1/2.3-Inch 10Mp CMOS Digital Image Sensor

MT9J001 and MT9J003

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

MT9J001 and MT9J003 Developer Guide

Table of Contents

Introduction 5
Identifying the MT9J001 and MT9J003 Using Register Settings 5
Sensor Initialization 5
Modes Recommended in the Register Addendum 6
Programming the Sensor PLL 7
PLL Programming for HiSPi 7
PLL Programming for the Parallel Interface 9
PLL Configuration for Low Power Mode 10
Image Sensor Array 12
Bayer Resampler 13
Recommended Register Settings for Sensor Initialization 14
Programming Summary for the Sensor on Start-up 14
Recommended Default Register Changes 14
Recommended Gain Table 15
Sensor Trimming 16
Programming Decision Tree to Determine Trimming Version 16
Configuration to Use Parallel or HiSPi Interface 18
Configuring the HiSPi Interface 18
Programming Sensor Data Pedestal and Disabling Embedded Data 18
Enabling Sensor Streaming 19
Recommended Register Sequence for Mode Changes 19
Using grouped_parameter_hold 19
Using frame_restart with grouped_parameter_hold 20
Appendix A: PLL Programming Tables 21
Appendix B: Sensor Mode HiSPi Programming Table 23
Appendix C: Sensor Mode Parallel Programming Table 24
Appendix D: High Frame Rate Modes 25
Appendix E: Subsampling Using Summing or Binning 26
Appendix F: Sensor Programming Examples 27
Syntax 27
Sensor Programming Example 1: Sensor Initialization 27
Sensor Programming Example 2: Programming a New Sensor Mode 29
Revision History 31

List of Figures

Figure 1: Start-up Sequence from Power-on to Streaming Images 5
Figure 2: HiSPi PLL – HiSPi Data and Clock Speed. 7
Figure 3: HiSPi PLL – Array Clock Calculation. 8
Figure 4: PLL Configuration for the Parallel Interface 9
Figure 5: PLL Configuration for the Parallel Interface Using Low Power Mode 10
Figure 6: PLL Configuration for the HiSPi Interface Using Low Power Mode 11
Figure 7: Standard Pixel Array and Configuration 12
Figure 8: Bayer Resampling. 13
Figure 9: Results of Resampling 13
Figure 10: MT9J001 and MT9J003 Decision Tree to Determine Trimming Version 16
Figure 11: Frame Timing Using the "grouped_parameter_hold" Feature 19

List of Tables

Table 1: Recommended Imaging Readout Modes 6
Table 2: Recommended Default Register Changes 14
Table 3: Recommended Gain Stages 15
Table 4: Recommended Sensor Trimming for the MT9J001 and MT9J003 Sensors 17
Table 5: Interface-Specific Configuration 18
Table 6: HiSPi Programming Registers 18
Table 7: Instructions to Program the Data Pedestal and to Disable the Embedded Data 18
Table 8: Enable Streaming Using the Parallel Interface 19
Table 9: HiSPi PLL Register Setup Examples 21
Table 10: Parallel PLL Register Setup Examples 21
Table 11: Sensor Mode HiSPi Programming Table 23
Table 12: Sensor Mode Parallel Programming Table 24
Table 13: Sensor Mode High Frame Rate Programming Table 25
Table 14: Description of "XY-SUM", "X-BIN Y-SUM", and "X-BIN and Y-BIN" 26
Table 15: Register Settings to Configure Video Modes to Use Binning 26

Introduction

The MT9J001 and the MT9J003 are 1/2.3-inch format CMOS active-pixel digital image sensors that have a pixel array of 3856H x 2764V. They incorporate sophisticated on-chip camera functions such as windowing, mirroring, binning and skip modes, and snapshot mode. They are programmable through a simple two-wire serial interface and have very low power consumption.

This Developer Guide is a reference tool for engineers developing applications, also providing information on working with chip registers. Use the data sheet along with this guide as a reference for specific register and usage information.

This document is relevant for two revisions of the 1/2.3-inch 10Mp CMOS image sensor that will be in mass production, Revision 2 and Revision 3.1. A Revision 3 of the sensor was created at Aptina but not released to vendors. The Revision 2 sensor is sold as part number MT9J001 and the Revision 3.1 will be sold as MT9J003.

Identifying the MT9J001 and MT9J003 Using Register Settings

The MT9J001 (Rev2) and MT9J003 (Rev3.1) can be identified through the sensor register settings:

MT9J001:
- Register 0x02 = 0x20
- Register 0x31FE = 0x02

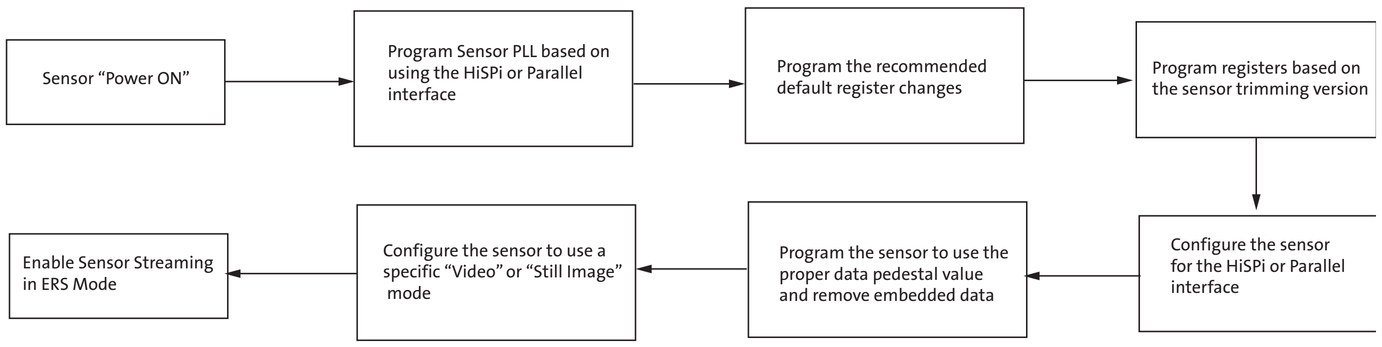
MT9J003: (Mass Production)
- Register 0x02 = 0x30
- Register 0x31FE = 0x03

MT9J003: (Mass Production)
- Register 0x02 = 0x20
- Register 0x31FE = 0x32

Sensor Initialization

The process from start-up to streaming images for the MT9J001 CMOS sensor includes numerous steps as shown in Figure 1:

Figure 1: Start-up Sequence from Power-on to Streaming Images


Sensor "Power ON"
Program Sensor PLL based on using the HiSPi or Parallel interface
Program the recommended default register changes
Program registers based on the sensor trimming version
Configure the sensor for the HiSPi or Parallel interface
Program the sensor to use the proper data pedestal value and remove embedded data
Configure the sensor to use a specific "Video" or "Still Image" mode
Enable Sensor Streaming in ERS Mode

Modes Recommended in the Register Addendum

The basic modes described in this developer guide are common full-resolution, video, and preview modes.The registers used to configure these modes are described in Appendix B and C.

Table 1: Recommended Imaging Readout Modes

Mode	Pixel Array Usage	Output Image Size	Subsampling	Frame Rate
10MP	3664 (V) x 2748 (H)	3664 (V) x 2748 (H)	None	HiSPi: 14.7 fps Parallel: 7.5 fps
1080p HDTV	3840 (V) x 2160 (H)	1920 (V) x 1080 (H)	2x2 Summing	HiSPi: 59.94 fps Parallel: 29.97 fps
720p + 10% EIS	1584 (V) x 2816 (H)	1408 (V) x 792 (H)	2x2 Summing	HiSPi: 59.94 fps
Monitor (Low Power Preview)	3664 (V) x 2748 (H)	916(V) x 687(H)	X – 2x skip + 2x sum Y – 2x skip + 2x sum	HiSPi or Parallel: 29.97 fps
VGA	3664 (V) x 2748 (H)	916 (V) x 687 (H)	X – 2x skip + 2x sum Y – 2x skip + 2x sum	HiSPi or Parallel: 29.97 fps
Full Array	3840 (V) x 2748 (H)	3840 (V) x 2748 (H)	None	HiSPi: 14.3 fps Parallel: 7.3 fps

Programming the Sensor PLL

The MT9J001 describes the sensor PLL in detail. This section will review how to program the sensor PLL for HiSPi and the Parallel interface.

