}$	Pixel clock to data valid		-	-	$0.6 \cdot PIXCLK$	ns	4
$t_{PFH}$	Pixel clock to frame valid high		-	-	$0.6 \cdot PIXCLK$	ns	
$t_{PLH}$	Pixel clock to frame valid low		-	-	$0.6 \cdot PIXCLK$	ns	
$t_{PFL}$	Pixel clock to line valid high		-	-	$0.6 \cdot PIXCLK$	ns	
$t_{PLL}$	Pixel clock to line valid low		-	-	$0.6 \cdot PIXCLK$	ns	
PIXCLK pin slew rate	Programmable slew = 7	$V_{DD\_IO} = 2.8\text{V}$ , Load = 45pF	-	1.2	-	V/ns	5
		$V_{DD\_IO} = 1.8\text{V}$ , Load = 45pF	-	0.6	-	V/ns	
	Programmable slew = 4	$V_{DD\_IO} = 2.8\text{V}$ , Load = 45pF	-	1	-	V/ns	
		$V_{DD\_IO} = 1.8\text{V}$ , Load = 45pF	-	0.5	-	V/ns	
	Programmable slew = 0	$V_{DD\_IO} = 2.8\text{V}$ , Load = 45pF	-	0.3	-	V/ns	
		$V_{DD\_IO} = 1.8\text{V}$ , Load = 45pF	-	0.15	-	V/ns	



**Table 27: AC Electricals**
 $f_{EXTCLK} = 6-54 \text{ MHz}$ ,  $V_{DD} = V_{DDIO\_TX} = V_{DD\_IO} = 1.8\text{V}$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8\text{V}$ ;  $T_j = 25^\circ$ 

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	Note
Output pin slew rate	Programmable slew = 7	$V_{DD\_IO} = 2.8\text{V}$ , $C_{load} = 45\text{pF}$	-	1.6	-	V/ns	5
		$V_{DD\_IO} = 1.8\text{V}$ , $C_{load} = 45\text{pF}$	-	0.8	-	V/ns	
	Programmable slew = 4	$V_{DD\_IO} = 2.8\text{V}$ , $C_{load} = 45\text{pF}$	-	1.25	-	V/ns	
		$V_{DD\_IO} = 1.8\text{V}$ , $C_{load} = 45\text{pF}$	-	0.55	-	V/ns	
	Programmable slew = 0	$V_{DD\_IO} = 2.8\text{V}$ , $C_{load} = 45\text{pF}$	-	0.3	-	V/ns	
		$V_{DD\_IO} = 1.8\text{V}$ , $C_{load} = 45\text{pF}$	-	0.15	-	V/ns	

- Notes:
1. Measured when the PLL is off. Specification not applicable when PLL is on, but input HIGH/LOW voltage should be within specification.
  2. VIH and VIL specifications apply to the over- and undershoot (ringing) present in the MCLK.
  3. Measurement done with PLL off.
  4. Valid for  $C_{load} < 20\text{pF}$  on PIXCLK, DOUT[9:0], LINE\_VALID, and FRAME\_VALID pads. Loads must be matched as closely as possible.
  5. PLL is off and EXTCLK is 24MHz.

**Table 28: DC Electricals**

Setup Conditions:  $f_{EXTCLK} = 6-54\text{MHz}$ ,  $V_{DD} = V_{DD\_IO\_TX} = V_{DD\_IO} = 1.8\text{V}$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8\text{V}$ ;  $T_j = 25^\circ\text{C}$ , unless stated otherwise

