1/3-Inch CMOS Digital Image Sensor

AR0331 Developer Guide, Rev. B

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

AR0331
Developer Guide

Table of Contents

List of Figures

List of Tables

Introduction

This Developer Guide provides detailed descriptions and usage guidelines for various features of the AR0331 HDR Sensor. Also provided are guidelines for optimal settings for various use cases. For detailed electrical and timing specifications or register descriptions, refer to the AR0331 Data Sheet and the AR0331 Register Reference documents, respectively.

AR0331 Optimal Setting Guidelines

The AR0331 HDR Sensor has many built-in features and is capable of many resolutions and frame rates. Guidelines for setting resolution and frame rate are provided in this section. Detailed settings for the many features are provided throughout the remainder of this Developer Guide.

Resolution

Aptina's AR0331 sensor is capable of a maximum resolution of 2048 x 1536 at up to 30fps. Registers y_addr_start, x_addr_start, y_addr_end, and x_addr_end are used to specify the image window. The minimum vale for x_addr_start is 6 and the maximum value is 2053. The minimum y_addr_start and maximum y_addr_end are 0 and 1543, respectively, but the maximum number of active rows must not exceed 1536.

Frame Rate

Achieving the desired frame rate at the proper resolution is a balancing act between row timing and the number of rows in the image. Integration time and the pixel clock frequency are additional factors. To avoid potential issues with shading, it is recommended that the line_length_pck be set below 1400 or above 2000. For example, with the AR0331 configured to run 1080p60, it is recommended that a 74.25MHz pixel clock be used with a line length (line_length_pck) of 1100. To run the same resolution at 30fps, the line_length_pck should be increased to 2200. In full resolution at 30fps (HiSPi), the line_length_pck should be set to 1120, with the number of lines (frame_length_lines) set to 2209. Refer to the AR0331 Data Sheet for minimum specifications for line_length_pck and frame_length_lines.

Integration Time

When running the AR0331 sensor in HDR mode, there are a few points to remember when setting integration time. The AR0331 sensor has 70 line buffers for the T2 exposure. The coarse integration time (R0x3012) represents the T1 (long) exposure time. To preserve the exposure ratio, it should be set to a value of (R * 70), where R is the T1/T2 exposure ratio. If the coarse integration time is increased, the exposure ratio will be calculated internally to properly linearize the combined image. When changing exposure ratios, there must be sufficient idle rows in the vertical blanking time to allow increases in integration time. For example, if the exposure ratio is 16 and coarse integration time is 128 rows, then T2 is 8 rows. If the exposure ratio is changed to 8 (coarse integration time stays the same), then the new T2 will increase by 8 rows to 16. There must be 8 idle rows in the frame time (frame_length_lines) to allow this jump. The coarse integration time is limited to approximately 4000 rows.

Pedestals

The AR0331 sensor has various pedestals throughout the data path. The appropriate pedestal settings depend on enabled features. Table 1 provides a summary of proper pedestal settings based on configuration. A description of each pedestal may be found in the appropriate section of the Developer Guide.

Table 1: Pedestal Settings

Mode	Motion Compensation	ALTM	Noise Pedestal (0x30FE)	Data Pedestal (0x301E)	ALTM Pedestal (0x2450)	ADACD Pedestal (0x320A)
Linear	N/A	ON	128 (0x0080)	0	0	128 (0x0080)
		OFF	128 (0x0080)	168 (0x00A8)	0	128 (0x0080)
HDR	ON	ON	0	0	0	0
		OFF	0	168 (0x00A8)	0	0
	OFF	ON	128 (0x0080)	0	0	128 (0x0080)
		OFF	128 (0x0080)	168 (0x00A8)	0	128 (0x0080)