PLL Programming for HiSPi

To program the sensor PLL for HiSP, do the following steps:

1. Program the PLL multiplier to supply the required data rate for the HiSPi serial interface.
2. Determine the pixel readout speed for the pixel array and the output clock to the sensor parallel or HiSPi interface.

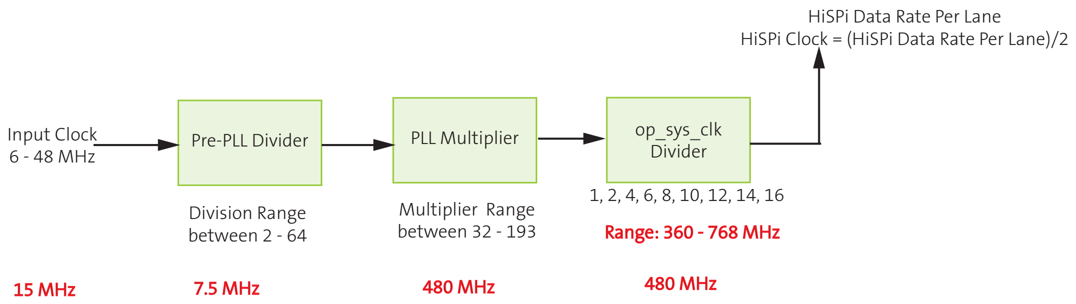
The HiSPi data rate is configured from by the PLL multiplier. The output of the PLL can be configured by the following equation:

$$\mathit{HiSPi\ Data\ Rate\ per\ Lane} = (\mathit{Input\ Clock} \times \mathit{PLL\ Multiplier}) / (\mathit{Pre\text{-}PLL\ Divider} \times \mathit{op\_sys\_clk\ divider})$$

The HiSPi clock speed is a double data-date (DDR) clock that is calculated as half of the HiSPi data rate per lane.

$$\mathit{HiSPi\ Clock\ Speed} = \mathit{HiSPi\ Data\ Rate\ per\ Lane} / 2$$

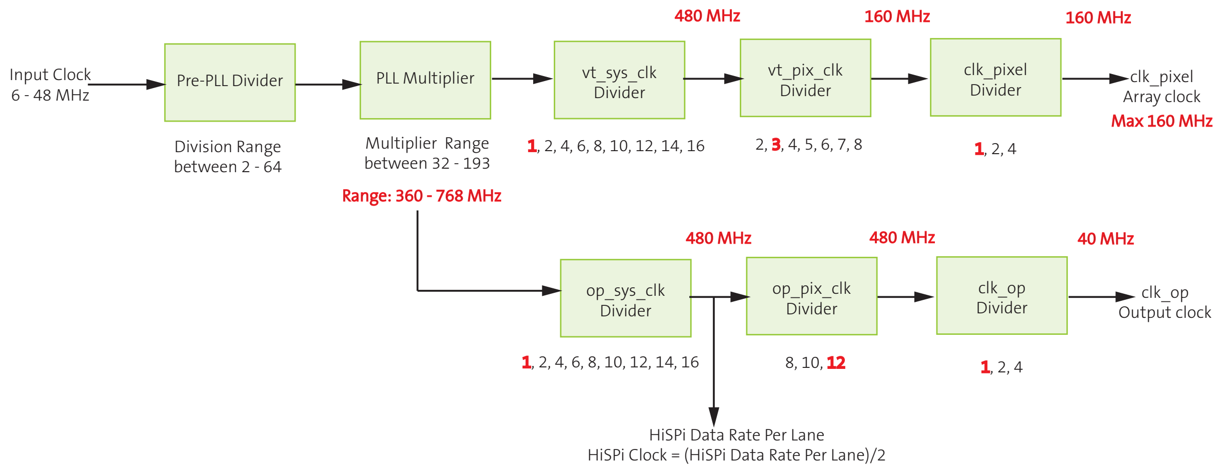
Figure 2: HiSPi PLL – HiSPi Data and Clock Speed



Notes:
1. The HiSPi clock is derived directly fro the PLL clock frequency.
   Input clock 8 Pre-PLL Divider * PLL Multiplier
   15 MHz * (1/2) * 64 = 480 MHz
   15 MHz * (1/2) * 48 = 360 MHz
   15 MHz * (1) * 48 * (1/2) = 720 MHz * (1/2) = 360 MHz
   12 MHz * (1) * 96* (1/2) = 568 MHz * (1/2) = 288 MHz
2. The clock seen by HiSpi is a bit clock.

The vt_pix_clk and op_pix_clk dividers are used to derive the pixel readout speeds for the sensor array and output interface. For the HiSPi interface, the vt_pix_clk divider is set to 1/3 while the op_pix_clk divider is set to 1/12.

Figure 3: HiSPi PLL – Array Clock Calculation



Notes:
1. The op_pix_clk divider is used to relate the output pixel clock to the bit-depth of the pixel:
   480 MHz * (1/1)*(1/12) = 40 MHz
   360 MHz * (1/1)*(1/12) = 30 MHz
   568 MHz * (1/2)*(1/12) = 288 MHz 8 (1/12) = 24 MHz
2. There are two requirements for the 4-lane HiSPi:
   2a. The output clock must be 1/4 of the array clock.
   2b. The value of op_pix_clk divider will set the bit-depth of the data output (for example,1/12 for12-bit).

The following can be determined from the diagram in Figure 3:

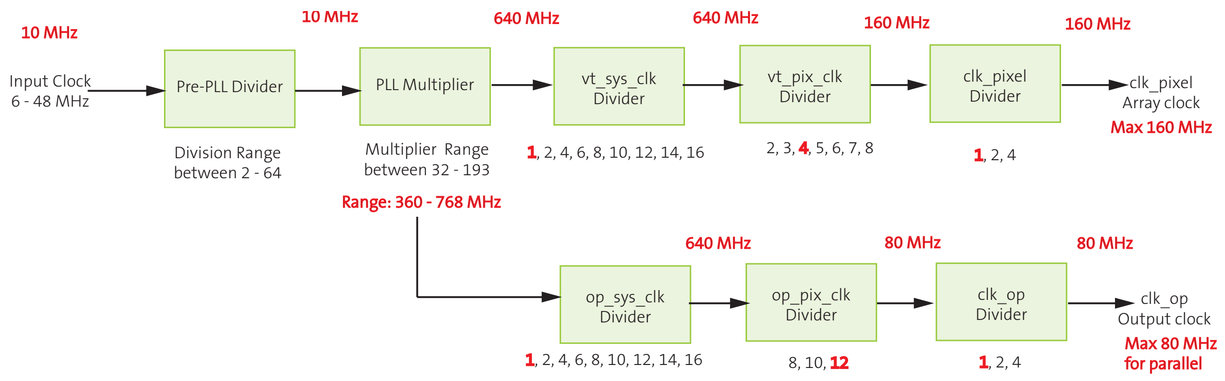
- The output of the vt_sys_clk divider at 480 MHz will set the HiSPi data rate per lane to 480 Mbps.
- The array clock is set to 160 MHz. This will configure the pixel read-out rate from the array to the analog signal chain to be 160Mp/s.
- The output clock is set to 40 MHz where each of the four HiSPi data lanes each output 40 Mp/s.

The combined output of 40 Mp/s from each of the four data lanes will match the pixel array readout of 160 Mp/s. The maximum speed for the array clock is 160 MHz.

PLL Programming for the Parallel Interface

The parallel implementation requires that the vt_pix_clk divider is set to 1/4 while the op_pix_clk divider is set to 1/8. This ensures that the output clock is set to half of the array clock.

