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

AR0542

ORDERING INFORMATION

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

General Description

The **onsemi** 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 **onsemi**'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)

*For additional information on our Pb−Free strategy and soldering details, please download the **onsemi** Soldering and Mounting Techniques Reference Manual, SOLDERRM/D.

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.

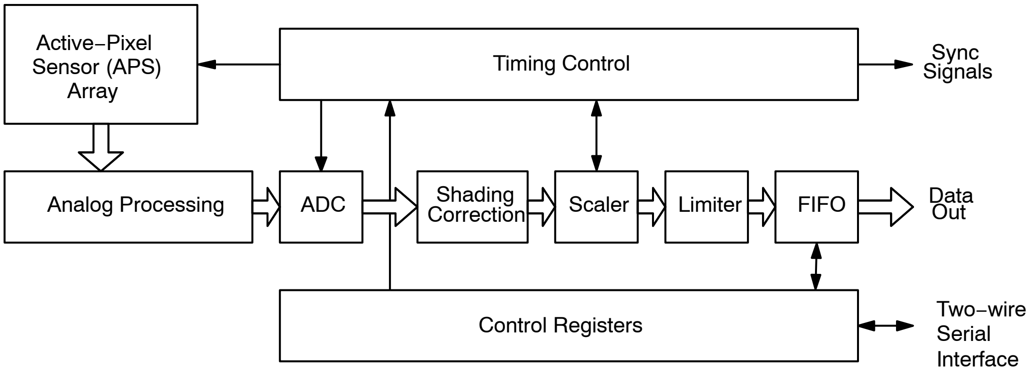


Figure 1. Block Diagram

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.

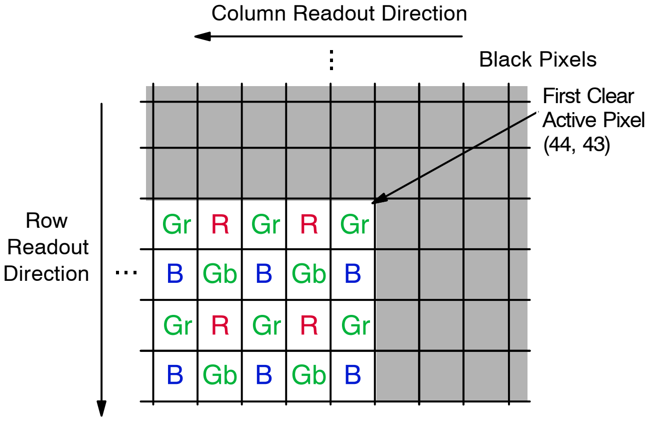


Figure 2. Pixel Color Pattern Detail (Top Right Corner)

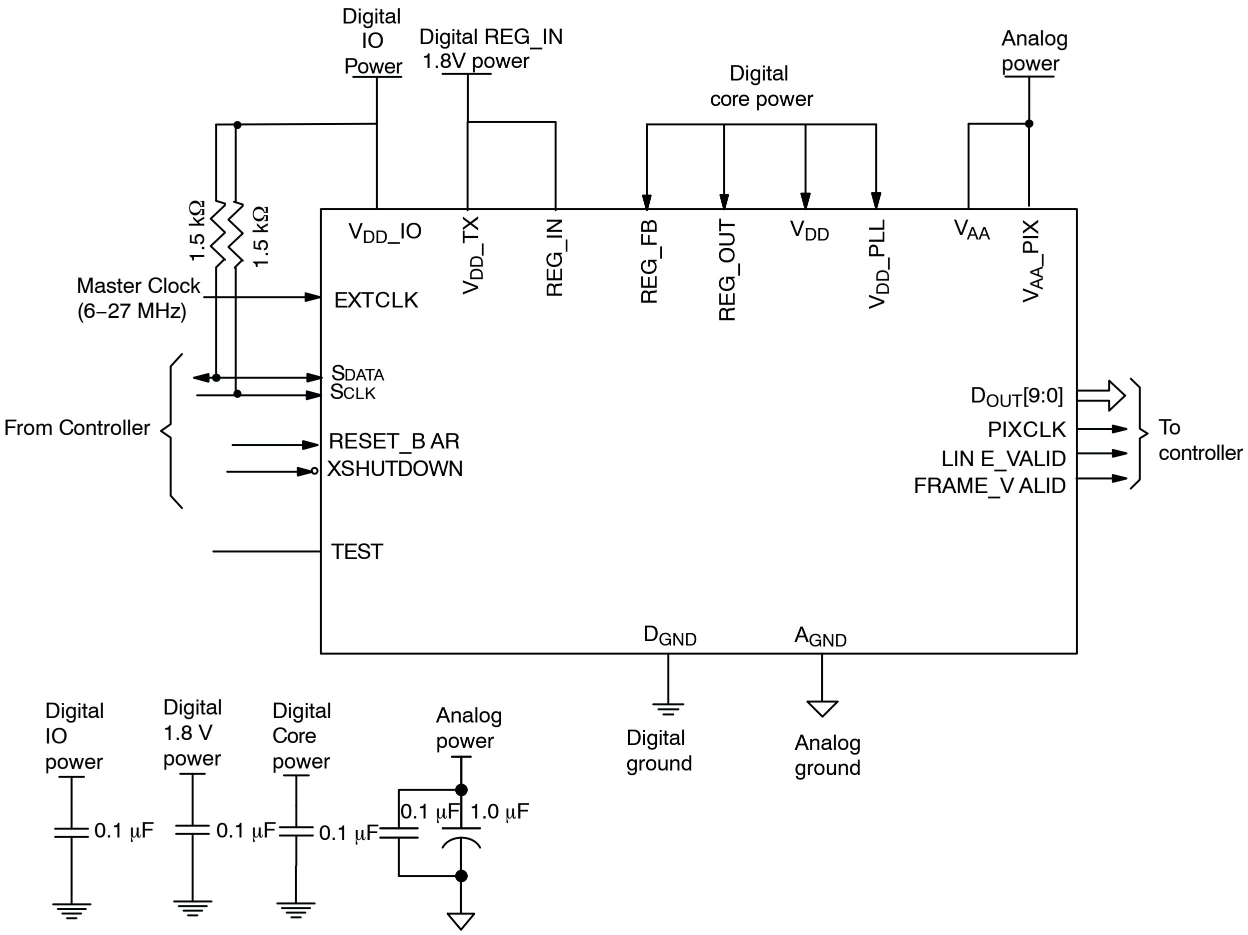
OPERATING MODES

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”.

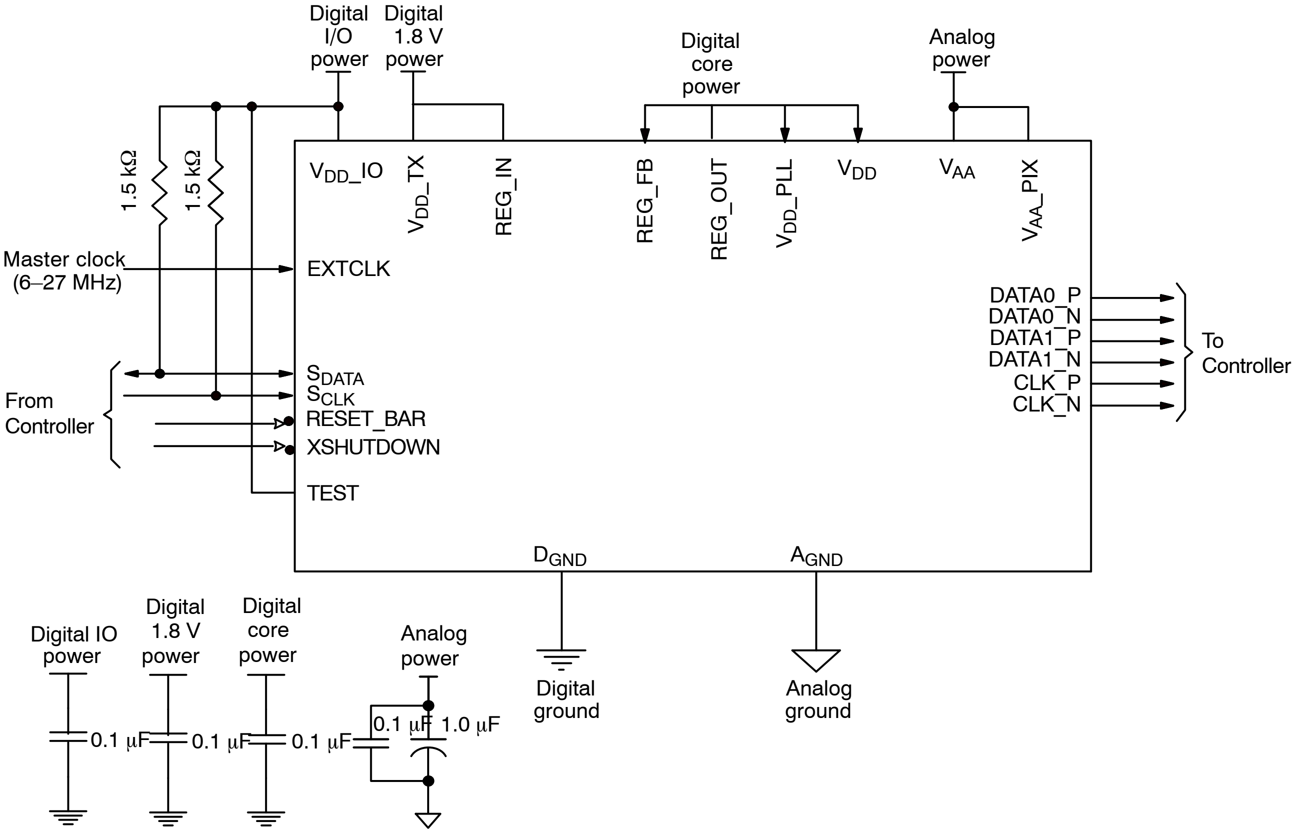
For low−noise operation, the AR0542 requires separate power supplies for analog and digital. Incoming digital and analog ground conductors can be tied together next to the die. Both power supply rails should be decoupled from the ground using capacitors as close as possible to the die.

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



1. All power supplies must be adequately decoupled.
2. **onsemi** 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. **onsemi** 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



1. All power supplies must be adequately decoupled.
2. **onsemi** 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. **onsemi** 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.

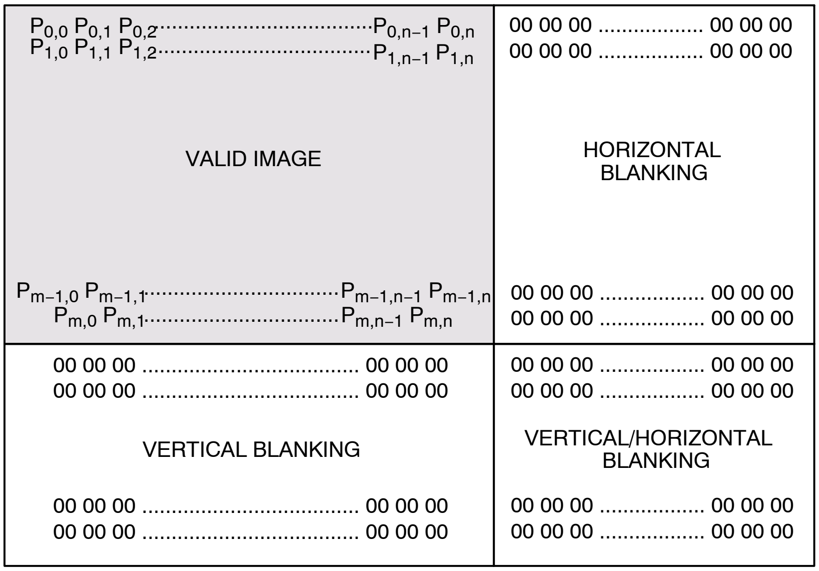
Table 3. SIGNAL DESCRIPTIONS

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 **onsemi** 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)”.



$P_{0,0}$ $P_{0,1}$ $P_{0,2}$ .... $P_{0,n-1}$ $P_{0,n}$ $P_{1,0}$ $P_{1,1}$ $P_{1,2}$ .... $P_{1,n-1}$ $P_{1,n}$ VALID IMAGE $P_{m-1,0}$ $P_{m-1,1}$ .... $P_{m-1,n-1}$ $P_{m-1,n}$ $P_{m,0}$ $P_{m,1}$ .... $P_{m,n-1}$ $P_{m,n}$	00 00 00 .... 00 00 00 00 00 00 .... 00 00 00 HORIZONTAL BLANKING 00 00 00 .... 00 00 00 00 00 00 .... 00 00 00
00 00 00 .... 00 00 00 00 00 00 .... 00 00 00 VERTICAL BLANKING 00 00 00 .... 00 00 00 00 00 00 .... 00 00 00	00 00 00 .... 00 00 00 00 00 00 .... 00 00 00 VERTICAL/HORIZONTAL BLANKING 00 00 00 .... 00 00 00 00 00 00 .... 00 00 00

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.

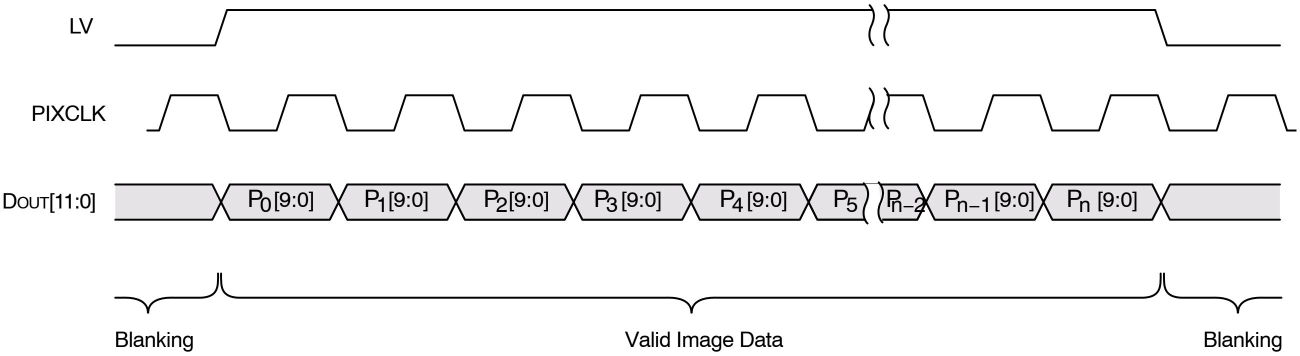


Figure 6. Pixel Data Timing Example

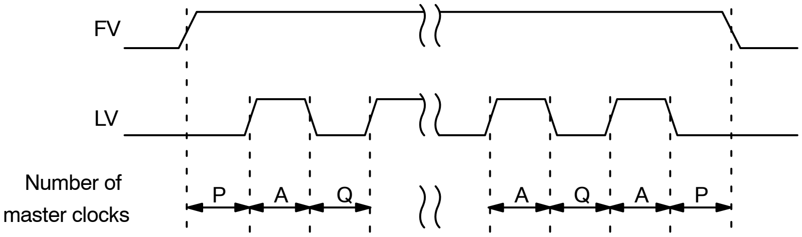


Figure 7. Row Timing and FV/LV Signals

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.

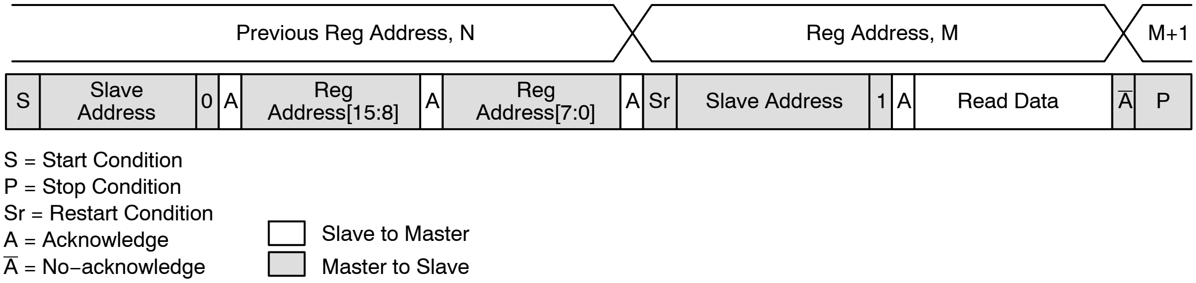


Figure 8. Single READ from Random Location

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.