Symbol	Parameter	Condition		Min	Typ	Max	Unit	Notes
VDD	Digital core supply voltage			1.7	1.8	1.95	V	
VDD_PLL	PLL supply voltage			2.5	2.8	3.1	V	
VAA	Analog supply voltage			2.5	2.8	3.1	V	
VAA_PIX	Pixel supply voltage			2.5	2.8	3.1	V	
VDD_IO	Digital IO supply voltage	For VDD_IO = 1.8V		1.7	1.8	1.95	V	
		For VDD_IO = 2.8V		2.5	2.8	3.1	V	
VDDIO_TX	MIPI supply voltage			1.7	1.8	1.95	V	
VPP	OTPM supply voltage			-	8	-	V	
IDD_IO	Digital IO supply current	Context A	VDD_IO = 1.8V	-	10	-	mA	1
			VDD_IO = 2.8V	-	15	-	mA	
		Context B	VDD_IO = 1.8V	-	12	-	mA	
			VDD_IO = 2.8V	-	20	-	mA	
IDD_PLL	PLL supply current	PLL is OFF		-	N/A	-	mA	
		PLL is ON		-	13	18	mA	2
IDD	Digital core supply current	Operating in Parallel mode	Context A	-	15	30	mA	
IAA	Analog supply current			-	40	50	mA	
IAA_PIX	Pixel supply current			-	1.5	3	mA	
IDDIO_TX	MIPI supply current			-	N/A	-	mA	3
IDD	Digital core supply current		Context B	-	25	52	mA	
IAA	Analog supply current			-	40	52	mA	
IAA_PIX	Pixel supply current			-	0.8	3	mA	
IDDIO_TX	MIPI supply current			-	N/A	-	mA	3

**Table 28: DC Electricals (continued)**

Setup Conditions:  $f_{EXTCLK} = 6-54\text{MHz}$ ,  $V_{DD} = V_{DD\_IO\_TX} = V_{DD\_IO} = 1.8\text{V}$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8\text{V}$ ;  $T_j = 25^\circ\text{C}$ , unless stated otherwise

Symbol	Parameter	Condition		Min	Typ	Max	Unit	Notes
IDD	Digital core supply current	Operating in Serial mode	Context A	-	23	50	mA	4
IAA	Analog supply current			-	40	50	mA	
IAA_PIX	Pixel supply current			-	1.5	3	mA	
IDDIO_TX	MIPI supply current			-	5	3	mA	
IDD	Digital core supply current		Context B	-	35	70	mA	
IAA	Analog supply current			-	40	70	mA	
IAA_PIX	Pixel supply current			-	0.8	3	mA	
IDDIO_TX	MIPI supply current			-	8	10	mA	
I <sub>HardStandby</sub>	Total Standby Current	STANDBY pin asserted, R0x0028=1		-	-	20	μA	5
I <sub>HardStandby</sub>		STANDBY pin asserted, R0x0028=0		-	-	90	μA	7
I <sub>SoftStandby</sub>		R0x0018[0]=1		-	-	90	μA	6
I <sub>SoftStandby</sub>		R0x0018[0]=1, R0x0028=0		-	-	2	mA	7
I <sub>SoftStandby</sub>		R0x0018[0]=1, R0x0028=1		-	-	3	mA	7

- Notes:
1.  $V_{DD\_IO}$  current is dependent on the output data rate.
  2. PLL is on with 85 MHz output frequency setting.
  3.  $V_{DD\_IO\_TX}$  current is only applicable in serial output mode.
  4. PLL is on with 480 MHz VCO frequency settings.
  5. Either with EXTCLK running or EXTCLK stopped and EXCLK pin is either pulled up or pulled down. Measured at  $T_j = 70^\circ\text{C}$ .
  6. EXTCLK is stopped and EXCLK pin is either pulled up or pulled down.
  7. EXTCLK running at 27 MHz.