Figure 4: PLL Configuration for the Parallel Interface



The following can be determined from the diagram in Figure 4:

- The array clock is set to 160 MHz where the pixel read-out rate will be 160 Mp/s.
- The output clock is set to 80 MHz where the parallel interface will have an output data rate for the interface of 80 Mp/s. The maximum pixel read-out rate for the parallel interface is 80 Mp/s.

The maximum pixel read-out rate of the parallel interface limits the maximum available frame-rate. An example of this limitation is the 1080p video mode that can achieve a frame-rate of 60fps using the HiSPi interface and 30fps using the parallel interface. Refer to table 1 for more examples.

It is recommended to enable the "Low Power Mode" when using the parallel interface with the sensor. This configuration will match the array clock to the parallel interface.

PLL Configuration for Low Power Mode

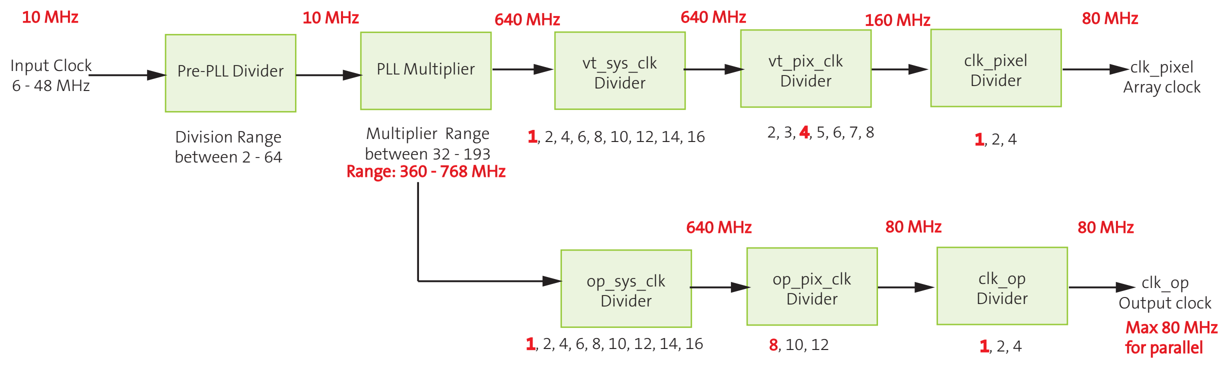
The sensor can implement the "Low Power Mode" where the pixel array is driven at half the speed as it would be in "Normal Power Mode". When enabling the "Low Power Mode" register bit 0x3040[9], the sensor will do the following:

- The configuration of the "Clk_pixel Divider" will automatically change from a value of "1" to a value of "2" or from a value of "2" to a value of "4". This will halve the array clock. The configuration of the rest of the PLL will not change as shown in Figure 5.
- The current-bias settings used in the analog circuitry will automatically be reduced to a value compatible with the reduced array clock.
- The "Low Power Mode" cannot be enabled if the "Clk_pixel Divider" is already configured to a value of "4".

The "Low Power Mode" will lower the analog power consumption of the sensor. Disabling the "Low Power Mode" to use the "Normal Power Mode" will undo the configuration changes of the clk_pixel divider and the current-bias settings within the analog circuitry.

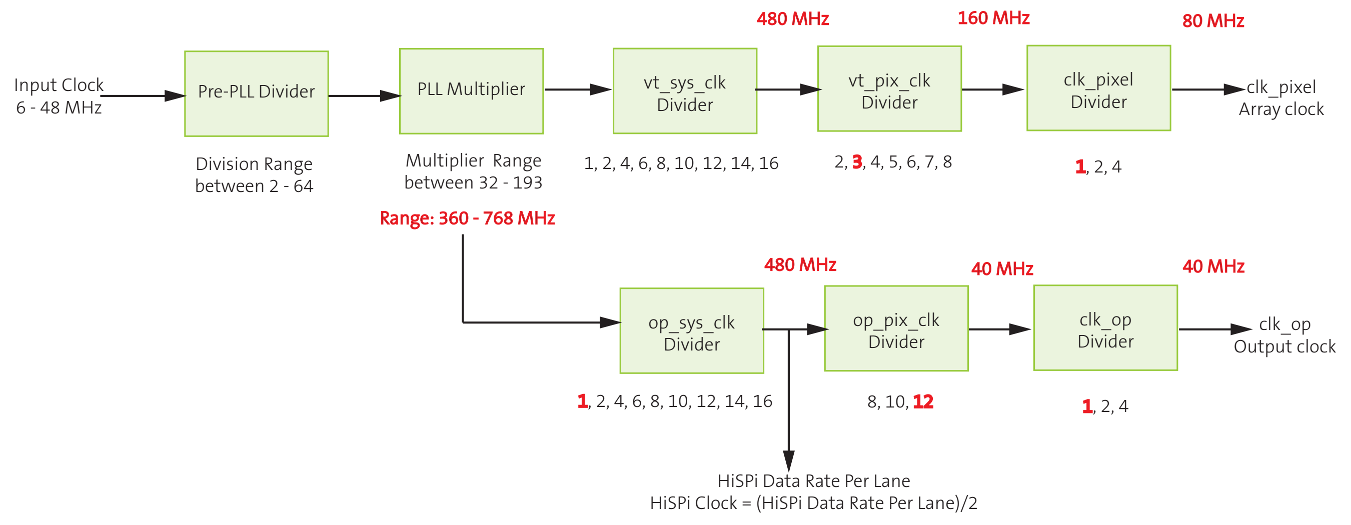
Implementing the "Low Power Mode" with the parallel interface will halve the array clock. This PLL configuration will match the array clock to the output clock of the parallel interface.

Figure 5: PLL Configuration for the Parallel Interface Using Low Power Mode



Configuring the sensor to use the "Low Power Mode" with the HiSPi interface will halve the array clock frequency but the HiSPi data rate per lane will remain the same. The transmitter and receiver will be required to operate at the same data rate per lane even though the array clock has been reduced in speed.

