Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

AR0542

AR0542 1/4-Inch 5 Mp CMOS Digital Image Sensor

ON Semiconductor®

www.onsemi.com

ORDERING INFORMATION

See detailed ordering and shipping information on page 3 of this data sheet.

General Description

The ON Semiconductor AR0542 is a 1/4-inch CMOS active-pixel digital image sensor with a pixel array of 2592 (H) x 1944 (V) (2608 (H) x 1960 (V) including border pixels). It incorporates sophisticated on-chip camera functions such as windowing, mirroring, column and row skip modes, and snapshot mode. It is programmable through a simple two-wire serial interface and has very low power consumption.

The AR0542 digital image sensor features ON Semiconductor'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 AR0542 sensor can generate full resolution image at up to 15 frames per second (fps). An on-chip analog-to-digital converter (ADC) generates a 10-bit value for each pixel.

Features

- Low Dark Current
- Simple Two-wire Serial Interface
- Auto Black Level Calibration
- Support for External LED or Xenon Flash
- High Frame Rate Preview Mode with Arbitrary Down-size Scaling from Maximum Resolution
- Programmable Controls: Gain, Horizontal and Vertical Blanking, Auto Black Level Offset Correction, Frame Size/Rate, Exposure, Left–right and Top–bottom Image Reversal, Window Size, and Panning
- Data Interfaces: Parallel or Single/Dual Lanes Serial Mobile Industry Processor Interface (MIPI)
- On-die Phase-locked Loop (PLL) Oscillator
- Bayer Pattern Down-size Scaler
- Superior Low-light Performance
- Integrated Position and Color-based Shading Correction
- 7.7 Kb One-time Programmable Memory (OTPM) for Storing Shading Correction Coefficients of Three Light Sources and Module Information
- Extended Flash Duration that is up to Start of Frame Readout
- On-chip VCM Driver

Applications

- Cellular Phones
- Digital Still Cameras
- PC Cameras
- PDAs

Table 1. KEY PERFORMANCE PARAMETERS

Parameter		Value
Optical Format		1/4-inch (4:3)
Active Imager Size		3.63 mm (H) x 2.72 (V):4.54 mm diagonal
Active Pixels		2592 (H) x 1944 (V)
Pixel Size		1.4 μm x 1.4 μm
Chief Ray Angle		25.0°
Color Filter Array		RGB Bayer Pattern
Shutter Type		Electronic Rolling Shutter (ERS)
Input Clock Frequency		6–27 MHz
Maximum Data Rate	Parallel	84 Mbps at 84 MHz PIXCLK
	MIPI	840 Mbps per Lane
Frame rate	Full Resolution (2592 x 1944)	15 fps
	1080P	19.8 fps (100% FOV, crop to 16:9) 30 fps (77% FOV, crop to 16:9)
	720P	30 fps (98% FOV, crop to 16:9, bin2) 60 fps (98% FOV, crop to 16:9, skip2)
	VGA (640 x 480)	60 fps (100% FOV, bin2skip2) 115 fps (100% FOV, skip4)
ADC Resolution		10-bit, on-die
Responsivity		0.82 V/lux-sec (550 nm)
Dynamic Range		66 dB
SNR_{MAX}		36.5 dB
Supply Voltage	Digital I/O	1.7–1.9 V (1.8 V nominal) or 2.4–3.1 V (2.8 V nominal)
	Digital Core	1.15–1.25 V(1.2 V nominal)
	Analog	2.6–3.1 V (2.8 V nominal)
	Digital 1.8 V	1.7–1.9 V (1.8 V nominal)
Power Consumption	Full Resolution	Parallel: 288.2 mW at 70°C (TYP)
		MIPI: 215 mW at 70°C (TYP)
	Standby*	25 μW at 70°C (TYP)
Package		Bare die
Operating Temperature		–30°C to +70°C (at Junction)

*Hard Standby by pulling XShutdown LOW.

ORDERING INFORMATION

Table 2. AVAILABLE PART NUMBERS

Part Number	Product Description	Orderable Product Attribute Description
AR0542MBSC25SUD20	5 MP 1/4” CIS	RGB Bare die
AR0542MBSC25SUFAD−GEVK	5 MP 1/4” CIS DK	Demo Kit
AR0542MBSC25SUFAH−GEVB	5 MP 1/4” CIS HB	Headboard

FUNCTIONAL OVERVIEW

The AR0542 is a progressive-scan sensor that generates a stream of pixel data at a constant frame rate. It uses an on-chip, phase-locked loop (PLL) to generate all internal clocks from a single master input clock running between 6 and 27 MHz. The maximum pixel rate is 84 Mp/s, corresponding to a pixel clock rate of 84 MHz. A block diagram of the sensor is shown in Figure 1.

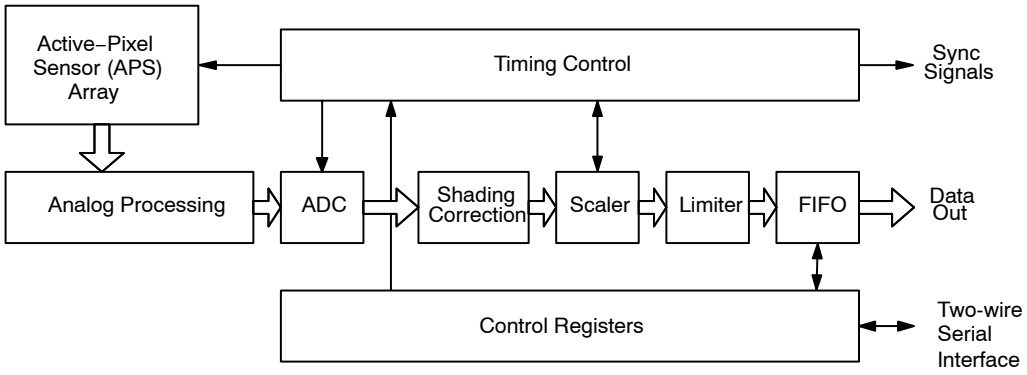


Figure 1. Block Diagram

The core of the sensor is a 5 Mp active-pixel array. The timing and control circuitry sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and reading that row, the pixels in the row integrate incident light. The exposure is controlled by varying the time interval between reset and readout. Once a row has been read, the data from the columns are sequenced through an analog signal chain (providing offset correction and gain), and then through an ADC. The output from the ADC is a 10-bit value for each pixel in the array. The ADC output passes through a digital processing signal chain (which provides further data path corrections and applies digital gain).

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

The sensor contains a set of control and status registers that can be used to control many aspects of the sensor behavior including the frame size, exposure, and gain setting. These registers can be accessed through a two-wire serial interface.

The output from the sensor is a Bayer pattern; alternate rows are a sequence of either green and red pixels or blue and green pixels. The offset and gain stages of the analog signal chain provide per-color control of the pixel data.

The control registers, timing and control, and digital processing functions shown in Figure 1 are partitioned into three logical parts:

- A sensor core that provides array control and data path corrections. The output of the sensor core is a 10-bit parallel pixel data stream qualified by an output data clock (PIXCLK), together with LINE_VALID (LV) and FRAME_VALID (FV) signals.
- A digital shading correction block to compensate for color/brightness shading introduced by the lens or chief ray angle (CRA) curve mismatch.
- Additional functionality is provided. This includes a horizontal and vertical image scaler, a limiter, a data compressor, an output FIFO, and a serializer.

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

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

Pixel Array

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

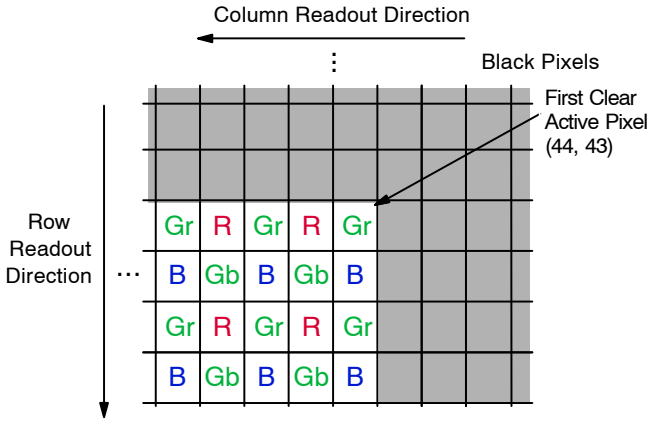


Figure 2. Pixel Color Pattern Detail (Top Right Corner)