Adaptive Local Tone Mapping

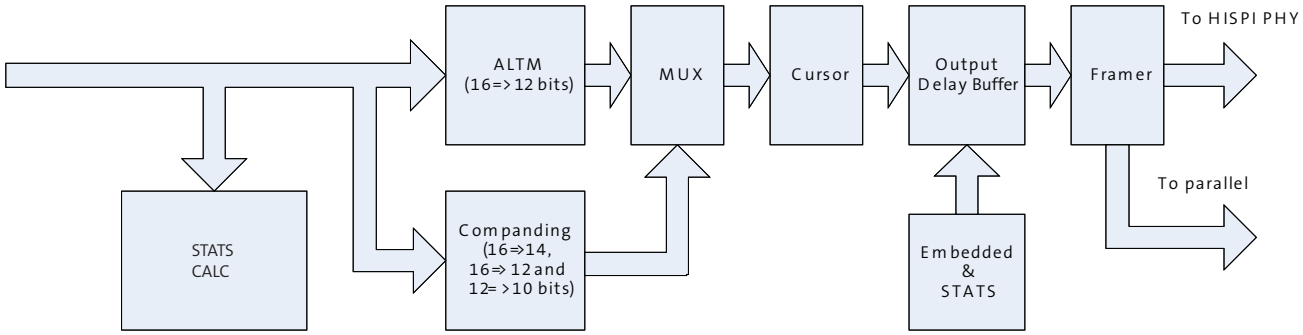
Real-world scenes often have a very high dynamic range (HDR) that far exceeds the electrical dynamic range of the imager. Dynamic range is defined as the luminance ratio between the brightest and the darkest objects in a scene. In recent years many technologies have been developed to capture the full dynamic range of real world scenes. The multi-exposure method is a widely-adapted technique for capturing high dynamic range images. This technique combines a series of low dynamic range images of the same scene taken under different exposure times into a single HDR image. Aptina's recently developed AR0331 is a multi-exposure-based HDR imaging sensor.

Even though digital imaging technology enables us to capture the full dynamic range, we are still limited by the low dynamic range of display devices. Today's typical LCD monitor has a contrast ratio of around 1,000:1, but it is not atypical for an HDR image to have a contrast ratio of around 250,000:1. Therefore, in order to reproduce HDR images on a low dynamic range display device, the captured high dynamic range must be compressed to the available range of the display device. This is commonly referred to as tone mapping.

Adaptive Local Tone Mapping (ALTM) uses a spatially varying mapping function determined by the neighborhood of a pixel, which allows an increase in the local contrast and the visibility of some details of the image. Local methods usually yield more pleasing results than global methods because they exploit the fact that human vision is more sensitive to local contrast.

The AR0331 implements ALTM in the Bayer domain. The advantage of doing tone mapping on the Bayer domain is that common 10-bit or 12-bit pipelines may be used for the HDR image rendering.Figure 1 shows the output data path with the location of the ALTM block.

Figure 1: Output Data Path



Using ALTM

ALTM should always be enabled when the AR0331 is in HDR mode. The only reason to disable ALTM in HDR mode is if a custom tone mapping algorithm is to be used off chip. When ALTM is enabled, companding should be disabled (0x31D0 = 0).

When controlling digital gains for white balance from an off chip controller, the digital gain values should be written back to the image sensor. The ALTM algorithm relies on a properly exposed and correctly white balanced image at the input to the ALTM block for optimal results.

If the AR0331 is in HDR mode and ALTM is disabled, a very dark image will be produced. This is because the 16-bit HDR image cannot be transmitted across a 12-bit parallel (or 12-bit HiSPi) interface. The lower four bits will be truncated, resulting in a greatly reduced dynamic range.

ALTM may be useful in linear mode for certain applications. ALTM may enhance dark regions (with increased noise) without the flicker associated with HDR multi-exposure imaging. This produces an “HDR-like” linear image. The default settings for ALTM in linear mode are the same as for HDR mode. Gamma in both cases should be set to 1.

Enabling ALTM

ALTM is enabled by setting altm_bypass, 0x2400[0] = 0, while setting altm_control_enable, 0x2400[1] = 1. The sensor should be placed in standby mode when writing to register R0x2400[0]. The data pedestal (R0x301E) should normally be set to 0 while ALTM is enabled. There are some situations, however, that setting the data pedestal to zero may cause a bit too much black level clipping. If this is the case, a slightly higher value such as 0x10 may help reduce clipping. When ALTM is disabled, the data pedestal value should be set to 168 (0x00A8).