Figure 6: PLL Configuration for the HiSPi Interface Using Low Power Mode

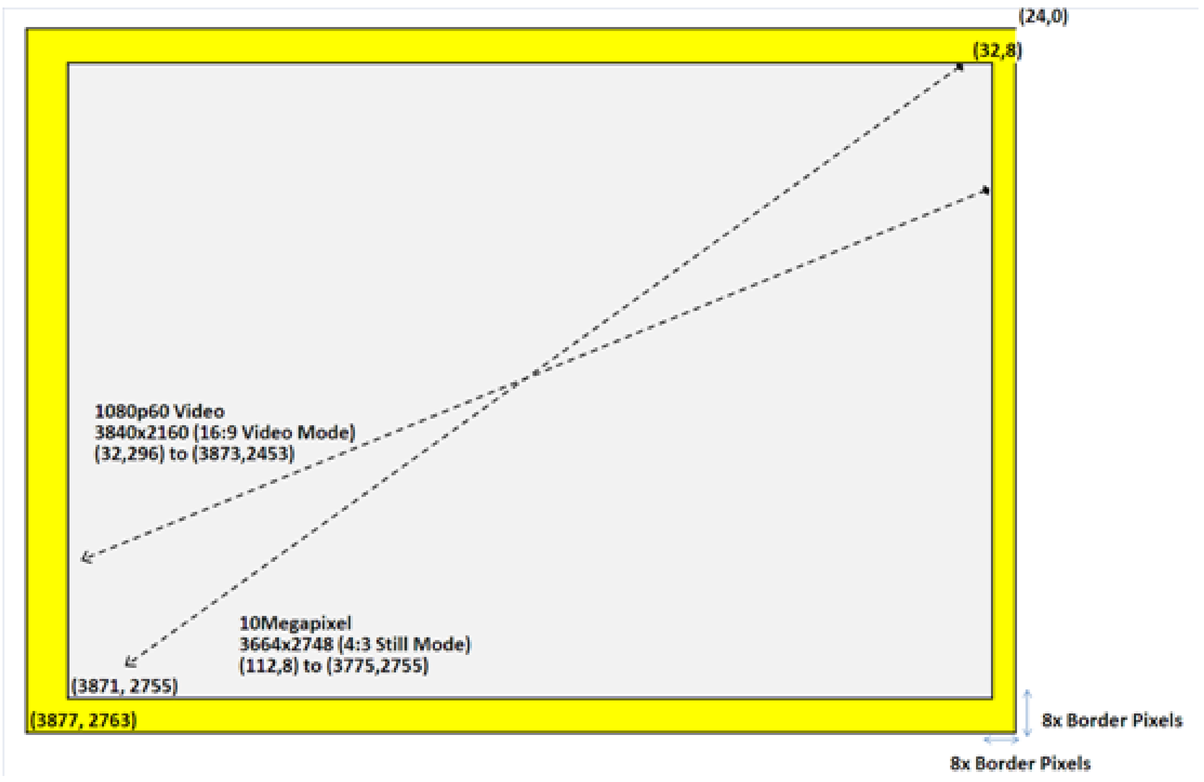


Notes:
1. The op_pix_clk divider is used to relate the output pixel clock to the bit-depth of the pixel:
   480 MHz * (1/1)*(1/12) = 40 MHz
   360 MHz * (1/1)*(1/12) = 30 MHz
   568 MHz * (1/2)*(1/12) = 288 MHz 8 (1/12) = 24 MHz
2. There are two requirements for the 4-lane HiSPi:
   2a. The output clock must be 1/4 of the array clock.
   2b. The value of op_pix_clk divider will set the bit-depth of the data output (for example,1/12 for12-bit).

Image Sensor Array

The sensor array has been designed to supply either "4:3" or "16:9" images. The typical array configurations are 3664 x 2748 to readout a 10Mp image and 3840 x 2160 combined with 2x2 summing or binning to readout a 1080p (1920 x 1080) image.

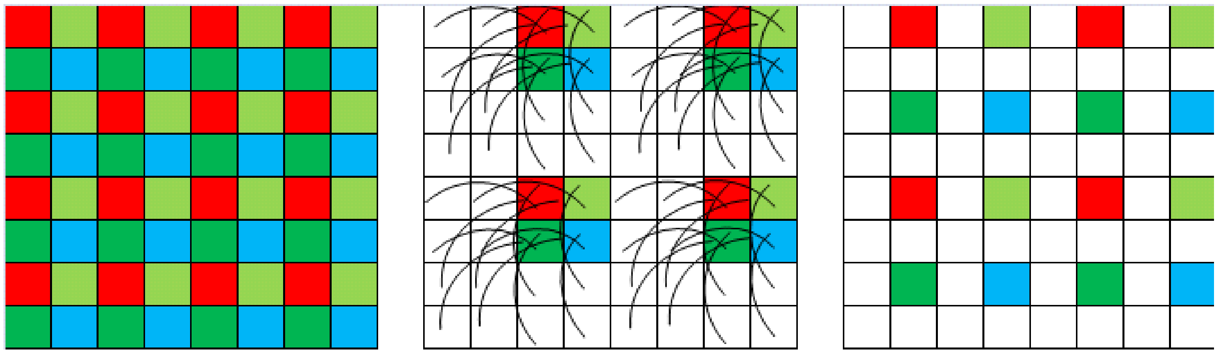
Figure 7: Standard Pixel Array and Configuration



Bayer Resampler

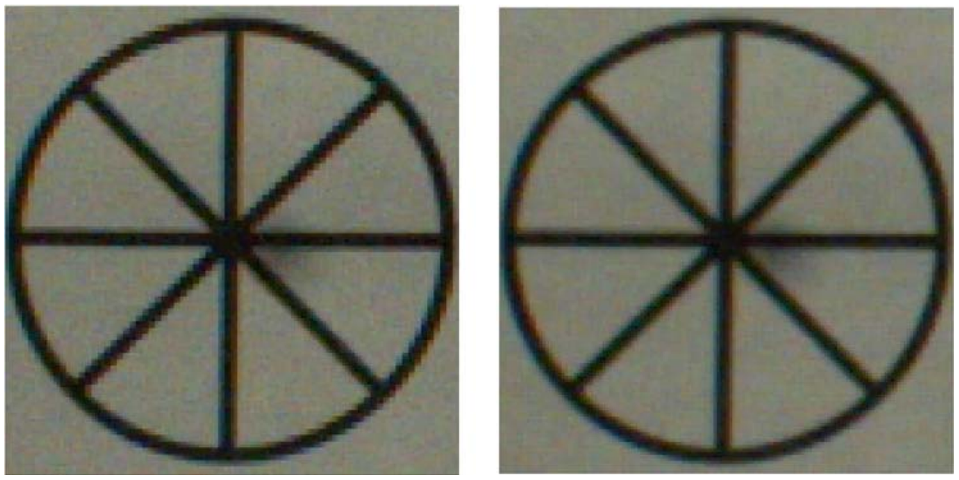
The imaging artifacts found from a 2x2 binning or summing will show image artifacts from aliasing. These can be corrected by resampling the sampled pixels in order to filter these artifacts. Figure 8 shows the pixel location resulting from 2x2 summing or binning located in the middle and the resulting pixel locations after the Bayer re-sampling function has been applied.

Figure 8: Bayer Resampling



The improvements from using the Bayer resampling feature can be seen in Figure 9 on page 13. In this example, image edges seen on a diagonal have smoother edges when the Bayer re-sampling feature is applied. This feature is only designed to be used with modes configured with 2x2 binning or summing. The feature will not remove aliasing artifacts that are caused skipping pixels.

Figure 9: Results of Resampling



To enable the Bayer resampling feature:

1. Set 0x0400 to 0x02 // Enable the on-chip scalar.
2. Set 0x306E to 0x90B0 // Configure the on-chip scalar to resample Bayer data.

To disable the Bayer resampling feature:

1. Set 0x0400 to 0x00 // Disable the on-chip scalar.
2. Set 0x306E to 0x9080 // Configure the on-chip scalar to resample Bayer data.

Recommended Register Settings for Sensor Initialization

The default register settings are determined by the sensor engineering team with the goal of optimizing the sensor for image quality and power consumption. These registers should be programmed to the sensor after initialization and before the sensor begins streaming.

Programming Summary for the Sensor on Start-up

The following order is recommended when programming the default register settings upon sensor initialization:

1. Configure the sensor PLL based on using the parallel or HiSPi interface. (Table 8 and 9)
2. Program recommended default register changes. (Table 2)
3. Program sensor registers that are dependent on the trimming version. (Table 3)
4. Configure the sensor for HiSPi or Parallel. (Table 4 and 5)
5. Configure the sensor data pedestal and disable embedded data. (Table 6)
6. Enable sensor streaming (Table 7 and 8)

Recommended Default Register Changes

The recommended default register changes are mentioned in Table 2. These registers should be programmed after the sensor is powered up and before sensor streaming is enabled.

Table 2: Recommended Default Register Changes

Register Address	Recommended Value	Notes
0x316C	0x0429	
0x3174	0x8000	
0x3E40	0xDC05	
0x3E42	0x6E22	
0x3E44	0xDC22	
0x3E46	0xFF00	Added to improve dark frame nonuniformity.
0x3ED4	0xF998	
0x3ED6	0x9789	
0x3EDE	0xE41A	Set to 0xE412 in low-power mode.
0x3EE0	0xA43F	
0x3EE2	0xA4BF	ADC Bias Voltage
0x3EEC	0x1221	Removes saturation noise that could be seen in Low Power Mode.
0x3EEE	0x1224	

Recommended Gain Table

Aptina recommends using the analog gain table listed below. This table limits the analog gain to 15.875x.

Table 3: Recommended Gain Stages

Desired Gain	Recommended Gain Register Setting
1–1.98	0x1040–0x107F
2–3.97	0x1840–0x187F
4–7.94	0x1C40–0x1C7F
8–15.875	0x1CC0–0x1CFF

Sensor Trimming

Trimming refers to the sensor calibration in the analog core that is stored on a small ROM in the sensor. The ROM stores the calibration as specific register values. The sensor will read these register values from the ROM on start-up and program the registers.