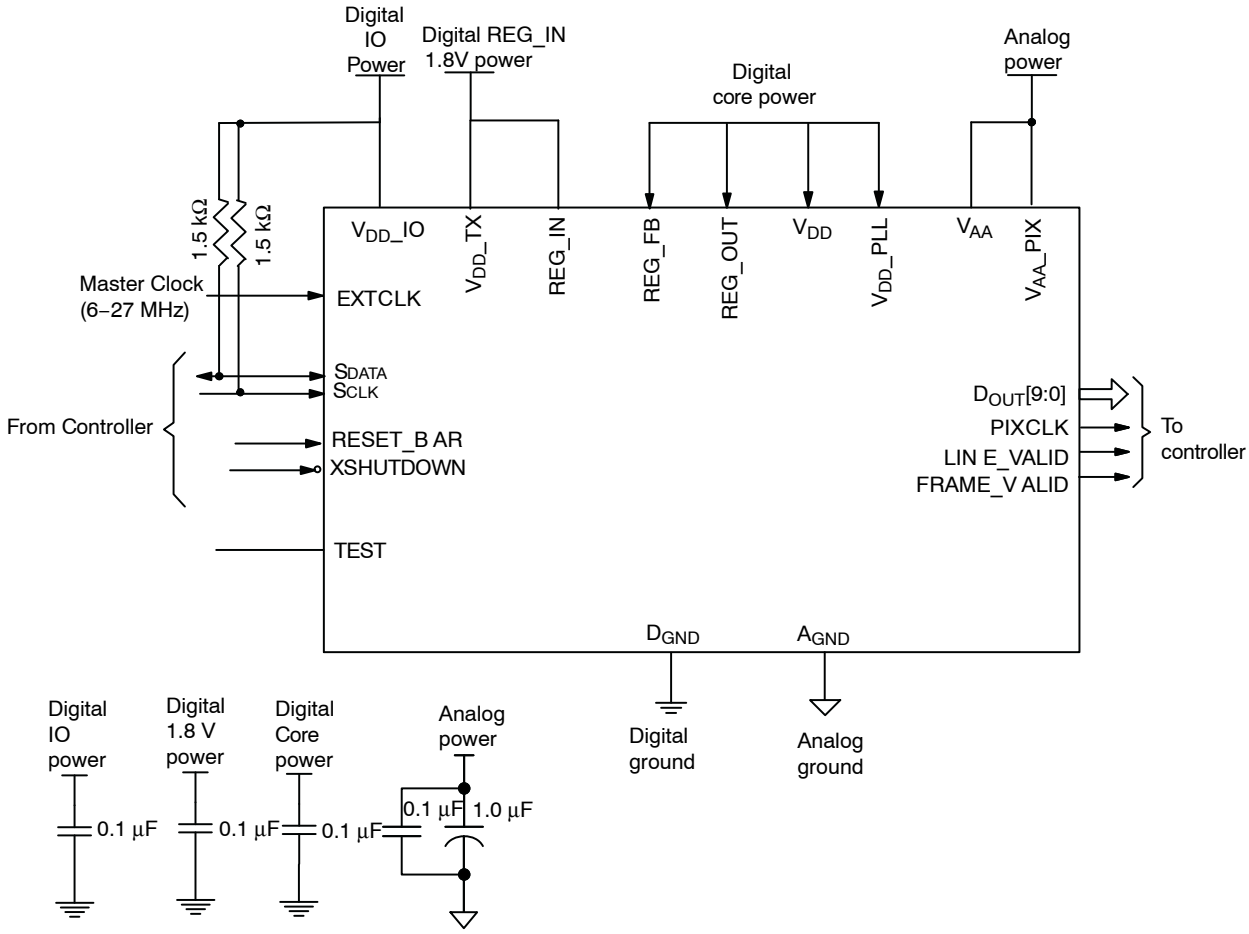
OPERATING MODES

By default, the AR0542 powers up with the serial pixel data interface enabled. The sensor can operate in serial MIPI mode. This mode is preconfigured at the factory. In either case, the sensor has a SMIA-compatible register interface while the two-wire serial device address is compliant with SMIA or MIPI requirements as appropriate. The reset level on the TEST pin must be tied in a way that is compatible with the configured serial interface of the sensor, for instance, TEST = 1 for MIPI.

The AR0542 also supports parallel data out in MIPI configuration. Typical configurations are shown in Figure 3, Note 15, and Figure 4. These operating modes are described in “Control of the Signal Interface”.

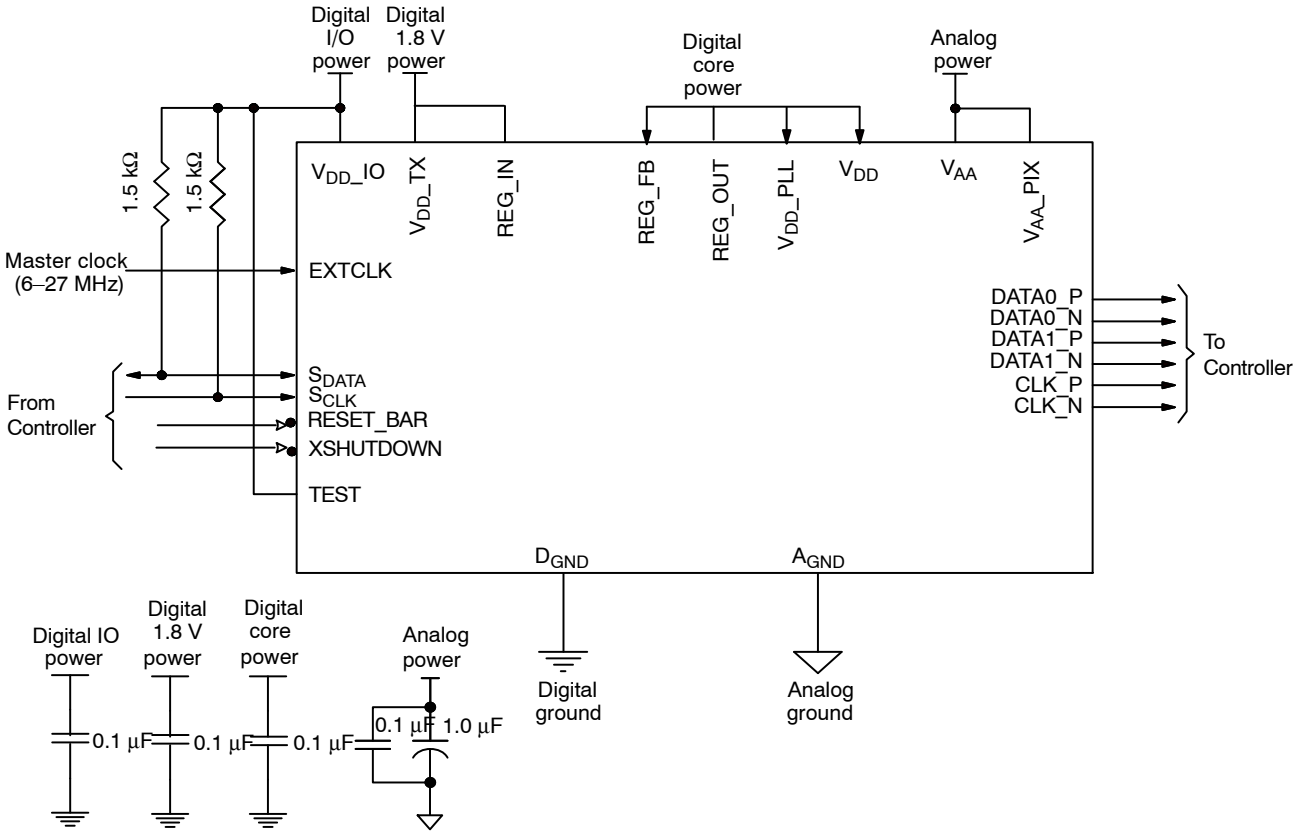
For low-noise operation, the AR0542 requires separate power supplies for analog and digital. Incoming digital and analog ground conductors can be tied together next to the die. Both power supply rails should be decoupled from the ground using capacitors as close as possible to the die.

CAUTION: ON Semiconductor does not recommend the use of inductance filters on the power supplies or output signals.



Note:
1. All power supplies must be adequately decoupled.
2. ON Semiconductor recommends a resistor value of 1.5 kΩ, but a greater value may be used for slower two-wire speed. This pull-up resistor is not required if the controller drives a valid logic level on SCLK at all times.
3. VDD_IO can be either 1.8 V (nominal) or 2.8 V (nominal). If VDD_IO is 1.8 V, VDD_IO can be tied to Digital REG_IN 1.8 V
4. VAA and VAA_PIX must be tied together.
5. VDD and VDD_PLL must be tied together
6. The serial interface output pads can be left unconnected if the parallel output interface is used.
7. ON Semiconductor recommends having 0.1 µF and 1.0 µF decoupling capacitors for analog power supply and 0.1 µF decoupling capacitor for other power supplies. Actual values and results may vary depending on layout and design considerations.
8. TEST can be tied to DGND (Device ID address = 0x20) or VDD_IO (Device ID address = 0x6C).
9. VDD _TX and REG_IN must be tied together.
10. Refer to the power-up sequence for XSHUTDOWN and RESET_BAR control.
11. The frequency range for EXTCLK must be 6–27 MHz.
12. VPP, which can be used during the module manufacturing process, is not shown in Figure 3. This pad is left unconnected during normal operation.
13. VCM_ISINK and VCM_GND, which can be used for internal VCM AF driver, are not shown in Figure 3. VCM_ISINK must be tied to the VCM actuator and VCM_GND must be tied to the DGND when the internal VCM is used. These pads are left unconnected if the internal VCM driver is not used.
14. The GPI[3:0] pins, which can be either statically pulled HIGH/LOW to be used as module IDs, or they can be programmed to perform special functions (TRIGGER, OE_BAR, SADDR, STANDBY) to be dynamically controlled, are not shown in Figure 3.
15. The FLASH, which can be used for flash control, is not shown in Figure 3.

Figure 3. Typical Configuration: Parallel Pixel Data Interface



Note:
1. All power supplies must be adequately decoupled.
2. ON Semiconductor recommends a resistor value of 1.5 kΩ, but a greater value may be used for slower two-wire speed. This pull-up resistor is not required if the controller drives a valid logic level on SCLK at all times.
3. VDD_IO can be either 1.8 V (nominal) or 2.8 V (nominal). If VDD_IO is 1.8 V, VDD_IO can be tied to Digital 1.8 V Power.
4. VAA and VAA_PIX must be tied together.
5. VDD and VDD_PLL must be tied together
6. The serial interface output pads can be left unconnected if the parallel output interface is used.
7. ON Semiconductor recommends having 0.1 µF and 1.0 µF decoupling capacitors for analog power supply and 0.1 µF decoupling capacitor for other power supplies. Actual values and results may vary depending on layout and design considerations.
8. TEST must be tied to VDD_IO for MIPI configuration (Device ID address = 0x6C).
9. VDD _TX and REG_IN must be tied together.
10. Refer to the power-up sequence for XSHUTDOWN and RESET_BAR control.
11. The frequency range for EXTCLK must be 6−27 MHz.
12. VPP, which can be used during the module manufacturing process, is not shown in Figure 4. This pad is left unconnected during normal operation.
13. VCM_ISINK and VCM_GND, which can be used for internal VCM AF driver, are not shown in Figure 4. VCM_ISINK must be tied to the VCM actuator and VCM_GND must be tied to the DGND when the internal VCM is used. These pads are left unconnected if the internal VCM driver is not used.
14. The GPI[3:0] pins, which can be either statically pulled HIGH/LOW to be used as module IDs, or they can be programmed to perform special functions (TRIGGER, OE_BAR, SADDR, STANDBY) to be dynamically controlled, are not shown in Figure 4.
15. The FLASH, which can be used for flash control, is not shown in Figure 4.

Figure 4. Typical Configuration: Serial Dual-Lane MIPI Pixel Data Interface

SIGNAL DESCRIPTIONS

Table 3 provides signal descriptions for AR0542 die. For pad location and aperture information, refer to the AR0542 die data sheet.