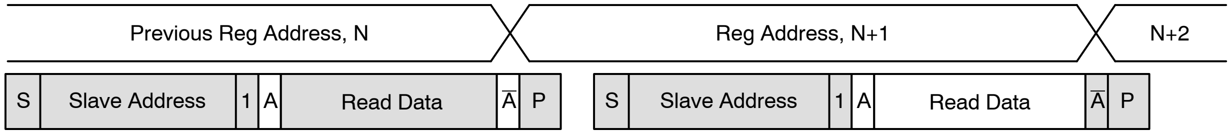


Figure 9. Single READ from Current Location

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.

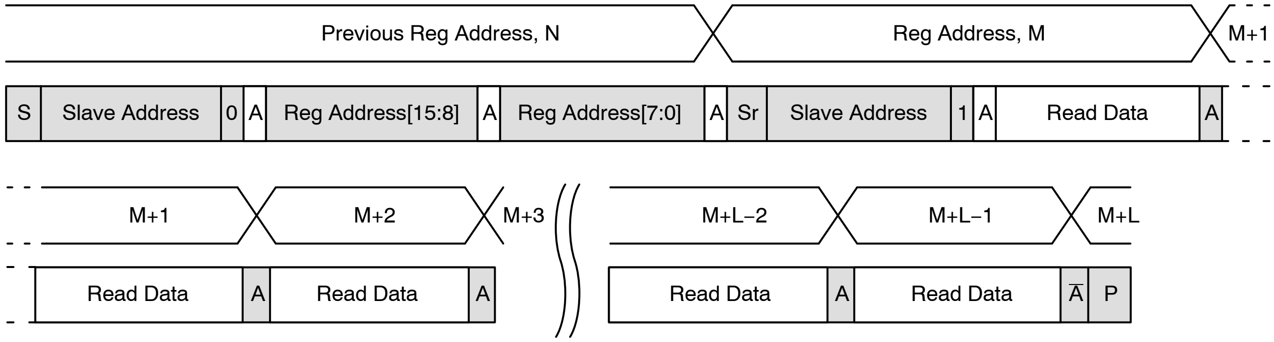


Figure 10. Sequential READ, Start from Random Location

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.

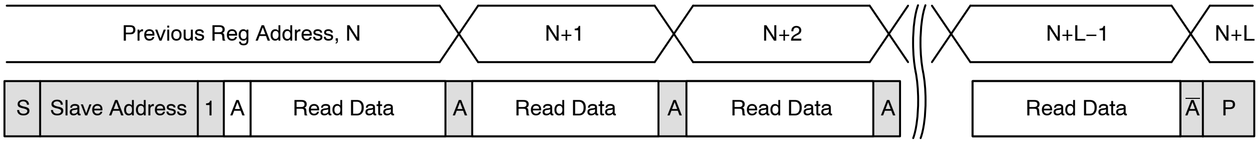


Figure 11. Sequential READ, Start from Current Location

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.

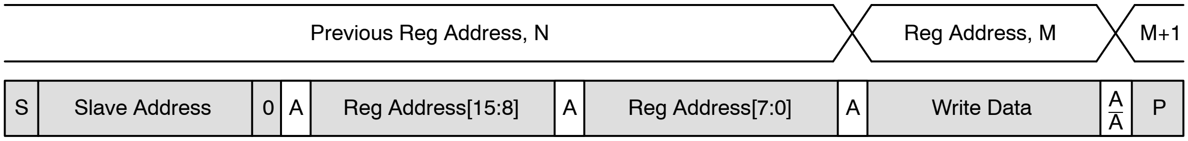


Figure 12. Single WRITE to Random Location

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.

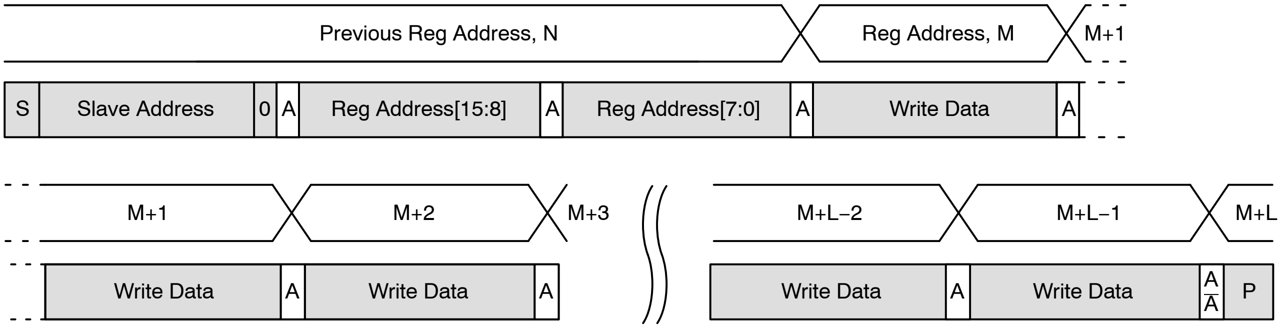


Figure 13. Sequential WRITE, Start at Random Location

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_m argin
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.

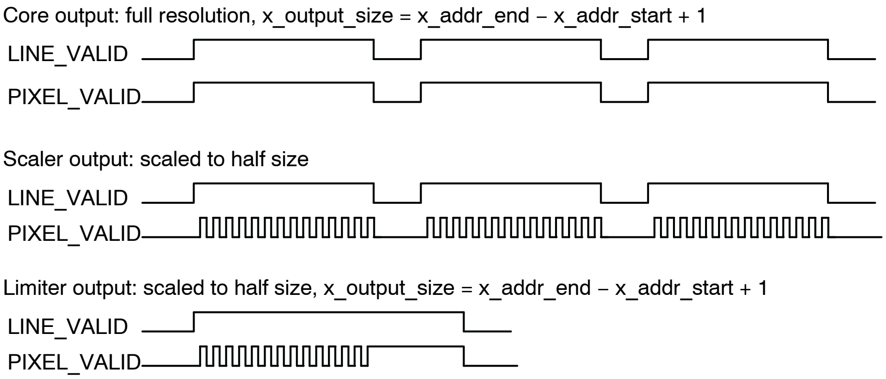

Core output: full resolution, x_output_size = x_addr_end – x_addr_start + 1
LINE_VALID
PIXEL_VALID
Scaler output: scaled to half size
LINE_VALID
PIXEL_VALID
Limiter output: scaled to half size, x_output_size = x_addr_end – x_addr_start + 1
LINE_VALID
PIXEL_VALID

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.

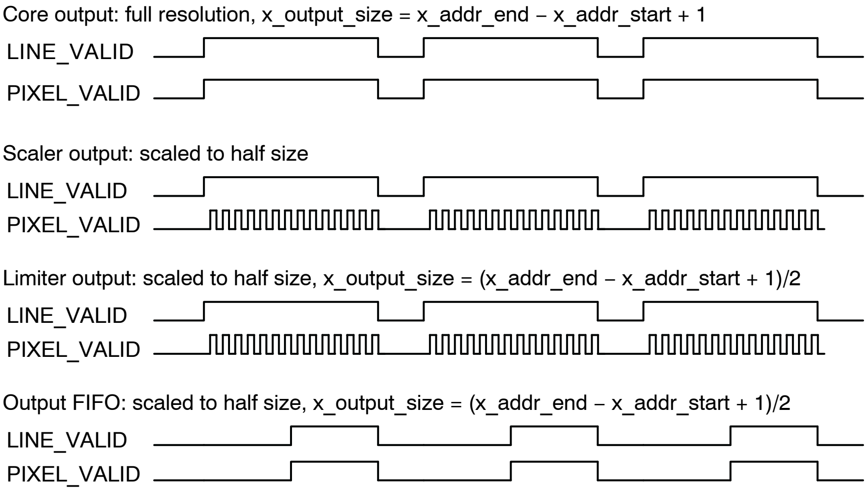

Core output: full resolution, x_output_size = x_addr_end − x_addr_start + 1
LINE_VALID
PIXEL_VALID
Scaler output: scaled to half size
LINE_VALID
PIXEL_VALID
Limiter output: scaled to half size, x_output_size = (x_addr_end − x_addr_start + 1)/2
LINE_VALID
PIXEL_VALID
Output FIFO: scaled to half size, x_output_size = (x_addr_end − x_addr_start + 1)/2
LINE_VALID
PIXEL_VALID

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 VDD_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.

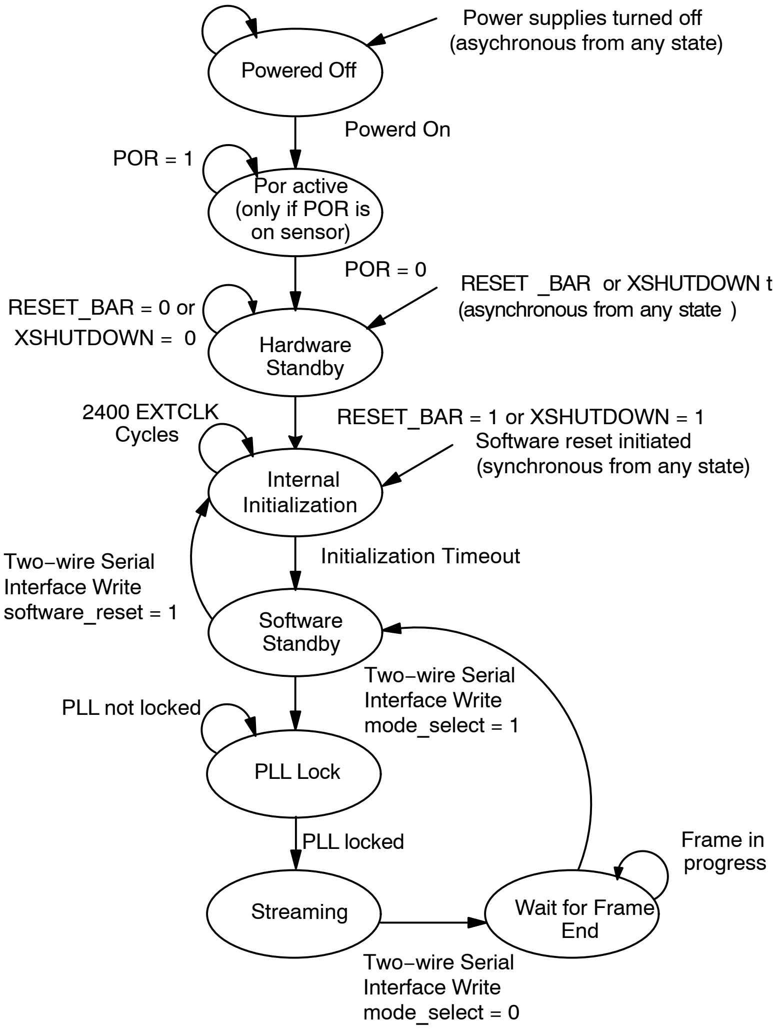


Figure 16. AR0542 System States

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 1 ms 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.

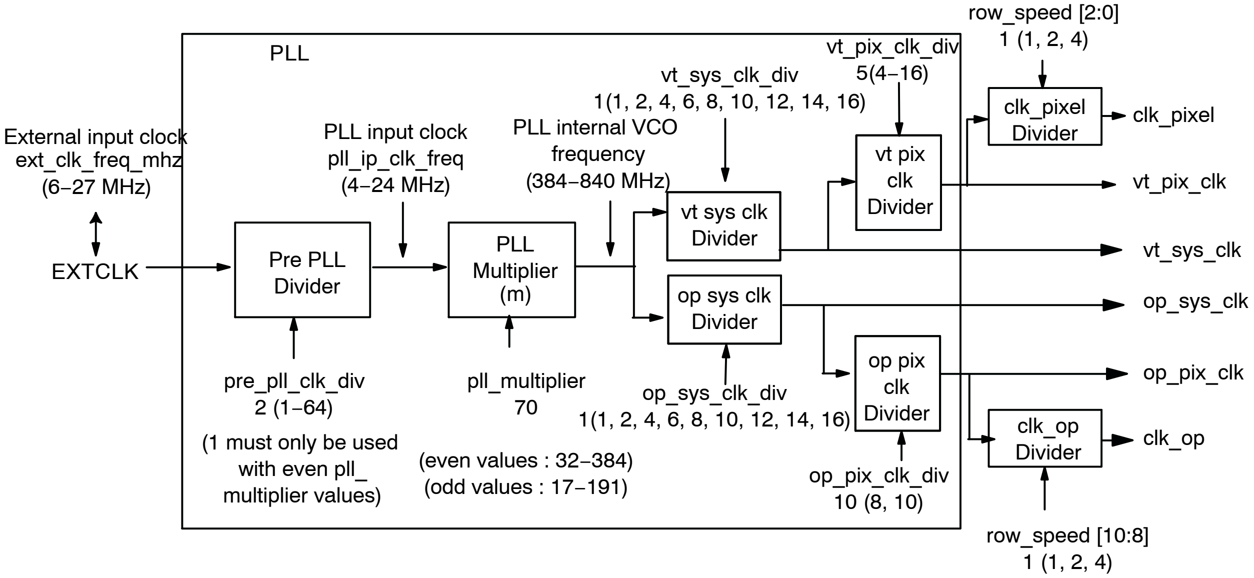


Figure 17. AR0542 Profile 1/2 Clocking Structure

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) 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:

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

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

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

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.

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.

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:

$$P_{corrected}(row, col) = P_{sensor}(row, col) \times f(row, col) \qquad \text{(eq. 4)}$$

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.

During the programming process, a dedicated high voltage pin (VPP) 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 VPP supply. Instantaneous VPP 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 (VPP) 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 VPP pin needs to be bonded out as a pin on the module, the trace for VPP 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 VPP 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 (VDD_IO, VAA, VAA_PIX, and Digital 1.8 V).
   - ♦ **onsemi** recommends setting VAA to 3.1 V during the programming process. All other supplies must be at their nominal voltage.

- ♦ Ensure that the VPP 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 VPP to 6.5 V. The recommended slew rate for VPP 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 (VPP) and float the VPP 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.

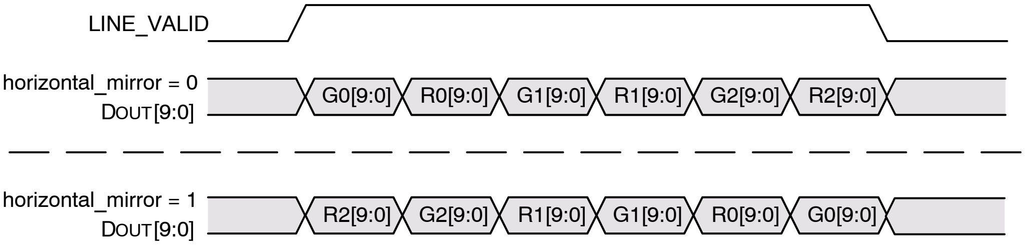


Figure 18. Effect of horizontal_mirror on Readout Order

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.

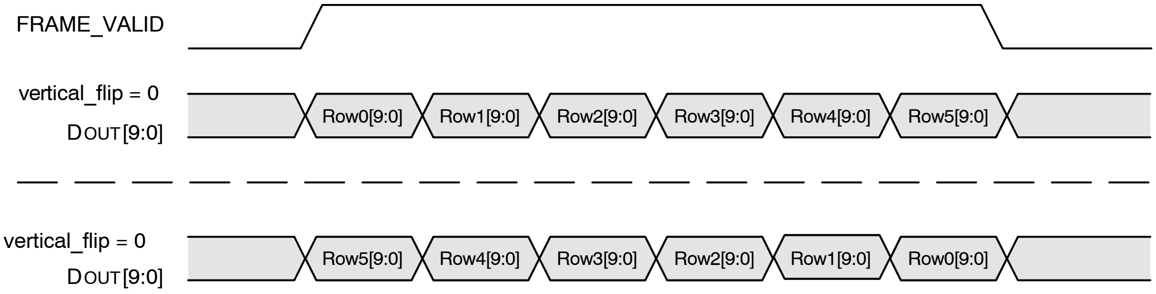


Figure 19. Effect of vertical_flip on Readout Order

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.