## Appendix A- VDD\_IO Current Addition

### Introduction

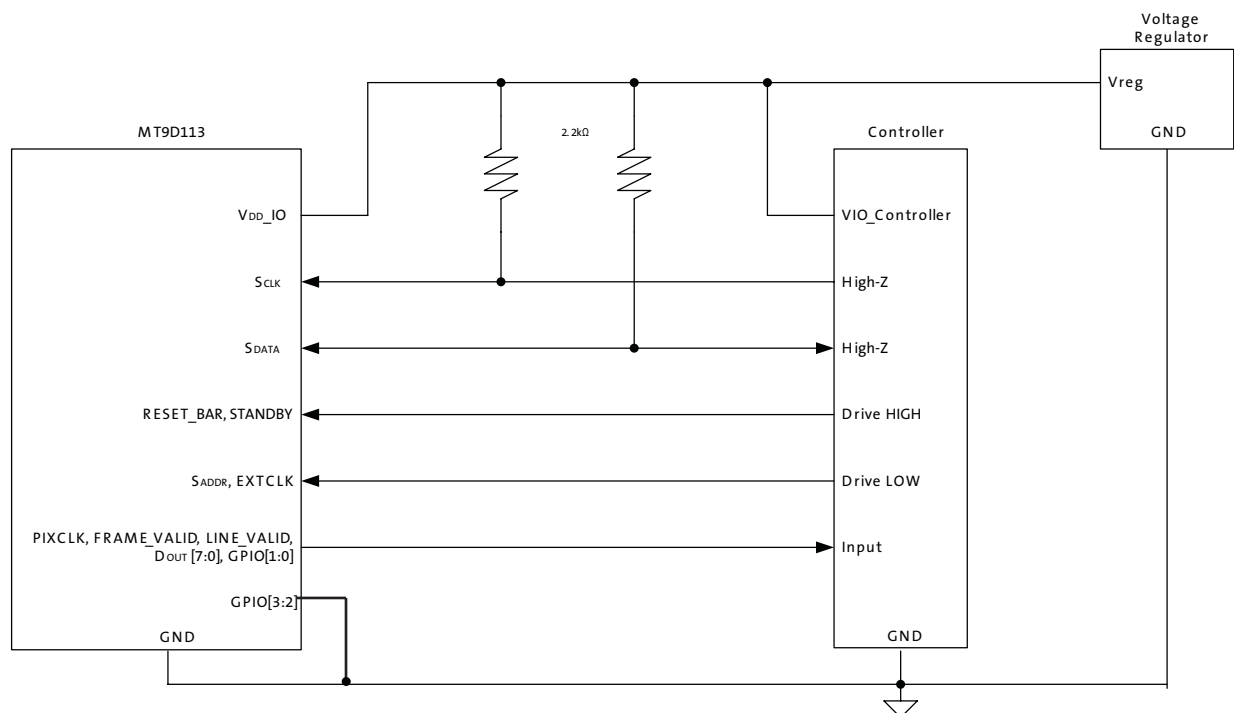
This appendix describes the system requirements to achieve lowest power consumption in the VDD\_IO domain during low power hardware standby mode in Aptina's MT9D115 CMOS digital image sensor.

### VDD\_IO Current

VDD\_IO current is used to measure the power consumption of the IO pad ring of our sensor. Therefore, it is extremely system-dependent and greatly affected by the external conditions as current can flow out of the chip through the IO ring and into the system. In this document, we outline requirements at the system level to achieve minimum power consumption during low power hardware standby mode.

To achieve the lowest VDD\_IO current, it is critical to match VDD\_IO level in the sensor and the controller. Otherwise, there can be elevated current drawn on the VDD\_IO due to mismatch in termination voltage level on the sensor pins. In Aptina system design, the IO voltage between the sensor and the controller is matched by using the same voltage regulator, as shown in Figure 35 on page 59. This is recommended for both the system design and setup to measure the VDD\_IO power consumption.

**Figure 35: Recommended System and Test Setup for Minimum VDD\_IO Power Consumption**





## IO Pin States

In addition to matching VDD\_IO levels between the controller chip and the sensor, the control signals to the input pins of the sensor must be at a specified level to achieve minimal VDD\_IO current draw. Table 29 shows the pin states recommended to achieve minimum VDD\_IO current during hardware standby mode.

It is always a good practice to measure the pin voltage during hard standby and try to match the recommended pin state.