Table 3. SIGNAL DESCRIPTIONS

Pad Name	Pad Type	Description
EXTCLK	Input	Master clock input, 6–27 MHz.
RESET_BAR	Input	Asynchronous active LOW reset. When asserted, data output stops and all internal registers are restored to their factory default settings
XSHUTDOWN	Input	Asynchronous active LOW reset. When asserted, data output stops and all internal registers are restored to their factory default settings. This pin will turn off the digital power domain and is the lowest power state of the sensor
SCLK	Input	Serial clock for access to control and status registers
GPI[3:0]	Input	General purpose inputs. After reset, these pads are powered-down by default; this means that it is not necessary to bond to these pads. Any of these pads can be configured to provide hardware control of the standby, output enable, SADDR select, and shutter trigger functions ON Semiconductor recommends that unused GPI pins be tied to DGND, but can also be left floating
TEST	Input	Enable manufacturing test modes. Connect to VDD_IO power for the MIPI-configured sensor
SDATA	I/O	Serial data from reads and writes to control and status registers
VCM_ISINK	I/O	Connected to VCM actuator. 100 mA max. 3.3 V max
VCM_GND	I/O	Connected to DGND
REG_OUT	I/O	1.2 V on-chip regulator output node
REG_IN	I/O	On-chip regulator input node. It needs to be connected to external 1.8 V
REG_FB	I/O	This pad is receiving the 1.2 V feedback from REG_OUT. It needs to be connected to REG_OUT
DATA0_P	Output	Differential MIPI (sub−LVDS) serial data (positive)
DATA0_N	Output	Differential MIPI (sub−LVDS) serial data (negative)
DATA1_P	Output	Differential MIPI (sub−LVDS) serial data 2nd lane (positive) Can be left floating when using one-lane MIPI serial interface
DATA1_N	Output	Differential MIPI (sub−LVDS) serial data second lane (negative) Can be left floating when using one-lane MIPI serial interface
CLK_P	Output	Differential MIPI (sub−LVDS) serial clock/strobe (positive)
CLK_N	Output	Differential MIPI (sub−LVDS) serial clock/strobe (negative)
LINE_VALID	Output	LINE_VALID (LV) output. Qualified by PIXCLK
FRAME_VALID	Output	FRAME_VALID (FV) output. Qualified by PIXCLK
DOUT[9:0]	Output	Parallel pixel data output. Qualified by PIXCLK
PIXCLK	Output	Pixel clock. Used to qualify the LV, FV, and DOUT[9:0] outputs
FLASH	Output	Flash output. Synchronization pulse for external light source. Can be left floating if not used
VPP	Supply	Power supply used to program one-time programmable (OTP) memory
VDD_TX	Supply	Digital PHY power supply. Digital power supply for the serial interface
VAA	Supply	Analog power supply
VAA_PIX	Supply	Analog power supply for the pixel array
AGND	Supply	Analog ground
VDD	Supply	Digital core power supply
VDD_IO	Supply	I/O power supply
DGND	Supply	Common ground for digital and I/O
VDD_PLL	Supply	PLL power supply

OUTPUT DATA FORMAT

Parallel Pixel Data Interface

AR0542 image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking, as shown in Figure 5. The amount of horizontal blanking and vertical blanking is programmable; LV is HIGH during the shaded region of the figure. FV timing is described in the “Output Data Timing (Parallel Pixel Data Interface)”.

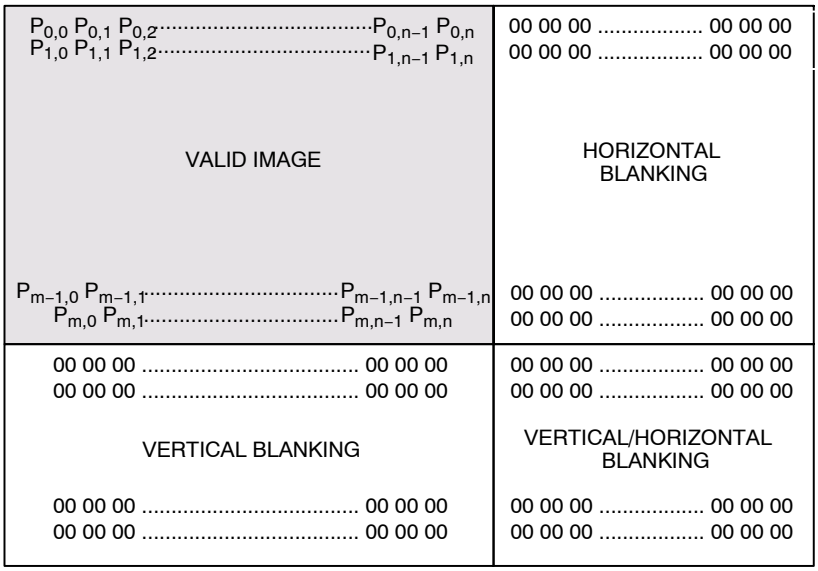


Figure 5. Spatial Illustration of Image Readout

Output Data Timing (Parallel Pixel Data Interface)

AR0542 output data is synchronized with the PIXCLK output. When LV is HIGH, one pixel value is output on the 10−bit DOUT output every PIXCLK period. The pixel clock frequency can be determined based on the sensor’s master input clock and internal PLL configuration. The rising edges on the PIXCLK signal occurs one−half of a pixel clock period after transitions on LV, FV, and DOUT (see Figure 6). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking period. The AR0542 can be programmed to delay the PIXCLK edge relative to the DOUT transitions. This can be achieved by programming the corresponding bits in the row_speed register. The parameters P, A, and Q in Figure 7 are defined in Table 4.

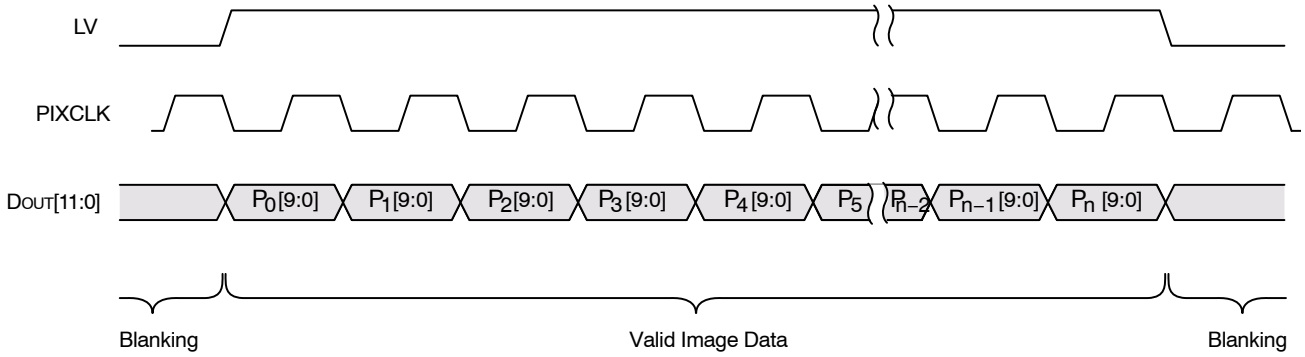


Figure 6. Pixel Data Timing Example

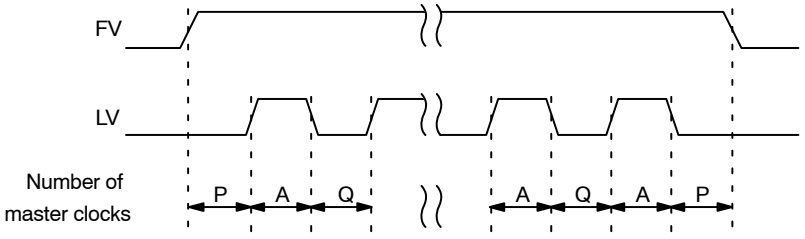


Figure 7. Row Timing and FV/LV Signals

Table 4. ROW TIMING

Parameter	Name	Equation	Default Timing
PIXCLK_PERIOD	Pixel Clock Period	R0x3016–7[2:0] / vt_pix_clk_freq_mhz	1 Pixel Clock = 11.9 ns
S	Skip (Subsampling) Factor	For x_odd_inc = y_odd_inc = 3, S = 2. For x_odd_inc = y_odd_inc = 7, S = 4. Otherwise, S = 1	1
A	Active Data Time	(x_addr_end – x_addr_start + x_odd_inc) * OP_PIX–CLK_PERIOD/S	30.85 µs
P	Frame Start/end Blanking	6 * PIXCLK_PERIOD	6 Pixel Clocks = 71.4 ns
Q	Horizontal Blanking	(line_length_pck * PIXCLK_PERIOD – A)	11.5 µs
A + Q	Row Time	line_length_pck * PIXCLK_PERIOD	42.4 µs
N	Number of Rows	(y_addr_end – y_addr_start + y_odd_inc)/S	1944 Rows
V	Vertical Blanking	((frame_length_lines – N) * (A+Q)) + Q – (2*P)	3.27 ms
T	Frame Valid Time	(N * (A + Q)) – Q + (2*P)	82.33 ms
F	Total Frame Time	line_length_pck * frame_length_lines * PIXCLK_PERIOD	85.60 ms

The sensor timing (Table 4) is shown in terms of pixel clock and master clock cycles (see Figure 6). The settings in Table 4 or the on-chip PLL generate an 84 MHz output pixel clock (op_pix_clk) given a 24-MHz input clock to the AR0542. Equations for calculating the frame rate are given in “Frame Rate Control”.

TWO-WIRE SERIAL REGISTER INTERFACE

The two-wire serial interface bus enables read/write access to control and status registers within the AR0542. This interface is designed to be compatible with the electrical characteristics and transfer protocols of the two-wire serial register interface specification.

The interface protocol uses a master/slave model in which a master controls one or more slave devices. The sensor acts as a slave device. The master generates a clock (SCLK) that is an input to the sensor and is used to synchronize transfers. Data is transferred between the master and the slave on a bidirectional signal (SDATA). SDATA is pulled up to VDD_IO off-chip by a 1.5 kΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The protocols described in the two-wire serial interface specification allow the slave device to drive SCLK LOW; the AR0542 uses SCLK as an input only and therefore never drives it LOW.