Disabling ALTM

If an off-chip tone mapping solution is desired, the ALTM block may be disabled and/or bypassed by setting altm_control_enable, 0x2400[1] = 0, and/or setting altm_bypass, 0x2400[0] = 1. Bypassing the ALTM block (rather than disabling) will save approximately 30-40mW of power. The sensor should be placed in standby mode when writing to register R0x2400[0]. If the interface is configured as 12 bits, the bottom 4 bits of the 16-bit HDR pixel data will be dropped. To prevent data loss due to dropped bits across

an interface less than 16 bits, the companding block should be enabled to compress the HDR image. The data should be decompanded, processed, and then passed through the off-chip tone mapping block for proper display.

Customizing ALTM Parameters

There are several registers provided to customize the ALTM algorithm.

Using Statistics Generation Exclusion Window

There are situations where it is not desired to include the entire image for statistics generation. Statistics include minimum and maximum luminance values and image averages. The ALTM block will extract statistics that are specific to the ALTM algorithm. These statistics are different than those generated from the auto exposure circuitry, and are generated from within the ALTM block. In general, statistics should only be extracted from the useful portion of the image. Four registers are used to set a window to be excluded from statistics generation. These set the x and y starting row/column, and the width and height of the excluded regions. See Table 2 on page 10 for more details.

Adjusting Settling Time

The altm_control_damper register (0x2440) may be used to set the settling time. This is useful in situations where the brightness of the scene may change and a smooth transition is desired. The damping may be adjusted using the altm_control_damper register. Setting a low value increases the response time, whereas a larger value results in rapid changes in brightness. Setting too high of a value may result in unwanted oscillations in brightness level. See Table 2 on page 10 for more details.

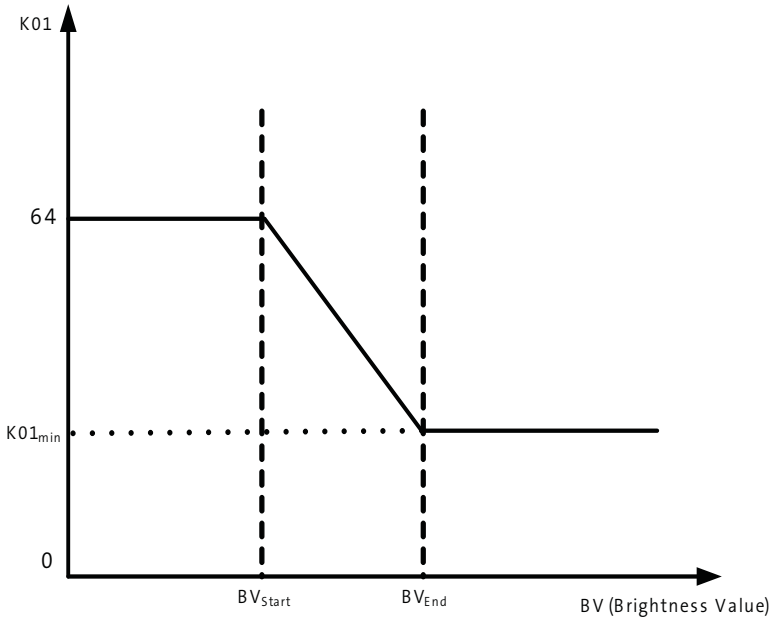
Adjusting Image Brightness

Several parameters control the brightness of the resulting tone-mapped image. The most important are altm_control_key_k0 (0x2442), altm_control_key_k01_lo (0x2444), and altm_control_key_k01_hi (0x2446). It is recommended that altm_control_key_k0 be held at the default value while adjusting altm_control_key_k01_lo and altm_control_key_k01_hi to control the brightness of the tone-mapped image. The parameter altm_control_key_k01 may be dynamically controlled based on the scene dynamic range and the scene brightness value estimated from the auto exposure statistics. The start and end points of the transition region (BVStart and BVEnd) may be programmed into the ISP. If the brightness value from the Auto Exposure statistics is below BVStart, then a lower value of altm_control_key_k01 should be used. If the scene brightness is above BVEnd, then a higher value for altm_control_key_k01 should be used to darken the image. The slope of the transition region can be set in the ISP based on the mapping of K01 values to the brightness value. See Figure 2. In general, increasing altm_control_key_k01_lo and altm_control_key_k01_hi will result in a darker image. Coarse adjustments may be made using altm_control_key_k01_hi, while fine tuning may be done with altm_control_key_k01_lo. It is recommended that altm_control_key_k01_hi be restricted to a decimal value of 64 or below. In low light scenes, it is recommended that altm_control_key_k01_hi be increased as shown in Figure 2. The values selected for BVStart and BVEnd will require a trade-off between the level of detail desired and the amount of noise that is acceptable.

