1/3.2-Inch 8Mp CMOS Digital Image Sensor

AR0833 Data Sheet

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

Features

- High-speed sensor supporting 8Mp (4:3) 30 fps still images and full HD 1080p30 video
- 1.4µ pixel with Aptina™ A-PixHS™ technology providing best-in-class low-light performance.
- Optional on-chip high-quality Bayer scaler to resize image to desired size
- Data interfaces: two-, three-, and four-lane serial mobile industry processor interface (MIPI)
- Bit-depth compression available for MIPI Interface: 10-8 and 10-6 to enable lower bandwidth receivers for full frame rate applications
- On-chip temperature sensor
- On-die phase-locked loop (PLL) oscillator
- 4Kb one-time programmable memory (OTPM) for storing module information
- On-chip 8-bit VCM driver
- 3D synchronization controls to enable stereo video capture
- Interlaced multi-exposure readout enabling High Dynamic Range (HDR) still and video applications
- 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
- Support for external mechanical shutter
- Support for external LED or Xenon flash

Applications

- Smart phones
- PC cameras
- Tablets

Ordering Information

Table 1: Available Part Numbers

Part Number	Description
AR0833CS3C29SMD20	Bare die

Table 2: Key Performance Parameters

Parameter		Typical Value
Array Format		3264 x 2448
Primary modes		Full Resolution: 4:3 - 8Mp at 30 fps
		16:9 - 6Mp at 30 fps
		16:9 - 1080p HD at 30 fps
Pixel size		1.4µm Back Side Illuminated (BSI)
Optical format		1/3.2"
Die size		6.86mm x 6.44mm (Area: 44.17mm^2)
Input Clock Frequency		6 - 27 MHz
Interface		MIPI CSI-2 (2-, 3-, 4-lane modes supported.) 800 MHz max MIPI clock speed per lane.
Subsampling modes		X - Bin2, Sum2 Skip: 2x, 4x
		Y - Sum2, Skip: 2x, 4x, 8x
Output data depth		10 bits raw or 8/6-bit DPCM
Analog gain		1x, 2x, 3x, 4x, 6x, 8x
High Quality Bayer Scalar		Adjustable scaling up to 1/6x scaling
Temperature sensor		10-bit, single instance on chip, controlled by two-wire serial I/F
VCM AF driver		8-bit resolution with slew rate control
3-D support		Frame rate and exposure synchronization
Supply Voltage	Analog	2.5 - 3.1 V (2.8V nominal)
	Digital	1.14 - 1.3 V (1.2V nominal)
	Pixel	2.5 - 3.1 V (2.8V nominal)
	OTPM Read	1.7 - 1.9 V (1.8V nominal)
	I/O	1.7 - 1.9 V (1.8V nominal) or 2.5 - 3.1 V (2.8V nominal)
	MIPI	1.14 - 1.3 V (1.2V nominal)
Power consumption		350mW at 30 fps, 8Mp
Responsivity		0.6 V/lux-sec
SNR_{MAX}		36 dB
Dynamic Range		64 dB
Operating Temperature Range (at junction) -T_J		-30°C to +70°C

Table 3: Modes of Operation and Power Consumption at 100% FOV

Modes	Active Readout Window (col x row)	Sensor Output Resolution (col x row)	Mode	FPS	Power Consumption [mW][5]
Snapshot Mode					
Full Resolution 4:3 - 8Mp[1]	3264 x 2448	3264 x 2448	Full mode	30	350
Full Resolution[1] 16:9 - 6Mp	3264 x 1836	3265 x 1836	Full mode	30	320
4:3 Video Mode					
VGA[2]	3264 x 2448	640 x 480	Scaling	30	290
VGA[3]	3264 x 2448	640 x 480	Bin2 + Scaling	30	250
VGA	3264 x 2448	640x 480	Bin2 + Scaling	60	290
16:9 Video Mode					
1080p[2]	3264 x 1836	1920 x 1080	Scaling	30	300
1080p[2]+ EIS[4]	3264 x 1836	2304 x 1296	Scaling	30	300
720p[2]	3264 x 1836	1280 x 720	Scaling	30	300
720p[2]	3264 x 1836	1280 x 720	Bin2 + Scaling	60	250
720p[2] + EIS	3264 x 1836	1536 x 864	Scaling	30	300
WVGA[2]	3264 x 1836	8546x 480	Scaling	30	290
WVGA[2]	3265 x 1836	856 x 480	Bin2 + Scaling	60	290

Notes:
1. 732 Mbps/lane MIPI data transfer rate.
2. Scaled image using internal High Quality Bayer Scaler.
3. Low power preview.
4. Electronic Image Stabilization.
5. Values measured at T=25°C and nominal voltages.

Table of Contents

List of Figures

List of Tables

General Description

The Aptina AR0833 is a 1/3.2-inch BSI (back side illuminated) CMOS active-pixel digital image sensor with a pixel array of 3264H x 2448V (3280H x 2464V including border pixels). It incorporates sophisticated on-chip camera functions such as mirroring, column and row skip modes, and context switching for zero shutter lag snapshot mode. It is programmable through a simple two-wire serial interface and has very low power consumption.

The AR0833 digital image sensor features Aptina's breakthrough low-noise 1.4µm pixel 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 AR0833 sensor can generate full resolution image at up to 30 frames per second (fps). An on-chip analog-to-digital converter (ADC) generates a 10-bit value for each pixel.

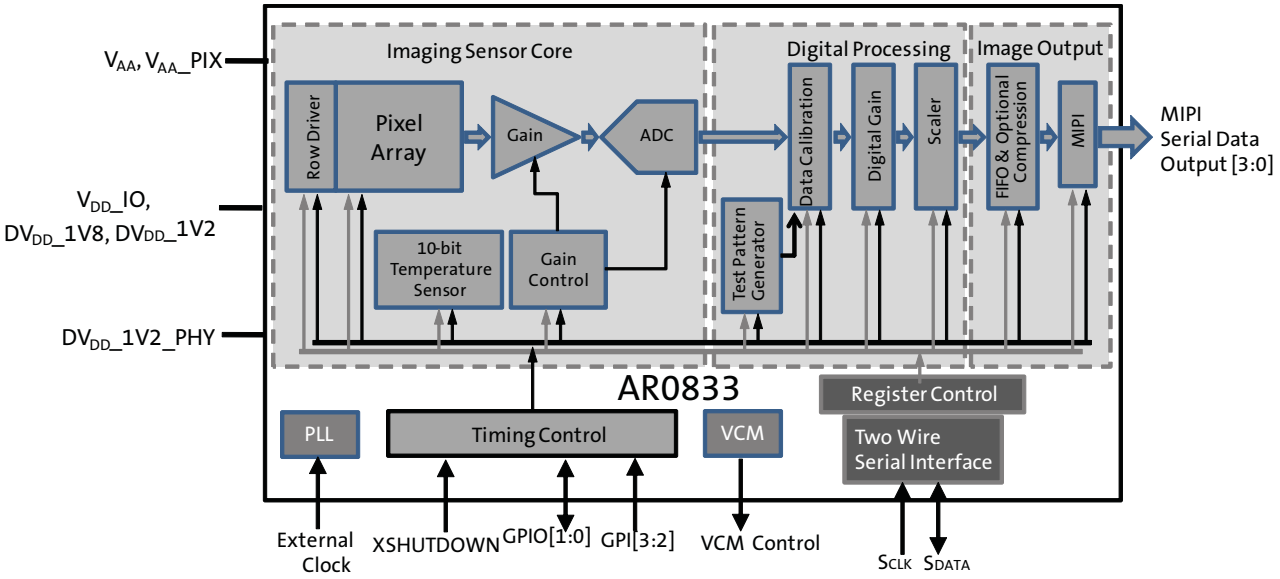
Functional Overview

In order to meet higher frame rates in AR0833 sensor, the architecture has been re-designed. The analog core has a column parallel architecture with 4 data paths. Digital block has been re-architected to have 4 data paths.

The chip is targeted to meet SMIA 85 module size. As a result, the final die size is 6.86mm x 6.44mm (singulated) which would fit in the intended module with two-sided pad frame.

Figure 1 shows the block diagram of the AR0833.

Figure 1: Top Level Block Diagram



The core of the sensor is an 8Mp active-pixel array. The timing and control circuitry sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and reading that row, the pixels in the row integrate incident light. The exposure is controlled by varying the time interval between reset and readout. Once a row has been read, the data from the columns is sequenced through an analog signal chain (providing 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.

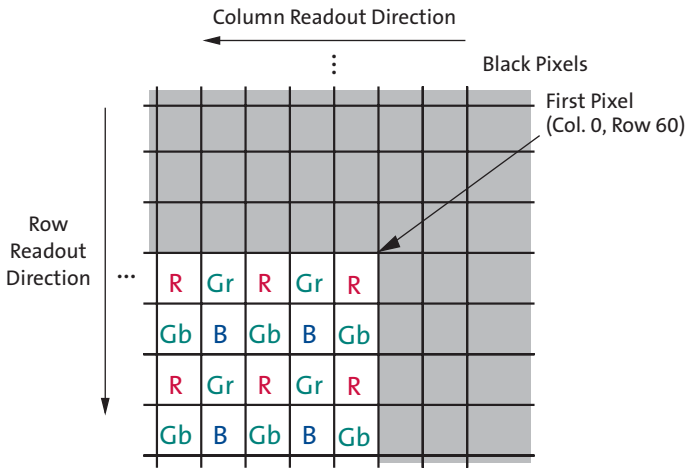
The output from the sensor is a Bayer pattern; alternate rows are a sequence of either green and red pixels or blue and green pixels. The offset and gain stages of the analog signal chain provide per-color control of the pixel data.

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

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 red and green pixels; odd-numbered columns contain blue and green pixels.

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

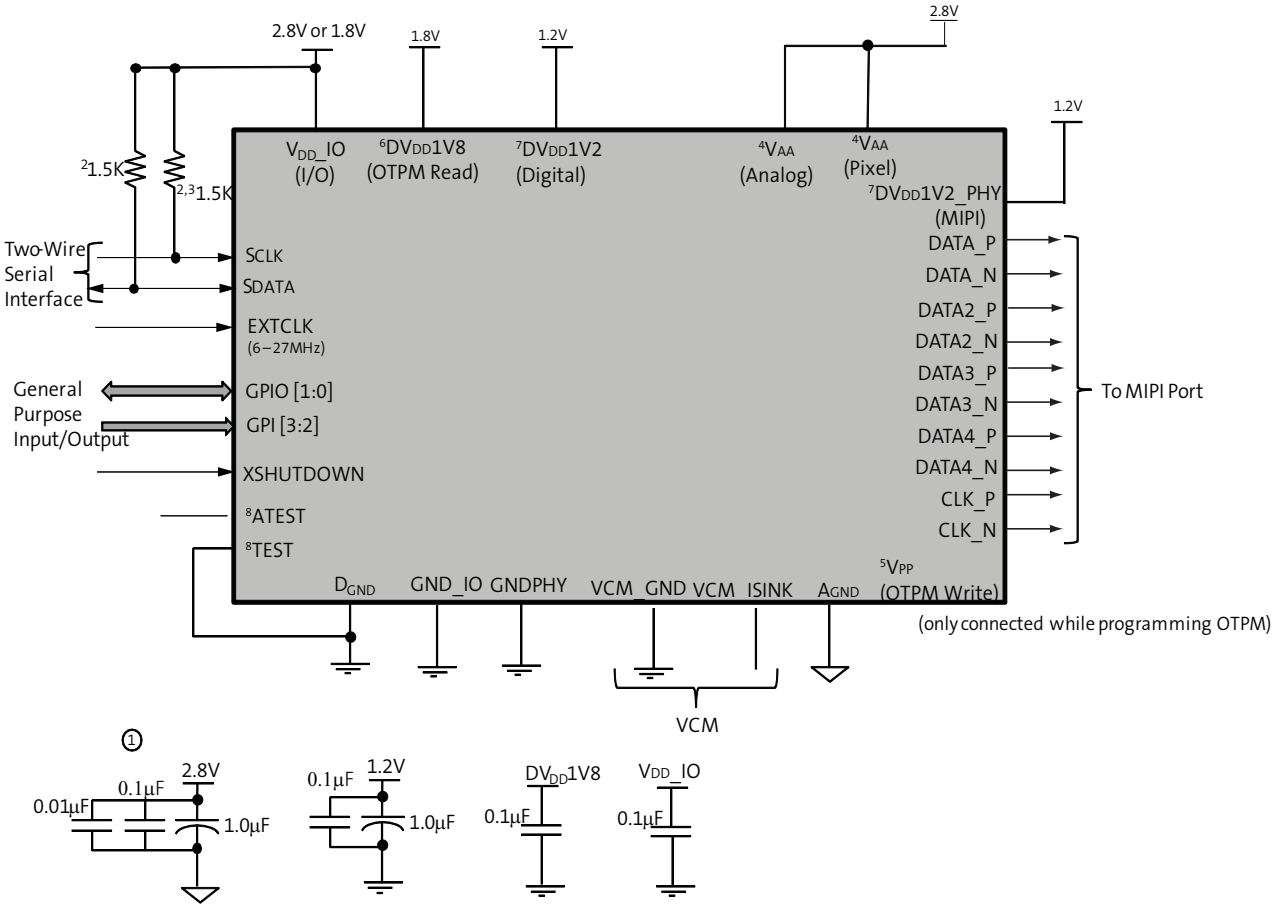


Note: By default the mirror bit is set, so the read-out direction is from left to right.

Typical Connections

The chip supports MIPI output protocol. MIPI can be configured in 4-, 3- or 2-lanes. There are no parallel data output ports.

Figure 3: Typical Application Circuit—MIPI Connection



For connectivity above:

1. All power supplies should be adequately decoupled; recommended cap values are:
   - 2.8v: 1.0µF, 0.1µF, and then 0.01µF
   - 1.2v: 1µF and 0.1µF
   - 1.8v: 0.1µF
2. Resistor value 1.5k is recommended, but may be greater for slower two-wire speed.
3. This pull-up resistor is not required if the controller drives a valid logic level on SCLK at all times.
4. VAA and VAA_PIX can be tied together. However, for noise immunity it is recommended to have them separate (i.e. two sets of 2.8V decoupling caps).
5. VPP, 6-7V, is used for programming OTPM. This pad is left unconnected if OTPM is not being programmed.
6. DVDD_1V8 can be combined with VDD_IO, if VDD_IO = 1.8V.
7. DVDD_1V2 and DVDD_1V2_PHY can be tied together.
8. ATEST can be left floating.

9. TEST pin must be tied to DGND.
10. DVDD_1V8 is the OTPM read voltage.

Signal Descriptions

AR0833 has 66 pads placed in a two sided pad frame. It has only serial outputs. The part may be configured as MIPI with different bit depths. The pad description is tabulated in Table 4:

Table 4: Pad Descriptions

Pad Name	Pad Type	Description
Sensor Control		
EXTCLK	Input	Master clock input; PLL input clock. 6 MHz - 27 MHz. This is a SMIA-compliant pad.
GPIO0	Input/Output	General Input and one Output function include: a. (Default Output) Flash b. (Input) all options in GPI2 High-Z before XSHUTDOWN going high; default value is '0' after all three voltages in place and XSHUTDOWN being high. After reset, this pad is not powered down since its default use is as Flash pin. If not used, can be left floating.
GPIO1	Input/Output	General Input and 2 Output functions include: a. (Default Output) Shutter b. (Output) 3-D daisy chain communication output c. (Input) all options in GPI2 High-Z before XSHUTDOWN going high; default value is '0' after all three voltages in place and XSHUTDOWN being high. After reset, this pad is not powered-down since its default use is as Shutter pin. If not used, can be left floating.
GPI2	Input	General Input; After reset, these pads are powered down by default; this means that it is not necessary to bond to these pads. Functions include: a. SADDR, switch to the second two-wire serial interface device address (see "Slave Address/Data Direction Byte" on page 19) b. Trigger signal for Slave Mode c. Standby If not used, can be left floating.
GPI3	Input	General Input; After reset, these pads are powered-down by default; this means that it is not necessary to bond to these pads. Functions include: a. 3-D daisy chain communication input b. All options in GPI2 If not used, can be left floating.
Two-wire Serial Interface		
SCLK	Input	Serial clock for access to control and status registers.
SDATA	I/O	Serial data for reads from and writes to control and status registers.
Serial Output		
DATA[4:1]P	Output	Differential serial data (positive).
DATA[4:1]N	Output	Differential serial data (negative).
CLK_P	Output	Differential serial clock/strobe (positive).
CLK_N	Output	Differential serial clock/strobe (negative).

Table 4: Pad Descriptions (continued)

Pad Name	Pad Type	Description
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.
VCM driver		
VCM_ISINK	Input/Output	VCM Driver current sink output. If not used, it could be left floating.
VCM_GND	Input/Output	Ground connection to VCM Driver. If not used, needs to be connected to ground (DGND). This ground must be separate from the other grounds.
Power		
VPP	Input/Output	High-voltage pin for programming OTPM, present on sensors with that capability. This pin can be left floating during normal operation.
VAA[7:1], VAA_PIX[2:1], AGND[9:1], VDD_IO_[4:1], GND_IO, DGND_[6:1] DVDD_1V2_[9:1], DVDD_1V2_PHY_[2:1], GNDPHY_[2:1], DVDD_1V8	Supply	Power supply. The domains are specified in the next table. The brackets indicate the number of individual pins.