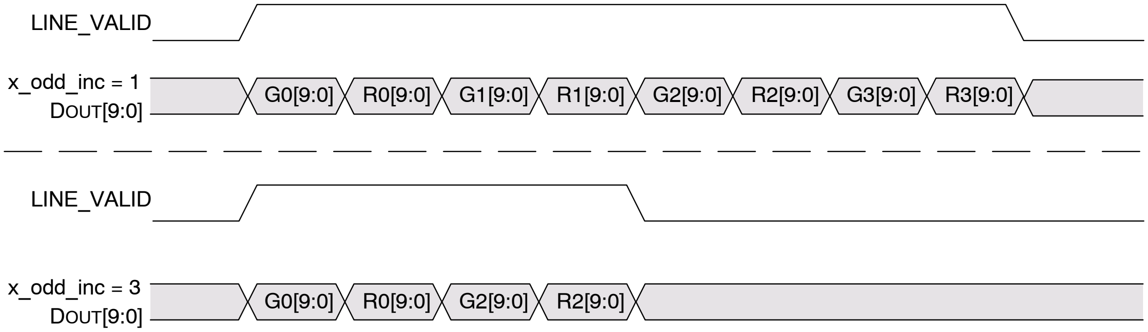


Figure 20. Effect of x_odd_inc = 3 on Readout Sequence

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.

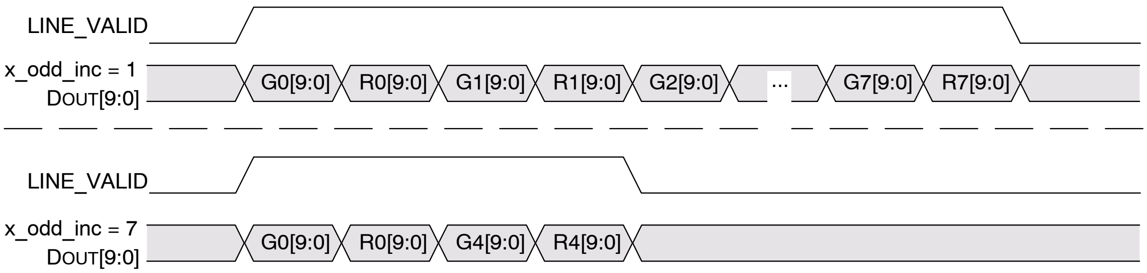


Figure 21. Effect of x_odd_inc = 7 on Readout Sequence

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

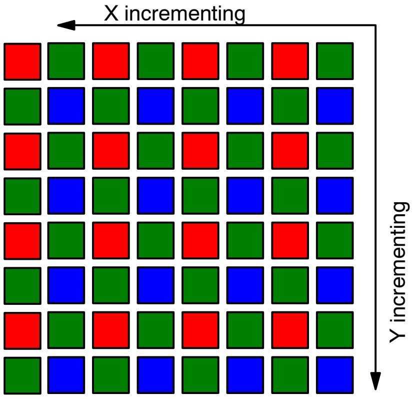


Figure 22. Pixel Readout (No Subsampling)

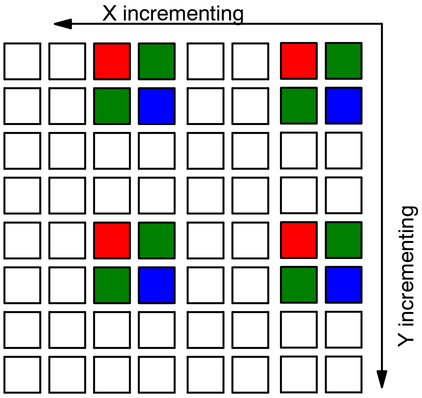


Figure 23. Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)

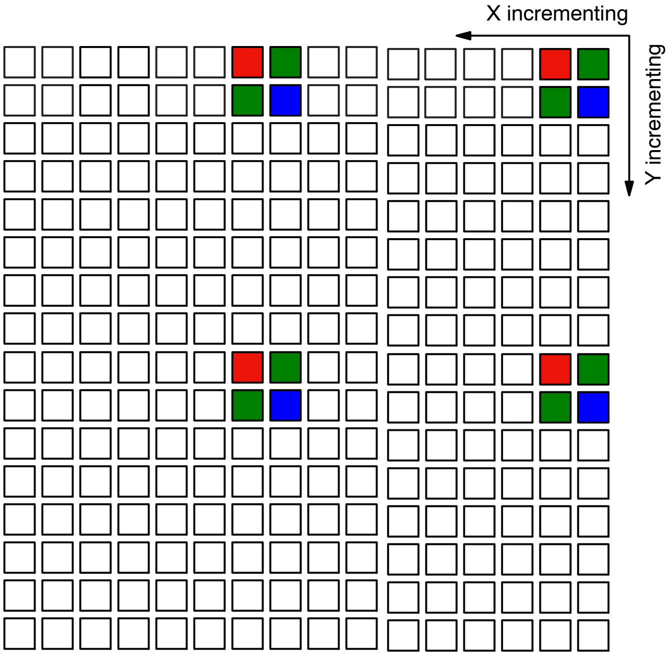


Figure 24. Pixel Readout (x_odd_inc = 7, y_odd_inc = 7)

Programming Restrictions when Subsampling

When subsampling is enabled as a viewfinder mode and the sensor is switched back and forth between full resolution and subsampling, **onsemi** recommends that line_length_pck be kept constant between the two modes. This allows the same integration times to be used in each mode.

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

x_skip_factor = (x_odd_inc + 1) / 2
x_skip_factor = (x_odd_inc + 1) / 2

- x_addr_start should be a multiple of x_skip_factor * 4
- (x_addr_end − x_addr_start + x_odd_inc) should be a multiple of x_skip_factor * 4
- (y_addr_end − y_addr_start + y_odd_inc) should be a multiple of y_skip_factor * 4

The number of columns/rows read out with subsampling can be found from the equation below:

- columns/rows = (addr_end − addr_start + odd_inc) / skip_factor

Example:

The sensor is set up to give out a full resolution 2592 x 1944 image:

- [full resolution starting address with (8,8)]

```
REG = 0x0104, 1     //GROUPED_PARAMETER_HOLD
REG = 0x0382, 1     //X_ODD_INC
REG = 0x0386, 1     //Y_ODD_INC
REG = 0x0344, 8     //X_ADDR_START
REG = 0x0346, 8     //Y_ADDR_START
REG = 0x0348, 2599  //X_ADDR_END
REG = 0x034A, 1951  //Y_ADDR_END
REG = 0x034C, 2592  //X_OUTPUT_SIZE
REG = 0x034E, 1944  //Y_OUTPUT_SIZE
REG = 0x0104, 0     //GROUPED_PARAMETER_HOLD
```

To halve the resolution in each direction (1296 x 972), the registers need to be reprogrammed as follows:

- [2 x 2 skipping starting address with (8,8)]

```
REG = 0x0104, 1     //GROUPED_PARAMETER_HOLD
REG = 0x0382, 3     //X_ODD_INC
REG = 0x0386, 3     //Y_ODD_INC
REG = 0x0344, 8     //X_ADDR_START
REG = 0x0346, 8     //Y_ADDR_START
REG = 0x0348, 2597  //X_ADDR_END
REG = 0x034A, 1949  //Y_ADDR_END
REG = 0x034C, 1296  //X_OUTPUT_SIZE
REG = 0x034E, 972   //Y_OUTPUT_SIZE
REG = 0x0104, 0     //GROUPED_PARAMETER_HOLD
```

To quarter the resolution in each direction (648 x 486), the registers need to be reprogrammed as follows:

- [4 x 4 skipping starting address with (8,8)]

```
REG = 0x0104, 1     //GROUPED_PARAMETER_HOLD
REG = 0x0382, 7     //X_ODD_INC
REG = 0x0386, 7     //Y_ODD_INC
REG = 0x0344, 8     //X_ADDR_START
REG = 0x0346, 8     //Y_ADDR_START
REG = 0x0348, 2593  //X_ADDR_END
REG = 0x034A, 1945  //Y_ADDR_END
REG = 0x034C, 648   //X_OUTPUT_SIZE
REG = 0x034E, 486   //Y_OUTPUT_SIZE
REG = 0x0104, 0     //GROUPED_PARAMETER_HOLD
```

Table 12 shows the row or column address sequencing for normal and subsampled readout. In the 2X skip case, there are two possible subsampling sequences (because the subsampling sequence only reads half of the pixels) depending upon the alignment of the start address. Similarly, there will be four possible subsampling sequences in the 4X skip case (though only the first two are shown in Table 12).

Table 12. ROW ADDRESS SEQUENCING DURING SUBSAMPLING

odd_inc = 1—Normal	odd_inc = 3, 2X Skip	odd_inc = 7, 4X Skip
start = 0	**start = 0**	**start = 0**
0	0	0
1	1	1
2		
3		
4	4	
5	5	
6		
7		
8	8	8
9	9	9
10		
11		
12	12	
13	13	
14		
15		

Binning

The AR0542 supports 2 x 1 (column binning, also called x−binning) and 2 x 2 analog binning (row/column binning, also called xy−binning). Binning has many of the same characteristics as subsampling, but because it gathers image data from all pixels in the active window (rather than a subset of them), it achieves superior image quality and avoids the aliasing artifacts that can be a characteristic side effect of subsampling.

Binning is enabled by selecting the appropriate subsampling settings (odd_inc = 3 and y_odd_inc = 1 for x−binning, x_odd_inc = 3 and y_odd_inc = 3 for xy−binning) and setting the appropriate binning bit in read_mode (R0x3040–1). As with subsampling, x_addr_end and y_addr_end may require adjustment when binning is enabled. It is the first of the two columns/rows binned together that should be the end column/row in binning, so the requirements to the end address are exactly the same as in non−binning subsampling mode. The effect of the different subsampling settings is shown in Figure 25 and Figure 26.

Binning can also be enabled when the 4X subsampling mode is enabled (x_odd_inc = 7 and y_odd_inc = 1 for x−binning, x_odd_inc = 7 and y_odd_inc = 7 for xy−binning). In this mode, however, not all pixels will be used so this is not a 4X binning implementation. An implementation providing a combination of skip2 and bin2 is used to achieve 4X subsampling with better image quality. The effect of this subsampling mode is shown in Figure 27.

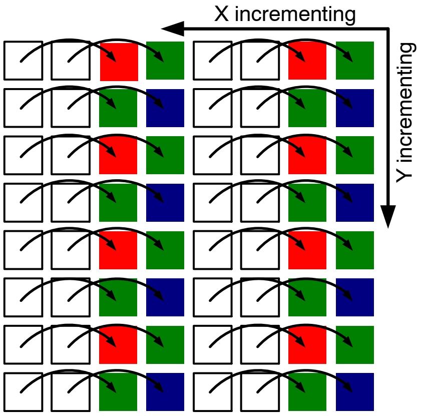


Figure 25. Pixel Readout (x_odd_inc = 3, y_odd_inc = 1, x_bin = 1)

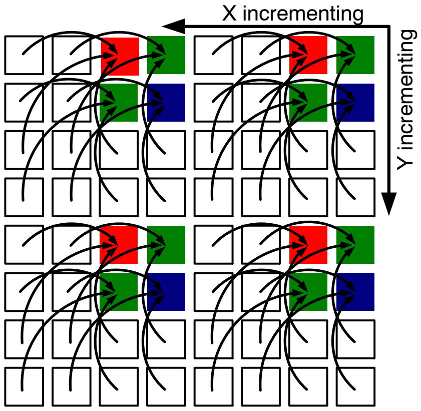


Figure 26. Pixel Readout (x_odd_inc = 3, y_odd_inc = 3, xy_bin = 1)

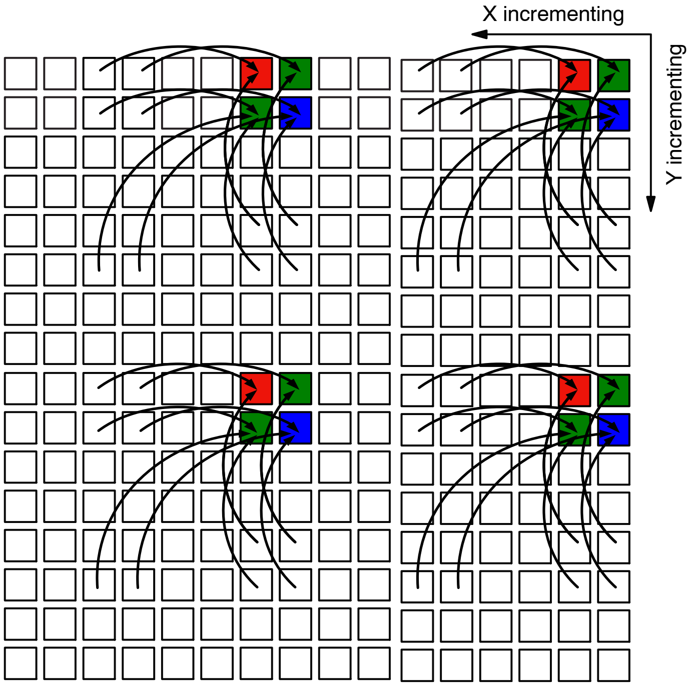


Figure 27. Pixel Readout (x_odd_inc = 7, y_odd_inc = 7, xy_bin = 1)

Binning address sequencing is a bit more complicated than during subsampling only, because of the implementation of the binning itself.

For a given column *n*, there is only one other column, n_bin, that can be binned with, because of physical limitations in the column readout circuitry. The possible address sequences are shown in Table 13.

Table 13. COLUMN ADDRESS SEQUENCING DURING BINNING

odd_inc = 1—Normal	odd_inc = 3, 2X Bin	odd_inc = 7, 2X Skip + 2X Bin
x_addr_start = 0	x_addr_start = 0	x_addr_start = 0
0	0/2	0/4
1	1/3	1/5
2		
3		
4	4/6	
5	5/7	
6		
7		
8	8/10	8/12
9	9/11	9/13
10		
11		
12	12/14	
13	13/15	
14		
15		

There are no physical limitations on what can be binned together in the row direction. A given row *n* will always be binned with row n+2 in 2X subsampling mode and with row n+4 in 4X subsampling mode. Therefore, which rows get binned together depends upon the alignment of y_addr_start. The possible sequences are shown in Table 14.

Table 14. ROW ADDRESS SEQUENCING DURING BINNING

odd_inc = 1—Normal	odd_inc = 3, 2X Bin	odd_inc = 7, 2X Skip + 2X Bin
x_addr_start = 0	x_addr_start = 0	x_addr_start = 0
0	0/2	0/4
1	1/3	1/5
2		
3		
4	4/6	
5	5/7	
6		
7		
8	8/10	8/12
9	9/11	9/13
10		
11		
12	12/14	
13	13/15	
14		
15		

Programming Restrictions when Binning

Binning requires different sequencing of the pixel array and imposes different timing limits on the operation of the sensor. In particular, xy−binning requires two read operations from the pixel array for each line of output data, which has the effect of increasing the minimum line blanking time. The SMIA specification cannot accommodate this variation because its parameter limit registers are defined as being static.

As a result, when xy−binning is enabled, some of the programming limits declared in the parameter limit registers are no longer valid. In addition, the default values for some of the manufacturer−specific registers need to be reprogrammed. See section ”Minimum Frame Time”, section ”Minimum Row Time”, and section ”Fine Integration Time Limits”.

Table 15. **READOUT MODES**

Readout Modes	x_odd_inc, y_odd_inc	xy_bin
2x skip	3	0
2x bin	3	1
4x skip	7	0
2x skip + 2x bin	7	1

Scaler

Scaling is a “zoom out” operation to reduce the size of the output image while covering the same extent as the original image. Each scaled output pixel is calculated by taking a weighted average of a group of input pixels which is composed of neighboring pixels. The input and output of the scaler is in Bayer format.

When compared to skipping, scaling is advantageous because it uses all pixel values to calculate the output image which helps avoid aliasing. Also, it is also more convenient than binning because the scale factor varies smoothly and the user is not limited to certain ratios of size reduction.

The AR0542 sensor is capable of horizontal scaling and full (horizontal and vertical) scaling.

$$\left(\text{Scale Factor} = \frac{\text{scale\_n}}{\text{scale\_m}} = \frac{16}{\text{scale\_m}}\right) \quad \text{(eq. 5)}$$

The scaling factor, programmable in 1/16 steps, is used for horizontal and vertical scalers.