**Table 29: Status of Signals During Standby State**

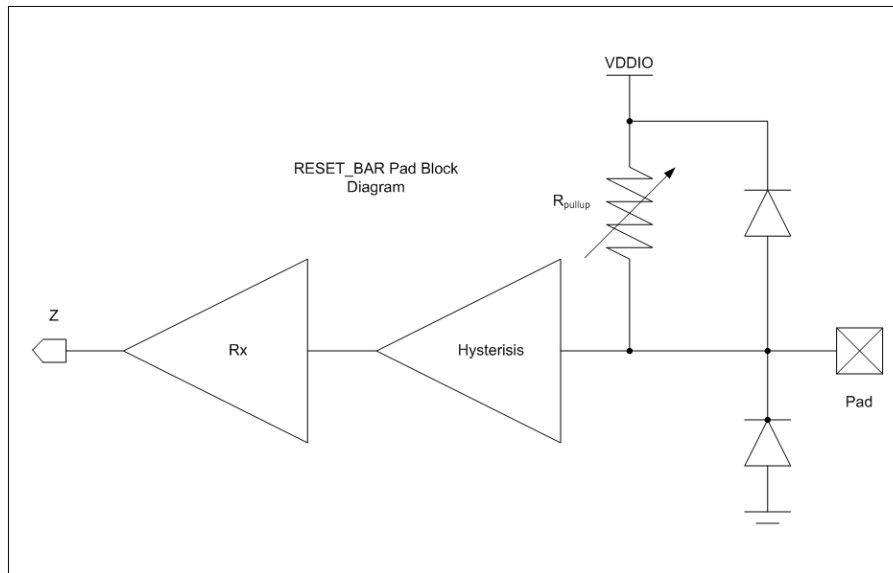
Signal	State on Sensor Pin	Aptina Test System	Note
DOUT[7:0]	High-Z by default (configurable through OE_BAR or two-wire serial interface register)	No control	4
PIXCLK	High-Z by default (configurable)		
LINE_VALID	High-Z by default (configurable)		
FRAME_VALID	High-Z by default (configurable)		
GPIO[3:0]	Depending on how the system uses them as DOUT_LSB1/DOUT_LSB0/FLASH/OE_BAR	Pulled to GND	8, 9
DOUT_N	0	Float	7
DOUT_P	0		
CLK_N	0		
CLK_P	0		
SADDR	Input	Pulled to GND	1, 6
EXTCLK	Input		
SDATA	Input	High Z	3
SCLK	Input		
RESET_N	Input	Pulled to VDD_IO Level	2, 5
STANDBY	Input		

- Notes:
1. VIL specification for input signal applies. Refer to Table 20 in the MT9D115 Data Sheet.
  2. VIH specification for input signal applies. Refer to Table 20 in the MT9D115 Data Sheet.
  3. These pins are not directly connected to VDD\_IO supply but through a voltage follower to the controller.
  4. The pins on the controller connected to these sensor pins are input pins.
  5. These pins are not directly connected to VDD\_IO supply but through the controller.
  6. These pins are not directly connected to GND but through the control signal on the controller.
  7. These pins are floating in parallel mode operation.
  8. Tie all the unused GPIO pins to DGND or VDD\_IO level in the module.
  9. If GPIO3 is connected to GPIO pin on the controller, program GPIO3 as an input before low power hard standby mode and drive GPIO3 externally from the controller to DGND level.

## RESET\_BAR with Internal Pull-UP

The RESET\_BAR pin has an internal pull-up device to VDD\_IO (see Figure 36).

**Figure 36: RESET\_BAR Pad Architecture**



Since  $R_{PULLUP}$  is active devices connected in triode state, the value of  $R_{PULLUP}$  is dependent on the difference between  $V_{DD\_IO}$  and  $V_{pad}$  Level (see Table 30).

**Table 30: Typical Mismatch Current in VDD\_IO Due to Mismatch in RESET\_BAR level and VDD\_IO Level**

Conditions:  $V_{DD} = V_{DD\_IO\_TX} = 1.8V$ ;  $V_{AA} = V_{AA\_PIX} = V_{DD\_PLL} = 2.8V$ ;  $T_j = 30^\circ C$ , EXTCLK is off and tied to GND level. All other pin states and levels are set according to Aptina E2E on pin states.

VDD_IO	Vpad	Min	Typ	Max	Unit
1.80	1.70	14	18	22	$\mu A$
	1.80	2	5	10	
	1.95	-13	-17	-21	
2.80	2.50	61	67	72	
	2.80	2	7	12	
	3.10	-54	-60	-66	