Protocol

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

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

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

Start Condition

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

Stop Condition

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

Data Transfer

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

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

Slave Address/Data Direction Byte

Bits [7:1] of this byte represent the device slave address and bit [0] indicates the data transfer direction. A "0" in bit [0] indicates a WRITE, and a "1" indicates a READ. The default slave addresses used by the AR0542 for the MIPI configured sensor are 0x6C (write address) and 0x6D (read address) in accordance with the MIPI specification. Alternate slave addresses of 0x6E (write address) and 0x6F (read address) can be selected by enabling and asserting the SADDR signal through the GPI pad.

An alternate slave address can also be programmed through R0x31FC.

Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data.

Acknowledge Bit

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

No-Acknowledge Bit

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

Typical Sequence

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

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

If the request was a READ, the master sends the 8-bit write slave address/data direction byte and 16-bit register address, the same way as with a WRITE request. The master then generates a (re)start condition and the 8-bit read slave address/data direction byte, and clocks out the register data, eight bits at a time. The master generates an acknowledge bit after each 8-bit transfer. The slave's internal register address

is automatically incremented after every 8 bits are transferred. The data transfer is stopped when the master sends a no-acknowledge bit.

Single READ from Random Location

This sequence (Figure 8) starts with a dummy WRITE to the 16-bit address that is to be used for the READ. The master terminates the WRITE by generating a restart condition. The master then sends the 8-bit read slave address/data direction byte and clocks out one byte of register data. The master terminates the READ by generating a no-acknowledge bit followed by a stop condition. Figure 8 shows how the internal register address maintained by the AR0542 is loaded and incremented as the sequence proceeds.

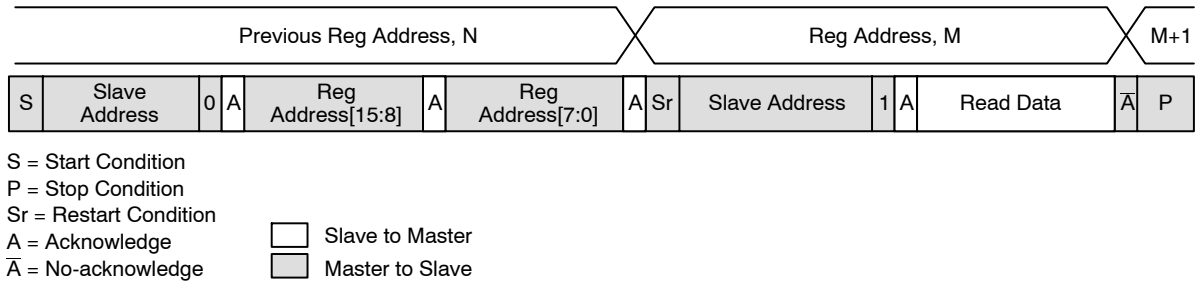


Figure 8. Single READ from Random Location

Single READ from Current Location

This sequence (Figure 9) performs a read using the current value of the AR0542 internal register address. The master terminates the READ by generating a no-acknowledge bit followed by a stop condition. The figure shows two independent READ sequences.

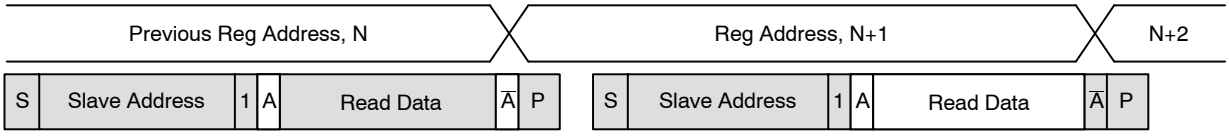


Figure 9. Single READ from Current Location

Sequential READ, Start from Random Location

This sequence (Figure 10) starts in the same way as the single READ from random location (Figure 8). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until “L” bytes have been read.

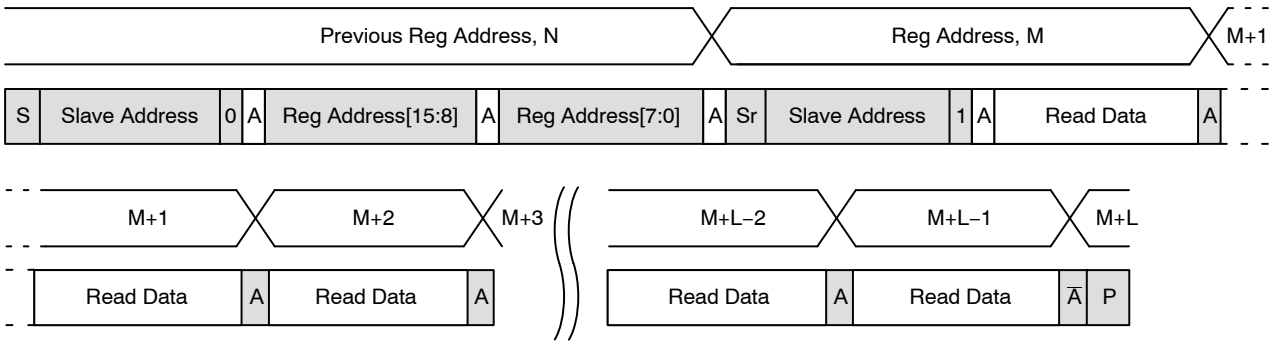


Figure 10. Sequential READ, Start from Random Location

Sequential READ, Start from Current Location

This sequence (Figure 11) starts in the same way as the single READ from current location (Figure 9). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until “L” bytes have been read.

Figure 11. Sequential READ, Start from Current Location

Single WRITE to Random Location

This sequence (Figure 12) begins with the master generating a start condition. The slave address/data direction byte signals a WRITE and is followed by the HIGH then LOW bytes of the register address that is to be written. The master follows this with the byte of write data. The WRITE is terminated by the master generating a stop condition.

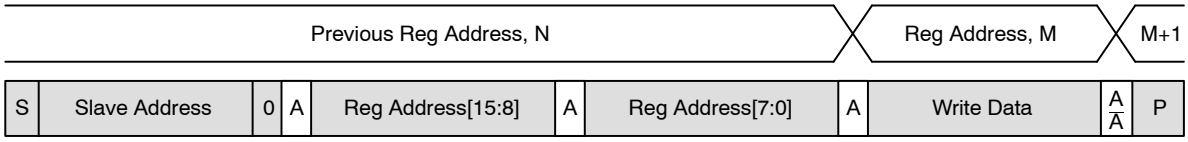


Figure 12. Single WRITE to Random Location

Sequential WRITE, Start at Random Location

This sequence (Figure 13) starts in the same way as the single WRITE to random location (Figure 12). Instead of generating a no-acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte WRITEs until “L” bytes have been written. The WRITE is terminated by the master generating a stop condition.

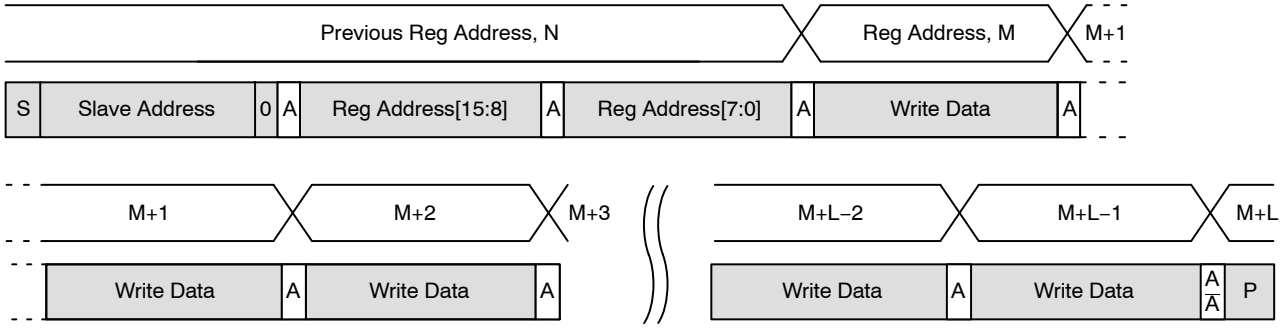


Figure 13. Sequential WRITE, Start at Random Location

REGISTERS

The AR0542 provides a 16-bit register address space accessed through a serial interface (“Two-Wire Serial Register Interface”). See the AR0542 Register Reference for details.

PROGRAMMING RESTRICTIONS

Table 6 shows a list of programming rules that must be adhered to for correct operation of the AR0542. It is recommended that these rules are encoded into the device driver stack-either implicitly or explicitly.

Table 5. DEFINITIONS FOR PROGRAMMING RULES

Name	Definition
xskip	xskip = 1 if x_odd_inc = 1; xskip = 2 if x_odd_inc = 3; xskip = 4 if x_odd_inc = 7
yskip	yskip = 1 if y_odd_inc = 1; yskip = 2 if y_odd_inc = 3; yskip = 4 if y_odd_inc = 7

Table 6. PROGRAMMING RULES

Parameter	Minimum Value	Maximum Value
coarse_integration_time	4 (8 is recommended)	frame_length_lines – coarse_integration_time_max_margin
fine_integration_time	fine_integration_time_min	line_length_pck – fine_integration_time_max_margin
digital_gain_* digital_gain_* is an integer multiple of digital_gain_step_size	digital_gain_min	digital_gain_max
frame_length_lines	min_frame_length_lines	max_frame_length_lines
line_length_pck	min_line_length_pck	max_line_length_pck
	((x_addr_end – x_addr_start + x_odd_inc)/xskip) + min_line_blanking_pck	
frame_length_lines	((y_addr_end – y_addr_start + y_odd_inc)/yskip) + min_frame_blanking_lines	
x_addr_start (must be an even number)	x_addr_min	x_addr_max
x_addr_end (must be an odd number)	x_addr_start	x_addr_max
(x_addr_end – x_addr_start + x_odd_inc)	must be positive	must be positive
y_addr_start (must be an even number)	y_addr_min	y_addr_max
y_addr_end (must be an odd number)	y_addr_start	y_addr_max
(y_addr_end – y_addr_start + y_odd_inc)	must be positive	must be positive
x_even_inc (must be an even number)	min_even_inc	max_even_inc
y_even_inc (must be an even number)	min_even_inc	max_even_inc
x_odd_inc (must be an odd number)	min_odd_inc	max_odd_inc
y_odd_inc (must be an odd number)	min_odd_inc	max_odd_inc
scale_m	scaler_m_min	scaler_m_max
scale_n	scaler_n_min	scaler_n_max
x_output_size (must be even number – this is enforced in hardware)	256	2608

Table 6. PROGRAMMING RULES (continued)

Parameter	Minimum Value	Maximum Value
y_output_size (must be even number – this is enforced in hardware)	2	frame_length_lines
With subsampling, start and end pixels must be addressed (impact on x/y start/end addresses, function of image orientation bits)		

Output Size Restrictions

When the parallel pixel data path is in use, the only restriction on x_output_size is that it must be even (x_output_size[0] = 0), and this restriction is enforced in hardware.