The scale factor is determined by:

- n, which is fixed at 16
- m, which is adjustable with register R0x0404
- Legal values for m are 16 through 256, giving the user the ability to scale from 1:1 (m = 16) to 1:16 (m = 256).

For example, when horizontal and vertical scaling is enabled for a 1:2 scale factor, an image is reduced by half in both the horizontal and vertical directions. This results in an output image that is one−fourth of the original image size. This can be achieved with the following register settings:

```
R0x0400 = 0x0002 // horizontal and vertical
scaling mode
R0x0402 = 0x0020 // scale factor m = 32
```

Frame Rate Control

The formulas for calculating the frame rate of the AR0542 are shown below.

The line length is programmed directly in pixel clock periods through register line_length_pck. For a specific window size, the minimum line length can be found from in Equation 6:

$$\text{minimum line\_length\_pck} = \left(\frac{\text{x\_addr\_end} - \text{x\_addr\_start} + 1}{\text{subsampling factor}} + \text{min\_line\_blanking\_pck}\right) \quad \text{(eq. 6)}$$

Note that line_length_pck also needs to meet the minimum line length requirement set in register min_line_length_pck. The row time can either be limited by the time it takes to sample and reset the pixel array for each row, or by the time it takes to sample and read out a row. Values for min_line_blanking_pck are provided in “Minimum Row Time”.

The frame length is programmed directly in number of lines in the register frame_line_length. For a specific window size, the minimum frame length can be found in Equation 7:

$$\text{minimum frame\_length\_lines} = \left(\frac{\text{y\_addr\_end} - \text{y\_addr\_start} + 1}{\text{subsampling factor}} + \text{min\_frame\_blanking\_lines}\right) \quad \text{(eq. 7)}$$

The frame rate can be calculated from these variables and the pixel clock speed as shown in Equation 8:

$$\text{frame rate} = \frac{\text{vt\_pixel\_clock\_mhz} \times 1 \times 10^6}{\text{line\_length\_pck} \times \text{frame\_length\_lines}} \quad \text{(eq. 8)}$$

If coarse_integration_time is set larger than frame_length_lines the frame size will be expanded to coarse_integration_time + 1.

Minimum Row Time

The minimum row time and blanking values with default register settings are shown in Table 16.

Table 16. MINIMUM ROW TIME AND BLANKING NUMBERS

	No Row Binning			Row Binning		
row_speed[2:0]	1	2	4	1	2	4
min_line_blanking_pck	0x044E	0x02B6	0x01E8	0x073C	0x040C	0x0274
min_line_length_pck	0x0590	0x03F8	0x0330	0x0940	0x0550	0x03B8

In addition, enough time must be given to the output FIFO so it can output all data at the set frequency within one row time.

There are therefore three checks that must all be met when programming line_length_pck:

- line_length_pck > min_line_length_pck in Table 16.
- line_length_pck > (x_addr_end – x_addr_start + x_odd_inc)/((1+x_odd_inc)/2) + min_line_blanking_pck in Table 16.
- The row time must allow the FIFO to output all data during each row. That is, line_length_pck > (x_output_size * 2 + 0x005E) * ”vt_pix_clk period” / ”op_pix_clk period”

Minimum Frame Time

The minimum number of rows in the image is 2, so min_frame_length_lines will always equal (min_frame_blanking_lines + 2).

Table 17. MINIMUM FRAME TIME AND BLANKING NUMBERS

	No Row Binning	Row Binning
min_frame_blanking_lines	0x004D	0x0049
min_frame_length_lines	0x005D	0x0059

Integration Time

The integration (exposure) time of the AR0542 is controlled by the fine_integration_time and coarse_integration_time registers.

The limits for the fine integration time are defined by:

$$\text{fine\_integration\_time\_min} \le \text{fine\_integration\_time} \le (\text{line\_length\_pck} - \text{fine\_integration\_time\_max\_margin}) \quad \text{(eq. 9)}$$

The limits for the coarse integration time are defined by:

$$\text{coarse\_integration\_time\_min} < \text{coarse\_integration\_time} \quad \text{(eq. 10)}$$

The actual integration time is given by:

$$\text{integration\_time} = \frac{((\text{coarse\_integration\_time} \times \text{line\_length\_pck}) + \text{fine\_integration\_time})}{(\text{vt\_pix\_clk\_freq\_mhz} \times 10^6)} \quad \text{(eq. 11)}$$

It is required that:

$$\text{coarse\_integration\_time} \le (\text{frame\_length\_lines} - \text{coarse\_integration\_time\_max\_margin}) \quad \text{(eq. 12)}$$

If this limit is broken, the frame time will automatically be extended to coarse_integration_time + coarse_integration_time_max_margin to accommodate the larger integration time.

In binning mode, frame_length_lines should be set larger than coarse_integration_time by at least 3 to avoid column imbalance artifact.

Fine Integration Time Limits

The limits for the *fine_integration_time* can be found from *fine_integration_time_min* and *fine_integration_time_max_margin*. Values for different mode combinations are shown in Table 18.

Table 18. fine_integration_time LIMITS

	No Row Binning			Row Binning		
row_speed[2:0]	1	2	4	1	2	4
fine_integration_time_min	0x02CE	0x0178	0x006E	0x0570	0x02C8	0x00C2
fine_integration_time_max_margin	0x0159	0x00AD	0x00AD	0x02B9	0x015D	0x0149

fine_correction

For the fine_integration_time limits, the fine_correction constant will change with the pixel clock speed and binning mode. It is necessary to change fine_correction (R0x3010) when binning is enabled or the pixel clock divider (row_speed[2:0]) is used. The corresponding fine_correction values are shown in Table 19.

Table 19. fine_correction VALUES

	No Row Binning			Row Binning		
row_speed[2:0]	1	2	4	1	2	4
fine_correction	0x00A0	0x004A	0x001F	0x0140	0x009A	0x0047

Flash Timing Control

The AR0542 supports both Xenon and LED flash timing through the FLASH output signal. The timing of the FLASH signal with the default settings is shown in Figures 28 (Xenon) and 29 (LED). The flash and flash_count registers allow the timing of the flash to be changed. The flash can be programmed to fire only once, delayed by a few frames when asserted, and (for Xenon flash) the flash duration can be programmed.

Enabling the LED flash will cause one bad frame, where several of the rows only have the flash on for part of their integration time. This can be avoided either by first enabling mask bad frames (write reset_register[9] = 1) before the enabling the flash or by forcing a restart (write reset_register[1] = 1) immediately after enabling the flash; the first bad frame will then be masked out, as shown in Figure 29. Read−only bit flash[14] is set during frames that are correctly integrated; the state of this bit is shown in Figures 28 and 29.

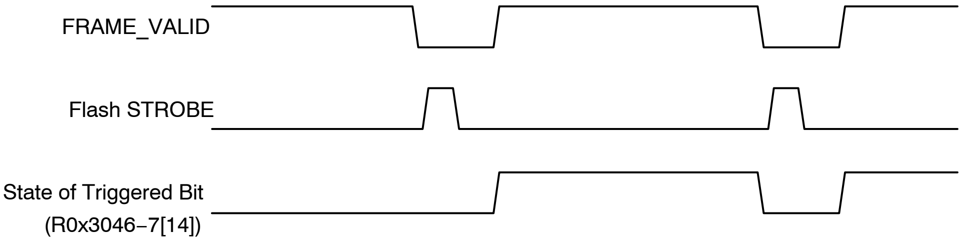
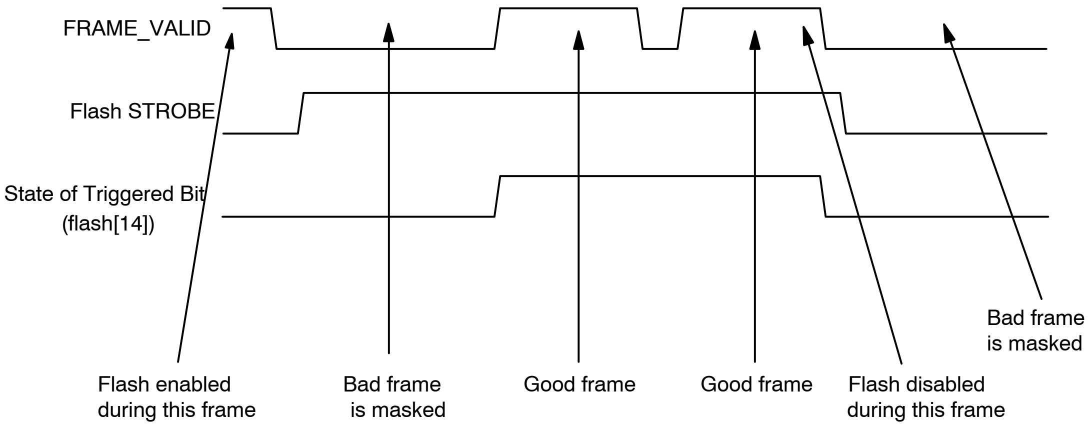


Figure 28. Xenon Flash Enabled



Note: An option to invert the flash output signal through R0x3046[7] is also available.

Figure 29. LED Flash Enabled

Analog Gain

The following sections describe the **onsemi** gain model for AR0542 and the different gain stages and gain control.

Using Per−color or Global Gain Control

The read−only analogue_gain_capability register returns a value of "1," indicating that the AR0542 provides per−color gain control. However, the AR0542 also provides the option of global gain control. Per−color and global gain control can be used interchangeably. A write to a global gain register is aliased as a write of the same data to the four associated color−dependent gain registers. A read from a global gain register is aliased to a read of the associated greenR gain register.

Table 20. GAIN REGISTERS

Register	Bits	Default	Name	Frame Sync'd	Bad Frame
12382 R0x305E	15:0	0x1050	**global_gain (R/W)**	N	N
	15:12	0x0001	**digital_gain** Digital Gain. Legal values 1−7.		
	11:10	0x0000	**col_gain** This is the column gain Valid values for bits[11:10] are: 00: 1x 01: 3x 10: 2x 11: 4x	Y	Y
	9:8	0x0000	**asc1_gain** This is the ASC1 gain Valid values for bits[9:8] are: 00: 1x 01: 1.3x 10: 2x 11: 4x		
	7	0x0000	**Reserved**	Y	N
	6:0	0x0050	**initial_gain** Initial gain = bits [6:0] * 1/32.		
	Gain = Column Gain*ASC1 Gain* Initial_gain			Y	Y

onsemi *Gain Model*

The **onsemi** gain model uses these registers to set the analog gain:

- global_gain
- green1_gain
- red_gain
- blue_gain
- green2_gain

The AR0542 uses 11 bits analog gain control. The analog gain is given by:

$$\begin{aligned}\text{Total gain} &= \text{Column\_gain} \times \text{ASC1\_gain} \times \text{Initial\_gain}\\ &= <\text{color}>\text{\_gain}[11:10] \times <\text{color}>\text{\_gain}[9:8] \times \frac{<\text{color}>\text{\_gain}[6:0]}{32}\end{aligned} \quad \text{(eq. 13)}$$

Table 21.

Valid Values	Column_gain(<color>_gain[11:10])	ASC_gain(<color>_gain[9:8])
2'b00	1X	1X
2'b01	3X	1.3X
2'b10	2X	2X
2'b11	4X	–

As a result, the step size varies depending upon which range the gain is in. Many of the possible gain settings can be achieved in different ways. However, the recommended gain setting is to use the Column_gain as much as possible instead of using ASC1_gain and Initial_gain for the desired gain setting, which will result lower noise. for the fine step, the Initial gain should be used with Column_gain and ASC1_gain.

The recommended minimum analog gain for AR0542 is 1.6x(R0x305E = 0x1127). Table 22 provides the gain usage table that is a guide to program a specific gain value while optimizing the noise performance from the sensor.

Table 22. GAIN USAGE

Total Gain	Column Gain	ASC1 Gain	Initial Gain
1.0≤Gain<1.33	1	1	1.0≤init<1.33
1.33≤Gain<2.0	1	1.33	1.0≤init<1.50
2.0≤Gain<2.66	2	1	1.0≤init<1.33
2.66≤Gain<3.0	2	1.33	1.0≤init<1.15
3.0≤Gain<4.0	3	1	1.0≤init<1.33
4.0≤Gain<5.3	4	1	1.0≤init<1.33
5.3≤Gain<8.0	4	1.33	1.0≤init<1.50
8.0≤Gain<32.0	4	2	1.0≤init<4.0

SENSOR CORE DIGITAL DATA PATH

Test Patterns

The AR0542 supports a number of test patterns to facilitate system debug. Test patterns are enabled using test_pattern_mode (R0x0600–1). The test patterns are listed in Table 23.

Table 23. TEST PATTERNS

test_pattern_mode	Description
0	Normal operation: no test pattern
1	Solid color
2	100% color bars
3	Fade−to−gray color bars
4	PN9 link integrity pattern (only on sensors with serial interface)
256	Walking 1s (10−bits)
257	Walking 1s (8−bits)

Test patterns 0–3 replace pixel data in the output image (the embedded data rows are still present). Test pattern 4 replaces all data in the output image (the embedded data rows are omitted and test pattern data replaces the pixel data).

For all of the test patterns, the AR0542 registers must be set appropriately to control the frame rate and output timing. This includes:

- All clock divisors
- x_addr_start
- x_addr_end
- y_addr_start
- y_addr_end
- frame_length_lines
- line_length_pck
- x_output_size
- y_output_size

Effect of Data Path Processing on Test Patterns

Test patterns are introduced early in the pixel data path. As a result, they can be affected by pixel processing that occurs within the data path. This includes:

- Noise cancellation
- Black pedestal adjustment
- Lens and color shading correction

These effects can be eliminated by the following register settings:

- R0x3044–5[10] = 0
- R0x30C0–1[0] = 1
- R0x30D4–5[15] = 0
- R0x31E0–1[0] = 0
- R0x3180–1[15] = 0
- R0x301A–B[3] = 0 (enable writes to data pedestal)
- R0x301E–F = 0x0000 (set data pedestal to “0”)
- R0x3780[15] = 0 (turn off lens/color shading correction)

Solid Color Test Pattern

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

100% Color Bars Test Pattern

In this test pattern, shown in Figure 30, all pixel data is replaced by a Bayer version of an 8−color, color−bar chart (white, yellow, cyan, green, magenta, red, blue, black). Each bar is 1/8 of the width of the pixel array (2592/8 = 324 pixels). The pattern repeats after 8 * 324 = 2592 pixels.

Each color component of each bar is set to either 0 (fully off) or 0x3FF (fully on for 10−bit data).

The pattern occupies the full height of the output image.

The image size is set by x_addr_start, x_addr_end, y_addr_start, y_addr_end and may be affected by the setting of x_output_size, y_output_size. The color−bar pattern is disconnected from the addressing of the pixel array, and will therefore always start on the first visible pixel, regardless of the value of x_addr_start. The number of colors that are visible in the output is dependent upon x_addr_end − x_addr_start and the setting of x_output_size: the width of each color bar is fixed at 324 pixels.

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

The effect of subsampling, binning and scaling of this test pattern is undefined. Test patterns should be analyzed at full resolution only.

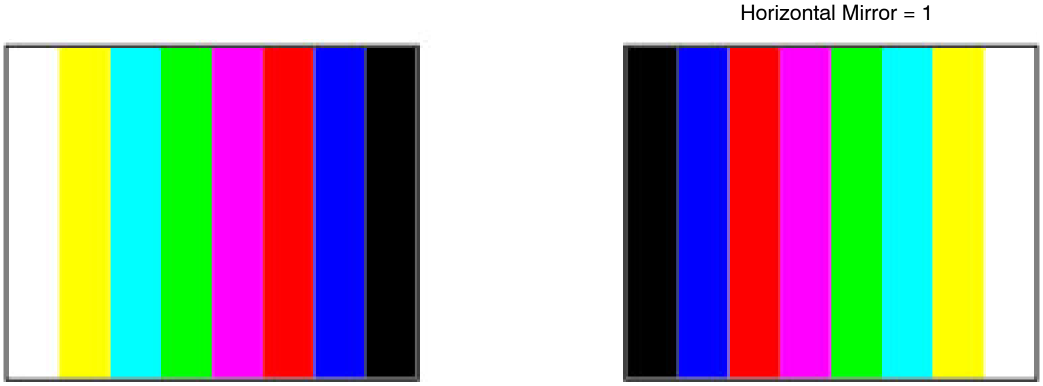


Figure 30. 100 Percent Color Bars Test Pattern

Fade−to−gray Color Bars Test Pattern

In this test pattern, shown in Figure 31, all pixel data is replaced by a Bayer version of an 8−color, color−bar chart (white, yellow, cyan, green, magenta, red, blue, black). Each bar is 1/8 of the width of the pixel array (2592/8 = 324 pixels). The test pattern repeats after 2592 pixels.