There are standard GPI and GPIO pads, 2 each. Chip can also be communicated to through the two-wire serial interface.

The chip has three unique power supply requirements: 1.2V, 1.8V,and 2.8V. These are further divided and in all there are seven power domains and five independent ground domains from the ESD perspective.

Table 5: Independent Power and Ground Domains

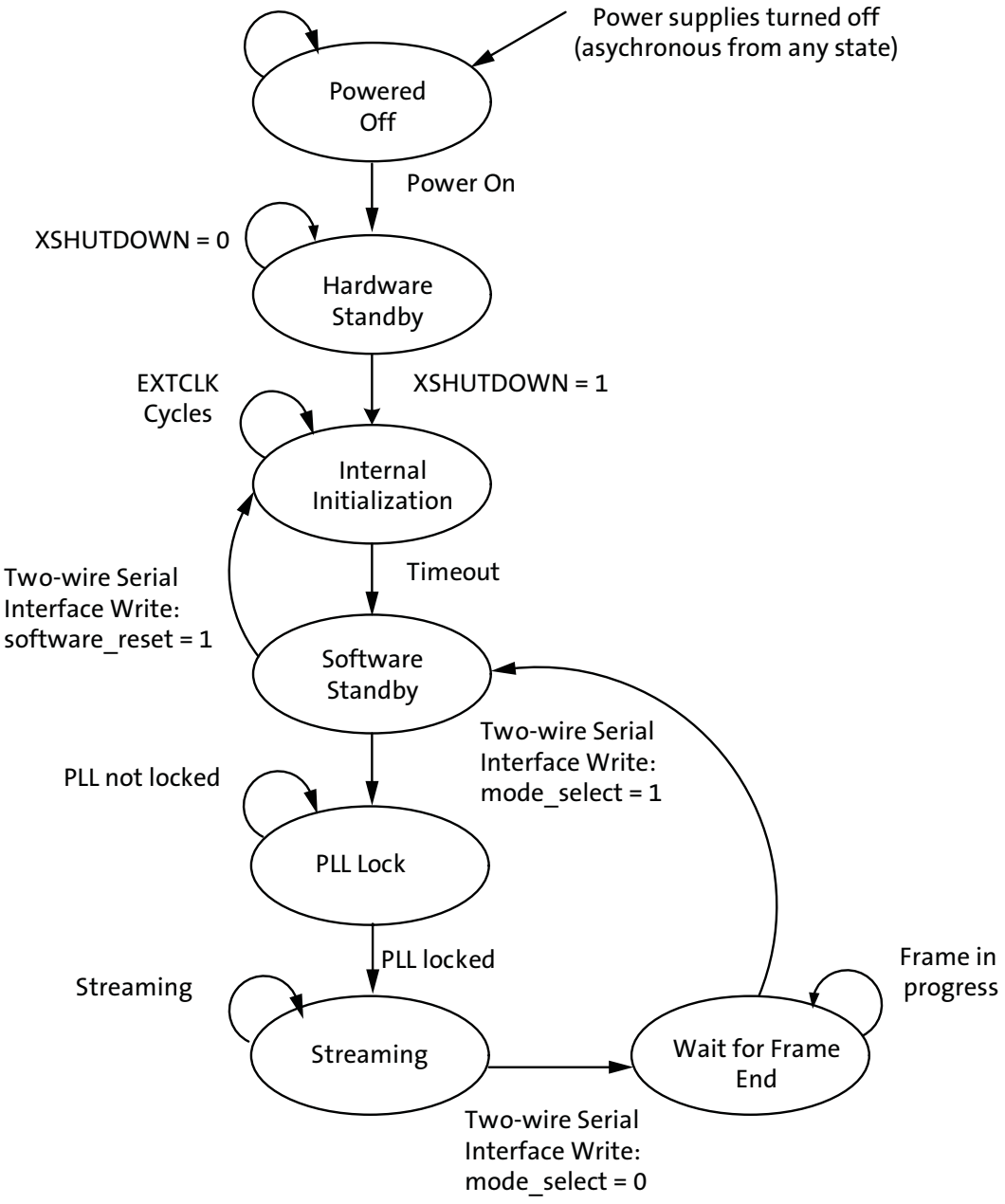
Pad Name	Power Supply	Description
Grounds		
DGND (GNDPHY, GND_IO)	0V	Digital
VCM_GND	0V	VCM driver
AGND	0V	Analog
Power		
VAA	2.8V	Analog/VCM driver/OTPM
VAA_PIX	2.8V	Pixel/Analog
DVDD_1V2	1.2V	Digital
VDD_IO	1.8v/2.8V	IO
DVDD_1V2_PHY	1.2V	MIPI
DVDD_1V8	1.8V	OTPM

System States

The system states of the AR0833 are represented as a state diagram in Figure 4 and described in subsequent sections.

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 Figure 4 page 13 and Figure 5 page 14.

Figure 4: System States



Sensor Initialization

Power-on Sequence

AR0833 has three voltage supplies divided into several domains. The three voltages are 1.2V, 1.8V, and 2.8V. For proper operation of the chip, a power-up sequence is recommended as shown in Figure 5.

The power sequence is governed by controlled vs controlling behavior of a power supply and the inrush current (ie current that exists when not all power supplies are present).

Table 6: Inrush consideration

XSHUTDOWN	1.2V	1.8V (VDDIO)	2.8V	Comment
x	Present	Absent	Absent	Not supported
x	Absent	Present	Absent	Supported
x	Absent	Absent	Present	Supported
x	Present	Present	Absent	Supported
x	Present	Absent	Present	Not supported
x	Absent	Present	Present	Supported
0	Present	Present	Present	Powered down state
1	Present	Present	Present	Powered up state

Since VDD_IO supply controls the XSHUTDOWN, it should be turned on first. The sequence of powering up the other two domains is not too critical. While turning on 2.8V supply before 1.2V supply shouldn't be an issue as shown in Table 1, it is still not recommended since the 2.8V domain is controlled by 1.2V signals. The dedicated 1.8V domain is used only for OTPM read function, so can turn on along with 1.8V supply.

Due to the above considerations, the suggested power-on sequence is as shown in Figure 5 on page 15:

Figure 5: Recommended Power-up Sequence

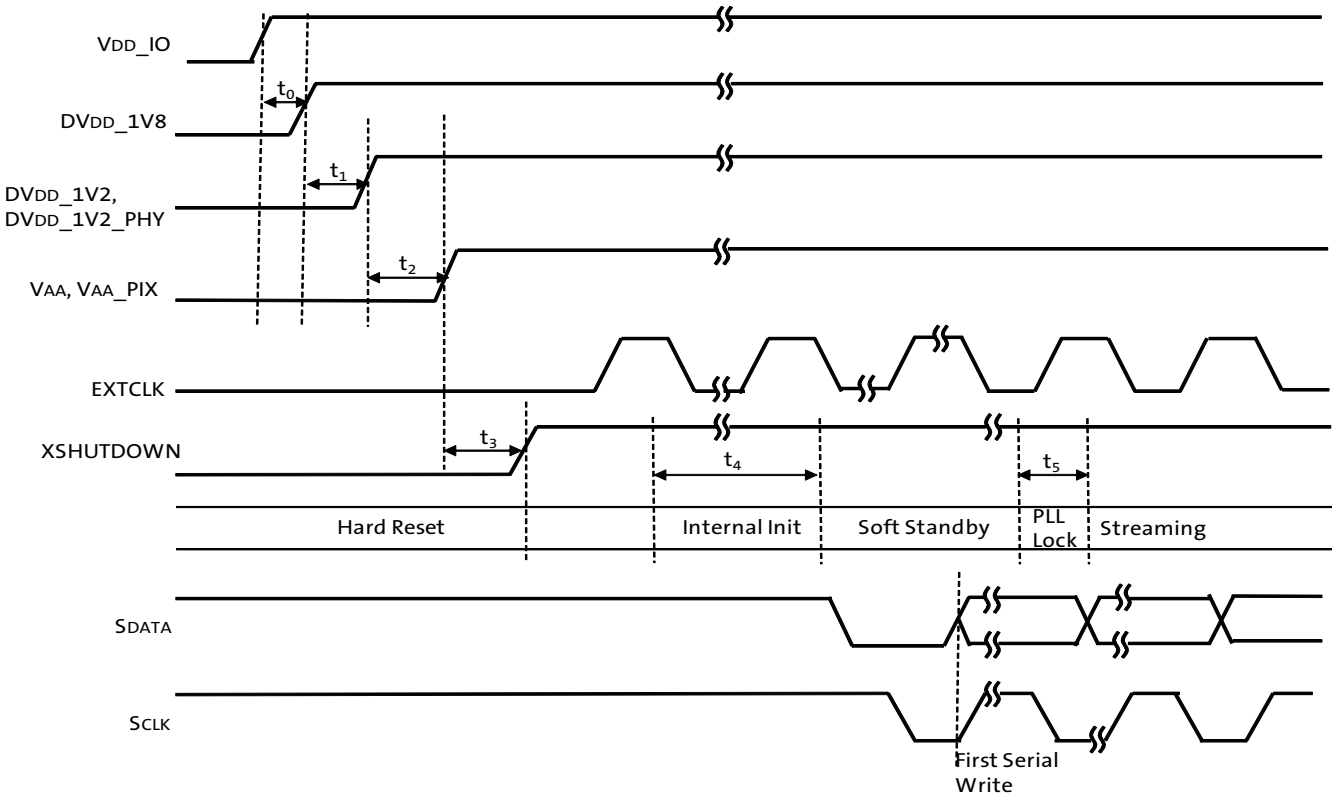


Table 7: Power-up Sequence

Definition	Symbol	Minimum	Typical	Maximum	Unit
VDD_IO to DVDD_1V8	t_0	–	–	500	ms
DVDD_1V8 to DVDD_1V2/DVDD_1V2_PHY	t_1	0.2	–	500	ms
SDVDD_1V2/DVDD_1V2_PHY to VAA/VAA_PIX	t_2	0.2	–	500	ms
Active Hard Reset	t_3	1	–	500	ms
Internal Initialization	t_4	2400	–	–	EXTCLKs
PLL Lock Time	t_5	1	–	5	ms

Power Down Sequence

The recommended power-down sequence for the AR0833 is shown in Figure 6. The three power supply domains (1.2V, 1.8V, and 2.8V) must have the separation specified below.

1. Disable streaming if output is active by setting standby R0x301a[2] = 0.
2. After disabling the internal clock EXTCLK, disable XSHUTDOWN.
3. After XSHUTDOWN is LOW disable the 2.8V/1.8V supply.
4. After the 2.8V/1.8V supply is LOW disable the 1.2V supply.
5. After the 1.2V supply is LOW disable the VDD_IO supply.

Figure 6: Recommended Power-down Sequence

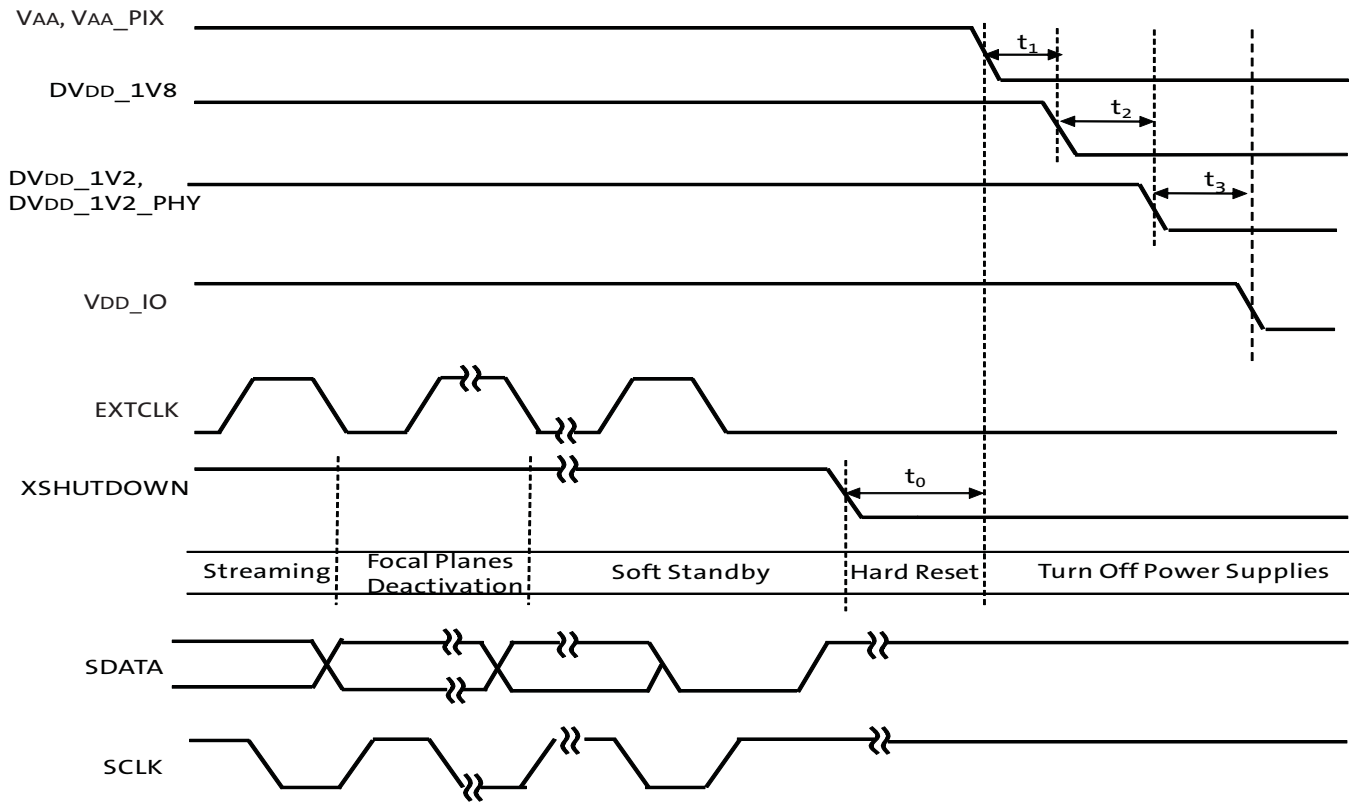


Table 8: Power-down Sequence

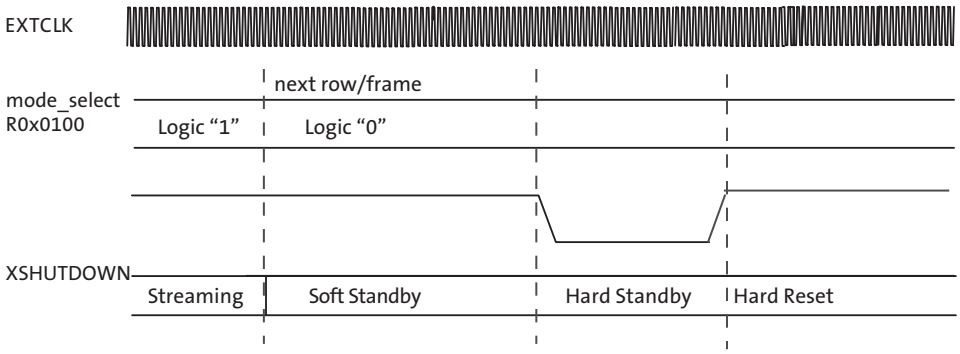
Definition	Symbol	Minimum	Typical	Maximum	Unit
EXTCLK to XSHUTDOWN	t_0	100	–	–	µs
XSHUTDOWN to supply 2.8V/1.8V	t_1	200	–	–	µs
Supply 2.8V/1.8V to supply 1.2V	t_2	0	200	–	µs
Supply 1.2V to VDD_IO	t_3	200	–	–	µs

Hard Standby and Hard Reset

The hard standby state is reached by the assertion of the XSHUTDOWN pad (hard reset). Register values are not retained by this action, and will be returned to their default values once hard reset is completed. The minimum power consumption is achieved by the hard standby state. The details of the sequence are described below and shown in Figure 7.

1. Disable streaming if output is active by setting mode_select 0x301A[2] = 0.
2. The soft standby state is reached after the current row or frame, depending on configuration, has ended.
3. Assert XSHUTDOWN (active LOW) to reset the sensor.
4. The sensor remains in hard standby state if XSHUTDOWN remains in the logic "0" state.

Figure 7: Hard Standby and Hard Reset



Soft Standby and Soft Reset

The AR0833 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. The details of the sequence are described below and shown in Figure 8 on page 18.