The altm_control_min_factor and altm_control_dark_floor may be used to stretch the lower end of the dynamic range. Similarly, altm_control_max_factor, and altm_control_bright_floor may be used to stretch the high end of the dynamic range. Further optimization may be done using altm_power_gain (R0x2410) and altm_power_offset (R0x2412). altm_power_gain may be used to adjust the brightness in

highlight regions of the image. Increasing the value of altm_power_gain will darken the bright regions of the image. Similarly, altm_power_offset may be used to control the brightness in low light regions of the image. Decreasing the value of altm_power_offset will brighten the dark regions of the image. Because of implementation constraints, altm_power_offset may also affect the bright regions of the image. For the default settings of altm_power_gain and altm_power_offset, the pipeline RGB gamma should be set to 1. See Table 2 for ALTM Registers, descriptions, and default settings.

Figure 2: Updating K01 with Brightness Value



Adjusting Black Level

The overall intensity level of the image may be adjusted by changing the value of the altm_out_pedestal (0x2450). The altm_out_pedestal is an adder applied to the pixel data at the very end of the output data path. It is recommended to set this register to 0x0000. Increasing the altm_out_pedestal will raise the black level and reduce the dynamic range of the image due to a reduction in the number of output ADC codes.

Table 2: ALTM Registers

Register/Field Name	Address	Default Value	Description
altm_control[1:0]	0x2400[0]	0x00	altm_bypass 0: Data goes through Adaptive Local Tone Mapping block. 1: Data does not go through the Adaptive Local Tone Mapping.
	0x2400[1]	0x01	altm_control_enable 0: Adaptive Local Tone Mapping disabled. 1: Adaptive Local Tone Mapping enabled.
altm_power_gain	0x2410	0x0010	Controls brightness of highlight regions. Legal values: [0, 127].
altm_power_offset	0x2412	0x0010	Controls brightness of low light regions. Legal values: [0, 63].
altm_fsharp_v	0x2420	0x0013	Reflectance sharpening strength. Legal values: [0, 63].

Table 2: ALTM Registers (continued)

Register/Field Name	Address	Default Value	Description
altm_stats_ex_win_x_start	0x2422	0x0000	First column of the exclusion window for ALTM statistics. Legal values: [0, 4095].
altm_stats_ex_win_width	0x2424	0x0000	Width of the exclusion window for ALTM statistics. Legal values: [0, 4095].
altm_stats_ex_win_y_start	0x2426	0x0000	First row of the exclusion window for ALTM statistics. Legal values: [0, 4095].
altm_stats_ex_win_height	0x2428	0x0000	Height of the exclusion window for ALTM statistics. Legal values: [0, 4095].
altm_control_min_factor	0x2438	0x0010	Adjustment to control how much to stretch the low end of the image. Legal values: [0, 255].
altm_control_max_factor	0x243A	0x0020	Adjustment to control how much to stretch the high end of the image. Legal values: [0, 255].
altm_control_dark_floor	0x243C	0x0000	Floor value in calculating the minimum illuminant. Legal values: [0, 65535].
altm_control_bright_floor	0x243E	0x0200	Floor value in calculating the maximum illuminant. Legal values: [0, 65535].
altm_control_damper	0x2440	0x02	Damping factor in changing luma values. Legal values: [0, 64].
altm_control_key_k0	0x2442	0x80	Parameter k0 to calculate the key that control brightness of the tone-mapped image. Legal values: [0, 255].
altm_control_key_k01_lo	0x2444	0x0000	Parameter k0*k1 to calculate the key that control brightness of the tone-mapped image. Legal values: [0, 65535].
altm_control_key_k01_hi	0x2446	0x04	Parameter k0*k1 to calculate the key that control brightness of the tone-mapped image. k01 = altm_control_key_k01_hi * 2^16 + altm_control_keyk01_lo. Legal values: [0, 64].
altm_out_pedestal	0x2450	0x0000	Pedestal added to the output of ALTM. Legal values: [0, 4095].