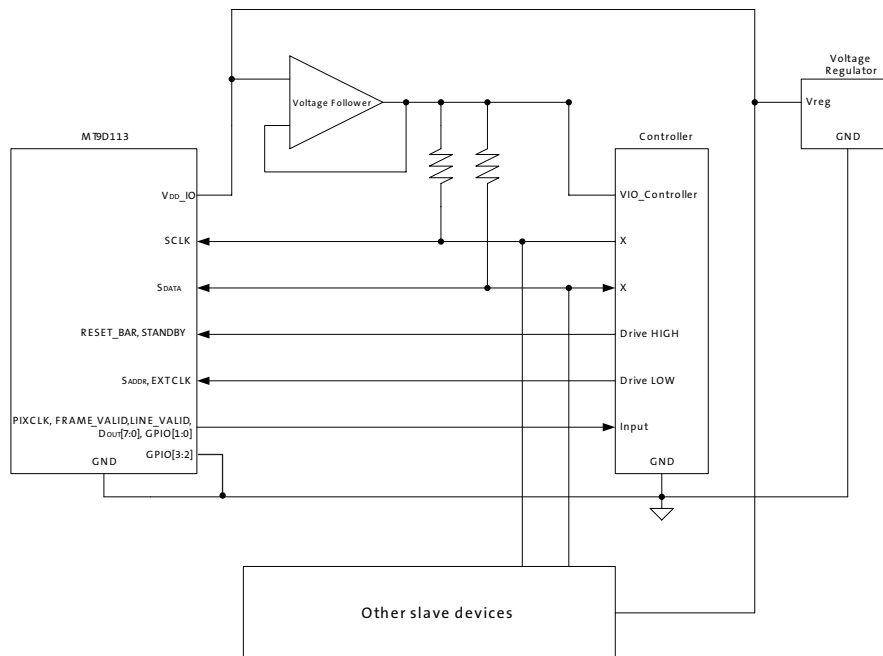
Note: The above data is for engineering purposes only. These represent typical values that can be expected.

## Recommended System and Test Setup with Multiple Serial Interface Slave Devices

The recommended system and test setup shown in Figure 1 is only valid for a system where MT9D115 is the only two-wire serial interface slave device connected to the controller chip. As the SCLK and SDATA are to be pulled high to the VDD\_IO level during the hardware standby to minimize the IO current consumption due to the data toggling in the two-wire serial interface, this design is not feasible if there is an additional slave device sharing the same two-wire serial interface on the controller chip.

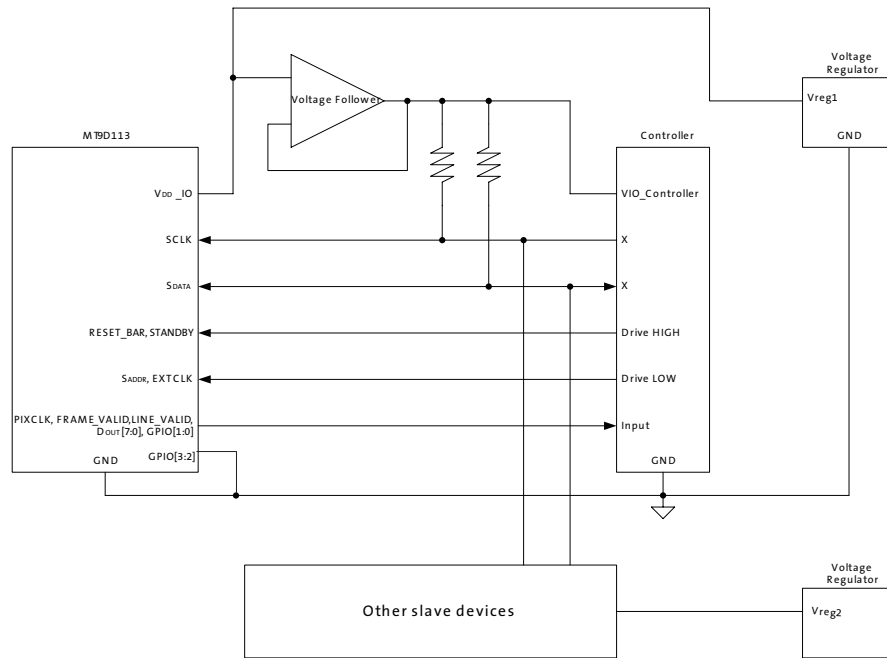
In such systems, it is recommended to have a voltage follower to isolate the pull-up resistors attached between the two-wire serial interface and VDD\_IO supply of MT9D115 as shown in Figure 37. Otherwise, the data toggling from the two-wire serial interface transactions to other could cause leakage from the VDD\_IO supply of MT9D115.

**Figure 37: Recommended System and Test Setup for Minimum VDD\_IO Power Consumption in the MT9D115 Sharing with Multiple Two-wire Serial Interface Devices**



In addition, in some system designs where the IO level of the additional devices sharing the same two-wire serial interface with MT9D115 is different from that of MT9D115, it is recommended that the IO voltage level between the controller and the MT9D115 be maintained to the same level by sourcing from the same voltage regular as shown in Figure 38 on page 63.