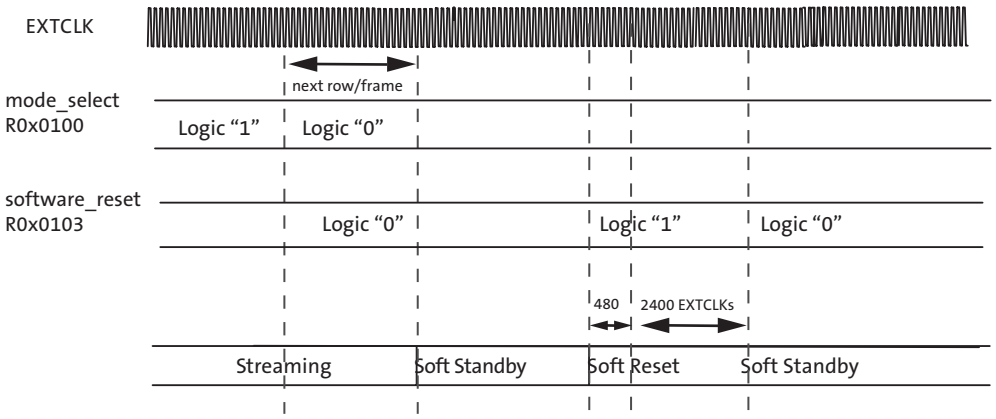
Soft Standby

1. Disable streaming if output is active by setting mode_select 0x301A[2] = 0.
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 (R0x3021) to start the internal initialization sequence.
3. After 2400 EXTCLKs (tentative), the internal initialization sequence is completed and the current state returns to soft standby automatically.

Figure 8: Soft Standby and Soft Reset



Two-Wire Serial Register Interface

A two-wire serial interface bus enables read/write access to control and status registers within the AR0833. The two-wire serial interface is fully compatible with the I^2C standard.

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 off-chip by a 1.5kΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The protocols described in the two-wire serial interface specification allow the slave device to drive SCLK LOW; the AR0833 uses SCLK as an input only and therefore never drives it LOW. The electrical and timing specifications are further detailed on "Two-Wire Serial Register Interface" on page 59.

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 AR0833 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.

The alternate slave addresses 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.

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.

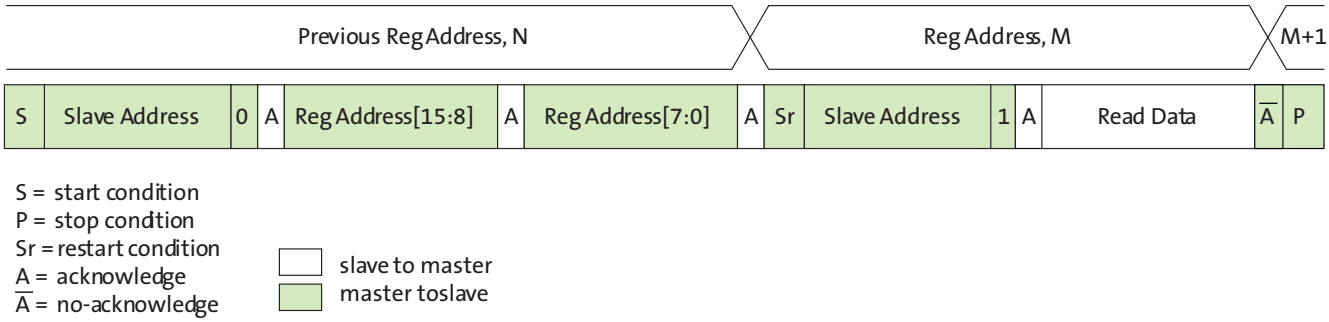
If the request was a WRITE, the master then transfers the 16-bit register address to which the WRITE should take place. This transfer takes place as two 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the 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 9 on page 21) 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 9 shows how the internal register address maintained by the AR0833 is loaded and incremented as the sequence proceeds.

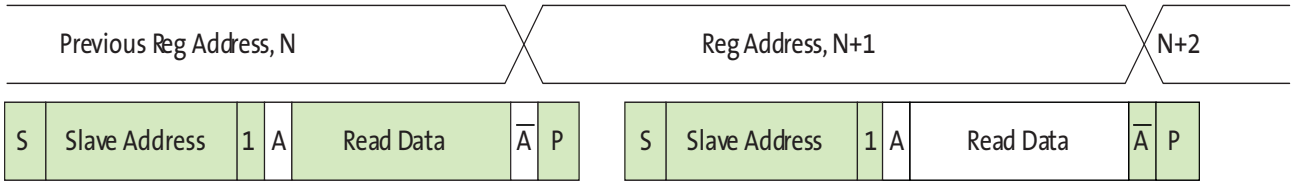
Figure 9: Single READ from Random Location



Single READ from Current Location

This sequence (Figure 10) performs a read using the current value of the AR0833 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.

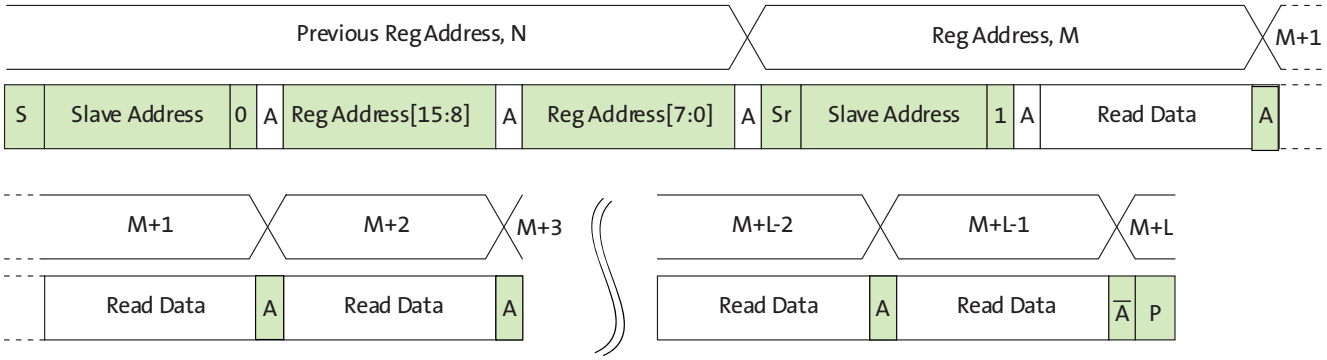
Figure 10: Single READ from Current Location



Sequential READ, Start from Random Location

This sequence (Figure 11) starts in the same way as the single READ from random 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.

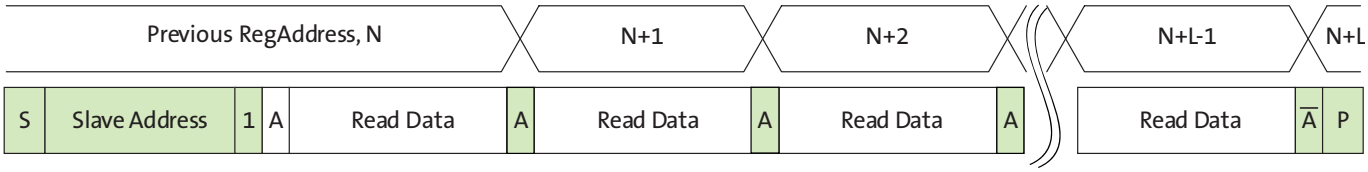
Figure 11: Sequential READ, Start from Random Location



Sequential READ, Start from Current Location

This sequence (Figure 12) starts in the same way as the single READ from current location (Figure 10 on page 21). 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.

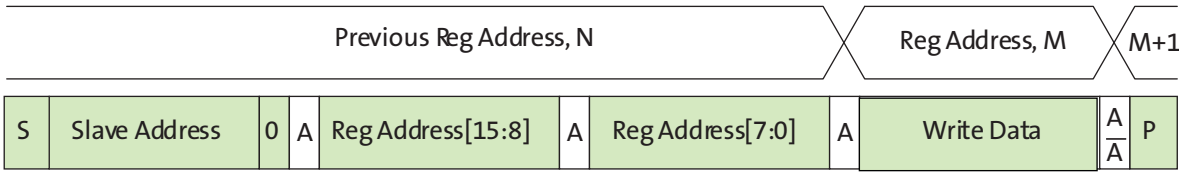
Figure 12: Sequential READ, Start from Current Location



Single WRITE to Random Location

This sequence (Figure 13) 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.

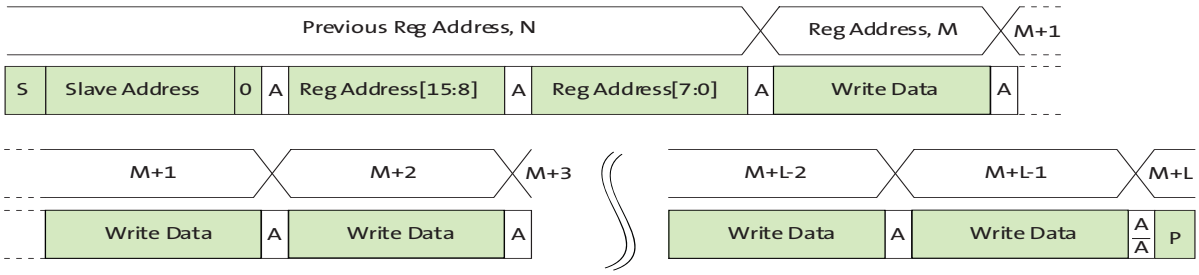
Figure 13: Single WRITE to Random Location



Sequential WRITE, Start at Random Location

This sequence (Figure 14) starts in the same way as the single WRITE to random location (Figure 13). 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 14: Sequential WRITE, Start at Random Location



Registers

The AR0833 provides a 16-bit register address space accessed through a serial interface ("Two-Wire Serial Register Interface" on page 19). Each register location is 8 or 16 bits in size.

The address space is divided into the five major regions shown in Table 9. The remainder of this section describes these registers in detail.

Table 9: Address Space Regions

Address Range	Description
0x0000–0x0FFF	Configuration registers (read-only and read-write dynamic registers)
0x1000–0x1FFF	Parameter limit registers (read-only static registers)
0x2000–0x2FFF	Image statistics registers (none currently defined)
0x3000–0x3FFF	Manufacturer-specific registers (read-only and read-write dynamic registers)

Register Notation

The underlying mechanism for reading and writing registers provides byte write capability. However, it is convenient to consider some registers as multiple adjacent bytes. The AR0833 uses 8-bit, 16-bit, and 32-bit registers, all implemented as 1 or more bytes at naturally aligned, contiguous locations in the address space.

In this document, registers are described either by address or by name. When registers are described by address, the size of the registers is explicit. For example, R0x3024 is a 2-bit register at address 0x3024, and R0x3000–1 is a 16-bit register at address 0x3000–0x3001. When registers are described by name, the size of the register is implicit. It is necessary to refer to the register table to determine that model_id is a 16-bit register.

Register Aliases

A consequence of the internal architecture of the AR0833 is that some registers are decoded at multiple addresses. Some registers in "configuration space" are also decoded in "manufacturer-specific space." To provide unique names for all registers, the name of the register within manufacturer-specific register space has a trailing underscore. For example, R0x0202 is coarse_integration_time and R0x3012 is coarse_integration_time_. The effect of reading or writing a register through any of its aliases is identical.

Bit Fields

Some registers provide control of several different pieces of related functionality, and this makes it necessary to refer to bit fields within registers. As an example of the notation used for this, the least significant 4 bits of the chip_version_reg register are referred to as chip_version_reg[3:0] or R0x0000–1[3:0].

Bit Field Aliases

In addition to the register aliases described above, some register fields are aliased in multiple places. For example, R0x0100 (mode_select) has only one operational bit, R0x0100[0]. This bit is aliased to R0x301A–B[2]. The effect of reading or writing a bit field through any of its aliases is identical.

Byte Ordering

Registers that occupy more than one byte of address space are shown with the lowest address in the highest-order byte lane to match the byte-ordering on the data bus. For example, the chip_version_reg register is R0x0000–1. In the register table the default value is shown as 0x4B00. This means that a read from address 0x0000 would return 0x4B, and a read from address 0x0001 would return 0x00. When reading this register as two 8-bit transfers on the serial interface, the 0x4B will appear on the serial interface first, followed by the 0x00.

Address Alignment

All register addresses are aligned naturally. Registers that occupy 2 bytes of address space are aligned to even 16-bit addresses, and registers that occupy 4 bytes of address space are aligned to 16-bit addresses that are an integer multiple of 4.

Bit Representation

For clarity, 32-bit hex numbers are shown with an underscore between the upper and lower 16 bits. For example: 0x3000_01AB.

Data Format

Most registers represent an unsigned binary value or set of bit fields. For all other register formats, the format is stated explicitly at the start of the register description. The notation for these formats is shown in Table 10.

Table 10: Data Formats

Name	Description
FIX16	Signed fixed-point, 16-bit number: two's complement number, 8 fractional bits. Examples: 0x0100 = 1.0, 0x8000 = –128, 0xFFFF = –0.0039065
UFIX16	Unsigned fixed-point, 16-bit number: 8.8 format. Examples: 0x0100 = 1.0, 0x280 = 2.5
FLP32	Signed floating-point, 32-bit number: IEEE 754 format. Example: 0x4280_0000 = 64.0

Register Behavior

Registers vary from "read-only," "read/write," and "read, write-1-to-clear."

Double-Buffered Registers

Some sensor settings cannot be changed during frame readout. For example, changing R0x3004–5 (x_addr_start) partway through frame readout would result in inconsistent row lengths within a frame. To avoid this, the AR0833 double-buffers many registers by implementing a "pending" and a "live" version. Reads and writes access the pending register. The live register controls the sensor operation.

The value in the pending register is transferred to a live register at a fixed point in the frame timing, called frame start. Frame start is defined as the point at which the first dark row is read out internally to the sensor. In the register tables the "Frame Sync'd" column shows which registers or register fields are double-buffered in this way.

Using grouped_parameter_hold

Register grouped_parameter_hold (R0x301A[15]) can be used to inhibit transfers from the pending to the live registers. When the AR0833 is in streaming mode, this register should be written to "1" before making changes to any group of registers where a set of changes is required to take effect simultaneously. When this register is written to "0," all transfers from pending to live registers take place on the next frame start.

An example of the consequences of failing to set this bit follows:

An external auto exposure algorithm might want to change both gain and integration time between two frames. If the next frame starts between these operations, it will have the new gain, but not the new integration time, which would return a frame with the wrong brightness that might lead to a feedback loop with the AE algorithm resulting in flickering.

Bad Frames

A bad frame is a frame where all rows do not have the same integration time or where offsets to the pixel values have changed during the frame.

Many changes to the sensor register settings can cause a bad frame. For example, when line_length_pck (R0x300C) is changed, the new register value does not affect sensor behavior until the next frame start. However, the frame that would be read out at that frame start will have been integrated using the old row width, so reading it out using the new row width would result in a frame with an incorrect integration time.

By default, bad frames are masked. If the masked bad frame option is enabled, both LV and FV are inhibited for these frames so that the vertical blanking time between frames is extended by the frame time.

In the register tables, the "Bad Frame" column shows where changing a register or register field will cause a bad frame. This notation is used:

N—No. Changing the register value will not produce a bad frame.
Y—Yes. Changing the register value might produce a bad frame.
YM—Yes; but the bad frame will be masked out when mask_corrupted_frames (R0x301A[9]) is set to "1."

Changes to Integration Time

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

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

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

Changes to Gain Settings

Usually, when the gain settings are changed, the gain is updated on the next frame start. When the integration time and the gain are changed at the same time, the gain update is held off by one frame so that the first frame output with the new integration time also has the new gain applied. In this case, a new gain should not be set during the extra frame delay. There is an option to turn off the extra frame delay by setting extra_delay (R0x3018).

Clocking

Default setup gives a physical 73.2 MHz internal clock for an external input clock of 24 MHz.

The sensor contains a phase-locked loop (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. The PLL structure is shown in Figure 15.