Each color bar fades vertically from zero or full intensity at the top of the image to 50% intensity (mid−gray) on the last row of the pattern. Each color bar is divided into a left and a right half, in which the left half fades smoothly and the right half fades in quantized steps.

The speed at which each color fades is dependent on the sensor's data width and the height of the pixel array. We want half of the data range (from 100 or 0 to 50%) difference between the top and bottom of the pattern. Because of the Bayer pattern, each state must be held for two rows.

The rate−of−fade of the Bayer pattern is set so that there is at least one full pattern within a full−sized image for the sensor. Factors that affect this are the resolution of the ADC (10−bit or 12−bit) and the image height.

The image size is set by x_addr_start, x_addr_end, y_addr_start, y_addr_end and may be affected by the setting of x_output_size, y_output_size. The color−bar pattern starts at the first column in the image, regardless of the value of x_addr_start. The number of colors that are visible in the output is dependent upon x_addr_end – x_addr_start and the setting of x_output_size: the width of each color bar is fixed at 324 pixels.

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

The effect of subsampling, binning, and scaling of this test pattern is undefined. Tst patterns should be analyzed at full resolution only.

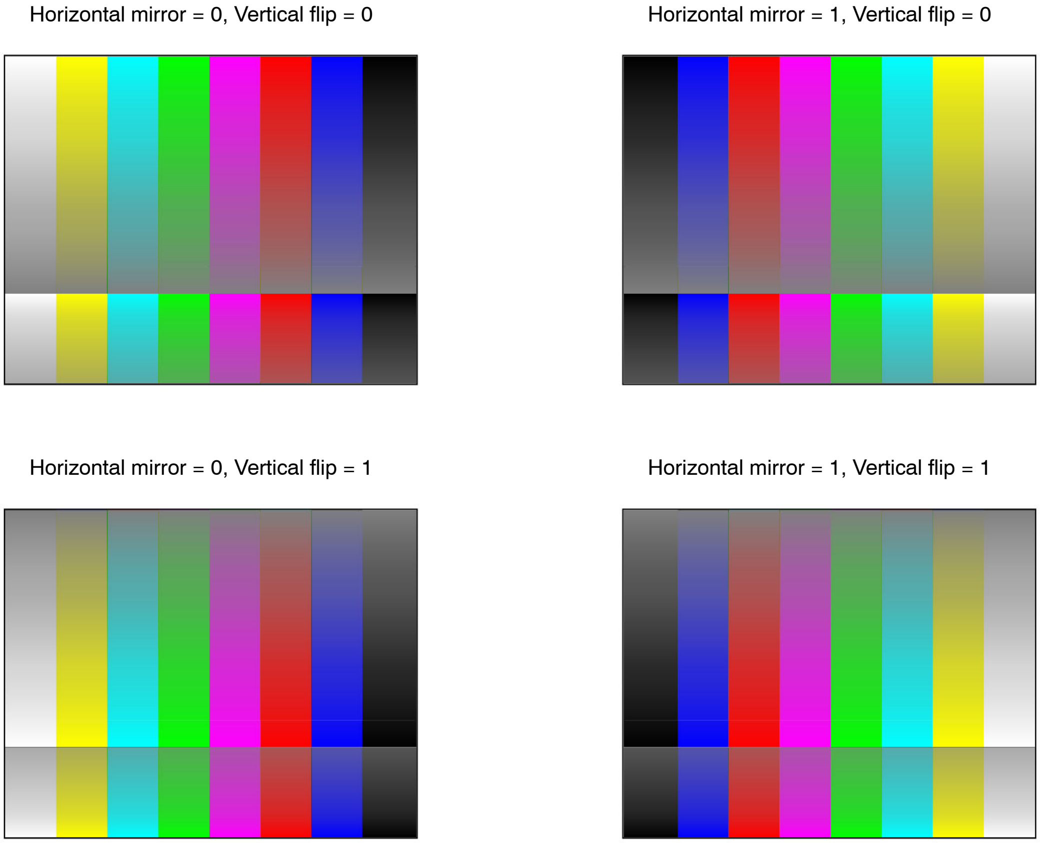

Horizontal mirror = 0, Vertical flip = 0
Horizontal mirror = 1, Vertical flip = 0
Horizontal mirror = 0, Vertical flip = 1
Horizontal mirror = 1, Vertical flip = 1

Figure 31. Fade−to−Gray Color Bars Test Pattern

PN9 Link Integrity Pattern

The PN9 link integrity pattern is intended to allow testing of a serial pixel data interface. Unlike the other test patterns, the position of this test pattern at the end of the data path means that it is not affected by other data path corrections (row noise, pixel defect correction and so on).

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

When this test pattern is enabled:

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

Before enabling this test pattern the clock divisors must be configured for RAW10 operation (op_pix_clk_div = 10).

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

Walking 1s

When selected, a walking 1s pattern will be sent through the digital pipeline. The first value in each row is 0. Each value will be valid for two pixels.

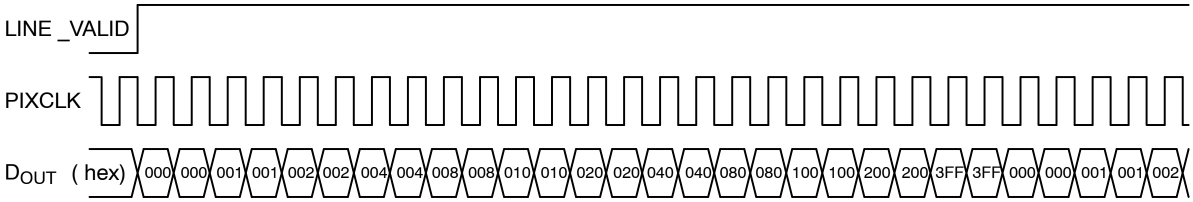

LINE _VALID
PIXCLK
D_{OUT} (hex)
000 000 001 001 002 002 004 004 008 008 010 010 020 020 040 040 080 080 100 100 200 200 3FF 3FF 000 000 001 001 002

Figure 32. Walking 1s 10−bit Pattern

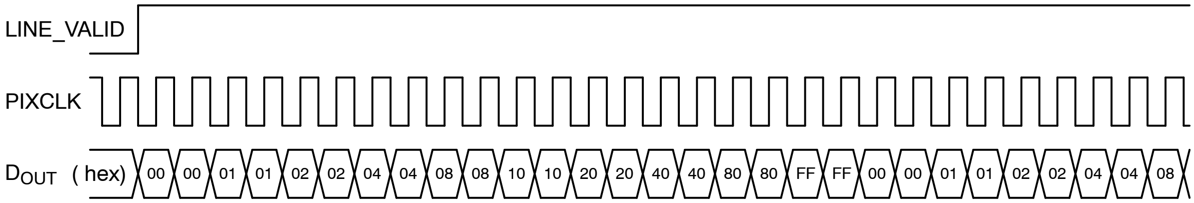

LINE_VALID
PIXCLK
D_{OUT} (hex)
00 00 01 01 02 02 04 04 08 08 10 10 20 20 40 40 80 80 FF FF 00 00 01 01 02 02 04 04 08

Figure 33. Walking 1s 8−bit Pattern

The walking 1s pattern was implemented to facilitate assembly testing of modules with a parallel interface.

The walking 1 test pattern is not active during the blanking periods; hence the output would reset to a value of 0x0. When the active period starts again, the pattern would restart from the beginning. The behavior of this test pattern is the same between full resolution and subsampling mode. RAW10 and RAW8 walking 1 modes are enabled by different test pattern codes.

Test Cursors

The AR0542 supports one horizontal and one vertical cursor, allowing a crosshair to be superimposed on the image or on test patterns 1–3. The position and width of each cursor are programmable in registers 0x31E8–0x31EE. Both even and odd cursor positions and widths are supported.

Each cursor can be inhibited by setting its width to 0. The programmed cursor position corresponds to the x and y addresses of the pixel array. For example, setting horizontal_cursor_position to the same value as y_addr_start would result in a horizontal cursor being drawn starting on the first row of the image. The cursors are opaque (they replace data from the imaged scene or test pattern). The color of each cursor is set by the values of the Bayer components in the test_data_red, test_data_greenR, test_data_blue and test_data_greenB registers. As a consequence, the cursors are the same color as test pattern 1 and are therefore invisible when test pattern 1 is selected.

When vertical_cursor_position = 0x0fff, the vertical cursor operates in an automatic mode in which its position advances every frame. In this mode the cursor starts at the column associated with x_addr_start = 0 and advances by a step−size of 8 columns each frame, until it reaches the column associated with x_addr_start = 2584, after which it wraps (324 steps). The width and color of the cursor in this automatic mode are controlled in the usual way.

The effect of enabling the test cursors when the image_orientation register is non−zero is not defined by the design specification. The behavior of the AR0542 is shown in Figure 34 and the test cursors are shown as translucent, for clarity. In practice, they are opaque (they overlay the imaged scene). The manner in which the test cursors are affected by the value of image_orientation can be understood from these implementation details:

- The test cursors are inserted last in the data path, the cursor is applied without any sensor corrections.
- The drawing of a cursor starts when the pixel array row or column address is within the address range of cursor start to cursor start + width.
- The cursor is independent of image orientation.

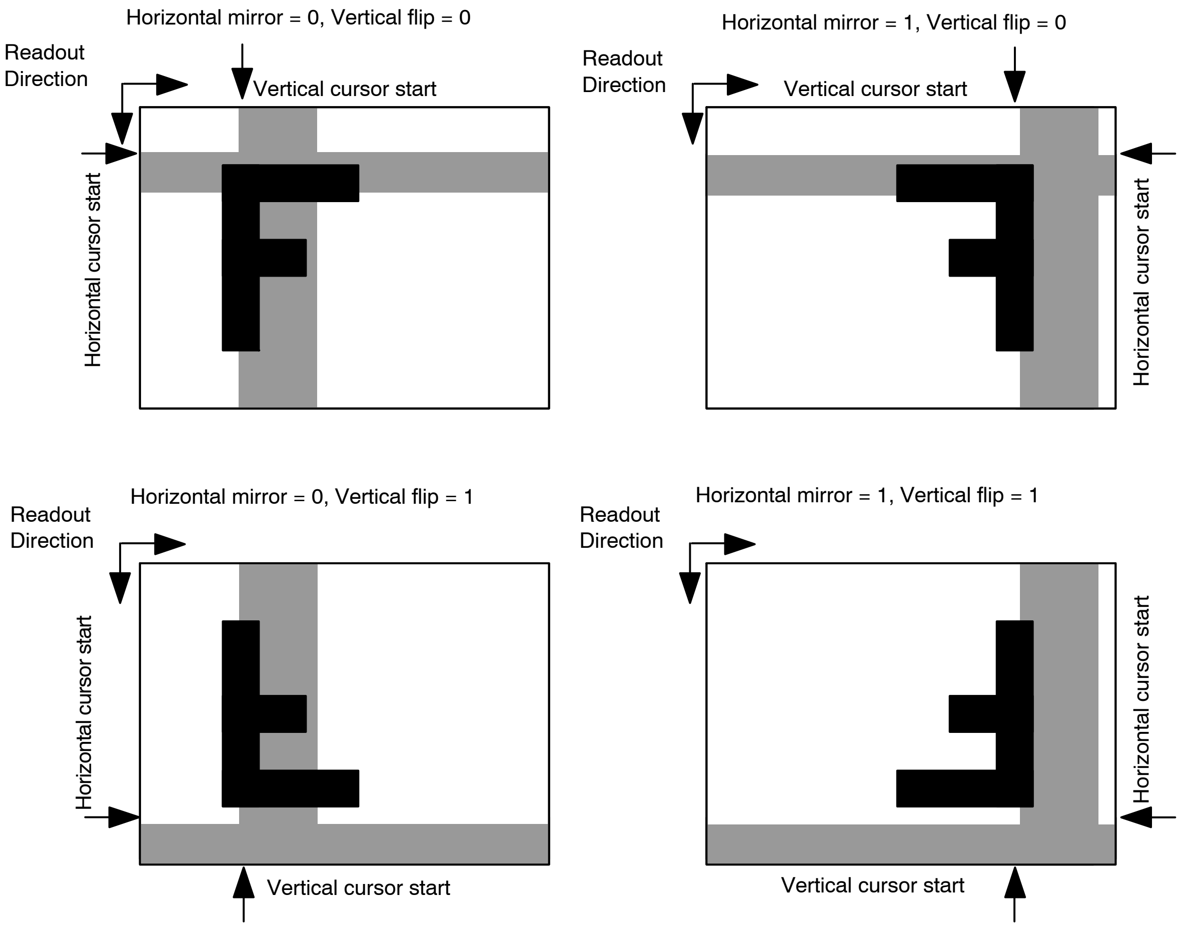


Figure 34. Test Cursor Behavior With Image Orientation

Digital Gain

Integer digital gains in the range 1–7 can be programmed.

Pedestal

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

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

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

DIGITAL DATA PATH

The digital data path after the sensor core is shown in Figure 35.

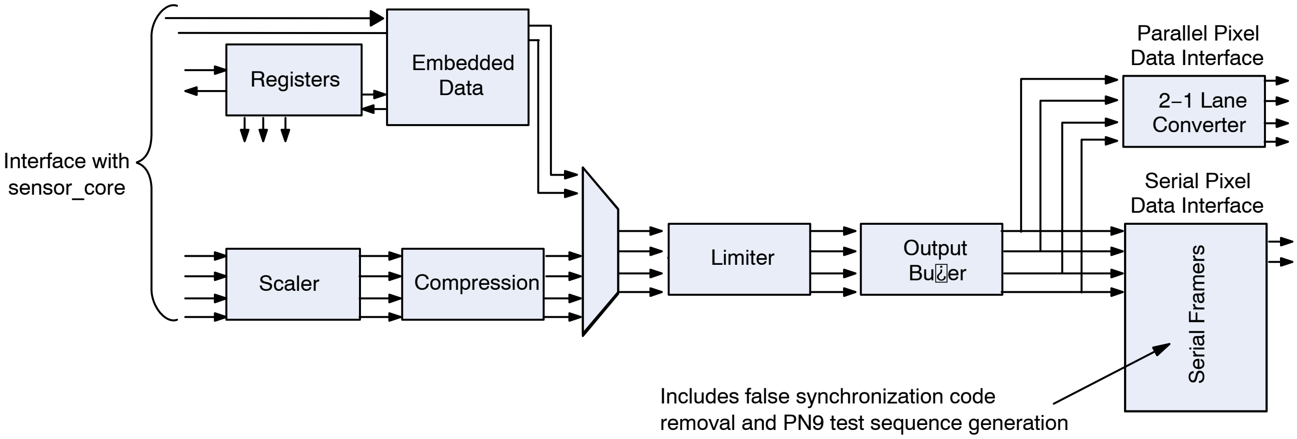


Figure 35. Data Path

Embedded Data Format and Control

When the serial pixel data path is selected, the first two rows of the output image contain register values that are appropriate for the image. The 12–bit format places the data byte in bits [11:4] and sets bits [3:0] to a constant value of 0101. Some register values are dynamic and may change from frame to frame. Additional information on the format of the embedded data can be located in the SMIA specification.