If the trimmed register values are overwritten by the camera, the calibrated register values used in these registers cannot be restored unless the sensor is reset.

For the MT9J001 sensor, there are two versions of the register trimming that have been used in manufactured parts:

- Version #1 where only R0x3ECE has been trimmed. Recommended values should be programmed to registers R0x3ED8, R0x3EDA, and R0x3EDC.
- Version #2 with improved trimming where R0x3ECE, R0x3ED8, R0x3EDA, and R0x3EDC have been trimmed.

The improved trimming had been used to increase the sensor yield. Manufactured sensors that have been sold to customers using either trimming Version #1 or #2 will have passed all quality tests at the factory. The sensors that were released in the "Alpha Samples" stage were not trimmed and are not guaranteed to meet all manufacturing specifications.

In MT9J003 ,the register R0x3ECE has been trimmed and should not be programmed. Registers R0x3ED8, R0x3EDA, and R0x3EDC should also be left at their initialized value.

Programming Decision Tree to Determine Trimming Version

The decision tree outlines how to determine the version of trimming that the sensor has been calibrated with. The decision tree can also determine if the chip is a MT9J003 and can have the improved power consumption using register 0x3EE2 applied to it.

These registers should be applied before streaming is enabled.

Figure 10: MT9J001 and MT9J003 Decision Tree to Determine Trimming Version

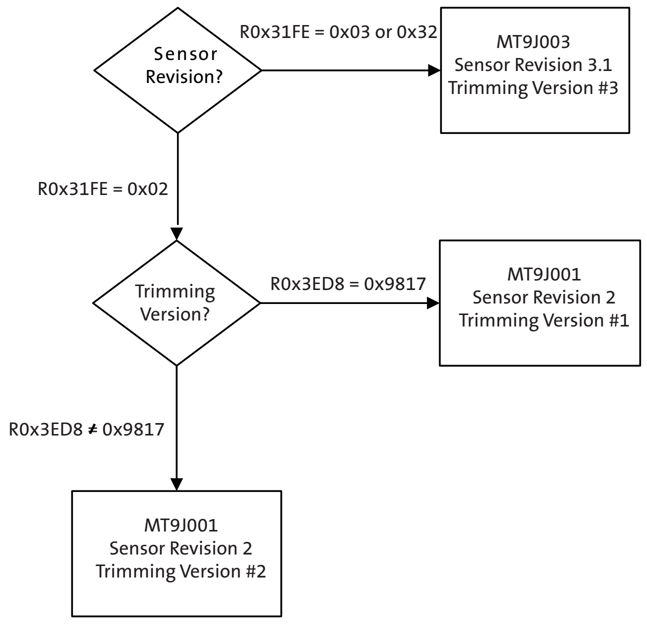

Sensor Revision?
R0x31FE = 0x03 or 0x32
MT9J003 Sensor Revision 3.1 Trimming Version #3
R0x31FE = 0x02
Trimming Version?
R0x3ED8 = 0x9817
MT9J001 Sensor Revision 2 Trimming Version #1
R0x3ED8 ≠ 0x9817
MT9J001 Sensor Revision 2 Trimming Version #2

Table 4 shows the values used for register trimming and their descriptions:

Table 4: Recommended Sensor Trimming for the MT9J001 and MT9J003 Sensors

Register Address	Recommended Value	Description
Sensor MT9J001 "Alpha Sample", Sensor is not trimmed: • Revision 2 sensor - MT9J001 • On-chip default value of Reg 0x3ED8 = 0x9817 (ADC Calibration)		
0x3ECE	0x141C	Analog Circuitry Calibration
0x3ED8	0x5803	ADC calibration
0x3EDA	0xD7C3	ADC Calibration
0x3EDC	0xD7E4	ADC Calibration
Sensor MT9J001, Trimming Version #1: • Revision 2 sensor - MT9J001 • On-chip default value of Reg 0x3ED8 = 0x9817 (ADC Calibration)		
0x3ECE	Use calibrated value. Do not program this register.	
0x3ED8	0x5803	ADC calibration
0x3EDA	0xD7C3	ADC Calibration
0x3EDC	0xD7E4	ADC Calibration
MT9J001, Trimming Version #2: • Revision 2 sensor - MT9J001 • On-chip default value of Reg 0x3ED8 ≠ 0x9817 (ADC Calibration)		
0x3ECE	Use calibrated value. Do not program these registers.	
0x3ED8		
0x3EDA		
0x3EDC		
MT9J003, Trimming Version #3: • Revision 3.1 sensor - MT9J003		
0x3ECE	Use calibrated value. Do not program this register.	
0x3ED8	Use calibrated values. Do not program these registers.	
0x3EDA		
0x3EDC		

Configuration to Use Parallel or HiSPi Interface

The sensor will need to be configured to use the Parallel or HiSPi interface. The sensor can be purchased as a HiSPi or Parallel sensor, however, the register configuration still needs to be programmed.

Table 5: Interface-Specific Configuration

Register Address	Recommended Value	Description
0x31AE	0x0301	Configure with the Parallel Sensor
0x31AE	0x0304	Configure with the HiSPi Sensor

Configuring the HiSPi Interface

The HiSPi register configuration is listed below. The HiSPi interface should be programmed based on the HiSPi receiver's requirements. This table lists registers that may need to be configured.

Table 6: HiSPi Programming Registers

Register	Mask	Value (On/Off)	Description
0x31C6	0x0001	0 - No Bar 1 - Bar	Vertical Left Bar Enable This feature adds an extra 4-bytes to the end of a SYNC code as described in the HiSPi adopter's specification.
	0x0002	0 - LSB 1 - MSB	Output MSB First

Programming Sensor Data Pedestal and Disabling Embedded Data

The sensor is recommended to be configured with a data pedestal value of 168. The data pedestal is a value added to each pixel that raises the black level from zero to the programmed value. The recommended value of 168 will slightly lower the dynamic range but will ensure that each pixel can be saturated.

The embedded data is enabled by default on the sensor. It is added by the digital block as an extra two lines at the top of the image. The extra two lines added to the image will not impact the sensor frame-rate. The format of this data follows the SMIA standard including gain, integration, and so on.

Table 7: Instructions to Program the Data Pedestal and to Disable the Embedded Data

Register	Setting	Description
R0x301A	0x0010	Unlock data pedestal register
R0x3064	0x0805	Disable embedded data
R0x301E	0x00A8	Set data pedestal to 168

Enabling Sensor Streaming

The sensor streaming can be enabled by configuring the reset_register of the sensor. The streaming can be disabled by setting the register to 0x0018, placing the sensor in a low power standby mode.

Table 8: Enable Streaming Using the Parallel Interface

Register Address	Recommended Value	Description
0x301A	0x001C	Enable HiSPi Streaming
0x301A	0x10DC	Enable Parallel Streaming

Recommended Register Sequence for Mode Changes

When configuring the sensor to output a new mode, the applied changes to the sensor should be implemented in a way that avoids "bad frames". A bad frame is a frame where all rows do not have the same integration or where offsets to the pixel values have changed during the frame. This can happen if a register change is written to the sensor that can be implemented at any time during a frame.

It is recommended that the "grouped_parameter_hold" function is used when changing modes. The function "grouped_parameter_hold" can be used to synchronize a group of register changes to the beginning of a new frame.

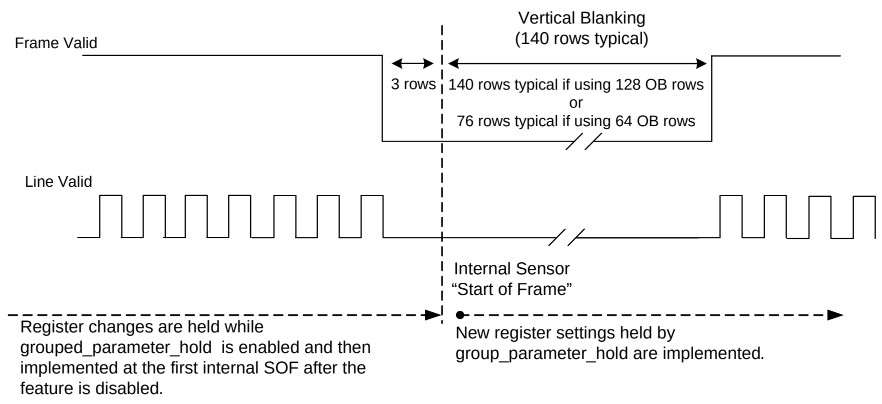
The "grouped_parameter_hold" function does not apply to all registers. The "MT9J001 Register Descriptions" document defines which registers can be held by listing them as "Frame Sync'd".