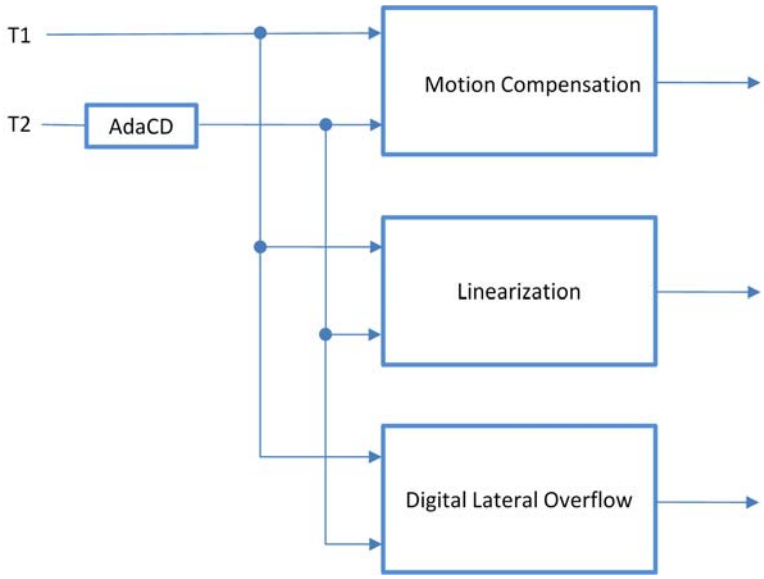
Motion Compensation and Linearization

The Aptina AR0331 image sensor is capable of capturing images with a high dynamic range. To extend the dynamic range of the sensor, the AR0331 incorporates a multiple exposure capture method. This involves capturing the same scene at two different exposures - one longer exposure (T1), and one shorter exposure (T2) that is determined by the selected exposure ratio. The two captured images are then processed by one of three digital blocks prior to output:

1. Motion Compensation,
2. Linearization, or
3. Digital Lateral Overflow (DLO)

The final output is a linearized (prior to tone mapping or companding) 16-bit image with increased dynamic range. The digital block used to process the 16-bit HDR image depends on the scene and customer preference. Figure 3 illustrates a portion of the data path when the AR0331 is configured in HDR mode. Only one of the three blocks is active at a time. Each of these options is described in more detail. The AdaCD block is described in the Adaptive Color Difference (AdaCD) section.

Figure 3: Portion of HDR Data Path



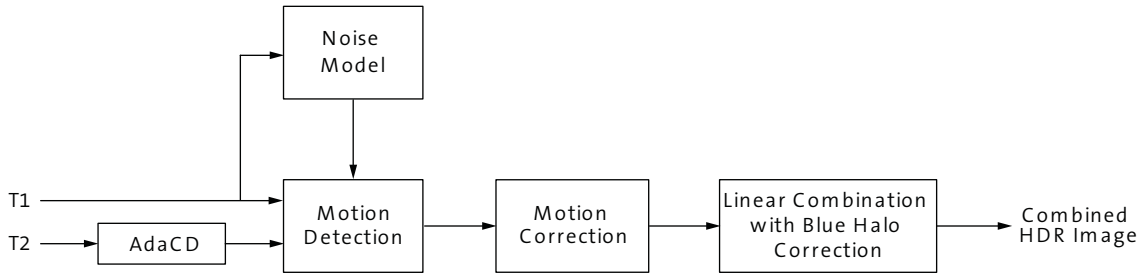
Motion Compensation

A potential problem occurs in a multiple exposure methodology when an object is in motion. This is especially problematic when a bright object is moving in front of a dark background, or vice versa. The object is in one location during the longer T1 exposure, but is in a different location during the shorter T2 exposure. This results in a badly formed combined image.

Since multiple exposure combination is in the Bayer domain, transition areas between T1 and T2 may contain some saturated pixels of one color channel where T2 pixel values are used. At the same time other pixels in other color channels may not be saturated and T1 pixel values are used. Image motion may cause significant color distortion in these transition areas.