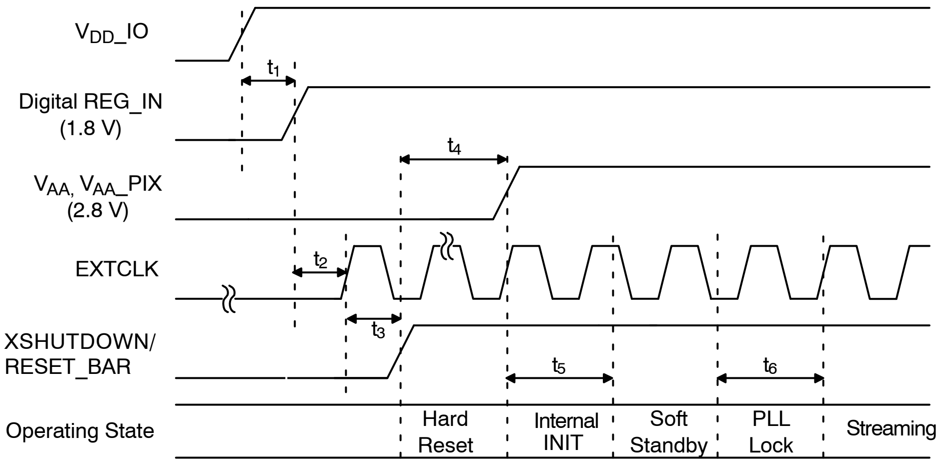
TIMING SPECIFICATIONS

Power−Up Sequence

Two power−up sequences are recommended for the AR0542 based on the XSHUTDOWN and RESET_BAR one−pin (pin−constrained mode) or two−pin (pin−unconstrained mode) control mode.

XSHUTDOWN/RESET_BAR Pin−Constrained Mode

1. Turn on V_{DD}_IO power supply.
2. After 0−10 ms, Turn on Digital REG_IN (1.8 V) power supply.
3. After 0−10 ms, enable EXTCLK.
4. After 0−100 ms, assert XSHUTDOWN/RESET_BAR (High).
5. After 1ms−500 ms, turn on V_{AA}/V_{AA}_PIX power supplies.
6. Wait 1 ms for internal initialization into soft standby.
7. Configure PLL, output and image settings to desired values.
8. Set mode_select = 1 (R0x0100).
9. Wait 1ms for the PLL to lock before streaming state is reached.



Note: If the AR0542 two−wire serial interface is also used for communication with other devices, the status of S_{DATA} during power−up needs to be considered at the system level due to the sensor's interaction during this time (t0 to t3) driving it to the low state; if the AR0542 two−wire serial interface is used for a dedicated point−point connection to the host, no additional considerations apply.

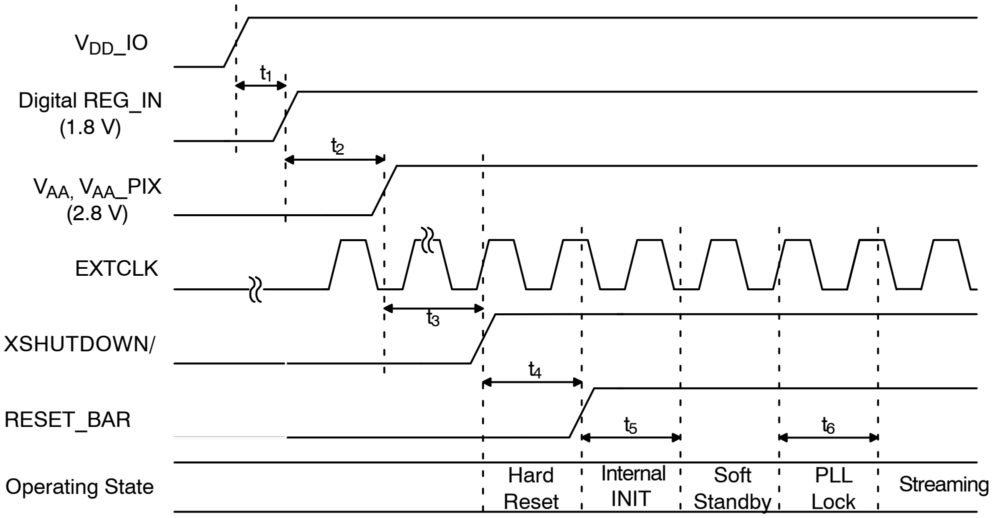
Figure 36. Power−Up Sequence with Pin−Constrained Mode

Table 24. POWER−UP SIGNAL TIMING WITH PIN−CONSTRAINED MODE

Parameter	Symbol	Min	Typ	Max	Unit
VDD_IO to Digital REG_IN 1.8 V	t1	0	−	10	ms
Digital REG_IN 1.8 V to Enable EXTCLK	t2	0	−	10	ms
Enable EXTCLK to Hard Reset Assertion	t3	0	−	100	ms
Hard Reset to VAA/VAA_PIX	t4	1	−	500	ms
Internal Initialization	t5	1	−	−	ms
PLL Lock Time	t6	1	−	−	ms

XSHUTDOWN/RESET_BAR Pin−unconstrained Mode

1. Turn on VDD_IO power supply.
2. After 0−10 ms, turn on Digital REG_IN power supply.
3. After 1−500 ms, turn on VAA/VAA_PIX power supplies and enable EXTCLK.
4. After 1 ms, assert XSHUTDOWN (High).
5. After 1 ms, assert RESET_BAR (High).
6. Wait 1ms for internal initialization into soft standby.
7. Configure PLL, output and image settings to desired values.
8. Set mode_select = 1 (R0x0100).
9. Wait 1ms for the PLL to lock before streaming state is reached.



Note: If the AR0542 two−wire serial interface is also used for communication with other devices, the status of SDATA during power−up needs to be considered at the system level due to the sensor's interaction during this time (t0 to t3) driving it to the low state; if the AR0542 two−wire serial interface is used for a dedicated point−point connection to the host, no additional considerations apply.

Figure 37. Power−Up Sequence with Pin−unconstrained Mode

Table 25. POWER−UP SIGNAL TIMING WITH PIN−UNCONSTRAINED MODE

Parameter	Symbol	Min	Typ	Max	Unit
VDD_IO to Digital REG_IN 1.8 V	t1	0	−	10	ms
Digital REG_IN (1.8V) to VAA, VAA_PIX (2.8 V)	t2	1	−	500	ms
Running EXTCLK to XSHUTDOWN Assertion	t3	1	−	−	ms
XSHUTDOWN High to RESET_BAR Assertion	t4	1	−	−	ms
Internal Initialization	t5	1	−	−	ms
PLL Lock Time	t6	1	−	−	ms

Power−Down Sequence

The recommended power−down sequence for the AR0542 is shown in Figure 38. The available power supplies—VDD_IO, Digital 1.8 V, VAA, VAA_PIX−can be turned off at the same time or have the separation specified below.

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. Assert hard reset by setting XSHUTDOWN/RESET_BAR to a logic “0.”
4. Turn off the VAA/VAA_PIX power supplies.
5. After 0–500 ms, turn off Digital 1.8 V power supply.
6. After 0–500 ms, turn off VDD_IO power supply.

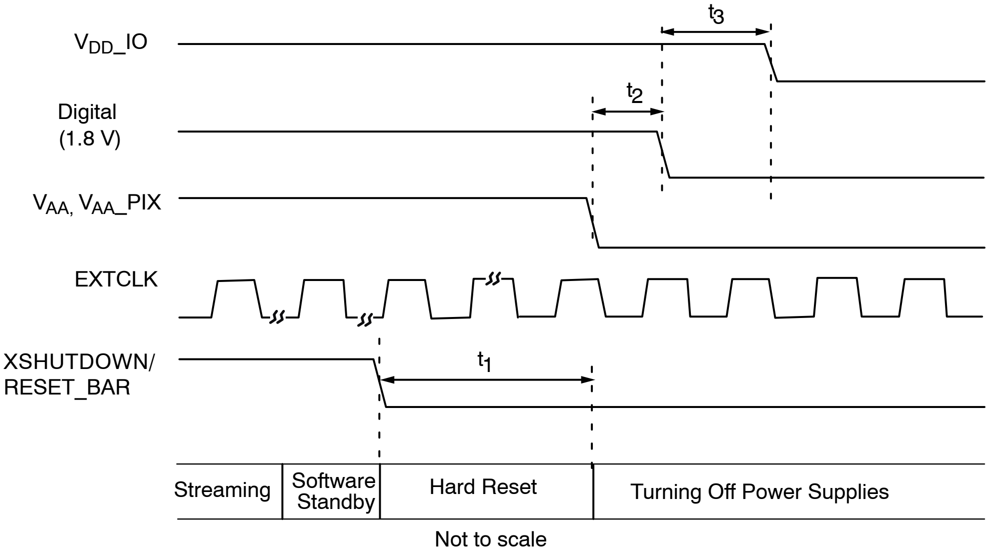


Figure 38. Power−Down Sequence

Table 26. POWER−DOWN SEQUENCE

Parameter	Symbol	Min	Typ	Max	Unit
XSHUTDOWN/RESET_BAR to VAA/VAA_PIX	t1	0	–	500	ms
VAA/VAA_PIX to Digital 1.8 V Time	t2	0	–	500	ms
Digital 1.8 V Time to VDD_IO	t3	0	–	500	ms

Hard Standby

The hard standby state is reached by the assertion of the XSHUTDOWN pad. There are two hard standby entering and exiting sequences for the AR0542 based on the XSHUTDOWN and RESET_BAR one−pin (pin−constrained mode) or two−pin (pin−unconstrained mode) control mode. Register values are not retained by this action, and will be returned to their default values once the sensor enters the hard standby state. The details of the sequence of the sequence for entering hard standby and exiting from hard standby are described below and shown in Figure 40 and 41.

XSHUTDOWN/RESET_BAR Pin−constrained Mode

< Entering Hard Standby >

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. De−assert XSHUTDOWN/RESET_BAR (Low) to enter the hard standby.
4. The sensor remains in hard standby state if XSHUTDOWN/RESET_BAR remains in the logic ”0” state.

< Exiting Hard Standby >

1. Turn off VAA/VAA_PIX power−supplies and enable EXTCLK if it was disabled.
2. After 1ms, assert XSHUTDOWN/RESET/BAR (High).
3. After 1ms, turn on VAA/VAA_PIX power−supplies.
4. Follow the pin−constrained power−up sequence from step 6 to 9 for output streaming.

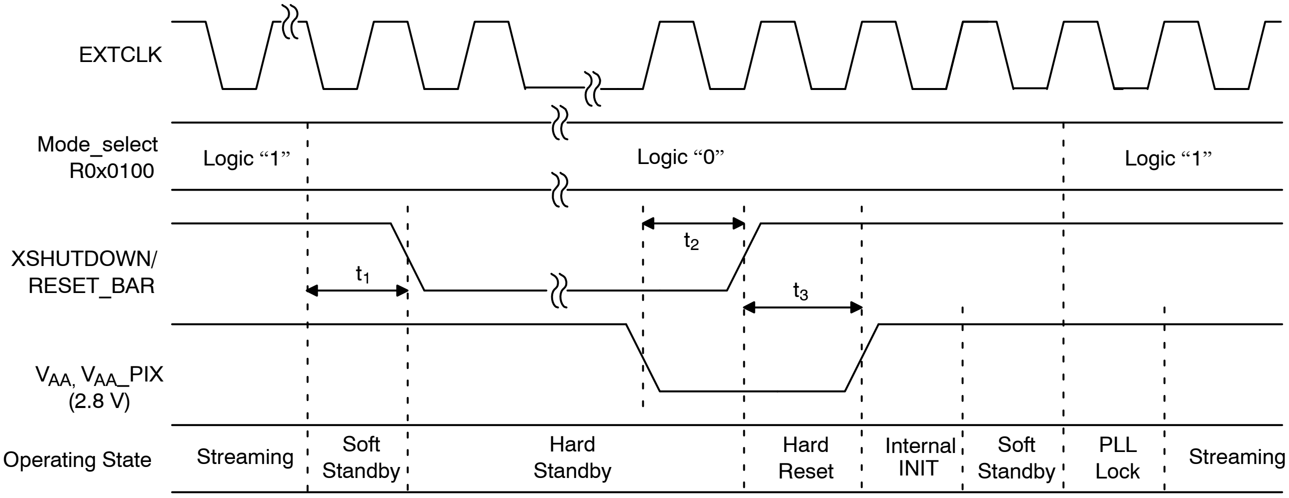


Figure 39. Hard Standby with Pin−constrained Mode

Table 27. HARD STANDBY WITH PIN−CONSTRAINED MODE

Parameter	Symbol	Min	Typ	Max	Unit
Enter Soft Standby to XSHUTDOWN/ RESET_BAR De−assertion	t1	1	–	–	ms
Turn off VAA/VAA_PIX to XSHUTDOWN/ RESET_BAR Assertion	t2	1	–	–	ms
XSHUTDOWN Assertion to Turn on VAA/ VAA_PIX Supplies	t3	1	–	–	ms

XSHUTDOWN/RESET_BAR Pin−unconstrained Mode

< Entering Hard Standby >

1. Disable streaming if output is active by setting mode_select = 0 (R0x0100).
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. De−assert XSHUTDOWN (Low) to enter the hard standby.
4. The sensor remains in hard standby state if XSHUTDOWN remains in the logic "0" state.

< Exiting Hard Standby >

1. De−assert RESET_BAR (Low) and enable EXTCLK if it was disabled.
2. After 1ms, assert XSHUTDOWN (High).
3. After 1ms, assert RESET_BAR (High).
4. Follow the pin−unconstrained power−up sequence from step 6 to 9 for output streaming.

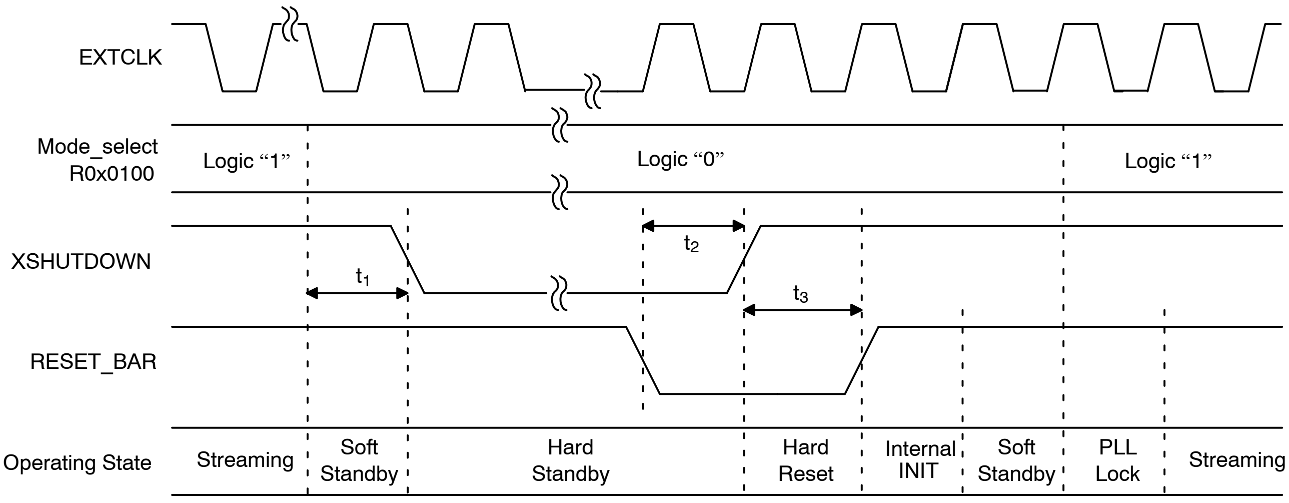


Figure 40. Hard Standby with Pin−unconstrained Mode

Table 28. HARD STANDBY WITH PIN−UNCONSTRAINED MODE

Parameter	Symbol	Min	Typ	Max	Unit
Enter Soft Standby to XSHUTDOWN De−assertion	t1	1	–	–	ms
RESET_BAR De−assertion to XSHUTDOWN Assertion	t2	1	–	–	ms
XSHUTDOWN Assertion to RESET_BAR Assertion	t3	1	–	–	ms

Soft Standby and Soft Reset

The AR0542 can reduce power consumption by switching to the soft standby state when the output is not needed. Register values are retained in the soft standby state. Once this state is reached, soft reset can be enabled optionally to return all register values to the default. The details of the sequence are described below and shown in Figure 41.

Soft Standby

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

Soft Reset

1. Follow the soft standby sequence list above.
2. Set software_reset = 1 (R0x0103) to start the internal initialization sequence.
3. After 2400 EXTCLKs, the internal initialization sequence is completed and the current state returns to soft standby automatically. All registers, including software_reset, return to their default values.