Using grouped_parameter_hold

The function "grouped_parameter_hold" can be enabled by setting either register 0x0104 or bit 0x301A[15] to "1". After the function is enabled, the grouped_parameter_hold can hold certain register changes until it is disabled.

After the grouped_parameter_hold is disabled, the sensor will wait until the internal sensor "Start of Frame" as described in the diagram below before implementing held register changes.

Figure 11: Frame Timing Using the "grouped_parameter_hold" Feature


Frame Valid
Vertical Blanking
(140 rows typical)
3 rows
140 rows typical if using 128 OB rows
or
76 rows typical if using 64 OB rows
Line Valid
Internal Sensor
"Start of Frame"
Register changes are held while grouped_parameter_hold is enabled and then implemented at the first internal SOF after the feature is disabled.
New register settings held by group_parameter_hold are implemented.

The recommended syntax for using the “grouped_parameter_hold” function is shown below:

grouped_parameter_hold = 1
<program register changes of the new mode>
grouped_parameter_hold = 0

This use-case will allow the user to implement a group of registers at the end of an uninterrupted frame. If the time delay between writing the group of registers and the end of the frame is unacceptable, the frame_restart function can be used to interrupt the current frame.

Using frame_restart with grouped_parameter_hold

The “frame_restart” function will end the current frame and begin a new frame. The beginning of the new frame will start as shown in Figure 11 following 140 lines of vertical blanking. After the new frame has begun, the frame_restart bit will reset from “1” to “0”.

The “frame_restart” can be used with the “grouped_parameter_hold” feature to hold the implementation of a number of registers when interrupting a frame.

The recommended syntax for using the “grouped_parameter_hold” and “frame_restart” functions together is shown below:

- grouped_parameter_hold = 1
- <program register changes of the new mode>
- frame_restart = 1
- grouped_parameter_hold = 0

Appendix A: PLL Programming Tables

Table 9: HiSPi PLL Register Setup Examples
Input clock = 10 MHz

Register Address	Recommended Value			Register Name
	Low Power Mode (Limited by array clock)	Normal Power Mode (Array clock matches output clock)		
0x0300	3	3	3	vt_pix_clk_div
0x0302	1	1	1	vt_sys_clk_div
0x0304	1	1	1	pre_pll_clk_div
0x0306	48	36	48	pll_multiplier
0x0308	12	12	12	op_pix_clk_div
0x030A	1	1	1	op_sys_clk_div
0x3016[2:0]	1	1	1	row_speed[pc_speed]
0x3016[10:8]	1	1	1	row_speed[op_speed]
HiSPi Data Rate per Lane	480Mbps	360 Mbps	480 Mbps	
Total HiSPi Data Rate	1.92 Gbps	1.44 Gbps	1.92 Gbps	
HiSPi Clock	240 MHz	160 MHz	240 MHz	
Array Clock	80 MHz	120 MHz	160 MHz	
Output Clock	40 MHz per data lane	30 MHz per data lane	40 MHz per data lane	
Max 10MP FPS	7.5 FPS	11 FPS	14.7 FPSs	
Max 1080p FPS	30 FPS	45 FPS	60 FPS	

Note: The total bandwidth supported by the HiSPi interface is the sum of four data lanes (that is, 160 MHz with an output clock of 40 Mhz).

Table 10: Parallel PLL Register Setup Examples
Input clock = 10 MHz

Register Address	Recommended Value		Register Name
	Low Power Mode (Array clock matches output clock)	Normal Power Mode (Limited by output clock)	
0x0300	4	4	vt_pix_clk_div
0x0302	1	1	vt_sys_clk_div
0x0304	1	1	pre_pll_clk_div
0x0306	64	64	pll_multiplier
0x0308	8	8	op_pix_clk_div
0x030A	1	1	op_sys_clk_div
0x3016[2:0]	1	1	row_speed[pc_speed] (refers to clk_pixel divider)
0x3016[10:8]	1	1	row_speed[op_speed] (refers to clk_op divider)
Array Clock	80 MHz	160 MHz	
Output Clock	80 MHz	80 MHz	
Max 10Mp FPS	7.5 FPS	7.5 FPS	
Max 1080p FPS	30 FPS	30 FPS	

As shown in Table 9 and Table 10, the clk_pixel divider is changed from “1” to “2” when low power mode is enabled. This change automatically happens when enabling low power mode and the divider will automatically switch from “2” to "“1” when low power mode is disabled.

Appendix B: Sensor Mode HiSPi Programming Table

Table 11: Sensor Mode HiSPi Programming Table

The frame-rates calculated in this table assume the PLL is configured for an array clock of 160MHz as described in Table 9.		**10MP ERS 14.7 FPS**	**1080p60 59.94 FPS**	**720p60 + 10%EIS 59.94 FPS**	**Monitor 29.97 FPS**	**VGA 59.94 FPS**	**Full Array 14 FPS**
		Normal Power Mode		**Low Power Mode**			Normal Power Mode
	FOV From Array						
0x3004	X_ADDR_START	112	32	640	112	112	32
0x3008	X_ADDR_END	3775	3873	3457	3769	3769	3871
0x3002	Y_ADDR_START	8	296	584	8	8	8
0x3006	Y_ADDR_END	2755	2457	2169	2753	2753	2755
	Binning/Summing						
0x3040[6:8]	X_ODD_INCREMENT	1	3	3	7	7	1
0x3040[0:5]	Y_ODD_INCREMENT	1	3	3	7	7	1
0x3040[10]	XY_BIN_ENABLE	0	0	0	0	0	0
0x3040[11]	X_BIN_ENABLE	0	1	1	1	1	0
0x3040[9]	LOW POWER MODE	0	0	1	1	1	0
0x3040[12]	Bin_Sum (not used)	0	0	0	0	0	0
0x3040[13]	Y Summing Enable	0	1	1	1	1	0
0x3EDE	**Reserved**	0xE41A	0xE41A	0xE412	0xE412	0xE412	0xE41A
0x3EDC[7]	Reserved	1	1	1	1	1	1
0x3178[4:5]	Reserved (enables X_SUM)	0	3	3	3	3	0
0x3178[6:7]	Reserved	0	1	1	1	1	0
	Scaling and Cropping						
0x400	SCALING MODE	2	2	2	2	2	2
0x404	M_SCALE	16	16	16	16	16	16
0x34C	X_OUTPUT_SIZE	3664	1920	1408	916	916	3840
0x34E	Y_OUTPUT_SIZE	2748	1080	792	688	688	2748
	Row Timing						
0x342	LINE_LENGTH_PCK	3776	2299	1528	1206	1206	3952
0x340	FRAME_LENGTH_LINES	2892	1161	873	2214	1106	2892
0x3010	FINE_CORRECTION	156	156	72	72	72	156
0x3014	FINE_INTEGRATION_TIME	1010	1010	522	522	522	1010
0x3018	EXTRA_DELAY	0	0	0	0	0	0
	Power Optimization						
0x3170	Register that is required to be set with Low Power Mode	0x00E5	0x00E5	0x0071	0x0071	0x0071	0x00E5
	Column Correction (0xB080–128, 0x9080–64)						
0X30D4	OB Rows	0xB080	0x9080	0x9080	0x9080	0xB080	0xB080

Note: The modes in this table can be referenced inTable 1 on page 6. The frame rate calculation assumes a PLL configuration where the array clock is set to 160 MHz. In Low Power Mode, the sensor will automatically halve the array clock speed to 80 MHz and lower the current-bias applied to the analog circuitry to reduce power consumption. The term "EIS" refers to extra pixels readout to support Electronic Image Stabilization.