When the serial pixel data path is in use, there is an additional restriction that x_output_size must be small enough such that the output row time (set by x_output_size, the framing and CRC overhead of 12 bytes and the output clock rate) must be less than the row time of the video array (set by line_length_pck and the video timing clock rate).

Effect of Scaler on Legal Range of Output Sizes

When the scaler is enabled, it is necessary to adjust the values of x_output_size and y_output_size to match the image size generated by the scaler. The AR0542 will operate incorrectly if the x_output_size and y_output_size are significantly larger than the output image.

To understand the reason for this, consider the situation where the sensor is operating at full resolution and the scaler is enabled with a scaling factor of 32 (half the number of pixels in each direction). This situation is shown in Figure 14.

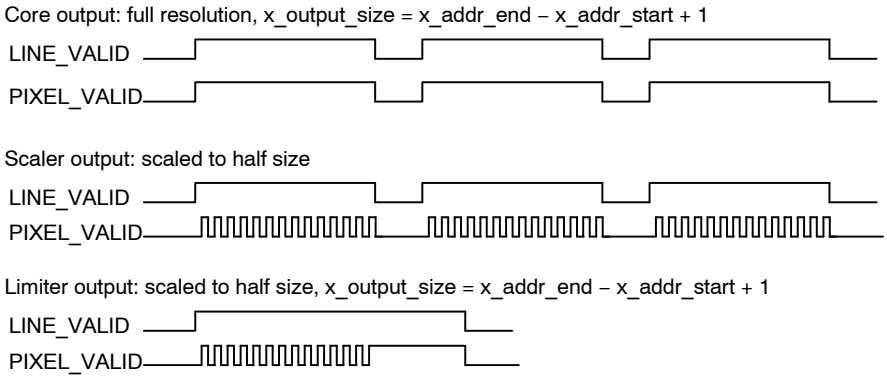


Figure 14. Effect of Limiter on the Data Path

In Figure 14, three different stages in the data path (see "Digital Data Path") are shown. The first stage is the output of the sensor core. The core is running at full resolution and x_output_size is set to match the active array size. The LINE_VALID signal is asserted once per row and remains asserted for *N* pixel times. The PIXEL_VALID signal toggles with the same timing as LINE_VALID, indicating that all pixels in the row are valid.

The second stage is the output of the scaler, when the scaler is set to reduce the image size by one-half in each dimension. The effect of the scaler is to combine groups of pixels. Therefore, the row time remains the same, but only half the pixels out of the scaler are valid. This is signaled by transitions in PIXEL_VALID. Overall, PIXEL_VALID is asserted for (*N*/2) pixel times per row.

The third stage is the output of the limiter when the x_output_size is still set to match the active array size. Because the scaler has reduced the amount of valid pixel data without reducing the row time, the limiter attempts to pad the row with (*N*/2) additional pixels. If this has the effect of extending LV across the whole of the horizontal blanking time, the AR0542 will cease to generate output frames.

A correct configuration is shown in Figure 15, in addition to showing the x_output_size reduced to match the output size of the scaler. In this configuration, the output of the limiter does not extend LV.

Figure 15 also shows the effect of the output FIFO, which forms the final stage in the data path. The output FIFO merges the intermittent pixel data back into a contiguous stream. Although not shown in this example, the output FIFO is also capable of operating with an output clock that is at a different frequency from its input clock.

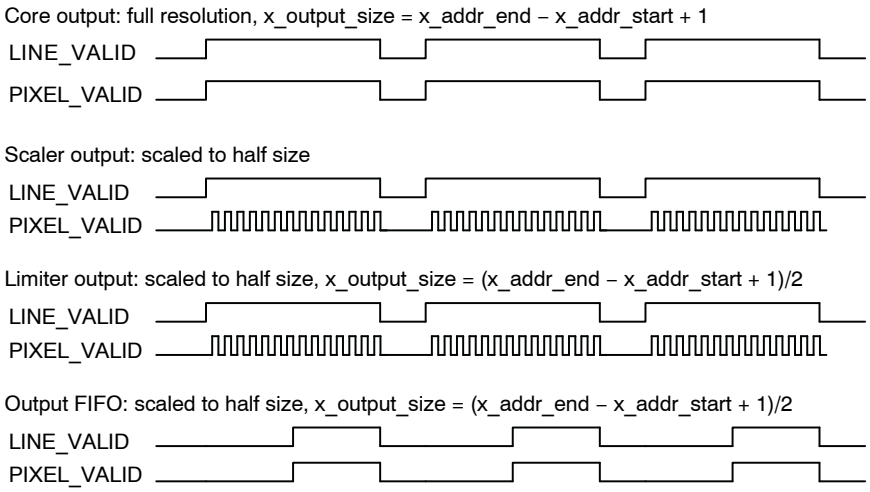


Figure 15. Timing of Data Path

Output Data Timing

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

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

A key constraint when configuring the clock for the output FIFO is that the frame rate out of the FIFO must exactly match the frame rate into the FIFO. When the scaler is disabled, this constraint can be met by imposing the rule that the row time on the serial data stream must be greater than or equal to the row time at the pixel array. The row time on the serial data stream is calculated from the x_output_size and the data_format (8 or 10 bits per pixel), and must include the time taken in the serial data stream for start of frame/row, end of row/frame and checksum symbols.

CAUTION: If this constraint is not met, the FIFO will either underrun or overrun. FIFO underrun or overrun is a fatal error condition that is signaled through the data path_status register (R0x306A).

Changing Registers while Streaming

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

- ccp_channel_identifier
- ccp_data_format
- ccp_signaling_mode
- vt_pix_clk_div
- vt_sys_clk_div
- pre_pll_clk_div
- pll_multiplier
- op_pix_clk_div
- op_sys_clk_div
- scale_m

Programming Restrictions When Using Global Reset

Interactions between the registers that control the global reset imposes some programming restrictions on the way in which they are used; these are discussed in ”Analog Gain”.

CONTROL OF THE SIGNAL INTERFACE

This section describes the operation of the signal interface in all functional modes.

Serial Register Interface

The serial register interface uses these signals:

- SCLK
- SDATA
- SADDR (through the GPI pad)

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

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

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

This interface is described in detail in ”Two-Wire Serial Register Interface”.

Default Power-Up State

The AR0542 sensor can provide two separate interfaces for pixel data: the MIPI serial interface and a parallel data interface.

At powerup and after a hard or soft reset, the reset state of the sensor is to enable serial interface when available.

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

- DATA0_P
- DATA0_N
- CLK_P
- CLK_N

The signal pairs are driven differentially using sub-LVDS switching levels. The serial pixel data interface is enabled by default at power up and after reset.

The DATA0_P, DATA0_N, CLK_P, and CLK_N pads are turned off if the SMIA serial disable bit is asserted (R0x301A-B[12]=1) or when the sensor is in the soft standby state.

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

When the serial pixel data interface is used, the LINE_VALID, FRAME_VALID, PIXCLK and DOUT[9:0] signals (if present) can be left unconnected.

MIPI Serial Pixel Data Interface

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

- DATA0_P
- DATA0_N
- DATA1_P
- DATA1_N
- CLK_P
- CLK_N

The signal pairs use both single-ended and differential signaling, in accordance with the MIPI specification. The serial pixel data interface is enabled by default at power up and after reset.

The DATA0_P, DATA0_N, DATA1_P, DATA1_N, CLK_P and CLK_N pads are set to the Ultra Low Power State (ULPS) if the SMIA serial disable bit is asserted (R0x301A-B[12]=1) or when the sensor is in the hardware standby or soft standby system states.

When the serial pixel data interface is used, the LINE_VALID, FRAME_VALID, PIXCLK and DOUT[9:0] signals (if present) can be left unconnected.

The ccp_data_format (R0x0112-3) register can be programmed to any of the following data format settings that are supported:

- 0x0A0A – Sensor supports RAW10 uncompressed data format. This mode is supported by discarding all but the upper 10 bits of a pixel value.
- 0x0808 – Sensor supports RAW8 uncompressed data format. This mode is supported by discarding all but the upper 8 bits of a pixel value.
- 0x0A08 – Sensor supports RAW8 data format in which an adaptive compression algorithm is used to perform 10-bit to 8-bit compression on the upper 10 bits of each pixel value

The serial_format register (R0x31AE) register controls which serial interface is in use when the serial interface is enabled (reset_register[12] = 0). The following serial formats are supported:

- 0x0201 – Sensor supports single-lane MIPI operation
- 0x0202 – Sensor supports dual-lane MIPI operation

Parallel Pixel Data Interface

The parallel pixel data interface uses these output-only signals:

- FV
- LV
- PIXCLK
- DOUT[9:0]

The parallel pixel data interface is disabled by default at power up and after reset. It can be enabled by programming R0x301A. Table 8 shows the recommended settings.

When the parallel pixel data interface is in use, the serial data output signals (DATA0_P, DATA0_N, DATA1_P, DATA1_N, CLK_P, and CLK_N) can be left unconnected.

Set reset_register[12] to disable the serializer while in parallel output mode.

To use the parallel interface, the V_{DD}_TX pad must be tied to a 1.8 V supply. For MIPI sensor, the VDD_IO supply can be set at 1.8 V or 2.8 V (nominal).

Output Enable Control

When the parallel pixel data interface is enabled, its signals can be switched asynchronously between the driven and High-Z under pin or register control, as shown in Table 7. Selection of a pin to use for the OE_N function is described in ”General Purpose Inputs”.

Table 7. OUTPUT ENABLE CONTROL

OE_N Pin	Drive Signals R0x301A–B[6]	Description
Disabled	0	Interface High-Z
Disabled	1	Interface Driven
1	0	Interface High-Z
X	1	Interface Driven
0	X	Interface Driven

Configuration of the Pixel Data Interface

Fields in R0x301A are used to configure the operation of the pixel data interface. The supported combinations are shown in Table 8.