**Figure 38: Recommended System and Test Setup for Minimum VDD\_IO Power Consumption in the MT9D115 Sharing with Multiple Two-wire Serial Interface Devices at Different IO Levels**

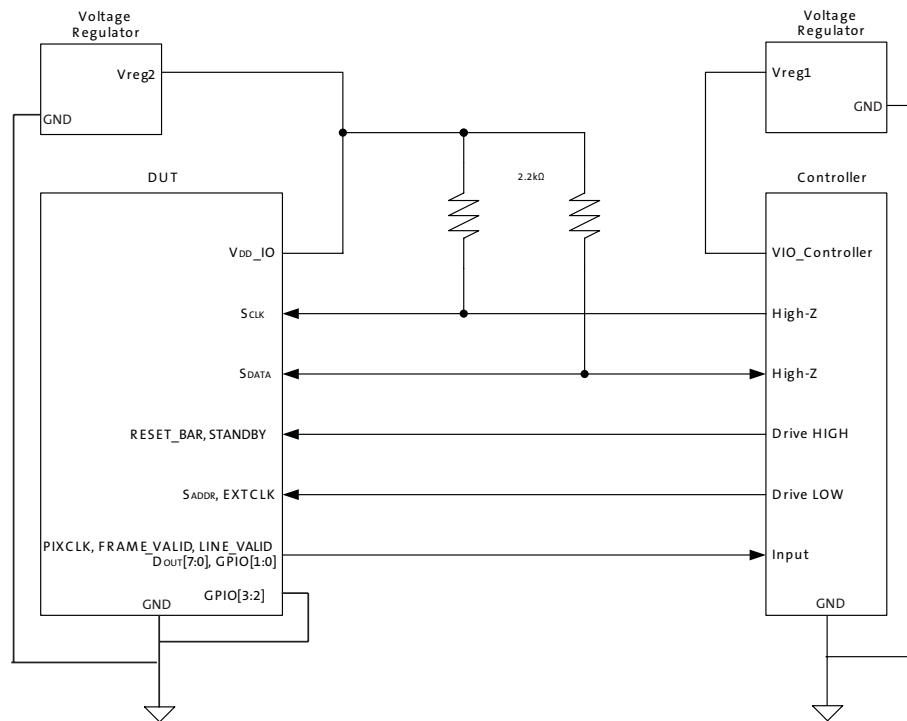


For a system which uses separate voltage regulators for MT9D115 sensor and the controller chip as shown in Figure 39, it is important to match the voltage levels between two regulators as close as possible. If the voltage levels are not matched, there can be additional current consumption in the VDD\_IO domain connected to the sensor. This additional current is generated in the internal pull up resistor present in RESET\_BAR pad and external resistors used for two-wire serial interface.

The current leakage from the voltage level mismatch in RESET\_BAR pad can be characterized as follows:

$$IDD\_IO_{mismatch} = (Vreg2 - Vreg1) / R_{pullupRESET\_BAR@VDD\_IO}$$

The internal pull up resistance is dependent on the VDD\_IO level. Typical levels of mismatch current at different VDD\_IO levels can be found in Table 30 on page 61.