Appendix C: Sensor Mode Parallel Programming Table

Table 12: Sensor Mode Parallel Programming Table

The frame-rates calculated in this table assume the PLL is configured for an array clock of 160 MHz as described in Table 10.		10MP ERS 7.5 FPS	1080p30 29.97 FPS	720p60 + 10% EIS 59.94 FPS	Monitor 29.97 FPS	VGA 59.94 FPS	Full Array 7 FPS
		Low Power Mode					
	FOV From Array						
0x3004	X_ADDR_START	112	32	640	112	112	32
0x3008	X_ADDR_END	3775	3873	3457	3769	3769	3871
0x3002	Y_ADDR_START	8	296	584	8	8	8
0x3006	Y_ADDR_END	2755	2457	2169	2753	2753	2755
	Binning/Summing						
0x3040[6:8]	X_ODD_INCREMENT	1	3	3	7	7	1
0x3040[0:5]	Y_ODD_INCREMENT	1	3	3	7	7	1
0x3040[10]	XY_BIN_ENABLE	0	0	0	0	0	0
0x3040[11]	X_BIN_ENABLE	0	1	1	1	1	0
0x3040[9]	LOW POWER MODE	1	1	1	1	1	1
0x3040[12]	Bin_Sum (not used)	0	0	0	0	0	0
0x3040[13]	Y Summing Enable	0	1	1	1	1	0
0xEDE	**Reserved**	0xE412	0xE412	0xE412	0xE412	0xE412	0xE412
0x3EDC[7]	Reserved	1	1	1	1	1	1
0x3178[4:5]	Reserved (enables X_SUM)	0	3	3	3	3	0
0x3178[6:7]	Reserved	0	1	1	1	1	0
	Scaling and Cropping						
0x400	SCALING MODE	2	2	2	2	2	2
0x404	M_SCALE	16	16	16	16	16	16
0x34C	X_OUTPUT_SIZE	3664	1920	1408	916	916	3840
0x34E	Y_OUTPUT_SIZE	2748	1080	792	688	688	2748
	Row Timing						
0x342	LINE_LENGTH_PCK	3776	2299	1528	1206	1206	3952
0x340	FRAME_LENGTH_LINES	2892	1161	873	2213	1106	2891
0x3010	FINE_CORRECTION	72	72	72	72	72	72
0x3014	FINE_INTEGRATION_TIME	522	522	522	522	522	522
0x3018	EXTRA_DELAY	0	0	0	0	0	0
	Power Optimization						
0x3170	Sensor Timing to enable Low Power Mode	0x0071	0x0071	0x0071	0x0071	0x0071	0x0071
	Column Correction (0xB080–128, 0x9080–64)						
0X30D4	OB Rows	0xB080	0x9080	0x9080	0xB080	0xB080	0xB080

Note: The modes in this table can be referenced in Table 1 on page 6. The frame rate calculation assumes a PLL configuration where the array clock is set to 160 MHz. In Low Power Mode, the sensor will automatically halve the array clock speed to 80 MHz and lower the current-bias applied to the analog circuitry to reduce power consumption. The term "EIS" refers to extra pixels readout to support Electronic Image Stabilization.

Appendix D: High Frame Rate Modes

Table 13: Sensor Mode High Frame Rate Programming Table

The frame rate calculated in this table uses an array clock of 160 MHz in normal power mode as described in Table 9 with the HiSPi interface or Table 10 with the parallel interface.		**FOV: 3664 x 2748 Output: 916 x 344 219 FPS**	**FOV: 2560 x 1920 Output: 640 x 240 290 FPS**	**FOV: 1280 x 960 Output: 320 x 240 290FPS**
	FOV From Array			
0x3004	X_ADDR_START	112	640	1280
0x3008	X_ADDR_END	3769	3193	2553
0x3002	Y_ADDR_START	8	872	1352
0x3006	Y_ADDR_END	2748	2777	2297
	Binning/Summing			
0x3040[6:8]	X_ODD_INCREMENT	7	7	7
0x3040[0:5]	Y_ODD_INCREMENT	15	15	7
0x3040[10]	XY_BIN_ENABLE	0	0	0
0x3040[11]	X_BIN_ENABLE	1	1	1
0x3040[9]	LOW POWER MODE	0	0	0
0x3040[12]	Bin_Sum (not used)	0	0	0
0x3040[13]	Y Summing Enable	0	0	0
0x3EDE	**Reserved**	0xE42A	0xE41A	0xE41A
0x3EDC[7]	Reserved	1	1	1
0x3178[4:5]	Reserved (enables X_SUM)	3	3	3
0x3178[6:7]	Reserved	1	1	1
	Scaling and Cropping			
0x400	SCALING MODE	2	2	2
0x404	M_SCALE	16	16	16
0x34C	X_OUTPUT_SIZE	916	640	320
0x34E	Y_OUTPUT_SIZE	344	240	240
	Row Timing			
0x342	LINE_LENGTH_PCK	1732	1732	1732
0x340	FRAME_LENGTH_LINES	424	320	320
0x3010	FINE_CORRECTION	156	156	156
0x3014	FINE_INTEGRATION_TIME	1010	1010	1010
0x3018	EXTRA_DELAY	0	0	0
	Power Optimization			
0x3170	Sensor Timing to enable Low Power Mode	0x00E5	0x00E5	0x00E5
	OB Rows for Column Correction			
0X30D4	OB Rows	0x9080	0x9080	0x9080

Note: The frame rate calculation assumes a PLL configuration where the array clock is set to 160 MHz. The frame rates for the modes listed in this table are limited by the minimum row readout time of the sensor.

Appendix E: Subsampling Using Summing or Binning

The "XY-SUM" summing feature will sum the pixel readout in the analog domain while the "X-BIN and Y-BIN" binning feature averages the pixels in the analog domain. The benefit of using the summing feature is a sensitivity improvement of 12dB (6dB X-SUM + 6dB Y-SUM) relative to the single pixel readout. This extra sensitivity can be used to improve image quality in low light while combination of binning and summing can be used when less sensitivity is required in brighter conditions.

These registers can be configured in any mode where x_odd_increment and y_odd_increment are either configured to a value of 3 or 7. A typical use case for these modes would be the 1080p, 720p+10%EIS, Monitor, or VGA modes described in Appendix B or C.

Table 14: Description of "XY-SUM", "X-BIN Y-SUM", and "X-BIN and Y-BIN"

	XY - SUM (+12dB)	X-BIN Y-SUM (+6dB)	X-BIN and Y-BIN
x_odd_increment = 3	Sum 2x	Bin 2x	Bin 2x
x_odd_increment = 7	Sum 2x + Skip 2x	Bin 2x + Skip 2x	Bin 2x + Skip 2x
y_odd_increment = 3	Sum 2y	Sum 2y	Bin 2y
y_oddd_increment = 7	Sum 2y + Skip 2y	Sum 2y + Skip 2y	Bin 2y + Skip 2y

If these registers are used with a mode described in Appendix B or C, the other register settings such as the Row Timing registers do not need to change providing that the values of x_odd_increment and y_odd_increment remain the same. This means that a video mode can be switched from using "XY-SUM" to either "X-BIN Y-SUM" or "X-BIN and Y-BIN" without a corrupted frame. Registers 0x3178 and 0x3ED0 are not synchronized by grouped_parameter_hold and the three registers will need to be written within a single frame.