Table 8. CONFIGURATION OF THE PIXEL DATA INTERFACE

Serializer Disable R0x301 A–B[12]	Parallel Enable R0x301A–B[7]	Standby End-of-Frame R0x301A–B[4]	Description
0	0	1	Power up default Serial pixel data interface and its clocks are enabled. Transitions to soft standby are synchronized to the end of frames on the serial pixel data interface
1	1	0	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of the current row readout on the parallel pixel data interface
1	1	1	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of frames in the parallel pixel data interface

System States

The system states of the AR0542 are represented as a state diagram in Figure 16 and described in subsequent sections. The effect of RESET_BAR on the system state and the configuration of the PLL in the different states are shown in Table 9.

The sensor's operation is broken down into three separate states: hardware standby, software standby, and streaming. The transition between these states might take a certain amount of clock cycles as outlined in Table 9.

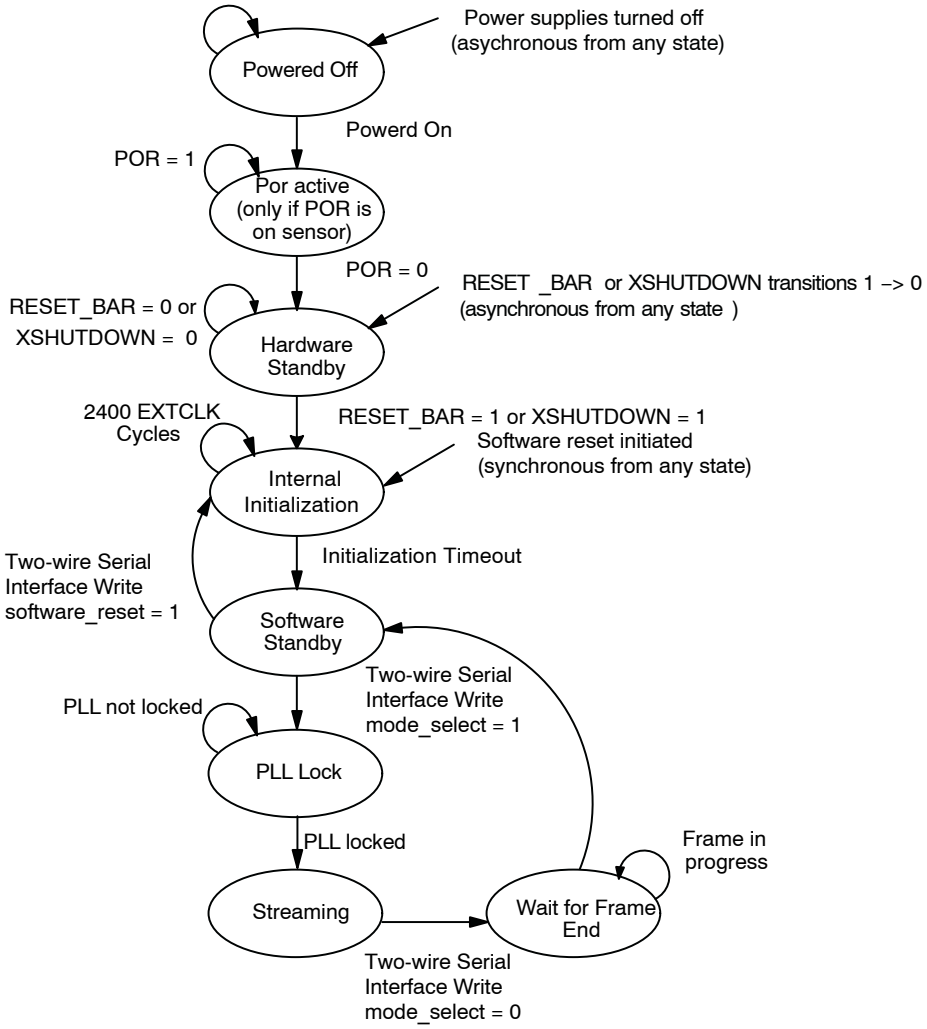


Figure 16. AR0542 System States

Table 9. XSHUTDOWN AND PLL IN SYSTEM STATES

State	XSHUTDOWN	PLL
Powered Off	x	VCO powered down
POR Active	x	
Hardware Standby	0	
Internal Initialization	1	
Software Standby		
PLL Lock		VCO powering up and locking, PLL output bypassed
Streaming		VCO running, PLL output active
Wait for Frame End		

Power-On Reset Sequence

When power is applied to the AR0542, it enters a low-power hardware standby state. Exit from this state is controlled by the later of two events:

- The negation of the XSHUTDOWN input.
- A timeout of the internal power-on reset circuit.

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

While XSHUTDOWN is asserted (or the internal power-on reset circuit is active) the AR0542 is in its lowest-powered, powered-up state; the internal PLL is disabled, the serializer is disabled and internal clocks are gated off.

When the sensor leaves the hardware standby state it performs an internal initialization sequence that takes 2400 EXTCLK cycles. After this, it enters a low-power software standby state. While the initialization sequence is in progress, the AR0542 will not respond to read transactions on its two-wire serial interface. Therefore, a method to determine when the initialization sequence has completed is to poll a sensor register; for example, R0x0000. While the initialization sequence is in progress, the sensor will not respond to its device address and reads from the sensor will result in a NACK on the two-wire serial interface bus. When the sequence has completed, reads will return the operational value for the register (0x4800 if R0x0000 is read).

When the sensor leaves software standby mode and enables the VCO, an internal delay will keep the PLL disconnected for up to 1ms so that the PLL can lock. The VCO lock time is 200 µs (typical), 1ms (maximum).

Soft Reset Sequence

The AR0542 can be reset under software control by writing “1” to software_reset (R0x0103). A software reset asynchronously resets the sensor, truncating any frame that is in progress. The sensor starts the internal initialization sequence, while the PLL and analog blocks are turned off. At this point, the behavior is exactly the same as for the power-on reset sequence.

Signal State During Reset

Table 10 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during software standby (after exit from hardware standby and before any registers within the sensor have been changed from their default power-up values).

Table 10. SIGNAL STATE DURING RESET

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Enabled. Must be driven to a valid logic level	
XSHUTDOWN/RESET_BAR	Input	Enabled. Must be driven to a valid logic level	
LINE_VALID	Output	High-Z. Can be left disconnected/floating	
FRAME_VALID	Output		
DOUT[9:0]	Output		
PIXCLK	Output		
SCLK	Input	Enabled. Must be pulled up or driven to a valid logic level	
SDATA	I/O	Enabled as an input. Must be pulled up or driven to a valid logic level	
FLASH	Output	High-Z	Logic 0
DATA0_P	Output	MIPI: Ultra Low-Power State (ULPS), represented as an LP-00 state on the wire (both wires at 0 V)	
DATA0_N	Output		
DATA1_P	Output		
DATA1_N	Output		
CLK_P	Output		
CLK_N	Output		
GPI[3:0]	Input	Powered down. Can be left disconnected/floating	
TEST	Input	Enabled. Must be driven to a logic 1 for a serial MIPI-configured sensor	

General Purpose Inputs

The AR0542 provides four general purpose inputs. After reset, the input pads associated with these signals are powered down by default, allowing the pads to be left disconnected/floating.

The general purpose inputs are enabled by setting reset_register[8] (R0x301A). 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 through gpi_status[3:0] (R0x3026).

In addition, each of the following functions can be associated with none, one, or more of the general purpose inputs so that the function can be directly controlled by a hardware input:

- Output enable (see “Output Enable Control”)
- Trigger (see the sections below)
- Standby functions
- SADDR selection (see “Serial Register Interface”)

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

Streaming/Standby Control

The AR0542 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 11. Selection of a pin to use for the STANDBY function is described in “General Purpose Inputs”. The state diagram for transitions between soft standby and streaming states is shown in Figure 16.

Table 11. STREAMING/STANDBY

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

CLOCKING

The AR0542 contains a PLL for timing generation and control. The PLL contains a prescaler to divide the input clock applied on EXTCLK, a VCO to multiply the prescaler output, and a set of dividers to generate the output clocks.

Both SMIA profile 0 and profile 1/2 clock schemes are supported. Sensor profile level represents an increasing level of data rate reduction for video applications, for example, viewfinder in full resolution. The clocking scheme can be selected by setting R0x306E–F[7] to 0 for profile 0 or to 1 for profile 1/2.

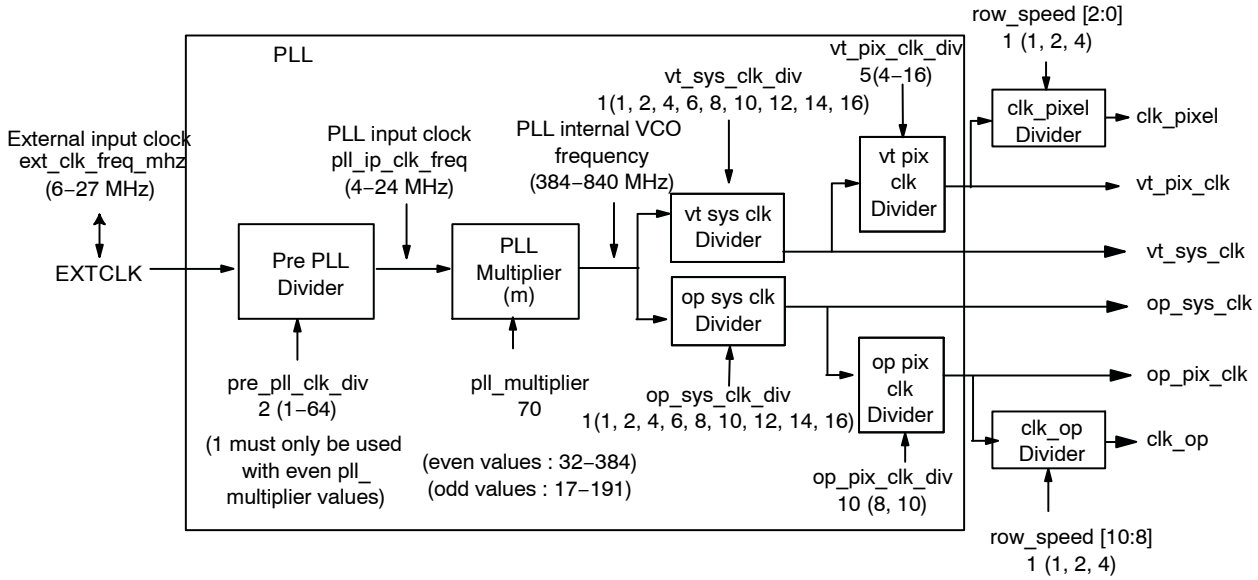


Figure 17. AR0542 Profile 1/2 Clocking Structure

Figure 17 shows the different clocks and the names of the registers that contain or are used to control their values. Also shown is the default setting for each divider/multipler

control register and the range of legal values for each divider/multiplier control register.