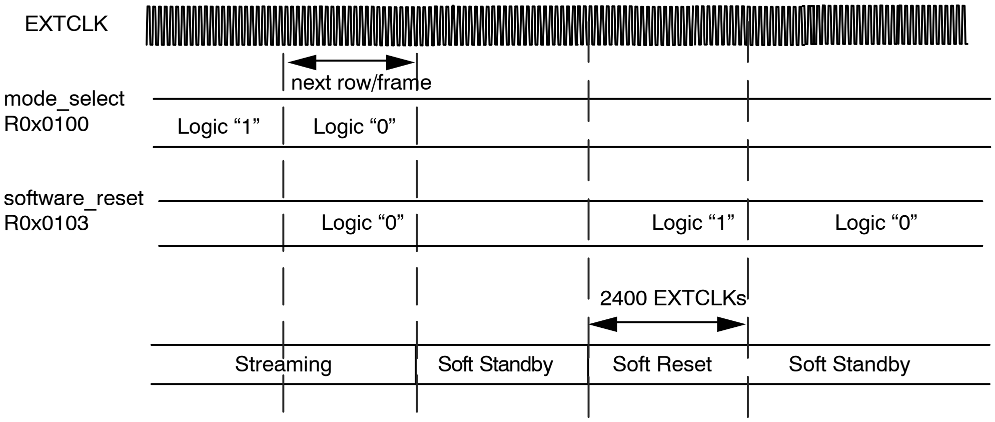


Figure 41. Soft Standby and Soft Reset

Internal VCM Driver

The AR0542 utilizes an internal Voice Coil Motor (VCM) driver. The VCM functions are register−controlled through the serial interface.

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

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

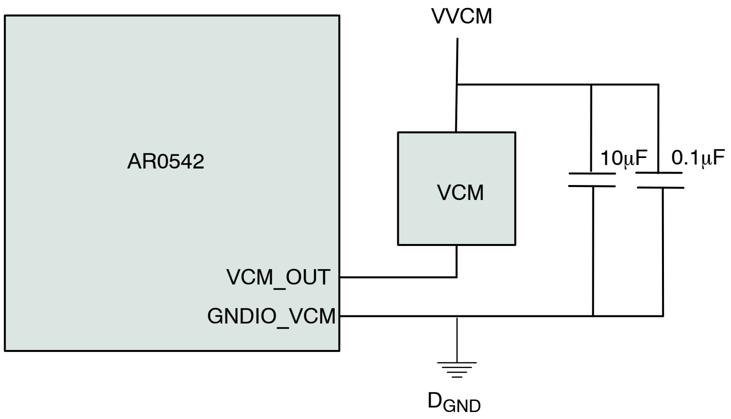
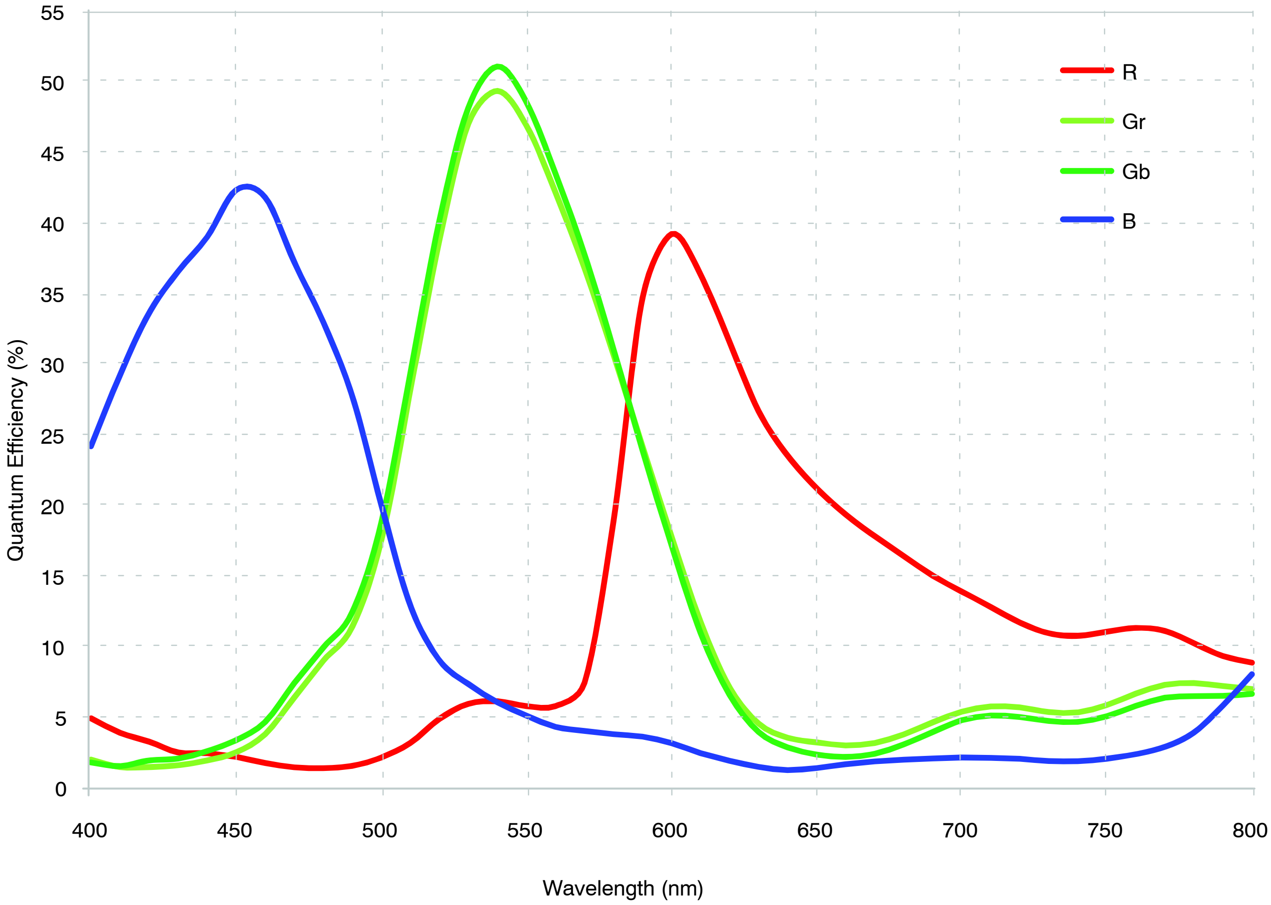


Figure 42. VCM Driver Typical Diagram

Table 29. VCM DRIVER TYPICAL

Characteristic	Parameter	Minimum	Typical	Maximum	Units
VCM_OUT	Voltage at VCM Current Sink	2.5	2.8	3.3	V
WVCM	Voltage at VCM Actuator	2.5	2.8	3.3	V
INL	Relative Accuracy	–	1.5(±)	4(±)	LSB
RES	Resolution	–	8	–	bits
DNL	Differential Nonlinearity	−1	–	1	LSB
IVCM	Output Current	88	100	110	mA

SPECTRAL CHARACTERISTICS



Note: **onsemi** recommends 670 ±5nm IR cut filter for AR0542. Refer to Table 30 for the IRCF specification.

Figure 43. Quantum Efficiency

Table 30. RECOMMENDED IR CUT LIMITS

Wavelength	Recommended Limits
<400 nm	Not specified
400−650 nm	>90%
Cut−off wavelength	670 ±5 nm
720−900 nm	Equal or less than 2%
900−1000 nm	Equal or less than 0.1%
1000−1050 nm	Equal or less than 0.03%
1050−1150 nm	Equal or less than 0.05%
>1150 nm	Not specified

CRA vs. Image Height Plot

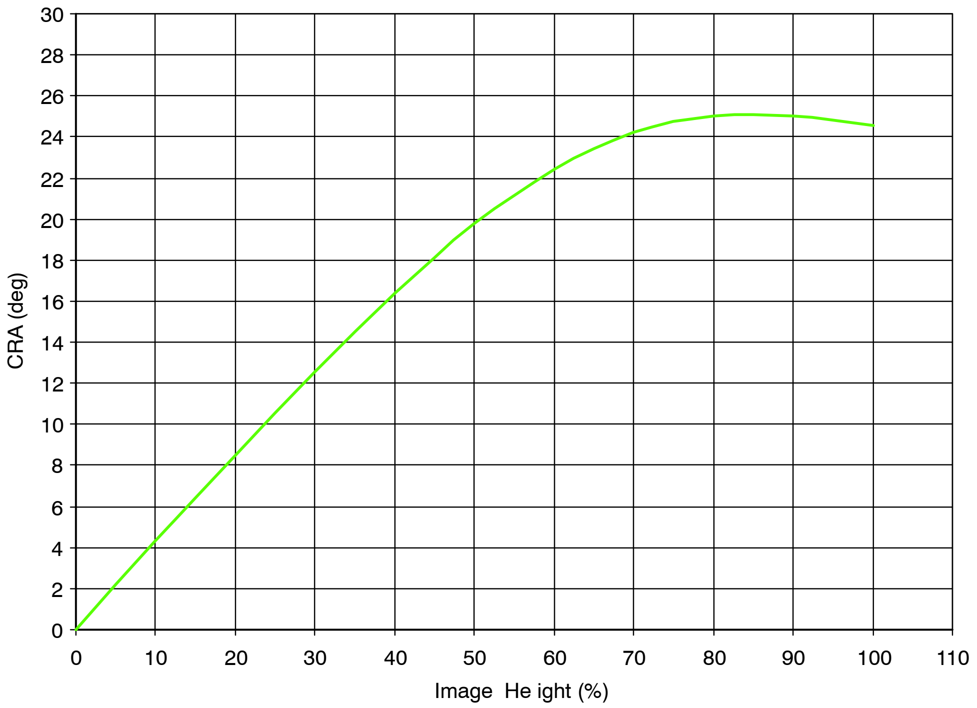


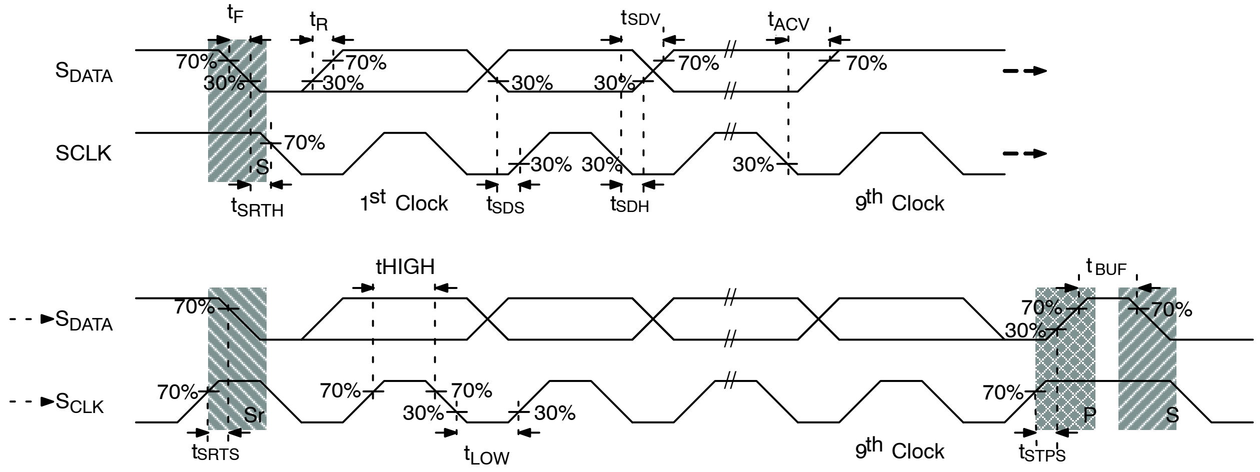
Figure 44. Chief Ray Angle (CRA) vs. Image Height

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

ELECTRICAL CHARACTERISTICS

Two–Wire Serial Register Interface

The electrical characteristics of the two–wire serial register interface (SCLK, SDATA) are shown in Figure 45 and Table 31. The SCLK and SDATA signals feature fail–safe input protection, Schmitt trigger input, and suppression of input pulses of less than 50 ns.



Note: Read sequence: For an 8–bit READ, read waveforms start after the WRITE command and register addresses are issued.

Figure 45. Two–Wire Serial Bus Timing Parameters

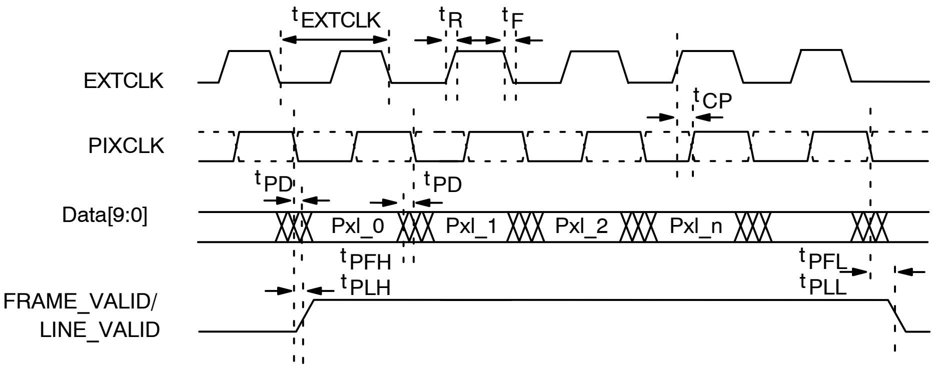
Table 31. TWO–WIRE SERIAL INTERFACE ELECTRICAL CHARACTERISTICS
(^{f}EXTCLK = 24 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output load = 68.5 pF; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
V_{IL}	Input LOW Voltage		0.85	0.898	0.96	V
I_{IL}	Input Leakage Current	No Pull Up Resistor; V_{IN} = V_{DD}_IO or DGND	10		14	μA
V_{OL}	Output LOW Voltage	At Specified 2 mA	0	0.054	0.58	V
I_{OL}	Output LOW Current	At specified V_{OL} 0.1 V			6	mA
C_{IN}	Input Pad Capacitance				6	pf
C_{LOAD}	Load Capacitance				N/A	pf

Table 32. TWO–WIRE SERIAL INTERFACE TIMING SPECIFICATION

Symbol	Parameter	Min	Max	Unit
f_{SCLK}	SCLK Frequency	0	400	kHz
t_{HIGH}	SCLK High Period	0.6		μs
t_{LOW}	SCLK Low Period	1.3		μs
t_{SRTS}	Start Setup Time	0.6		μs
t_{SRTH}	Start Hold Time	0.6		μs
t_{SDS}	Data Setup Time	100		ns
t_{SDH}	Data Hold Time	0	(Note 1)	μs
t_{SDV}	Data Valid Time		0.9	μs
t_{ACV}	Data Valid Acknowledge Time		0.9	μs
t_{STPS}	Stop Setup Time	0.6		μs
t_{BUF}	Bus Free Time between STOP and START	1.3		μs
t_R	SCLK and SDATA Rise Time		300	ns
t_F	SCLK and SDATA Fall Time		300	ns

1. Maximum t_{SDH} could be 0.9 μs, but must be less than maximum of t_{SDV} and t_{ACV} by a transition time.



Note: FRAME_VALID assertion leads LINE_VALID assertion by 6 PIXCLK periods

Note: FRAME_VALID negation trails LINE_VALID negation by 6 PIXCLKs.

Note: PLL disabled for ^{t}CP.

Figure 46. Parallel Data Output Timing Diagram

EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 33. The EXTCLK input supports an AC−coupled sine−wave input clock or a DC−coupled square−wave input clock.

If EXTCLK is AC−coupled to the AR0542 and the clock is stopped, the EXTCLK input to the AR0542 must be driven to ground or to VDD_IO. Failure to do this will result in excessive current consumption within the EXTCLK input receiver.