Table 15: Register Settings to Configure Video Modes to Use Binning

Register	Register Description	XY-SUM		X-BIN Y-SUM		X-BIN and Y-BIN	
0x3040[6:8]	X_ODD_INCREMENT	3	7	3	7	3	7
0x3040[6:8]	X_ODD_INCREMENT	3	7	3	7	3	7
0x3040[0:5]	Y_ODD_INCREMENT	3	7	3	7	3	7
0x3040[10]	XY_BIN_ENABLE	0	0	0	0	0	0
0x3040[11]	X_BIN_ENABLE	1	1	1	1	1	1
0x3040[9]	LOW POWER MODE	1	1	1	1	1	1
0x3040[12]	Bin_Sum (Not Used)	0	0	0	0	0	0
0x3040[13]	Y Summing Enable	1	1	1	1	0	0
0x3040	Read_Mode Register	0x2AC3	0x2BC7	0x2AC3	0x2BC7	0x0AC3	0x0BC7
0x3178	Control for xy-summing	0x0070		0x0050		0x1152	
0x3ED0	Required for Y-Binning	0x1B24		0x1B24		0x5BE4	

Appendix F: Sensor Programming Examples

This section describes a programming example of the MT9J001 CMOS Image sensor. For further examples, refer to the sensor INI file included with the Aptina Devware demo software installation.

Syntax

```
REG      = <address>, <value>
BITFIELD = <address>, <mask>, <value>
```

<address>	the register address
<value>	the new value to assign to the register
<mask>	is the part of a register value that needs to be updated with a new value

Sensor Programming Example 1: Sensor Initialization

The following sequence details the sensor start-up sequence. The sequence programs the sensor to stream 10Mp images using the HiSPi interface at 14.7 FPS.

```
//Program the PLL using a 10MHz input clock
REG=0x0300, 4// vt_pix_clk_div
REG=0x0302, 1// vt_sys_clk_div
REG=0x0304, 1  // pre_pll_clk_div
REG=0x0306, 64// pll_multiplier
REG=0x0308, 8// op_pix_clk_div
REG=0x030A, 1// op_sys_clk_div
BITFIELD=0x3016,0x007,1 // row_speed[2:0] (pc_speed)
BITFIELD=0x3016,0x700,1 // row_speed[10:8] (op_speed)

//Default Recommended Register Changes
REG=0x316C,0x0429//Reserved
REG=0x3174,0x8000//Reserved
REG=0x3E40,0xDC05//Reserved
REG=0x3E42,0x6E22//Reserved
REG=0x3E44,0xDC22//Reserved
REG=0x3E46,0xFF00//Reserved
REG=0x3ED4,0xF998//Reserved
REG=0x3ED6,0x9789//Reserved
REG=0x3EDE,0xE41A//Reserved
REG=0x3EE0,0xA43F//Reserved
```

```
REG=0x3EE2,0xA4BF//Reserved
REG=0x3EEC,0x1221//Reserved
REG=0x3EEE, 0x1224 // Reserved

//***Check for TRIM values
LOAD=TRIMS

//***Change the data pedestal and SMIA encoded data
REG= 0x301A, 0x0010// Reserved
REG= 0x3064, 0x0805// Reserved
REG= 0x301E, 0x00A8// Reserved
BITFIELD=0x3064,0x0100,0 //Disable SMIA Encoding

//**Configure the sensor for 10M Readout

//ARRAY READOUT SETTINGS
REG=0x3004, 112   // X_ADDR_START
REG=0x3008, 3775// X_ADDR_END
REG=0x3002, 8    // Y_ADDR_START
REG=0x3006, 2755// Y_ADDR_END
BITFIELD=0x3040,0x01C0,1// X_ODD_INCREMENT
BITFIELD=0x3040,0x003F,1// Y_ODD_INCREMENT
BITFIELD=0x3040,0x400,0 // XY_BIN_ENABLE
BITFIELD=0x3040,0x800,0 // X_BIN_ENABLE
BITFIELD=0x3040,0x200,1// Low Power Mode
BITFIELD=0x3040,0x1000,0// Binning Summing Enable
BITFIELD=0x3040,0x2000,0// Y SUM Enable

REG=0x3170,0x0071//Reserved - Low Power Mode

//OUTPUT DATA Path Settings
REG=0,0x400, 0// SCALING MODE (vertical and horizontal scaling)
REG=0,0x404, 16// M_SCALE
REG=0x034C, 3664// X_OUTPUT_SIZE
REG=0x034E, 2748// Y_OUTPUT_SIZE
```

```
//Frame and Integration Time Settings
REG=0x0342, 3776// LINE_LENGTH_PCK
REG=0x0340, 2892// FRAME_LENGTH_LINES
REG=0x3010, 72// FINE_CORRECTION Changed
REG=0x3014, 522// FINE_INTEGRATION_TIME
REG=0x3018, 0// EXTRA_DELAY

REG= 0x30D4, 0xB080// 128 OB Column Sample

//***Enable Streaming
REG= 0x301A, 0x10DC // Enable Streaming WIth the Parallel Interface
```

Sensor Programming Example 2: Programming a New Sensor Mode

The following sequence is an example of how to program the sensor to use the 1080p imaging using the HiSPi interface at 59.94 FPS.

```
//Enable the group_parameter_hold feature
BITFIELD=0x301A,0x8000,1// GROUPED_PARAMETER_HOLD

//Program the sensor mode
BITFIELD=0x3178, 0x0030, 0x0003 //Reserved
BITFIELD=0x3178, 0x00C0, 0x0001 //Reserved

//ARRAY READOUT SETTINGS
REG=0x3004, 32   // X_ADDR_START
REG=0x3008, 3873// X_ADDR_END
REG=0x3002, 296   // Y_ADDR_START
REG=0x3006, 2457// Y_ADDR_END
BITFIELD=0x3040,0x01C0,3// X_ODD_INCREMENT
BITFIELD=0x3040,0x003F,3// Y_ODD_INCREMENT
BITFIELD=0x3040,0x400,0 // XY_BIN_ENABLE
BITFIELD=0x3040,0x800,1 // X_BIN_ENABLE
BITFIELD=0x3040,0x200,0 // Low Power Mode
BITFIELD=0x3040,0x1000,0// Binning Summing Enable
BITFIELD=0x3040,0x2000,1// Y Summing Enable
```

```
REG=0x3170,0x00E5//Reserved - Enabled for all Normal Power Mode

//OUTPUT DATA Path Settings
REG=0,0x400, 2// SCALING MODE (vertical and horizontal scaling)
REG=0,0x404, 16// M_SCALE
REG=0x034C, 1920// X_OUTPUT_SIZE
REG=0x034E, 1080// Y_OUTPUT_SIZE

//Frame and Integration Time Settings
REG=0x0342, 2299// LINE_LENGTH_PCK
REG=0x0340, 1161// FRAME_LENGTH_LINES
REG=0x3010, 156// FINE_CORRECTION Changed
REG=0x3014, 1010// FINE_INTEGRATION_TIME
REG=0x3018, 0// EXTRA_DELAY

REG= 0x30D4, 0x9080// 128 OB Column Sample

//Sync Settings and Restart Frame
BITFIELD=0x301A,0x002,1 // RESTART FRAME
BITFIELD=0x301A,0x8000,0// GROUPED_PARAMETER_HOLD
```

Revision History

Rev. B 7/28/10

- Updated “Introduction” on page 5
- Updated “Identifying the MT9J001 and MT9J003 Using Register Settings” on page 5
- Updated Table 1, “Recommended Imaging Readout Modes,” on page 6
- Replaced Figures 2 and 3.
- Updated Figure 4: “PLL Configuration for the Parallel Interface,” on page 9, Figure 5: “PLL Configuration for the Parallel Interface Using Low Power Mode,” on page 10, and Figure 6: “PLL Configuration for the HiSPi Interface Using Low Power Mode,” on page 11
- Updated “Recommended Register Settings for Sensor Initialization” on page 14
- Updated Figure 10: “MT9J001 and MT9J003 Decision Tree to Determine Trimming Version,” on page 16
- Updated Table 11, “Sensor Mode HiSPi Programming Table,” on page 23
- Updated Table 12, “Sensor Mode Parallel Programming Table,” on page 24
- Added Table 14, “Description of "XY-SUM", "X-BIN Y-SUM", and "X-BIN and Y-BIN",” on page 26
- Updated “Sensor Programming Example 1: Sensor Initialization” on page 27
- Updated “Sensor Programming Example 2: Programming a New Sensor Mode” on page 29

Rev. A 6/23/09

- Initial release