Figure 15: Clocking Configuration

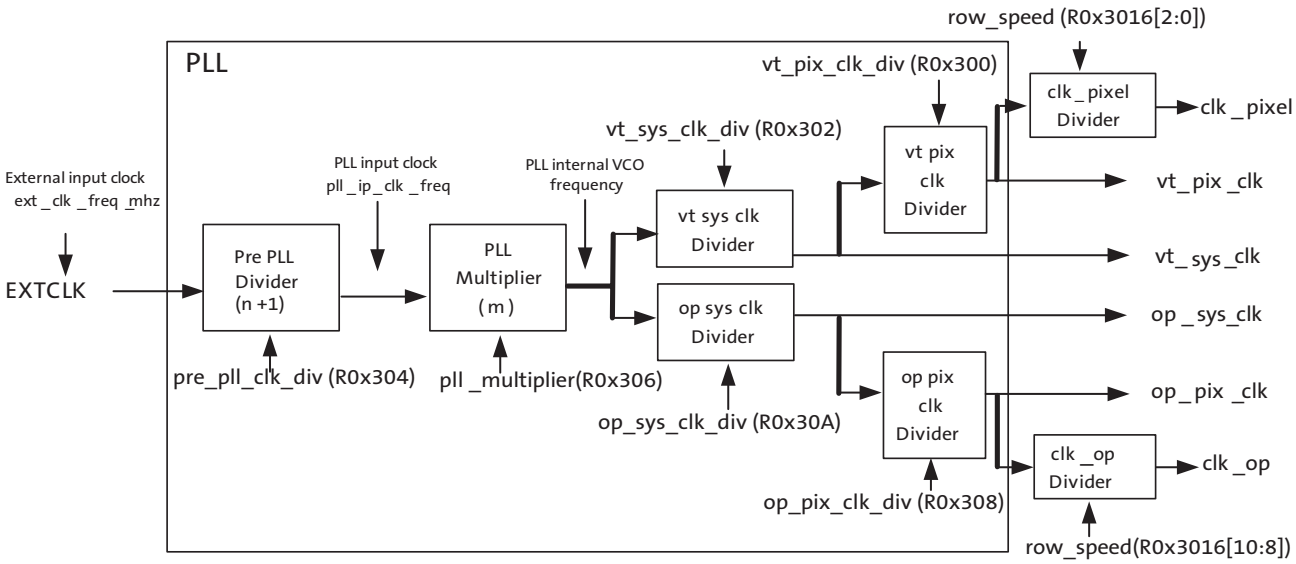


Figure 15 shows the different clocks and the names of the registers that contain or are used to control their values. The vt_pix_clk is divided by two to compensate for the fact that the design has 2 digital data paths. This divider should always remain turned on.

AR0833 has 10-to-8 and 10-to-6 compression. The Framer IP packs two 6-bit pixels into one 12-bit data and sends it to Physical Layer. The Framer takes the action to divide word clock into half speed. The word clock should be divided by 6 from VCO at PLL in order to match Physical Layer considering one data having 12 bit-clocks.

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) is controlled in increments of the clk_pixel period.
- clk_op (op_pix_clk / row_speed[10:8]) is used to load parallel pixel data from the output FIFO (see Figure 40 on page 54) to the serializer. The output FIFO generates one pixel each op_pix_clk period.
- 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.

The pixel frequency can be calculated in general as:

$$pixel\ clock\ mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times vt\_sys\_clk\_div \times 2 \times vt\_pix\_clk\_div \times row\_speed[2:0]} \quad \text{(EQ 1)}$$

The output clock frequency can be calculated as:

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

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

PLL Clocking

The PLL divisors should be programmed while the AR0833 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 AR0833 is in the streaming state is undefined.

Clock Control

The AR0833 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 AR0833 enters a soft standby 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.

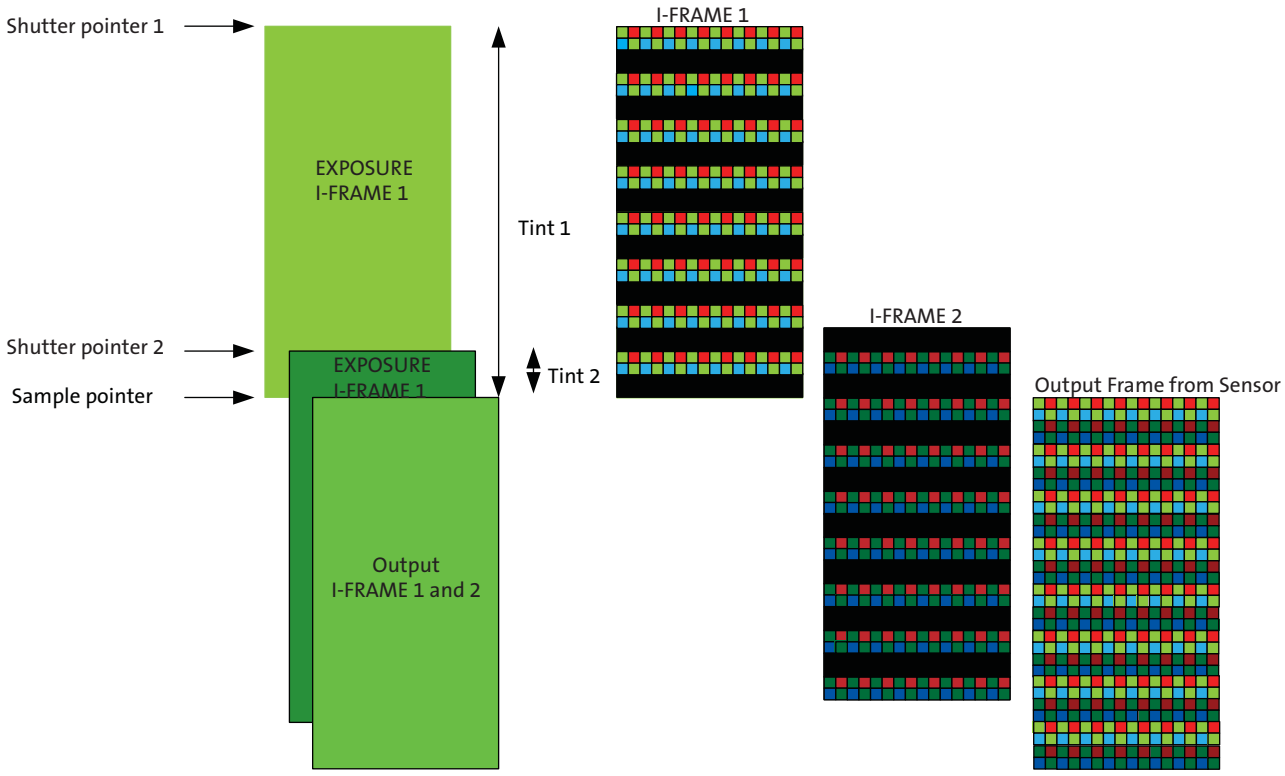
Features

Interlaced HDR Readout

The sensor enables HDR by outputting frames where even and odd row pairs within a single frame are captured at different integration times. This output is then matched with an algorithm designed to reconstruct this output into an HDR still image or video.

The sensor HDR is controlled by two shutter pointers (Shutter pointer1, Shutter pointer2) that control the integration of the odd (Shutter pointer1) and even (Shutter pointer 2) row pairs.

Figure 16: HDR Integration Time



Integration Time for Interlaced HDR Readout

Tint1 (integration time 1) and Tint2 (integration time 2)

The limits for the coarse integration time are defined by:

$$coarse\_integration\_time\_min \le coarse\_integration\_time \le (frame\_length\_lines - coarse\_integration\_time\_max\_margin) \quad \text{(EQ 4)}$$

$$coarse\_integration\_time2\_min \le coarse\_integration\_time2 \le (frame\_length\_lines - coarse\_integration\_time2\_max\_margin) \quad \text{(EQ 5)}$$

The actual integration time is given by:

$$integration\_time = \frac{(coarse\_integration\_time \times line\_length\_pck)}{vt\_pix\_clk\_freq\_mhz \times 10^6} \quad \textit{(EQ 6)}$$

$$integration\_time2 = \frac{(coarse\_integration\_time2 \times line\_length\_pck)}{vt\_pix\_clk\_freq\_mhz \times 10^6} \quad \text{(EQ 7)}$$

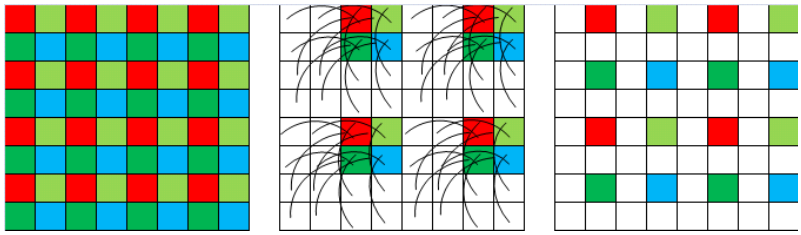
If this limit is broken, the frame time will automatically be extended to *(coarse_integration_time + coarse_integration_time_max_margin)* to accommodate the larger integration time.

The ratio between even and odd rows is typically adjusted to 1x, 2x, 4x, and 8x.

Bayer Resampler

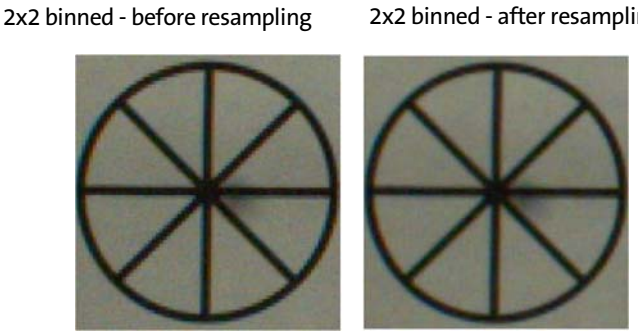
The imaging artifacts found from a 2x2 binning or summing will show image artifacts from aliasing. These can be corrected by resampling the sampled pixels in order to filter these artifacts. Figure 17 shows the pixel location resulting from 2x2 summing or binning located in the middle and the resulting pixel locations after the Bayer re-sampling function has been applied.

Figure 17: Bayer Resampling



The improvements from using the Bayer resampling feature can be seen in Figure 18. In this example, image edges seen on a diagonal have smoother edges when the Bayer re-sampling feature is applied. This feature is only designed to be used with modes configured with 2x2 binning or summing. The feature will not remove aliasing artifacts that are caused skipping pixels.

Figure 18: Results of Resampling



To enable the Bayer resampling feature:

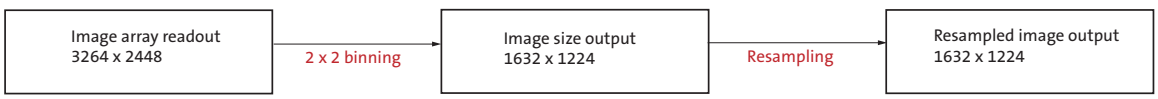
1. Set R0x400 = 2 // Enable the on-chip scalar.
2. Set R0x306E to 0x90B0 // Configure the on-chip scalar to resample Bayer data.

To disable the Bayer resampling feature:

1. Set R0x400 = 0 // Disable the on-chip scalar.
2. Set R0x306E to 0x9080 // Configure the on-chip scalar to resample Bayer data.

Note: The image readout (rows and columns) has to have two extra rows and two extra columns when using the resample feature.

Figure 19: Illustration of Resampling Operation



One-Time Programmable Memory (OTPM)

The AR0833 features 4Kb of one-time programmable memory (OTPM) for storing shading correction coefficients, individual module, and customer-specific information. The user may program the data before shipping. OTPM can be accessed through two-wire serial interface. The AR0833 uses the auto mode for fast OTPM programming and read operations.

To read out the OTPM, 1.8V supply is required. As a result, a dedicated DVDD_1V8 pad has been implemented.During the programming process, a dedicated pin for high voltage needs to be provided to perform the anti-fusing operation. This voltage (VPP) would need to be 6.5V. The completion of the programming process will be communicated by a register through the two-wire serial interface.

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 and Verifying the OTPM

The procedure for programming and verifying the AR0833 OTPM follows:

1. Apply power to all the power rails of the sensor (VDD_IO, VAA, VAA_PIX, DVDD_1V2, DVDD_1V2_PHY, and DVDD_1V8). VAA must be set to 2.8V during the programming process. VPP must be initially floating. All other supplies must be at their nominal voltage.
2. Provide a 12-MHz EXTCLK clock input.
3. Set R0x301A = 0x18, to put sensor in the soft standby mode.
4. Set R0x3130 = 0xFF01 (Timing configuration)
5. Set R0x304C[15:8] = Record type (E.g. 0x30)
6. Set R0x304C[7:0] = Length of the record which is the number of OTPM data registers that are filled in.
7. Set R0x3054[9] = 0 to ensure that the error checking and correction is enabled.
8. Write data into all the OTPM data registers: R0x3800-R0x39FE.
9. Ramp up VPP to 6.5V.
10. Set the otpm_control_auto_wr_start bit in the otpm_control register R0x304A[0] = 1, to initiate the auto program sequence. The sensor will now program the data into the OTPM.
11. Poll otpm_control_auto_wr_end (R0x304A [1]) to determine when the sensor is finished programming the word.
12. Verify that the otpm_control_auto_wr_success(0x304A[2]) bit is set.

13. If the above bits are not set to 1, then examine otpm_status register R0x304E[9] to verify if the OTPM memory is full and 0x304E[10] to verify if OTPM memory is insufficient.
14. Remove the high voltage (VPP) and float VPP pin.

Reading the OTPM

1. Apply power to all the power rails of the sensor (VDD_IO, VAA, VAA_PIX, DVDD_1V2, DVDD_1V2_PHY, and DVDD_1V8) at their nominal voltage.
2. Set EXTCLK to normal operating frequency.
3. Perform proper reset sequence to the sensor.
4. Set R0x3134=0xCD95 (Timing Configuration)
5. Set R0x304C[15:8] = Record Type (for example, 0x30)
6. Set R0x304C[7:0] = Length of the record which is the number of data registers to be read back. This could be set to 0 during OTPM auto read if length is unknown.
7. Set R0x3054 = 0x0400
8. Initiate the auto read sequence by setting the otpm_control_auto_read_start bit (R0x304A[4]) = 1.
9. Poll the otpm_control_auto_rd_end bit (R0x304A[5]) to determine when the sensor is finished reading the word(s). When this bit becomes 1, the otpm_control_auto_rd_success bit (R0x304A[6]) will indicate whether the memory was read successfully or not.
10. Data can now be read back from the otpm_data registers (R0x3800-R0x39FE).

Image Acquisition Modes

The AR0833 supports two image acquisition modes:

1. Electronic rolling shutter (ERS) mode.
   This is the normal mode of operation. When the AR0833 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 AR0833 switches cleanly from the old integration time to the new while only generating frames with uniform integration. See “Changes to Integration Time” on page 26.
2. Global reset release (GRR) mode.
   This mode can be used to acquire a single image at the current resolution. In this mode, the end point of the pixel integration time is controlled by an external electromechanical shutter, and the AR0833 provides control signals to interface to that shutter. The operation of this mode is described in detail in “Global Reset Release (GRR)” on page 48.

The benefit for the use of an external electromechanical shutter is that it eliminates the visual artifacts associated with ERS operation. Visual artifacts arise in ERS operation, particularly at low frame rates, because an ERS image effectively integrates each row of the pixel array at a different point in time.

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. The output image size is controlled by the x_output_size and y_output_size registers.

Pixel Border

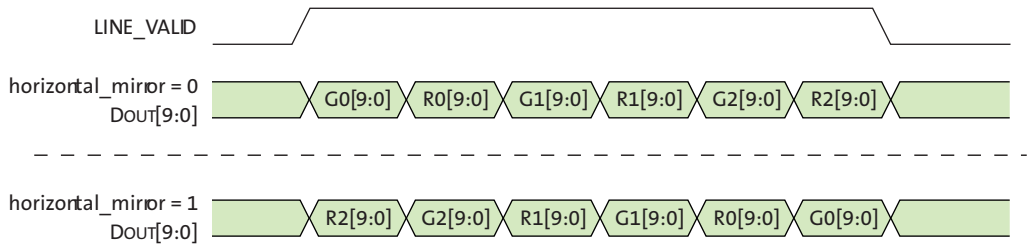
The default settings of the sensor provide a 3264H x 2448V image. A border of up to 8 pixels (4 in binning) on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end, y_addr_end, x_output_size, and y_output_size registers accordingly. These border pixels can be used but are disabled by default.

Readout Modes

Horizontal Mirror

The horizontal_mirror bit in the image_orientation register is set by default. The result of this is that 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 20 on page 35 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.

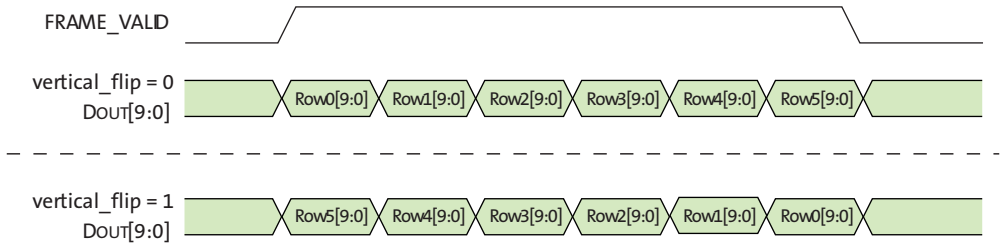
Figure 20: Effect of horizontal_mirror on Readout Order



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 21 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.