The parameter limit register space contains registers that declare the minimum and maximum allowable values for:

- The frequency allowable on each clock
- The divisors that are used to control each clock

These factors determine what are valid values, or combinations of valid values, for the divider/multiplier control registers:

- The minimum/maximum frequency limits for the associated clock must be met pll_ip_clk_freq must be in the range 4–24 MHz. Higher frequencies are preferred. PLL internal VCO frequency must be in the range 384–840 MHz.
- The minimum/maximum value for the divider/multiplier must be met.
  Range for m: 17 –384. (In addition odd values between 17–191 and even values between 32–384 are accepted.)
  Range for n: 0–63. Range for (n+1): 1–64.
- clk_op must never run faster than the clk_pixel to ensure that the output data stream is contiguous.
- Given the maximum programmed line length, the minimum blanking time, the maximum image width, the available PLL divisor/multiplier values, and the requirement that the output line time (including the necessary blanking) must be output in a time equal to or less than the time defined by line_length_pck.

Although the PLL VCO input frequency range is advertised as 4–24 MHz, superior performance is obtained by keeping the VCO input frequency as high as possible.

The usage of the output clocks is shown below:

- clk_pixel (vt_pix_clk / row_speed[2:0]) is used by the sensor core to readout and control the timing of the pixel array. The sensor core produces one 10-bit pixel each vt_pix_clk period. The line length (line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of the vt_pix_clk period.
- clk_op (op_pix_clk / row_speed[10:8]) is used to load parallel pixel data from the output FIFO (see Figure 35 on page 40) to the serializer. The output FIFO generates one pixel each op_pix_clk period. The pixel is either 8-bit or 10-bit, depending upon the output data format, controlled by R0x0112–3 (ccpdata_format).
- op_sys_clk is used to generate the serial data stream on the output. The relationship between this clock frequency and the op_pix_clk frequency is dependent upon the output data format.

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

$$clk\_pix\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier \times clk\_pixel\_divN}{pre\_pll\_clk\_div \times vt\_sys\_clk\_div \times vt\_pix\_clk\_div \times row\_speed[2:0]} \quad \text{(eq. 1)}$$

$$clk\_op\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times op\_sys\_clk\_div \times op\_pix\_clk\_div \times row\_speed[10:8]} \quad \text{(eq. 2)}$$

$$op\_sys\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times op\_sys\_clk\_div} \quad \text{(eq. 3)}$$

NOTE: For dual-lane MIPI interface, clk_pixel_divN = 1. For other interfaces (parallel and single-lane MIPI), clk_pixel_divN = 2.

In Profile 0, RAW10 data format is required. As a result, op_pix_clk_div should be set to 10. Also, due to the inherent design of the AR0542 sensor, vt_pix_clk_div should be set to 5 for profile 0 mode.

PLL Clocking

The PLL divisors should be programmed while the AR0542 is in the software standby state. After programming the divisors, it is necessary to wait for the VCO lock time before enabling the PLL. The PLL is enabled by entering the streaming state.

An external timer will need to delay the entrance of the streaming mode by 1 millisecond so that the PLL can lock.

The effect of programming the PLL divisors while the AR0542 is in the streaming state is undefined.

Influence of ccp_data_format

R0x0112–3 (ccp_data_format) controls whether the pixel data interface will generate 10 or 8 bits per pixel.

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

Influence of ccp2_signalling_mode

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

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

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

This behavior is a result of the implementation of the CCP serializer and the PLL. When the serializer is using data and strobe signaling, it uses both edges of the op_sys_clk, and therefore that clock runs at one half of the bit rate. All of the programmed divisors are set up to make this behavior invisible. For example, when the divisors are programmed

to generate a PLL output of 640 MHz, the actual PLL output is 320 MHz, but both edges are used.

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

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

FEATURES

Shading Correction (SC)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing fixed pattern signal gradients in images captured by image sensors. The cumulative result of all these factors is known as image shading. The AR0542 has an embedded shading correction module that can be programmed to counter the shading effects on each individual Red, GreenB, GreenR, and Blue color signal.

The Correction Function

Color-dependent solutions are calibrated using the sensor, lens system and an image of an evenly illuminated, featureless gray calibration field. From the resulting image, register values for the color correction function (coefficients) can be derived.

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

$$Pcorrected(row, col) = Psensor(row, col) \times f(row, col) \quad \text{(eq. 4)}$$

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

Each function includes a set of color-dependent coefficients defined by registers R0x3600–3726. The function's origin is the center point of the function used in the calculation of the coefficients. Using an origin near the central point of symmetry of the sensor response provides the best results. The center point of the function is determined by ORIGIN_C (R0x3782) and ORIGIN_R (R0x3784) and can be used to counter an offset in the system lens from the center of the sensor array.

One-Time Programmable Memory (OTPM)

The AR0542 features 7.7 Kb of one-time programmable memory (OTPM) for storing shading correction coefficients, individual module ID, and sensor specific information. It takes roughly 5 Kb (102 registers x 16-bits x 3 sets = 4896 bits) to store three sets of illumination-dependent shading coefficients. The OTPM array has a total of 201 accessible row-addresses, with each row having two 20-bit words per row. In each word, 16 bits are used for data storage, while the remaining 4 bits are used by the error detection and correction scheme. OTP memory can be accessed through two-wire serial interface. The AR0542 uses the auto mode for fast OTPM programming and read operations.

Clock Control

The AR0542 uses an aggressive clock-gating methodology to reduce power consumption. The clocked logic is divided into a number of separate domains, each of which is only clocked when required.

When the AR0542 enters a low-power state, almost all of the internal clocks are stopped. The only exception is that a small amount of logic is clocked so that the two-wire serial interface continues to respond to read and write requests.

During the programming process, a dedicated high voltage pin (V_{PP}) needs to be supplied with a 6.5 V +3% voltage to perform the anti-fusing operation, and a slew rate of 1 V/µs or slower is recommended for V_{PP} supply. Instantaneous V_{PP} cannot exceed 9 V at any time. The completion of the programming process will be communicated by a register through the two-wire serial interface.

Because this programming pin needs to sustain a higher voltage than other input/output pins, having a dedicated high voltage pin (V_{PP}) minimizes the design risk. If the module manufacturing process can probe the sensor at the die or PCB level (that is, supply all the power rails, clocks, and two-wire serial interface signals), then this dedicated high voltage pin does not need to be assigned to the module connector pinout. However, if the V_{PP} pin needs to be bonded out as a pin on the module, the trace for V_{PP} needs to carry a maximum of 1 mA – for programming only. This pin should be left floating once the module is integrated to a design. If the V_{PP} pin does not need to be bonded-out as a pin on the module, it should be left floating inside the module.

The programming of the OTPM requires the sensor to be fully powered and remain in software standby with its clock input applied. The information will be programmed through the use of the two-wire serial interface, and once the data is written to an internal register, the programming host machine will apply a high voltage to the programming pin, and send a program command to initiate the anti-fusing process. After the sensor has finished programming the OTPM, a status bit will be set to indicate the end of the programming cycle, and the host machine can poll the setting of the status bit through the two-wire serial interface. Only one programming cycle for the 16-bit word can be performed.

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.

Programming the OTPM

Program the AR0542 OTPM as follows:

1. Apply power to all the power rails of the sensor (V_{DD}_IO, V_{AA}, V_{AA}_PIX, and Digital 1.8 V).

– On Semiconductor recommends setting V_{AA} to 3.1 V during the programming process. All other supplies must be at their nominal voltage.
– Ensure that the V_{PP} pin is floating during sensor power-up.

2. Provide an EXTCLK clock input (12 MHz is recommended).
3. Set R0x301A = 0x10D8, to put sensor in the soft standby mode.
4. Set R0x3064[9] =1 to bypass PLL.
5. Set R0x3054[8]=1
6. Write data (102 words for one set of LSC coefficients) into the OTPM data registers (R0x3800–R0x38CA for one set of LSC coefficients).
7. Set OTPM start address register R0x3050[15:8] = 0 to program the array with the first batch of data.

NOTE: When programming the second batch of data, set the start address to 128 (considering that all the previous 0–127 locations are already written to by the data registers 0–255), otherwise the start address should be set accordingly.

8. Set R0x3054[9] = 0 to ensure that the error checking and correction is enabled.
9. Set the length register (R0x304C [7:0]) accordingly, depending on the number of OTM data registers that are filled in (0x66 for 102 words). It may take about 500 ms for one set of LSC (102 words).
10. Set R0x3052 = 0x2504 (OTPM_CONFIG)
11. Ramp up V_{PP} to 6.5 V. The recommended slew rate for V_{PP} is 1 V/µs or slower.
12. Set the otpm_control_auto_wr_start bit in the otpm_manual_control register R0x304A[0] = 1, to initiate the auto program sequence. The sensor will now program the data into the OTPM starting with the location specified by the start address.
13. Poll OTPM_Control_Auto_WR_end (R0x304A [1]) to determine when the sensor is finished programming the word.
14. Repeat steps 13 and 14.
15. Remove the high voltage (V_{PP}) and float the V_{PP} pin.

Reading the OTPM

Read the AR0542 OTPM as follows:

1. Perform the proper reset sequence to the sensor by setting R0x0103 = 1.
2. Set OTPM_CONFIG register R0x3052 = 0x2704.
3. Set R0x3054[8] = 1.
4. Program R0x3050[15:8] with the appropriate value to specify the start address (0x0 for address 0).
5. Program R0x304C [7:0] with the appropriate value to specify the length (number of data registers to be read back, starting from the specified start address – 0x66 for 102 words).
6. Initiate the auto read sequence by setting the otpm_control_auto_read_start bit R0x304A[4] = 1.
7. Poll the otpm_control_auto_rd_end bit (R0x304A[5]) to determine when the sensor is finished reading the word(s).
   Data can now be read back from the otpm_data registers (R0x3800–R0x39FE).
8. Verify that the read data from the OTPM_DATA registers are the expected data.