**Figure 39: System Setup with Independent Voltage Sources for MT9D115 and Controller Chip**




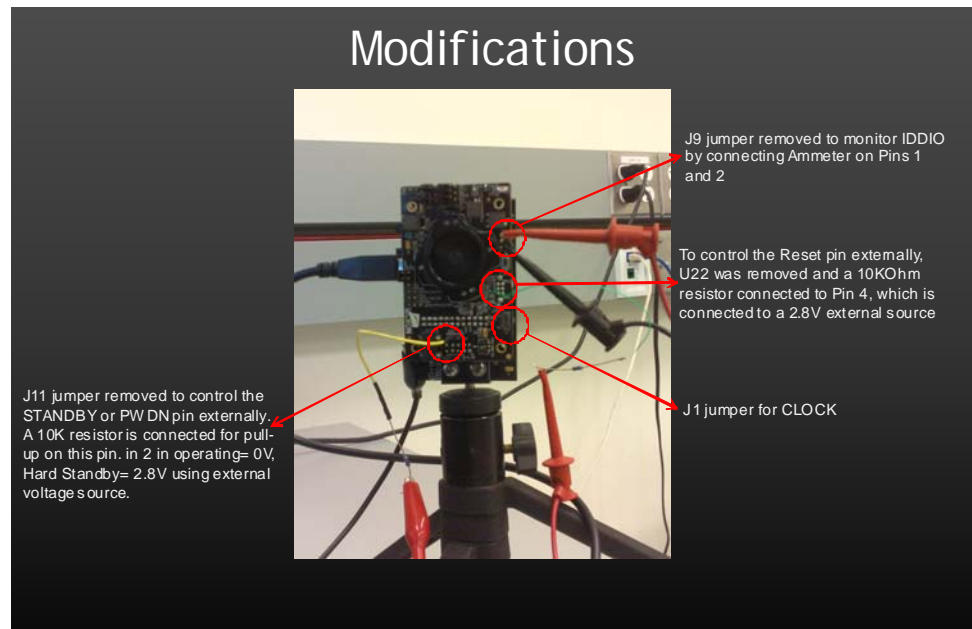
## Test Sequence for Measuring VDD\_IO Current at Hard Standby Mode

Follow the procedure defined within the sensor data sheet. The following test sequence takes the MT9D115 as a reference. Contact your local Aptina Applications Engineer for any specific product.

To measure VDD\_IO current at hard standby mode:

1. Supply PWRDN (also known as STANDBY) pin (J11 pin2) with 0 volt externally.
2. Supply the RESET\_BAR pin (U22 pin4) with VDD\_IO volt externally.
3. Connect an ammeter between IDD\_IO (J9) and DGND.
4. Use jumpers J5, J17, and J20 to make sure GPIO pins are connected to either GND or VDD\_IO.  
At this point, the part is in operating mode and it is possible to load an .INI file to get an image with DevWare.
5. Supply the PWRDN pin with VDD\_IO volt externally and wait for one frame + 50 clock. At this stage, the part is in hard standby.
6. By removing J1, the EXTCLK is disconnected to the part and then connect EXTCLK pin (J1 pin2) to either DGND or VDD\_IO.
7. Now the part is in hard standby with EXTCLK off. The user can read off the VDD\_IO current from the ammeter at step 3 above.

**Figure 40: Modifications to the Demo2 Sensor Head Board for VDD\_IO Current Measurement**



## Conclusion

It is important from the system perspective to ensure IO levels between MT9D115 and the controller chip at the same potential level to achieve minimum IO power consumption in MT9D115. A system configuration to achieve this is proposed in this document. In addition, suggestions on how to approach system designs with multiple two-wire serial interface devices and multiple IO level devices are provided.



## Revision History

<b>Rev. C</b> .....	3/14/12
<ul style="list-style-type: none"> <li>• Updated notes for Table 3, “Signal Description and Direction,” on page 9</li> <li>• Updated Figure 6: “Two-Wire Serial Control Bus Timing,” on page 13</li> <li>• Updated “Parallel and MIPI Output” on page 32</li> <li>• Added Figure 31: “Recommended Power Down Sequence,” on page 46</li> <li>• Added Table 16, “Power Down Signal Timing,” on page 46</li> <li>• Updated Table 22, “I/O Parameters,” on page 54</li> <li>• Updated Table 25, “MIPI Low-Power Transmitter DC Characteristics,” on page 55</li> <li>• Updated Table 28, “DC Electricals,” on page 57</li> </ul>	
<b>Rev. B</b> .....	7/7/11
<ul style="list-style-type: none"> <li>• Updated to Production</li> <li>• Updated “Features” on page 1</li> <li>• Updated Table 1, “Key Performance Parameters,” on page 1</li> <li>• Updated Table 2, “Available Part Numbers,” on page 1</li> <li>• Deleted “Overview”</li> <li>• Added “Functional Description” on page 6</li> <li>• Added “Architecture Overview” on page 6</li> <li>• Updated Figure 11: “Six Rows in Normal and Row Mirror Readout Modes,” on page 19</li> </ul>	
<b>Rev. A</b> .....	3/16/10
<ul style="list-style-type: none"> <li>• Initial release.</li> </ul>	