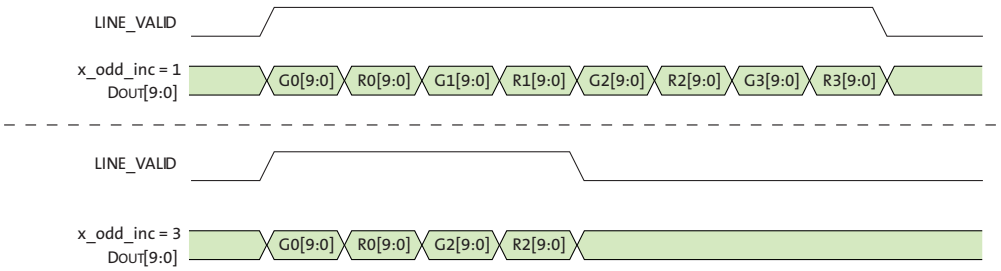
Figure 21: Effect of vertical_flip on Readout Order



Subsampling

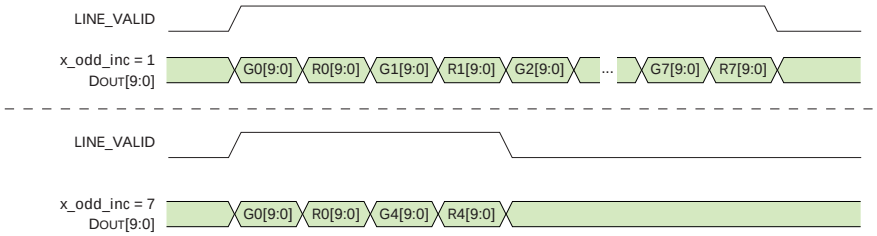
The AR0833 supports subsampling to reduce the amount of data processed by the signal chains in the AR0833, thereby allowing the frame rate to be increased and power consumption reduced. Subsampling is enabled by setting x_odd_inc and/or y_odd_inc. Values of 1, 3, and 7 can be supported. Setting both 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 AR0833. Setting x_odd_inc = 3 and y_odd_inc = 3 results in a quarter reduction in output image size. Figure 22 on page 36 shows a sequence of 8 columns being read out with x_odd_inc = 3 and y_odd_inc = 1.

Figure 22: 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 AR0833. Figure 23 shows a sequence of 16 columns being read out with x_odd_inc = 7 and y_odd_inc = 1.

Figure 23: 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 24 through Figure 26 on page 38.

Figure 24: Pixel Readout (No Subsampling)

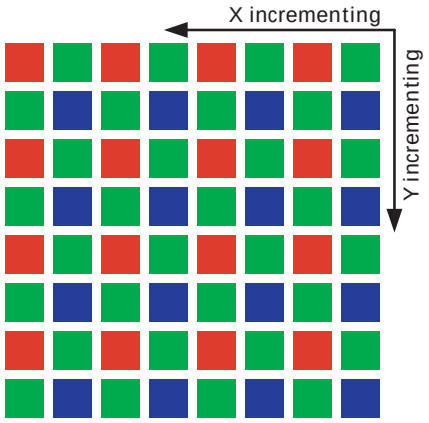


Figure 25: Skip2 Pixel Readout (x_odd_inc = 3, y_odd_inc = 3)

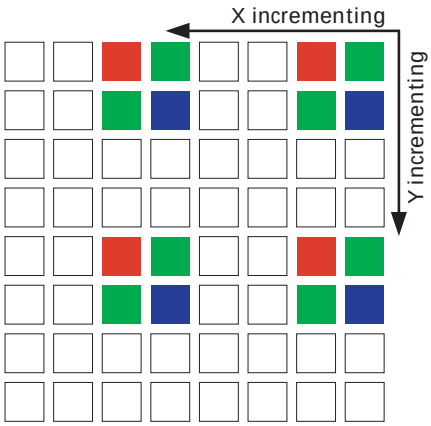
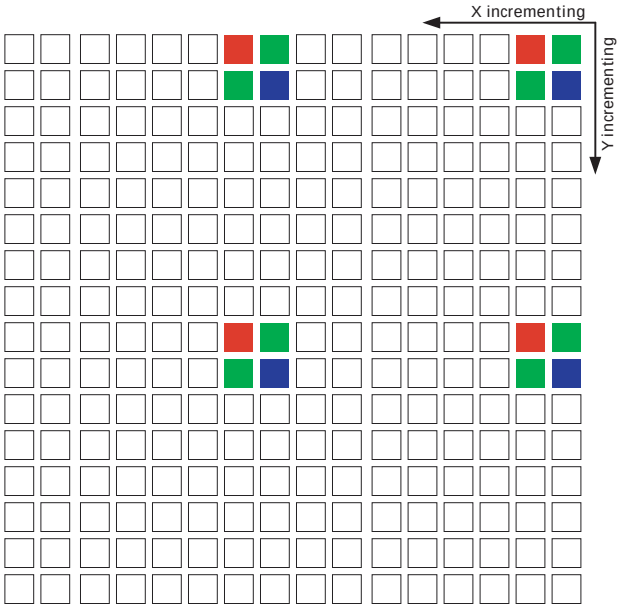


Figure 26: Skip4 Pixel Readout (x_odd_inc = 7, y_odd_inc = 7)



Programming Restrictions when Subsampling

When subsampling is enabled and the sensor is switched back and forth between full resolution and subsampling, Aptina 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_start, x_addr_end, 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
y_skip_factor = (y_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

Table 11 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 11).

Table 11: 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 AR0833 supports 2 x 1 (column binning, also called x-binning). Binning has many of the same characteristics as skipping, 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 skipping.

Binning is enabled by selecting the appropriate subsampling settings (in read_mode, the sub-register x_odd_inc = 3 and y_odd_inc = 1 for x-binning and setting the appropriate binning bit in read_mode R0x3040[11] = 1 for x_bin_enable). As with skipping, 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 skipping mode. The effect of the different binning is shown in Figure 27 below and Figure 28 on page 41.

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 4X 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 28 on page 41.

Figure 27: Bin2 Pixel Readout (x_odd_inc = 3, y_odd_inc = 1, x_bin = 1)

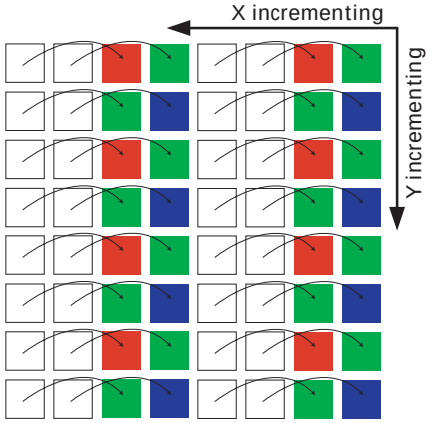
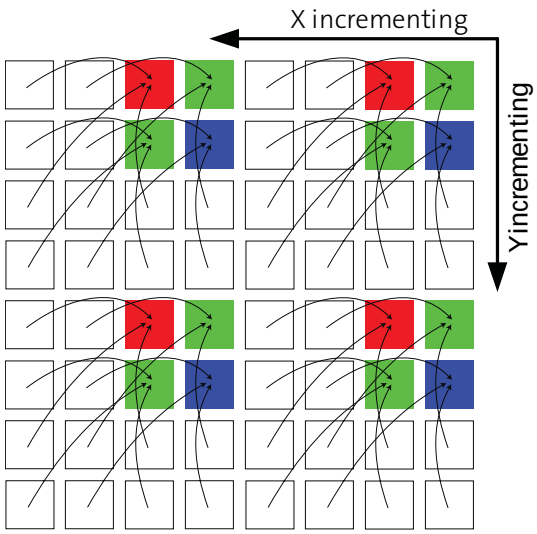


Figure 28: Bin2Pixel Readout (x_odd_inc = 3, y_odd_inc = 3, x_bin = 1)



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 12.

Table 12: Column Address Sequencing During Binning

odd_inc = 1 (Normal)	odd_inc = 3 (2X Bin)	odd_inc = 7 (2X Skip + 2XBin)
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 13 on page 42.

Table 13: 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 and Summing

Binning and summing require different sequencing of the pixel array and impose different timing limits on the operation of the sensor.

As a result, when xy-subsampling 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 “Minimum Frame Time” and "Minimum Row Time" on page 45.

Subsampling / binning options:

1. XskipYskip
   - R0x3040[11], x_bin_en: 0
   - R0x3040[13], row_sum: 0
     - R0x0382: x_odd_inc = 3 (xskip2) or 7 (xskip4)
     - R0x0386: y_odd_inc = 3 (yskip2), 7 (yskip4) or 15 (yskip8)
2. XbinYskip
   - R0x3040[11], x_bin_en: 1
   - R0x3040[13], row_sum: 0
     - R0x0382: x_odd_inc = 3 (xbin2)
     - R0x0386: y_odd_inc = 3 (yskip2), 7 (yskip4) or 15 (yskip8)
3. XskipYsum
   - R0x3040[11], x_bin_en: 0
   - R0x3040[13], row_sum: 1
     - R0x0382: x_odd_inc = 3 (xskip2) or 7 (xskip4)
     - R0x0386: y_odd_inc = 3 (ysum2)
4. XbinYsum
   - R0x3040[11], x_bin_en: 1
   - R0x3040[13], row_sum: 1
     - R0x0382: x_odd_inc = 3 (xbin2)
     - R0x0386: y_odd_inc = 3 (ysum2)
5. XsumYsum
   - R0x3040[11], x_bin_en: 1
   - R0x3040[13], row_sum: 1
   - R0x3EE4[0], sreg_colamp_sum2: 1
   - (cannot write to this bit when streaming - have to write to entire register)
     - R0x0382: x_odd_inc = 3 (xsum2)
     - R0x0386: y_odd_inc = 3 (ysum2)

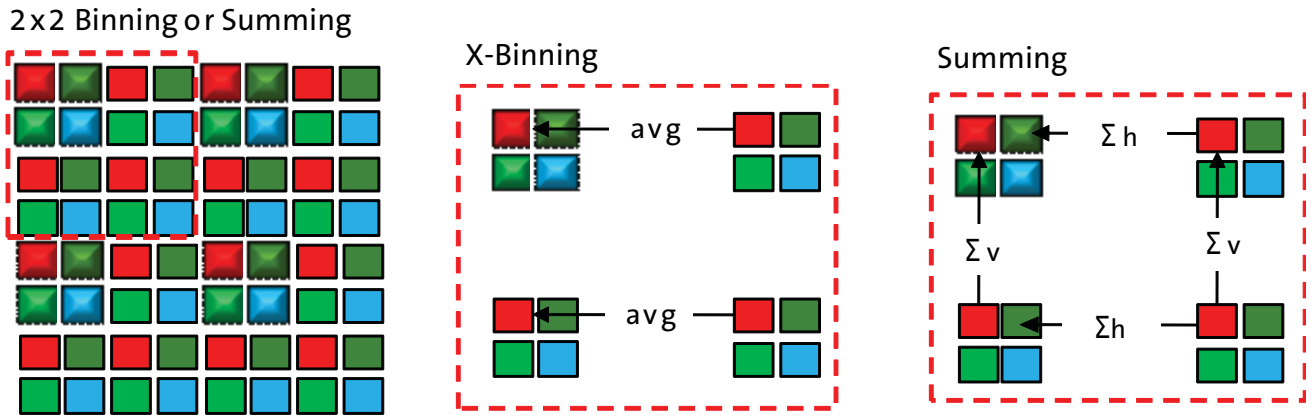
Binning, Skipping, and Summing Mode

Summing, skipping, and binning can be combined in the modes listed in Table 14. Unlike binning mode where the values of adjacent same color pixels are averaged together, summing adds the pixel values together resulting in better sensor sensitivity. Summing is supposed to provide two times the sensitivity compared to the binning only mode.

Table 14: Available Skip, Bin, and Sum Modes in the AR0833 Sensor

Subsampling Method	Horizontal	Vertical
Skipping	2x, 4x	2x, 4x, 8x
Binning	2x	
Summing	2x	2x

Figure 29: Pixel Binning and Summing



Scaler

Scaling reduces the size of the output image while maintaining the same field-of-view. The input and output of the scaler is in Bayer format.

When compared to skipping, scaling is advantageous as it avoids aliasing. The scaling factor, programmable in 1/16 steps, is used for horizontal and vertical scalers.

The AR0833 sensor is capable of horizontal scaling and full (horizontal and vertical) scaling.

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 96, giving the user the ability to scale from 1:1 (m=16) to 1:6 (m=96).

Frame Rate Control

The formulas for calculating the frame rate of the AR0833 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 Equation 8:

$$minimum\ line\_length\_pck = \left(\frac{x\_addr\_end - x\_addr\_start\ + 1}{subsampling\ factor} + min\_line\_blanking\_pck\right) \quad \text{(EQ 8)}$$

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” on page 45.

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 9:

$$minimum\ frame\_length\_lines = \left(\frac{y\_addr\_end - y\_addr\_start\ + 1}{subsampling\ factor} + min\_frame\_blanking\_lines\right) \quad \text{(EQ 9)}$$

The frame rate can be calculated from these variables and the pixel clock speed as shown in Equation 10:

$$frame\ rate = \frac{vt\_pixel\_clock\_mhz\ x\ 1\ x\ 10^6}{line\_length\_pck\_x\ frame\_length\_lines} \quad \text{(EQ 10)}$$

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

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 two checks that must all be met when programming line_length_pck:

- line_length_pck ≥ min_line_length_pck in Table 15.
- 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 1, so min_frame_length_lines will always equal (min_frame_blanking_lines + 1).

Table 15: Minimum Frame Time and Blanking Numbers

min_frame_blanking_lines	0x008F
min_frame_length_lines	0x0A1F

Integration Time

The integration (exposure) time of the AR0833 is controlled by the coarse_integration_time register.

The limits for the coarse integration time are defined by:

$$coarse\_integration\_time\_min \leq coarse\_integration\_time \quad (EQ\ 11)$$

The actual integration time is given by:

$$integration\_time = \frac{(coarse\_integration\_time * line\_length\_pck)}{(vt\_pix\_clk\_freq\_mhz * 10^6)} \quad (EQ\ 12)$$

It is required that:

$$coarse\_integration\_time \leq (frame\_length\_lines - coarse\_integration\_time\_max\_margin) \quad (EQ\ 13)$$

If this limit is broken, the frame time will automatically be extended to (coarse_integration_time + coarse_integartion_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.

Flash Timing Control

The AR0833 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 Figure 30 (xenon) and Figure 31 on page 47 (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 (R0x301A[9] = 1) before the enabling the flash or by forcing a restart (R0x301A[1] = 1) immediately after enabling the flash; the first bad frame will then be masked out, as shown in Figure 31. Read-only bit flash[14] is set during frames that are correctly integrated; the state of this bit is shown in Figures 30 and Figure 31.

Figure 30: Xenon Flash Enabled

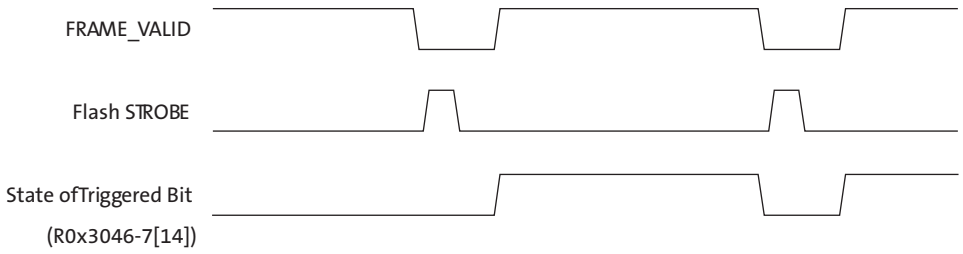
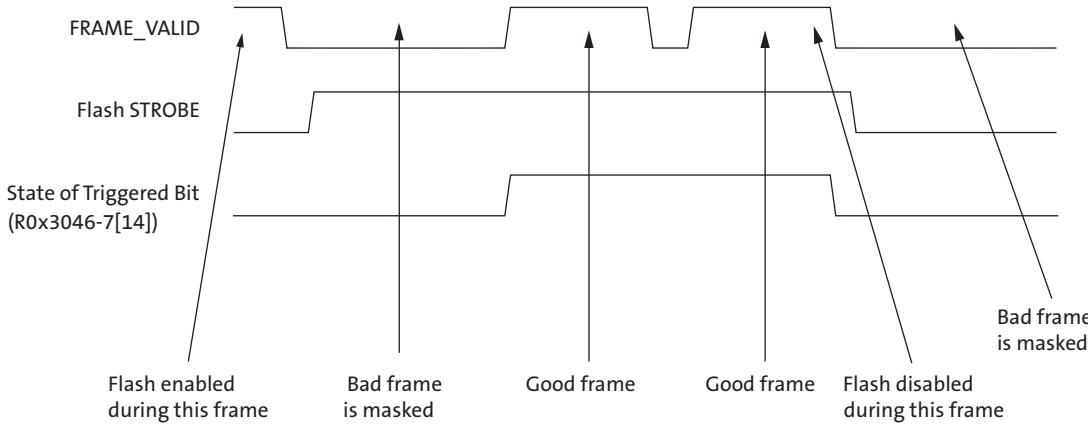


Figure 31: LED Flash Enabled



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

Global Reset Release (GRR)

Global reset release mode allows the integration time of the AR0833 to be controlled by an external electromechanical shutter. GRR mode is generally used in conjunction with ERS mode. The ERS mode is used to provide viewfinder information, the sensor is switched into GRR mode to capture a single frame, and the sensor is then returned to ERS mode to restore viewfinder operation.