Table 33. ELECTRICAL CHARACTERISTICS (EXTCLK)
(^{f}EXTCLK = 24 MHz; ^{f}PIXCLK = 84 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output load = 68.5 pF; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
^{f}EXTCLK1	Input Clock Frequency	PLL Enabled	6		27	MHz
tEXTCLK1	Input Clock Period	PLL Enabled	37		167	ns
^{t}R	Input Clock Rise Slew Rate			2.9	8*	ns
^{t}F	Input Clock Fall Slew Rate			2.7	8*	ns
VIN_AC	Input Clock Minimum Voltage Swing (AC Coupled)		0.5			Vpp
VIN_DC	Input Clock Maximum Voltage Swing (DC Coupled)				2.3	V
^{f}CLKMAX(AC)	Input Clock Signaling Frequency (Low Amplitude)	VIN = VIN_AC (MIN)			12	MHz
^{f}CLKMAX(DC)	Input Clock Signaling Frequency (Full Amplitude)	VIN = VDD_IO			27	MHz
	Clock Duty Cycle		35	50	65	%
^{t}JITTER	Input Clock Jitter	Cycle−to−cycle			600	ps
^{t}LOCK	PLL VCO Lock Time			0.2	1	ms
CIN	Input Pad Capacitance			3		pF
IIH	Input HIGH Leakage Current		1.36	1.89	3	mA
VIH	Input HIGH Voltage		1.26		2.3	V
VIL	Input LOW Voltage		−0.5		0.5	V

*For additional information on our Pb−Free strategy and soldering details, please download the **onsemi** Soldering and Mounting Techniques Reference Manual, SOLDERRM/D.

Parallel Pixel Data Interface

The electrical characteristics of the parallel pixel data interface (FV, LV, DOUT[9:0], PIXCLK, SHUTTER, and FLASH outputs) are shown in Table 34.

Table 34. ELECTRICAL CHARACTERISTICS (PARALLEL PIXEL DATA INTERFACE)
(^{f}EXTCLK = 24 MHz; ^{f}PIXCLK = 84 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output load = 68.5 pF; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
VOH	Output HIGH Voltage			–		V
VOL	Output LOW Voltage			–		V
IOH	Output HIGH Current			–		mA
IOL	Output LOW Current			–		mA
IOZ	Tri−state Output Leakage Current			–		mA
tCP	EXTCLK to PIXCLK Propagation Delay	PLL Bypass, EXTCLK = 27 MHz		26.519		ns
	Output Pin Slew (Rising)	CLOAD = 25 pF		1.4		V/ns
	Output Pin Slew (Falling)	CLOAD = 250 pF		1.5		V/ns
^{t}PD	PIXCLK to Data Valid	PLL Bypass, EXTCLK = 27 MHz		1		ns
^{f}PIXCLK	PIXCLK Frequency	Default		48		MHz

Table 34. ELECTRICAL CHARACTERISTICS (PARALLEL PIXEL DATA INTERFACE) (continued)
(^{f}EXTCLK = 24 MHz; ^{f}PIXCLK = 84 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output load = 68.5 pF; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
^{t}PFH	PIXCLK to FV HIGH	PLL Bypass, EXTCLK = 27 MHz		1		ns
^{t}PLH	PIXCLK to LV HIGH	PLL Bypass, EXTCLK = 27 MHz		1		ns
^{t}PFL	PIXCLK to FV LOW	PLL Bypass, EXTCLK = 27 MHz		1		ns
^{t}PLL	PIXCLK to LV LOW	PLL Bypass, EXTCLK = 27 MHz		1		ns

Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLK_P, CLK_N,DATA0_P, DATA1_P, DATA0_N, and DATA1_N) are shown in Table 35 and Table 36.

To operate the serial pixel data interface within the electrical limits of the CSI−2 specification, V_{DD}_IO (I/O digital voltage) is restricted to operate in the range 1.7–1.9 V. All MIPI specifications are with sensor operation using on−chip internal regulator.

Table 35. HS TRANSMITTER DC SPECIFICATIONS

Symbol	Parameter	Min	Nom	Max	Unit
V_{CMTX} (Note 2)	HS transmit static common−mode voltage	150	200	250	mV
$^{\Delta V}$CMTX$_{(1,0)}$ (Note 3)	V_{CMTX} mismatch when output is Differential−1 or Differential−0			5	mV
ΔV_{OD} (Note 2)	HS transmit differential voltage	140	200	270	mV
ΔV_{OD} (Note 3)	V_{OD} mismatch when output is Differential−1 or Differential−0			10	mV
V_{OHHS} (Note 2)	HS output high voltage			360	mV
Z_{OS}	Single ended output impedance	40	50	62.5	Ω
|ΔZOS|	Single ended ouput impedance mismatch			20	%

2. Value when driving into load impedance anywhere in the Z_{ID} range.
3. It is recommended that the implementer minimize ΔV_{OD} and $\Delta V_{CMTX(1,0)}$ in order to minimize radiation and optimize signal integrity.

Table 36. HS TRANSMITTER AC SPECIFICATIONS

Symbol	Parameter	Min	Nom	Max	Unit
$^{\Delta V}$CMTX$_{(HF)}$	HS transmit static common−mode voltage			15	mV$_{RMS}$
$^{\Delta V}$CMTX$_{(LF)}$	V_{CMTX} mismatch when output is Differential−1 or Differential−0			25	mV$_{PEAK}$
^{t}R and ^{t}F	20%−80% rise time and fall time (Note 5)			0.3	UI
		150			ps

4. UI is equal to 1/(2*fh).
5. Excess capacitance not to exceed 4 pF on each pin.

Table 37. LP TRANSMITTER DC SPECIFICATIONS

Symbol	Parameter	Min	Nom	Max	Unit
V_{OH}	HS transmit static common−mode voltage	1.1	1.2	1.3	V
V_{OL}	V_{CMTX} mismatch when output is Differential−1 or Differential−0	−50		50	mV
Z_{OLP}	20%−80% rise time and fall time (Note 6)	110			Ω

6. Though no maximum value for Z_{OLP} is specified, the LP transmitter output impedance shall ensure the T_{RLP}/T_{FLP} is met.

Table 38. LP TRANSMITTER AC SPECIFICATIONS

Parameter	Description	Min	Max	Unit
TRLP/TFLP	15%−80% rise time and fall time (Note 7)		25	ns
TREOT	30%−85% rise time and fall time (Notes 7, 11, 12)		35	ns
σV/σtsr	Slew rate @ CLOAD = 70 pF (Falling edge only), (Notes 7, 9, 13, 14)		150	mV/ns
	Slew rate @ CLOAD = 70 pF (Rising edge only), (Notes 7, 8, 9)			mV/ns

7. CLOAD includes the low−frequency equivalent transmission line capacitance. The capacitance of TX and RX are assumed to always be <10 pF. The disturbed line capacitance can up to 50 pF for a transmission line with 2 ns delay.
8. When the ouput voltage is between 400 mV and 930 mV.
9. Measured as average across any 50 V segment of the output signal transition.
10. This parameter value can be lower than TLPX due to differences in the rise vs. fall signal slopes and trip levels and mismatches between Dp and Dn transmitters. ANY LP transmitters. Any LP exclusive−OR pulse observed during HS EoT (transition from HS level to LP−1) is glitch behavior.
11. The rise time of TREOT starts from the HS common−Level at the moment the differential amplitude drops below 70 mV, due to stopping the differential drive.
12. With an additional load capacitance CCM between 0 and 60 pF on the termination center tap at RX side of the Lane.
13. This value represents a corner point in a piecewise linear curve.
14. When the output voltage is in the range specified by $V_{PIN(absmax)}$
15. When the output voltage is between 400 mV and 700 mV
16. When VOINST is the instantaneous output voltage, VDP or VDN in millivolts.
17. When the output voltage is between 700 mV and 930 mV

High Speed Clock Timing

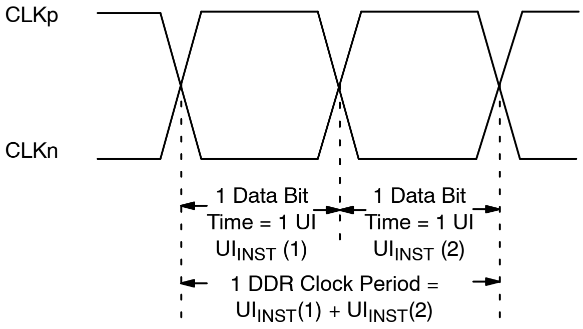


Figure 47. High Speed Clock Timing

Table 39. DC ELECTRICAL CHARACTERISTICS (CONTROL INTERFACE)

Parameter	Description	Min	Typ	Max	Unit
UI Instantaneous (Note 18, 19)	UIINST			12.5	ns

18. This value corresponds to a minimum 80 Mbps data rate.
19. The minimum UI shall not be violated for any single bit period, for example any DDR half cycle within a data burst.

Data Clock Timing Specification

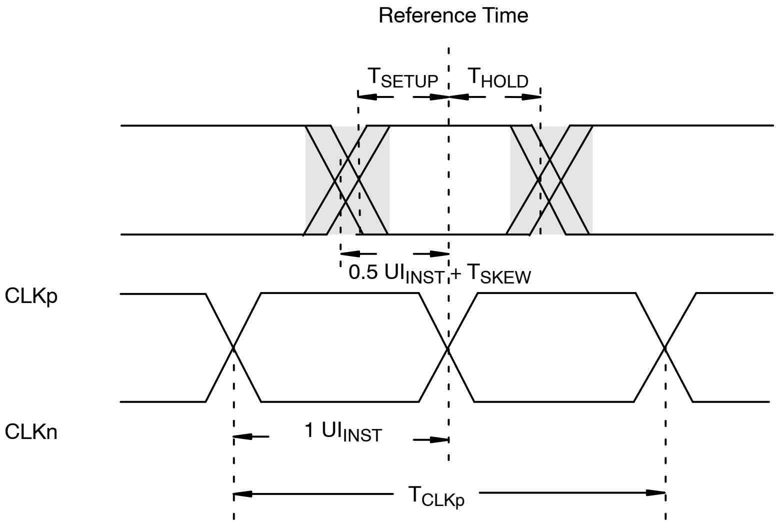


Figure 48. Data Clock Timing

Table 40. DATA−CLOCK TIMING SPECIFICATIONS

Parameter	Symbol	Min	Typ	Max	Unit
Data to Clock Skew (Measured at Transmitter)	$T_{SKEW[TX]}$	−0.15		0.15	UI_{INST}

20. Total silicon and package delay of 0.3*UI_{INST}.

Control Interfaces

The electrical characteristics of the control interface (RESET_BAR, TEST, GPI0, GPI1, GPI2, and GPI3) are shown in Table 41.

Table 41. DC ELECTRICAL CHARACTERISTICS (CONTROL INTERFACE)

(^{f}EXTCLK = 24 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output load = 68.5 pF; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
V_{IH}	Input HIGH Voltage		1.26		2.3	V
V_{IL}	Input LOW Voltage		−0.5		0.5	V
I_{IN}	Input Leakage Current	No Pull−up Resistor; V_{IN} = V_{DD}_IO or D_{GND}			10	μA
C_{IN}	Input Pad Capacitance			3		pF

Operating Voltages

VAA and VAA_PIX must be at the same potential for correct operation of the AR0542.

Table 42. DC ELECTRICAL DEFINITIONS AND CHARACTERISTICS

(^{f}EXTCLK = 24 MHz; REG_IN = 1.8 V; V_{DD}_TX = 1.8 V; V_{DD}_IO = 1.8 V; V_{AA} = 2.8 V; V_{AA}_PIX = 2.8 V; Output Load = 68.5 pF; Using internal Regulator; T_J = 70°C)

Symbol	Parameter	Condition	MIN	TYP	MAX	Unit
REG_IN	1.8 V Supply Voltage		1.7	1.8	1.9	
VDD_TX	PHY Digital Voltage		1.7	1.8	1.9	V
VDD_IO	I/O Digital Voltage	Parallel pixel data interface	1.7	1.8	1.9	V
			2.4	2.8	3.1	V
VAA	Analog Voltage		2.6	2.8	3.1	V
VAA_PIX	Pixel Supply Voltage		2.6	2.8	3.1	V
I_REGIN/TX	1.8 V Digital Current	Streaming, full resolution Parallel 15 FPS	29	35	44	mA
IDD_IO(1.8V)	I/O Digital Current		20	24	38	
IDD_IO(2.8)	I/O Digital Current		30	45	67	
IAA/IAA_PIX	Analog Current		50	65	85	
I_REGIN/TX	1.8 V Digital Current	Streaming, full resolution MIPI 15 FPS	24	26.5	44	mA
IDD_IO	I/O Digital Current		0.007	0.04	0.08	
IAA/IAA_PIX	Analog Current		45	60	85	
I_REGIN/TX	1.8 V Digital Current	Streaming, 1296x972 (xy_bin) resolution Parallel 30 FPS	21	23.5	30	mA
IDD_IO(1.8V)	I/O Digital Current		12	13.5	16	
IDD_IO(2.8)	I/O Digital Current		15	22	31	
IAA/IAA_PIX	Analog Current		50	65	85	
I_REGIN/TX	1.8 V Digital Current	Streaming, 1296x972 (xy_bin) resolution MIPI 30 FPS	15	18.5	30	mA
IDD_IO	I/O Digital Current		0.007	0.03	0.08	
IAA/IAA_PIX	Analog Current		50	65	85	
	Hard Standby (Clock on at 24 MHz)	STANDBY current when asserting XSHUTDOWN signal				
	Analog Current		0.3	1	4	μA
	Digital Current		1.5	2	6	μA
	Hard Standby (Clock Off)					
	Analog Current		0.3	1	4	μA
	Digital Current		1.5	2	6	μA
	Soft Standby (Clock On at 24 MHz)	STANDBY current when asserting R0x100 = 1				
	Analog Current		15	41	90	μA
	Digital Current		4	4.8	7.5	mA
	Soft Standby (Clock Off)					
	Analog Current		15	41	90	μA
	Digital Current		3.5	4.2	7	mA

21. Digital Current includes REG_IN, as the regulator is still operating in soft standby mode.

Table 43. **ABSOLUTE MAXIMUM VALUES**

Symbol	Parameter	MIN	MAX	Unit
VDD1V8(REG_IN)	1.8 V Digital Voltage	–0.3	2.1	V
VDD_TX	PHY Digital Voltage	–0.3	2.1	V
VDD_IO	I/O Digital Voltage	–0.3	3.5	V
VAA	Analog Supply Voltage	–0.3	3.5	V
VAA_PIX	Pixel Supply Voltage	–0.3	3.5	V
T_OP	Operating Temperature Measured at Junction	–30	70	°C
T_STG	Storage Temperature	–40	85	°C

Stresses exceeding those listed in the Maximum Ratings table may damage the device. If any of these limits are exceeded, device functionality should not be assumed, damage may occur and reliability may be affected.

SMIA and MIPI Specification Reference

The sensor design and this documentation is based on the following reference documents:

- SMIA Specifications:
  - SMIA 1.0 Part 1: Functional Specification (Version 1.0 dated 30 June 2004)
  - SMIA 1.0 Part 1: Functional Specification ECR0001 (Version 1.0 dated 11 Feb 2005)
- MIPI Specifications:
  - MIPI Alliance Standard for CSI−2 version 1.0
  - MIPI Alliance Specification for D−PHY Version 1.00.00 – 14 May 2009

onsemi, onsemi, , and other names, marks, and brands are registered and/or common law trademarks of Semiconductor Components Industries, LLC dba “**onsemi**” or its affiliates and/or subsidiaries in the United States and/or other countries. **onsemi** owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of **onsemi**’s product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent–Marking.pdf. **onsemi** reserves the right to make changes at any time to any products or information herein, without notice. The information herein is provided “as–is” and **onsemi** makes no warranty, representation or guarantee regarding the accuracy of the information, product features, availability, functionality, or suitability of its products for any particular purpose, nor does **onsemi** assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using **onsemi** products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by **onsemi**. “Typical” parameters which may be provided in **onsemi** data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. **onsemi** does not convey any license under any of its intellectual property rights nor the rights of others. **onsemi** products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use **onsemi** products for any such unintended or unauthorized application, Buyer shall indemnify and hold **onsemi** and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that **onsemi** was negligent regarding the design or manufacture of the part. **onsemi** is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner.

ADDITIONAL INFORMATION

TECHNICAL PUBLICATIONS:
Technical Library: www.onsemi.com/design/resources/technical–documentation
onsemi Website: www.onsemi.com

ONLINE SUPPORT: www.onsemi.com/support
For additional information, please contact your local Sales Representative at
www.onsemi.com/support/sales

◊