Image Acquisition Mode

The AR0542 supports the electronic rolling shutter (ERS) mode. This is the normal mode of operation. When the AR0542 is streaming, it generates frames at a fixed rate, and each frame is integrated (exposed) using the ERS. When the ERS is in use, timing and control logic within the sensor sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and subsequently reading that row, the pixels in the row integrate incident light. The integration (exposure) time is controlled by varying the time between row reset and row readout. For each row in a frame, the time between row reset and row readout is fixed, leading to a uniform integration time across the frame. When the integration time is changed (by using the two-wire serial interface to change register settings), the timing and control logic controls the transition from old to new integration time in such a way that the stream of output frames from the AR0542 switches cleanly from the old integration time to the new while only generating frames with uniform integration. See “Changes to Integration time” in the AR0542 Register Reference.

Window Control

The sequencing of the pixel array is controlled by the x_addr_start, y_addr_start, x_addr_end, and y_addr_end registers. For both parallel and serial MIPI interfaces, the output image size is controlled by the x_output_size and y_output_size registers.

Pixel Border

The default settings of the sensor provide a 2592 (H) x 1944 (V) image. A border of up to 8 pixels (4 in binning) on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end, y_addr_end, x_output_size, and y_output_size registers accordingly.

Readout Modes

Horizontal Mirror

When the horizontal_mirror bit is set in the image_orientation register, the order of pixel readout within

a row is reversed, so that readout starts from x_addr_end and ends at x_addr_start. Figure 18 shows a sequence of 6 pixels being read out with horizontal_mirror = 0 and horizontal_mirror = 1. Changing horizontal_mirror causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.

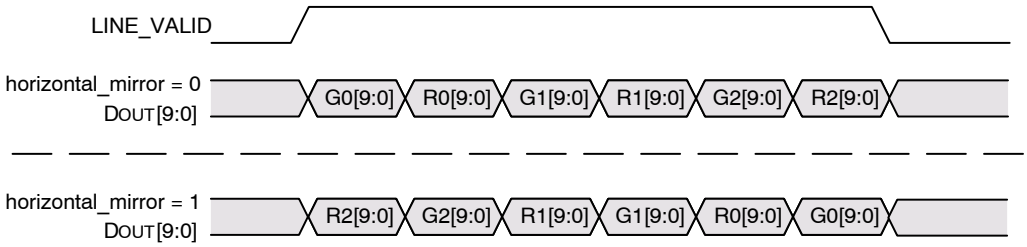


Figure 18. Effect of horizontal_mirror on Readout Order

Vertical Flip

When the vertical_flip bit is set in the image_orientation register, the order in which pixel rows are read out is reversed, so that row readout starts from y_addr_end and ends at y_addr_start. Figure 19 shows a sequence of 6 rows being read out with vertical_flip = 0 and vertical_flip = 1. Changing vertical_flip causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.

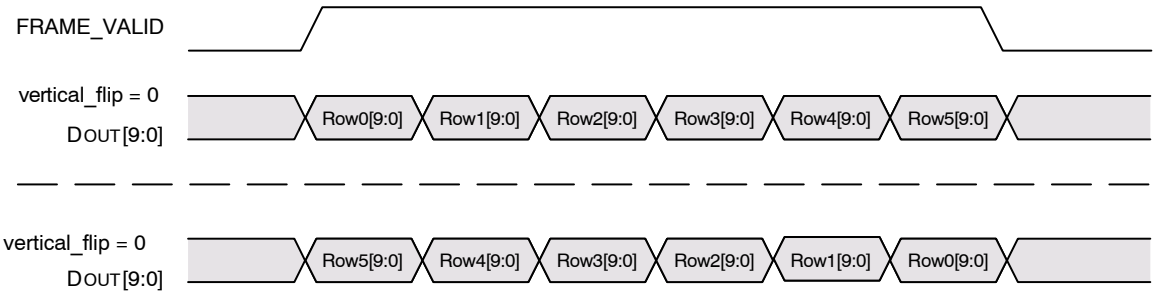


Figure 19. Effect of vertical_flip on Readout Order

Subsampling

The AR0542 supports subsampling. Subsampling reduces the amount of data processed by the analog signal chain in the AR0542 thereby allowing the frame rate to be increased. Subsampling is enabled by setting x_odd_inc and/or y_odd_inc. Values of 1, 3, and 7 can be supported. Setting both of these variables to 3 reduces the amount of row and column data processed and is equivalent to the 2 x 2 skipping readout mode provided by the AR0542. Setting x_odd_inc = 3 and y_odd_inc = 3 results in a quarter reduction in output image size. Figure 20 shows a sequence of 8 columns being read out with x_odd_inc = 3 and y_odd_inc = 1.

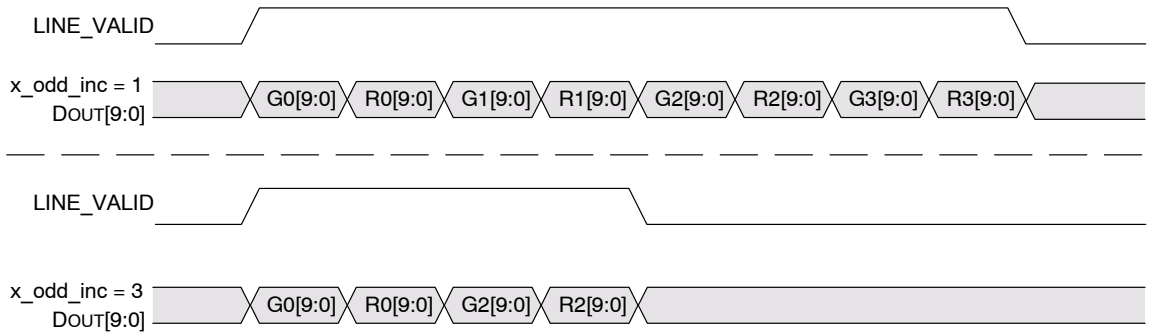


Figure 20. Effect of x_odd_inc = 3 on Readout Sequence

A 1/16 reduction in resolution is achieved by setting both x_odd_inc and y_odd_inc to 7. This is equivalent to 4 x 4 skipping readout mode provided by the AR0542. Figure 21

shows a sequence of 16 columns being read out with x_odd_inc = 7 and y_odd_inc = 1.

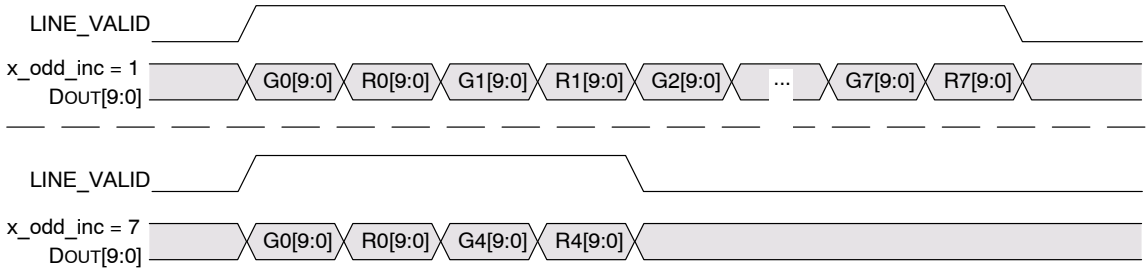


Figure 21. Effect of x_odd_inc = 7 on Readout Sequence

The effect of the different subsampling settings on the pixel array readout is shown in Figure 22 through Figure 24.

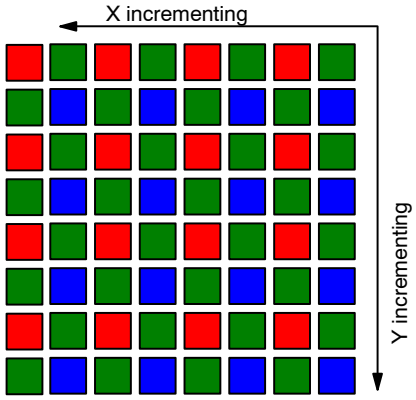


Figure 22. Pixel Readout (No Subsampling)

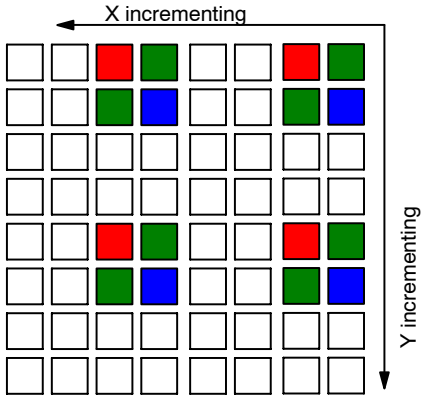


Figure 23. Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)

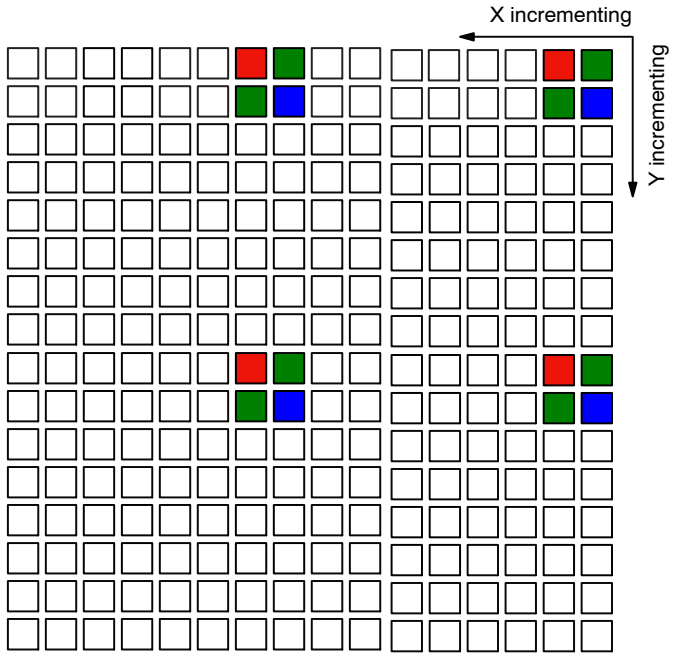


Figure 24. Pixel Readout (x_odd_inc = 7, y_odd_inc = 7)