Overview of Global Reset Release Sequence

The basic elements of the GRR sequence are:

1. By default, the sensor operates in ERS mode and the SHUTTER output signal is LOW. The electromechanical shutter must be open to allow light to fall on the pixel array. Integration time is controlled by the coarse_integration_time register.
2. A global reset sequence is triggered.
3. All of the rows of the pixel array are placed in reset.
4. All of the rows of the pixel array are taken out of reset simultaneously. All rows start to integrate incident light. The electromechanical shutter may be open or closed at this time.
5. If the electromechanical shutter has been closed, it is opened.
6. After the desired integration time (controlled internally or externally to the AR0833), the electromechanical shutter is closed.
7. A single output frame is generated by the sensor with the usual LV, FV, PIXCLK, and DOUT timing. As soon as the output frame has completed (FV negates), the electromechanical shutter may be opened again.
8. The sensor automatically resumes operation in ERS mode.

This sequence is shown in Figure 32. The following sections expand to show how the timing of this sequence is controlled.

Figure 32: Overview of Global Reset Sequence

Entering and Leaving the Global Reset Sequence

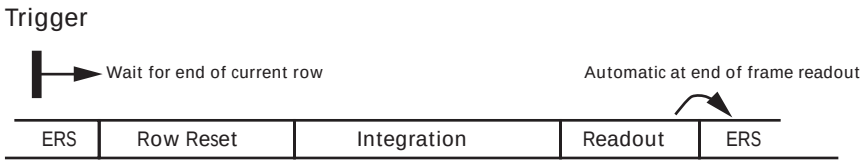
A global reset sequence can be triggered by a register write to R0x315E global_seq_trigger[0] (global trigger, to transition this bit from a 0 to a 1) or by a rising edge on a suitably-configured GPI input).

When a global reset sequence is triggered, the sensor waits for the end of the current row. When LV negates for that row, FV is negated 6 PIXCLK periods later, potentially truncating the frame that was in progress.

The global reset sequence completes with a frame readout. At the end of this readout phase, the sensor automatically resumes operation in ERS mode. The first frame integrated with ERS will be generated after a delay of approximately ((13 + coarse_integration_time) * line_length_pck). This sequence is shown in Figure 33.

While operating in ERS mode, double-buffered registers ("Double-Buffered Registers" on page 25) are updated at the start of each frame in the usual way. During the global reset sequence, double-buffered registers are updated just before the start of the readout phase.

Figure 33: Entering and Leaving a Global Reset Sequence



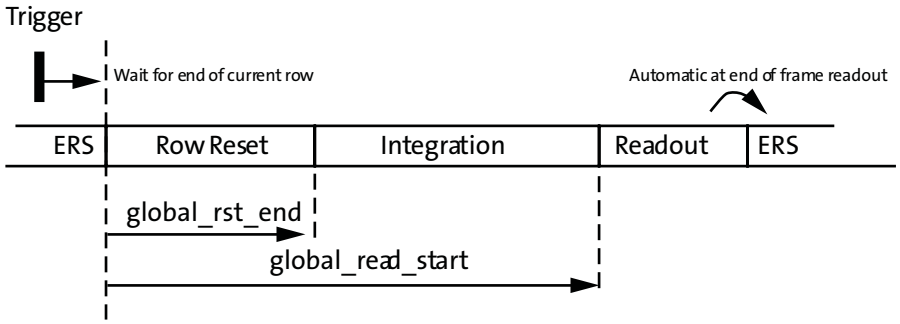
Programmable Settings

The registers global_rst_end and global_read_start allow the duration of the row reset phase and the integration phase to be controlled, as shown in Figure 34. The duration of the readout phase is determined by the active image size.

The recommended setting for global_rst_end is 0x3160 (for example, 512 µs total reset time) with default vt_pix_clk. This allows sufficient time for all rows of the pixel array to be set to the correct reset voltage level. The row reset phase takes a finite amount of time due to the capacitance of the pixel array and the capability of the internal voltage booster circuit that is used to generate the reset voltage level.

As soon as the global_rst_end count has expired, all rows in the pixel array are taken out of reset simultaneously and the pixel array begins to integrate incident light.

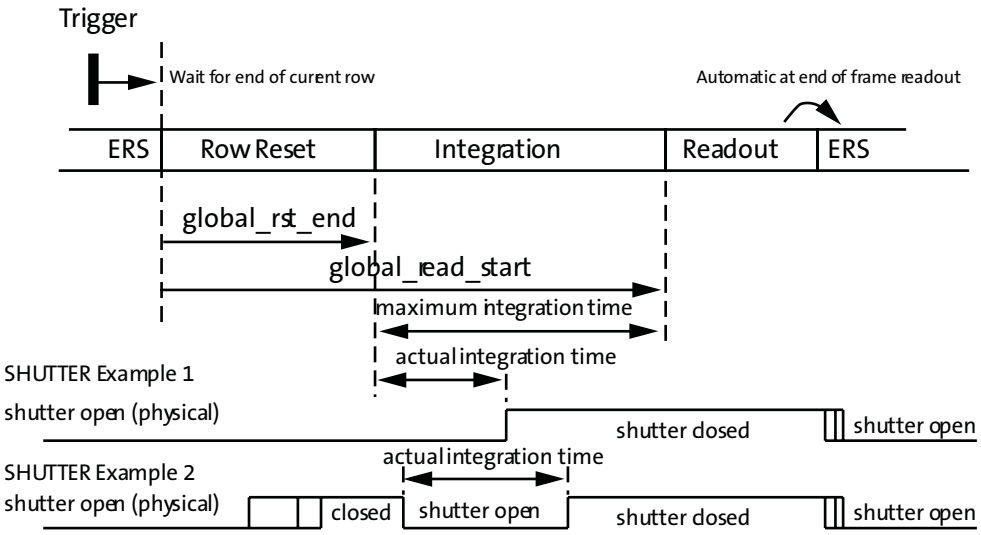
Figure 34: Controlling the Reset and Integration Phases of the Global Reset Sequence



Control of the Electromechanical Shutter

Figure 35 on page 50 shows two different ways in which a shutter can be controlled during the global reset sequence. In both cases, the maximum integration time is set by the difference between global_read_start and global_rst_end. In shutter example 1, the shutter is open during the initial ERS sequence and during the row reset phase. The shutter closes during the integration phase. The pixel array is integrating incident light from the start of the integration phase to the point at which the shutter closes. Finally, the shutter opens again after the end of the readout phase. In shutter example 2, the shutter is open during the initial ERS sequence and closes sometime during the row reset phase. The shutter both opens and closes during the integration phase. The pixel array is integrating incident light for the part of the integration phase during which the shutter is open. As for the previous example, the shutter opens again after the end of the readout phase.

Figure 35: Control of the Electromechanical Shutter



It is essential that the shutter remains closed during the entire row readout phase (that is, until FV has negated for the frame readout); otherwise, some rows of data will be corrupted (over-integrated).

It is essential that the shutter closes before the end of the integration phase. If the row readout phase is allowed to start before the shutter closes, each row in turn will be integrated for one row-time longer than the previous row.

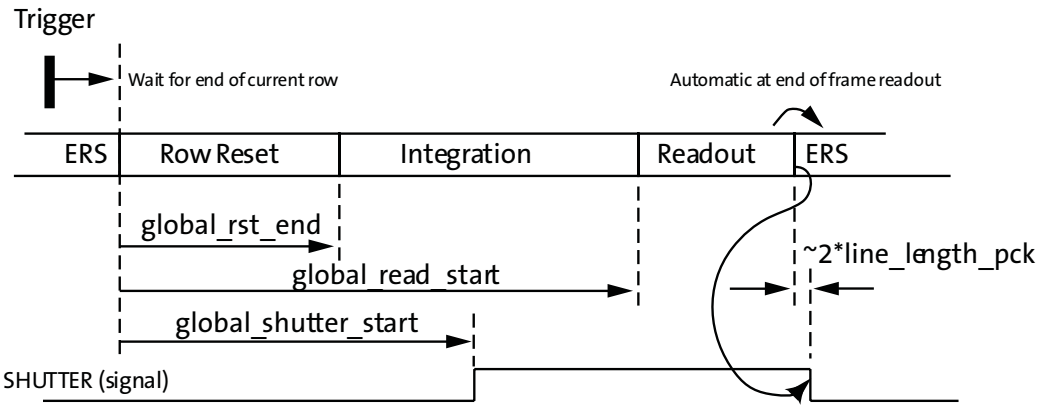
After FV negates to signal the completion of the readout phase, there is a time delay of approximately (10 * line_length_pck) before the sensor starts to integrate light-sensitive rows for the next ERS frame. It is essential that the shutter be opened at some point in this time window; otherwise, the first ERS frame will not be uniformly integrated.

The AR0833 provides a SHUTTER output signal to control (or help the host system control) the electromechanical shutter. The timing of the SHUTTER output is shown in Figure 36 on page 51. SHUTTER is negated by default. The point at which it asserts is controlled by the programming of global_shutter_start. At the end of the global reset readout phase, SHUTTER negates approximately (2 * line_length_pck) after the negation of FV.

This programming restriction must be met for correct operation:

- global_read_start > global_shutter_start.

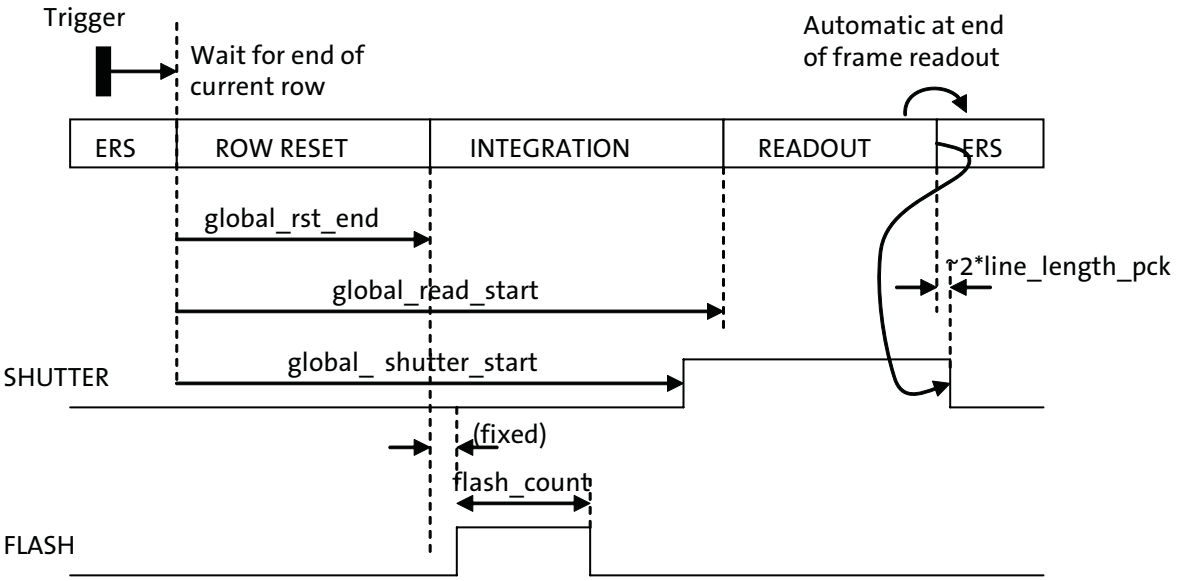
Figure 36: Controlling the SHUTTER Output



Using FLASH with Global Reset

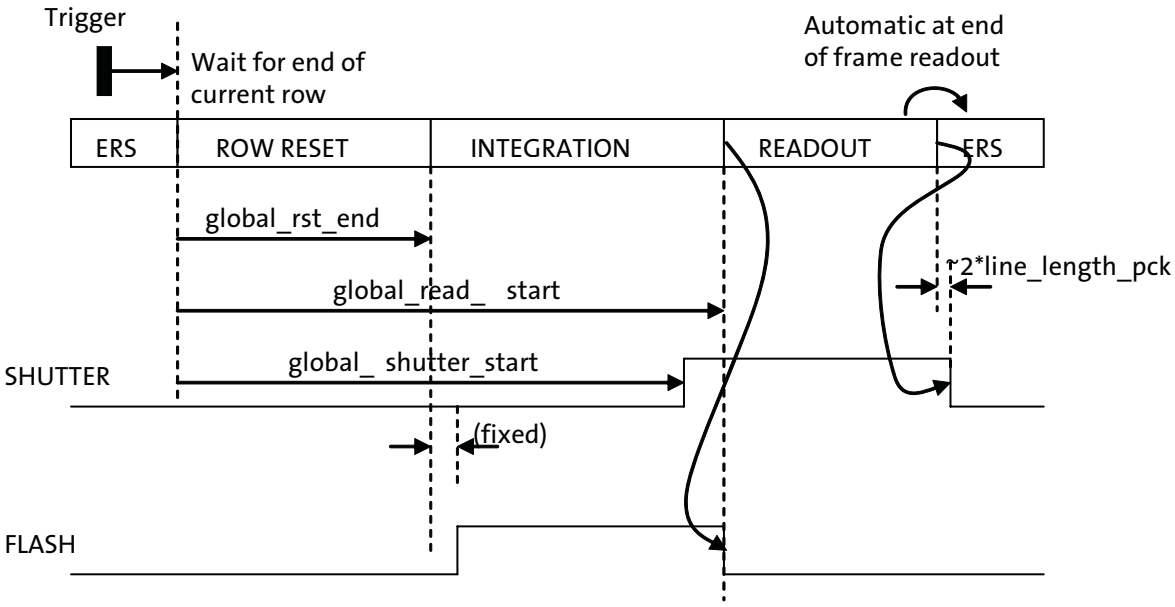
If R0x315E global_seq_trigger[2] = 1 (global flash enabled) when a global reset sequence is triggered, the FLASH output signal will be pulsed during the integration phase of the global reset sequence. The FLASH output will assert a fixed number of cycles after the start of the integration phase and will remain asserted for a time that is controlled by the value of the flash_count register. When flash_count is programmed for value N, (where N is 0–0x3FE) the resulting flash duration is given by N * 512 * (1/vt_pix_clk_freq_mhz), as shown in Figure 37.

Figure 37: Using FLASH with Global Reset



When the flash_count = 0x3FF, the flash signal will be maximized and goes LOW when readout starts, as shown in Figure 38 on page 52. This would be preferred if the latency in closing the shutter is longer than the latency for turning off the flash. This guarantees that the flash stays on while the shutter is open.

Figure 38: Extending FLASH Duration in Global Reset (Reference Readout Start)



External Control of Integration Time

If global_seq_trigger[1] = 1 (global bulb enabled) when a global reset sequence is triggered, the end of the integration phase is controlled by the level of trigger (global_seq_trigger[0] or the associated GPI input). This allows the integration time to be controlled directly by an input to the sensor.

This operation corresponds to the shutter "B" setting on a traditional camera, where "B" originally stood for "Bulb" (the shutter setting used for synchronization with a magnesium foil flash bulb) and was later considered to stand for "Brief" (an exposure that was longer than the shutter could automatically accommodate).

When the trigger is de-asserted to end integration, the integration phase is extended by a further time given by *global_read_start – global_shutter_start*. Usually this means that global_read_start should be set to *global_shutter_start + 1*.

The operation of this mode is shown in Figure 39 on page 53. The figure shows the global reset sequence being triggered by the GPI2 input, but it could be triggered by any of the GPI inputs or by the setting and subsequence clearing of the global_seq_trigger[0] under software control.

The integration time of the GRR sequence is defined as:

$$Integration\ Time = \frac{global\_scale \times [global\_read\_start - global\_shutter\_start - global\_rst\_end]}{vt\_pix\_clk\_freq\_mhz} \quad \text{(EQ 14)}$$

Where:

$$global\_read\_start = (2^{16} \times global\_read\_start2[7:0] + global\_read\_start1[15:0]) \quad \text{(EQ 15)}$$

$$global\_shutter\_start = (2^{16} \times global\_shutter\_start2[7:0] + global\_shutter\_start1[15:0]) \quad \text{(EQ 16)}$$

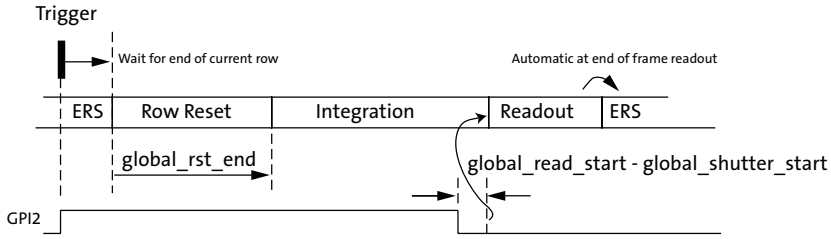
The integration equation allows for 24-bit precision when calculating both the shutter and readout of the image. The global_rst_end has only 16-bit as the array reset function and requires a short amount of time.

The integration time can also be scaled using global_scale. The variable can be set to 0–512, 1–2048, 2–128, and 3–32.

These programming restrictions must be met for correct operation of bulb exposures:

- global_read_start > global_shutter_start
- global_shutter_start > global_rst_end
- global_shutter_start must be smaller than the exposure time (that is, this counter must expire before the trigger is de-asserted)

Figure 39: Global Reset Bulb



Retriggering the Global Reset Sequence

The trigger for the global reset sequence is edge-sensitive; the global reset sequence cannot be retriggered until the global trigger bit (in the R0x315E global_seq_trigger register) has been returned to "0," and the GPI (if any) associated with the trigger function has been negated.

The earliest time that the global reset sequence can be retriggered is the point at which the SHUTTER output negates; this occurs approximately (2 * line_length_pck) after the negation of FV for the global reset readout phase.

Using Global Reset with SMIA Data Path

When a global reset sequence is triggered, it usually results in the frame in progress being truncated (at the end of the current output line). The SMIA data path limiter function (see Figure 40 on page 54) attempts to extend (pad) all frames to the programmed value of y_output_size. If this padding is still in progress when the global reset readout phase starts, the SMIA data path will not detect the start of the frame correctly. Therefore, to use global reset with the serial data path, this timing scenario must be avoided. One possible way of doing this would be to synchronize (under software control) the assertion of trigger to an end-of-frame marker on the serial data stream.

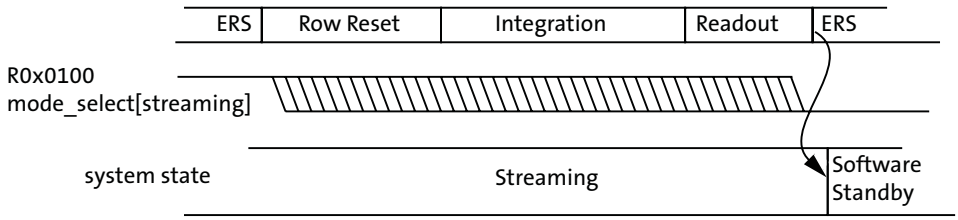
At the end of the readout phase of the global reset sequence, the sensor automatically resumes operation in ERS mode.

The frame that is read out of the sensor during the global reset readout phase has exactly the same format as any other frame out of the serial pixel data interface, including the addition of two lines of embedded data. The value of the coarse_integration_time register within the embedded data matches the programmed values of those registers and does not reflect the integration time used during the global reset sequence.

Global Reset and Soft Standby

If the R0x301A[2] mode_select[stream] bit is cleared while a global reset sequence is in progress, the AR0833 will remain in streaming state until the global reset sequence (including frame readout) has completed, as shown in Figure 40 on page 54.

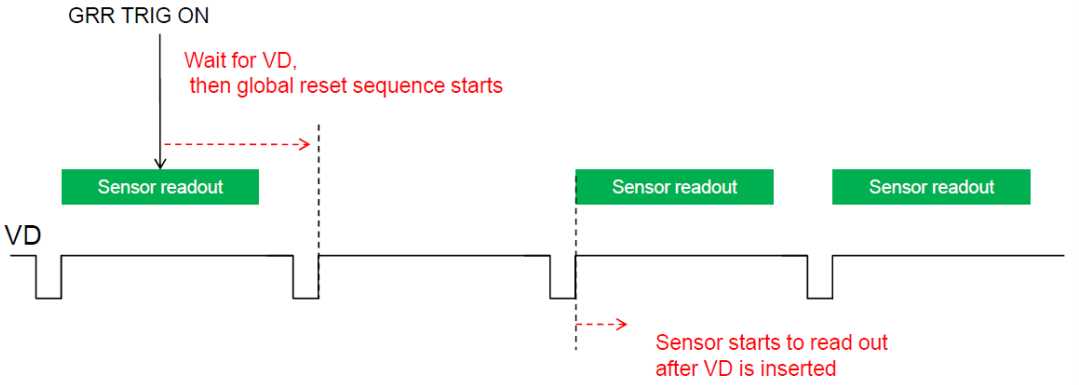
Figure 40: Entering Soft Standby During a Global Reset Sequence



Slave Mode

Slave mode is to ensure having an ERS-GRR-ERS transition without a broken ERS frame before GRR. It requests to trigger/end the GRR sequence through the pin which is similar to the GRR Bulb mode. The major difference to our existing sensor is to start the GRR sequence after the end of the current frame instead of to start immediately in the next following row.

Figure 41: Slave Mode Transition



Gain

AR0833 supports both analog and digital gain.

Analog gain

Analog gain is provided by colamp and ADC reference scaling (there is no ASC gain due to column parallel nature of architecture). Only global (not per-color) coarse gain can be set by analog gain. Global gain register (R0x305E) sets the analog gain. Bits [1:0] set the colamp gain while bits [4:2] are reserved for ADC gain. While the 2-bit colamp gain provides up to 4x analog gain, only LSB (bit [2]) of ADC gain bits is utilized to support 2x ADC gain. Table 16 is the recommended gain setting:

Table 16: Recommended Analog Gain Setting

Colamp Gain Codes (R0x305E[1:0])		ADC Gain Codes (R0x305E[4:2])			Colamp Gain	ADC Gain	Total Gain
0	0	0	0	0	1	1	1
0	1	0	0	0	2	1	2
1	0	0	0	0	3	1	3
1	1	0	0	0	4	1	4
1	0	0	0	1	3	2	6
1	1	0	0	1	4	2	8

Digital Gain

Digital gain provides both per-color and fine (sub 1x) gain. The analog and digital gains are multiplicative to give the total gain. Digital gain is set by setting bits R0x305E[15:5] to set global gain or by individually setting digital color gain R0x3056-C[15:5] where these 11 bits are designed in 4p7 format i.e. 4 MSB provide gain up to 15x in step of 1x while 7 LSB provide sub-1x gain with a step size of 1/128. This sub-1x gain provides the fine gain control for the sensor.

Total Gain

Max. total gain required by design spec is 8x (analog) and 16x (digital) with min. step size of 1/8. The total gain equation can be formulated as:

$$Total\ gain = (1 + dec(R0x305E[1:0])) \times (1 + R0x305E[2]) \times \frac{dec(R0x305X[15:5])}{128} \quad \text{(EQ 17)}$$

where X is 6, 8, A, C, for Gr, B, R and Gb, respectively

Note: Aptina recommends using the registers mentioned above for gain settings. Avoid R0x3028 to R0x3038 unless their mapping to above registers is well understood and taken into account.

Temperature Sensor

A standalone PTAT based temperature sensor has been implemented. The block is controlled independent of sensor timing and all communication happens through the two-wire serial interface.

Internal VCM Driver

The AR0833 utilizes an internal Voice Coil Motor (VCM) driver. The VCM functions are register-controlled through the serial interface.

There are two output ports, VCM_ISINK and VCM_GND, 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 connection. Appropriate filtering would also be required on the actuator supply. Typical values would be a 0.1µF and 10µF in parallel.

Figure 42: VCM Driver Typical Diagram

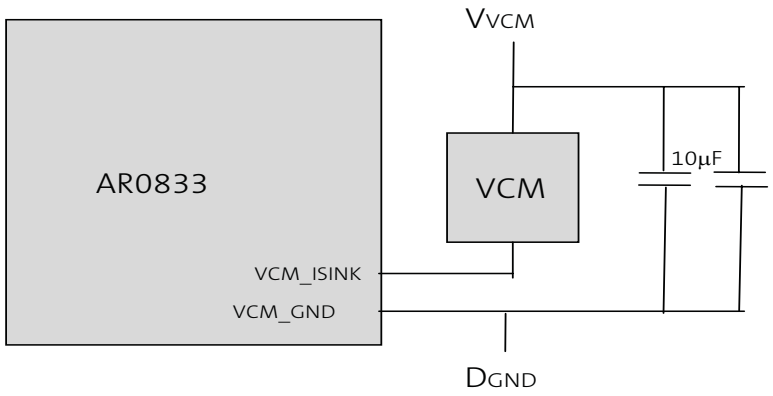


Table 17: VCM Driver Typical

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	90	100	110	mA
	Slew rate (user programmable)	–	–	13	mA/ms

Spectral Characteristics

Figure 43: Quantum Efficiency

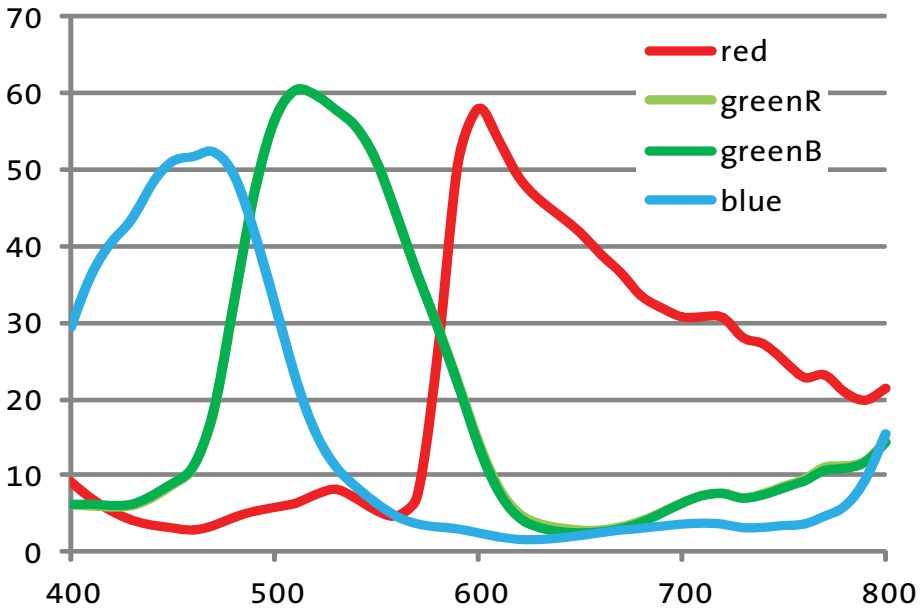


Figure 44: Chief Ray Angle (CRA) vs. Image Height

CRA vs. Image Height Plot

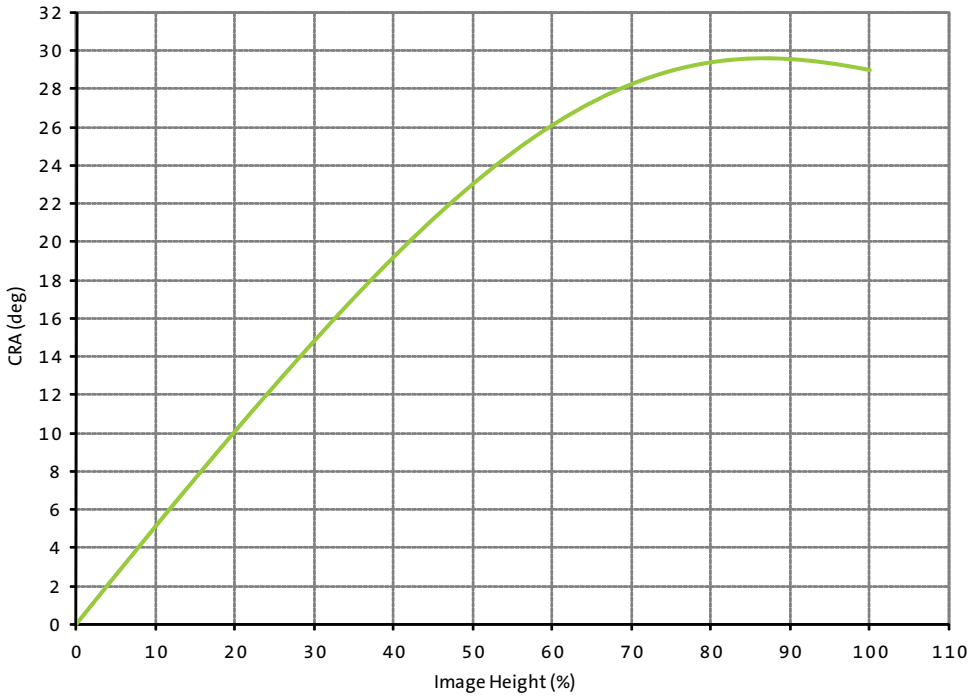


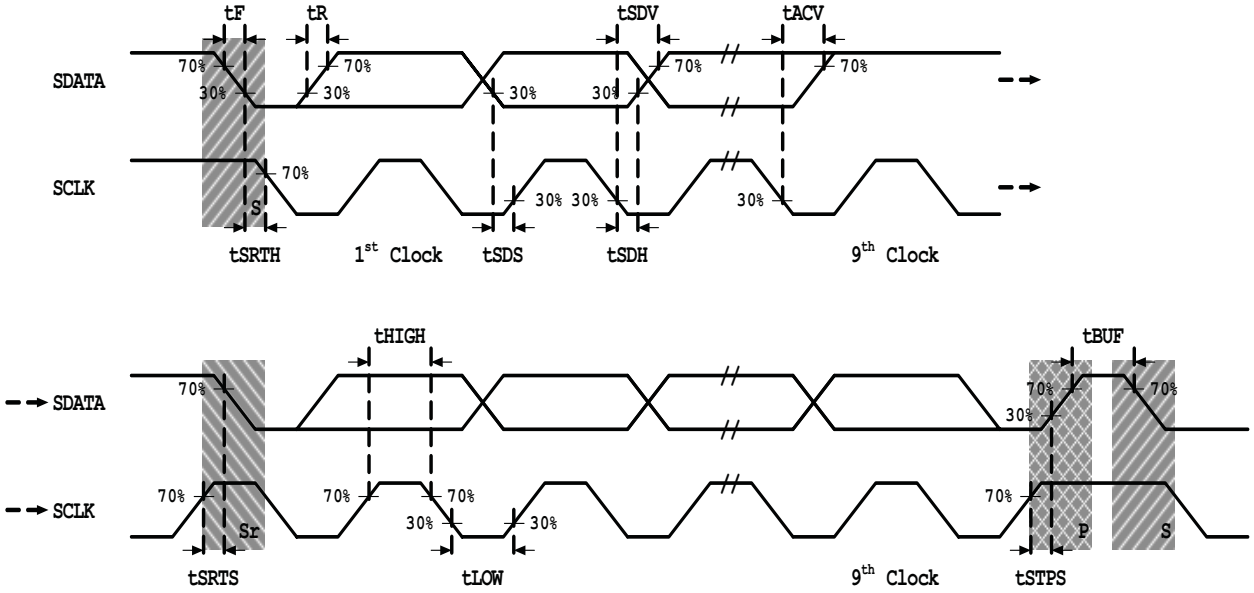
Image Height		CRA
(%)	(mm)	(deg)
0	0	0
5	0.143	2.58
10	0.286	5.12
15	0.428	7.61
20	0.571	10.07
25	0.714	12.47
30	0.857	14.80
35	1.000	17.05
40	1.142	19.18
45	1.285	21.17
50	1.428	23.00
55	1.571	24.64
60	1.714	26.07
65	1.856	27.26
70	1.999	28.21
75	2.142	28.91
80	2.285	29.35
85	2.428	29.55
90	2.570	29.52
95	2.713	29.30
100	2.856	28.94

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 18. Table 19 on page 60 shows the timing specification for the two-wire serial interface.

Figure 45: Two-Wire Serial Bus Timing Parameters



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

Table 18: Two-Wire Serial Register Interface Electrical Characteristics

^{f}EXTCLK = 25 MHz; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_IO} = 1.8V; V_{DD} (digital core) = 1.2V; V_{DD_PLL} = 1.2V; V_{DD_TX} = 1.8V; Output load = 68.5pF; T_J = 55°C

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IL}	Input LOW voltage		-0.5	–	0.3 x V_{DD_IO}	V
V_{IH}	Input HIGH voltage		0.7 x V_{DD_IO}	–	V_{DD_IO} + 0.5	V
I_{IN}	Input leakage current	No pull up resistor; V_{IN} = V_{DD_IO} or D_{GND}	10	–	14	µA
V_{OL}	Output LOW voltage	At specified 2mA	0.11	–	0.3	V
I_{OL}	Output LOW current	At specified V_{OL} 0.1V	–	–	6	mA
C_{IN}	Input pad capacitance		–	–	6	pF
C_{LOAD}	Load capacitance		–	–	N/A	pF

Table 19: Two-Wire Serial Interface Timing Specifications
VDD = 1.7-1.9V; VAA = 2.4 -3.1V; Environment temperature = -30°C to 50°C

Symbol	Definition	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	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
tr	SCLK and SDATA Rise Time	–	300	ns
tf	SCLK and SDATA Fall Time	–	300	ns

EXTCLK

The electrical characteristics of the EXTCLK input are shown in Table 20. 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 AR0833 and the clock is stopped, the EXTCLK input to the AR0833 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 20: Electrical Characteristics (EXTCLK)
f_{EXTCLK} = 24 MHz; VAA = 2.8V; VAA_PIX = 2.8V; VDD_IO = 1.8V; DVDD_1V2 = 1.2V;
Output load = 68.5pF; TJ = 55°C

Symbol	Parameter	Condition	Min	Typ	Max	Unit
^{f}EXTCLK1	Input clock frequency	PLL enabled	6	24	27	MHz
^{t}R	Input clock rise slew rate	CLOAD<20pF	–	2.883	–	ns
^{t}F	Input clock fall slew rate	CLOAD<20pF	–	2.687	–	ns
VIN_AC	Input clock minimum voltage swing (AC coupled)	–	0.5	–	–	V (p-p)
VIN_DC	Input clock maximum voltage swing (DC coupled)	–	–	–	VDD_IO + 0.5	V
^{f}CLKMAX(AC)	Input clock signalling frequency (low amplitude)	VIN = VIN_AC (MIN)	–	–	25	MHz
^{f}CLKMAX(DC)	Input clock signalling frequency (full amplitude)	VIN = VDD_IO	–	–	48	MHz
	Clock duty cycle	–	45	50	55	%
^{t}JITTER	Input clock jitter	cycle-to-cycle	–	545	600	ps
^{t}LOCK	PLL VCO lock time	–	–	0.2	2	ms
CIN	Input pad capacitance	–	–	6	–	pF
IIH	Input HIGH leakage current	–	0	–	10	µA
VIH	Input HIGH voltage	0.7 x VDD_IO	–	VDD_IO + 0.5	–	V
VIL	Input LOW voltage	-0.5	–	0.3 x VDD_IO	–	V

Figure 46: Fall Slew Rates (Cap Load = 25pF)

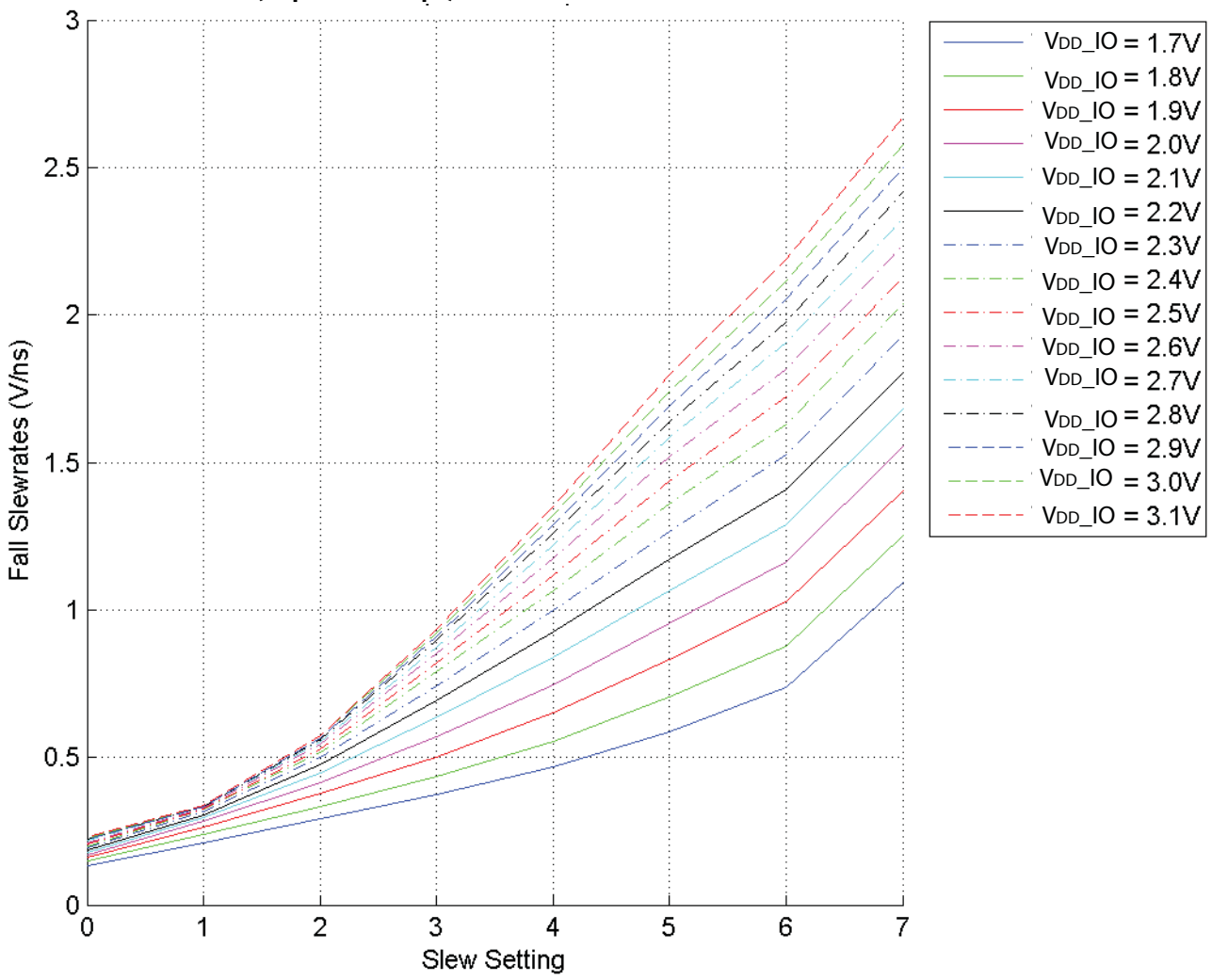
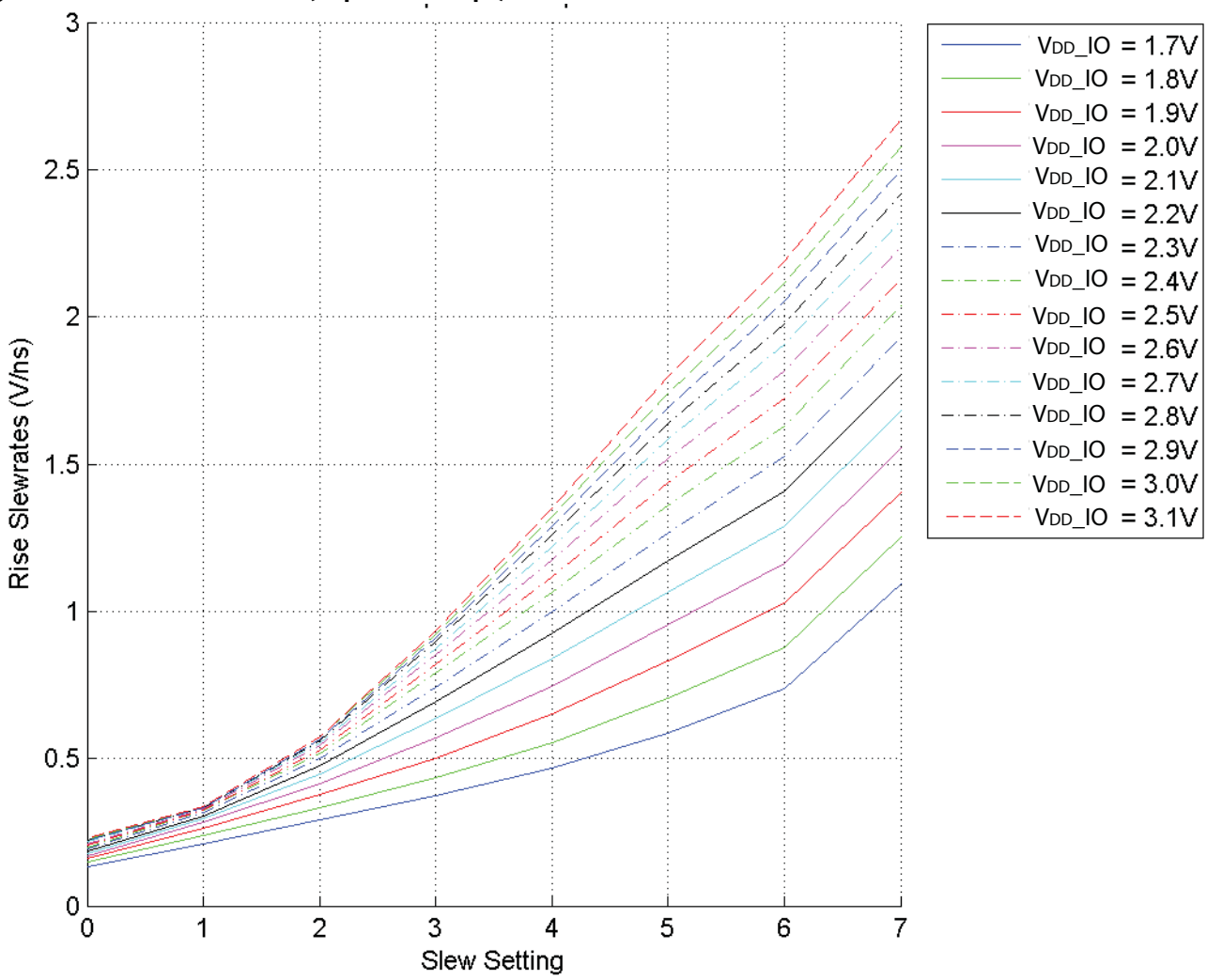


Figure 47: Rise Slew Rates (Cap Load = 25pF)



Serial Pixel Data Interface

The electrical characteristics of the serial pixel data interface (CLK_P, CLK_N, DATA[4:1]_P, and DATA[4:1]_N) are shown in Table 21.

Table 21: Electrical Characteristics (Serial MIPI Pixel Data Interface)

Symbol	Parameter	Min	Typ	Max	Unit
V_{OD}	High speed transmit differential voltage	140		270	mV
V_{CMTX}	High speed transmit static common-mode voltage	150	200	250	mV
ΔV_{OD}	V_{OD} mismatch when output is Differential-1 or Differential-0		<10		mV
Z_{OS}	Single ended output impedance	40	50	62.5	Ω
ΔZ_{OS}	Single ended output impedance mismatch		<14		%
$\Delta V_{CMTX(L,F)}$	Common-level variation between 50–450 MHz		<25		mV
^{t}R	Rise time (20–80%)	150	350	380	ps
^{t}F	Fall time (20–80%)	150	350	380	ps
V_{OL}	LP Output LOW level		–50 to 50		mV
V_{OH}	LP Output HIGH level	1.1		1.3	V
Z_{OLP}	Output impedance of low power parameter	140			Ω
T_{RLP}	15–85% rise time			25	ns
T_{FLP}	15–85% fall time			25	ns
Δv/Δdtsr	Slew rate (C_{LOAD} = 20–70pF)	30			mV/ns

Control Interface

The electrical characteristics of the control interface (RESET_BAR, TEST, GPIO0, GPIO1, GPI2, and GPI3) are shown in Table 22.

Table 22: DC Electrical Characteristics (Control Interface)
f_{EXTCLK} = 24 MHz; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_IO} = 1.8V; DV_{DD_1V2} = 1.2V;
Output load = 68.5pF; T_J = 55°C

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IH}	Input HIGH voltage		0.7 x V_{DD_IO}	–	V_{DD_IO} + 0.5	V
V_{IL}	Input LOW voltage		–0.5	–	V_{DD_IO} x 0.3	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		–	6	–	pF

Operating Voltages

V_{AA} and V_{AA_PIX} must be at the same potential for correct operation of the AR0833.

Table 23: DC Electrical Definitions and Characteristics

f_{EXTCLK} = 24 MHz; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; V_{DD_IO} = 1.8V; DV_{DD_1V2} = 1.2V;
Output load = 68.5pF; T_J = 70°C; Mode = Full Resolution (3264x2488); Frame rate = 30 fps

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{AA}	Analog voltage		2.5	2.8	3.1	V
V_{AA_PIX}	Pixel supply voltage		2.5	2.8	3.1	V
DV_{DD_1V2}	Digital voltage		1.14	1.2	1.3	V
DV_{DD_1V8}	PHY digital voltage		1.7	1.8	1.9	V
V_{DD_IO}	I/O digital voltage		1.7	1.8	1.9	V
			2.5	2.8	3.1	V
$DV_{DD\_1V2\_PHY}$	MIPI supply		1.14	1.2	1.3	V
H/W Standby Current Consumption			–	–	30	µA
Output Driving Strength			10	–	–	mA
Slew Rate			–	0.7	––	µV/sec
Programming voltage for OTPM (V_{PP})			6	6.5	7	V

Typical Operating Current Consumption (MIPI)

Table 24: Typical Operating Current Consumption (MIPI)

f_{EXTCLK} = 24 MHz; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; DV_{DD_1V8} = 1.8V; V_{DD_IO} = 1.8V; DV_{DD_1V2} = 1.2V; Tj = 25°C

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{AA}	Analog supply current	Full resolution 4:3 - 30fps	--	69.76	--	mA
		Full resolution 16:9 - 30fps	--	69.26	--	mA
		1080p, 30fps	--	69.28	--	mA
		720p, 60fps	--	69.26	--	mA
		VGA, 60fps	--	69.78	--	mA
I_{AA_PIX}	Pixel supply current	Full resolution 4:3 - 30fps	--	12.04	--	mA
		Full resolution 16:9 - 30fps	--	9.19	--	mA
		1080p, 30fps	--	9.19	--	mA
		720p, 60fps	--	9.73	--	mA
		VGA, 60fps	--	12.10	--	mA
I_{DD_1V8}	OTPM read supply current	Full resolution 4:3 - 30fps	--	40.39	--	mA
		Full resolution 16:9 - 30fps	--	35.66	--	mA
		1080p, 30fps	--	35.67	--	mA
		720p, 60fps	--	29.03	--	mA
		VGA, 60fps	--	31.08	--	mA
I_{DD_IO}	I/O supply current	Full resolution 4:3 - 30fps	--	0.02	--	mA
		Full resolution 16:9 - 30fps	--	0.02	--	mA
		1080p, 30fps	--	0.03	--	mA
		720p, 60fps	--	0.02	--	mA
		VGA, 60fps	--	0.02	--	mA

Table 24: Typical Operating Current Consumption (MIPI) (continued)
f_{EXTCLK} = 24 MHz; V_{AA} = 2.8V; V_{AA_PIX} = 2.8V; DV_{DD_1V8} = 1.8V; V_{DD_IO} = 1.8V; DV_{DD_1V2} = 1.2V; Tj = 25°C

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{DD_1V2}	MIPI supply current	Full resolution 4:3 - 30fps	--	81.94	--	mA
		Full resolution 16:9 - 30fps	--	74.00	--	mA
		1080p, 30fps	--	70.00	--	mA
		720p, 60fps	--	60.00	--	mA
		VGA, 60fps	--	59.27	--	mA

Absolute Maximum Ratings

Caution **Stresses greater than those listed in Table 25 may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect reliability. This is a stress rating only, and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.**

Table 25: Absolute Max Voltages

Symbol	Parameter	Condition	Min	Max	Unit
V_{DD_1V2}	Digital/Analog voltage		-0.3	1.5	V
DV_{DD_1V8}	OTPM		-0.3	2.1	V
V_{DD_IO}	I/O digital voltage		-0.3	3.5	V
V_{AA}	Analog voltage		-0.3	3.5	V
V_{AA_PIX}	Pixel supply voltage		-0.3	3.5	V
$DV_{DD\_1V2\_PHY}$	MIPI supply		-0.3	1.5	V

MIPI Specification Reference

The sensor design and this documentation is based on the following MIPI Specifications:

- MIPI Alliance Standard for CSI-2 version 1.0
- MIPI Alliance Standard for D-PHY version 1.0

Revision History

Rev. E 7/12/12

- Updated Power consumption in Table 2, “Key Performance Parameters,” on page 1

Rev. D 7/9/12

- Updated “Features” on page 1
- Updated Table 2, “Key Performance Parameters,” on page 1
- Updated Table 3, “Modes of Operation and Power Consumption at 100% FOV,” on page 2
- Updated Figure 3: “Typical Application Circuit—MIPI Connection,” on page 10
- Added “System States” on page 13
- Updated Figure 5: “Recommended Power-up Sequence,” on page 15
- Updated Table 7: “Power-up Sequence,” on page 15
- Updated Figure 6: “Recommended Power-down Sequence,” on page 16
- Updated Figure 7: “Hard Standby and Hard Reset,” on page 17
- Updated Figure 15: “Clocking Configuration,” on page 28
- Added “Interlaced HDR Readout” on page 30
- Updated “Binning, Skipping, and Summing Mode” on page 44
- Added Table 14, “Available Skip, Bin, and Sum Modes in the AR0833 Sensor,” on page 44
- Updated “Scaler” on page 45
- Updated Table 16, “Recommended Analog Gain Setting,” on page 55
- Updated “Internal VCM Driver” on page 56
- Added Table 24, “Typical Operating Current Consumption (MIPI),” on page 64

Rev. C 2/28/12

- Updated dynamic range in Table 2, “Key Performance Parameters,” on page 1

Rev. B 2/14/12

- Updated to Preliminary
- Updated headers
- Updated trademarks
- Updated “Features” on page 1
- Updated Figure 1: “Top Level Block Diagram,” on page 8
- Updated Figure 44: “Chief Ray Angle (CRA) vs. Image Height,” on page 58

Rev. A, Advance 10/18/11

- Initial release

10 Eunos Road 8 13-40, Singapore Post Center, Singapore 408600 prodmktg@aptina.com www.aptina.com