The AR0331 motion compensation algorithm is designed to reduce the artifacts described above. The difference between the T1 and T2 exposures is detected, and corrected by replacing T1 with T2 in motion areas. The T1 exposure is then combined with the T2 exposure in the linear combination block. Blue halo correction and a noise filter complete the signal path (See Figure 4). The following sections describe how to enable and configure the motion compensation registers.

Figure 4: Motion Compensation Block Diagram



Enabling Motion Compensation

The motion compensation algorithm is enabled by setting register R0x318C[14] = 1. When enabling motion compensation, all associated features should also be enabled. For example, the smoothing filter (R0x318C[15] = 1) and blue halo correction (R0x318C[0] = 1) should also be enabled. The noise pedestal (R0x30FE) and AdaCD pedestals (R0x320A) should be set to 0x0 while motion compensation is enabled. When disabled, the noise pedestal and AdaCD pedestals should be set to 0x0080.

Configuring Motion Compensation

Several registers are provided to configure the motion compensation algorithm. The most important of these are the motion_detect_q1 (R0x3198[7:0]) and the motion_detect_q2 (R0x3198[15:8]) parameters. Motion_detect_q1 may be thought of as a motion detection threshold. Anything above this threshold will be detected as motion. The range of values is from 0 to 255, with a default value of 30. A value of 0 will detect everything as motion, while a value of 255 effectively disables the motion detection. The selected value of motion_detect_q1 should be adjusted based on applied analog gain. This is to compensate for the increased noise inherent in high gain configurations. The motion_detect_q1 parameter should be configured as indicated in equation 1:

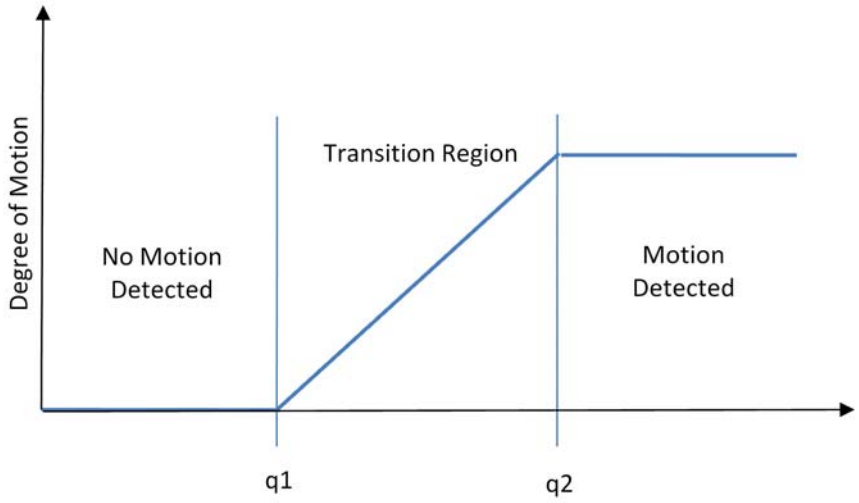
$$motion\_detect\_q1 = motion\_detect\_q1\ (default) \times \sqrt{analog\ gain} \quad \text{(EQ 1)}$$

Example: Using the default value of q1, and an analog gain of 8x, the value of motion_detect_q1 should be changed to 30 x $\sqrt{8}$ = 85 to account for the increase in noise at 8x gain. This value is approximate, and a higher value may be required depending on operating conditions to correctly compensate for increased noise.

The motion_detect_q2 parameter determines the detection range (sensitivity) allowing a smooth transition between regions of detected motion and regions in which no motion is detected (see Figure 5 on page 14). It acts as a coefficient, or scaling factor relative to q1. The default value of q2 is 6. Setting motion_detect_q2 too low may result in motion artifacts since there would be very little transition between regions of detected

and non-detected motion. Setting motion_detect_q2 too high may result in motion being undercompensated. It is recommended to use the default value of motion_detect_q2, and to adjust motion_detect_q1 as described earlier.

Figure 5: Illustration of q1 and q2 Parameters



Following motion detection is the motion correction block. This proprietary algorithm will smoothly replace T1 pixels with T2 pixels in areas of detected motion. After motion correction, the T1 and T2 exposures are linearized and combined into a 16-bit HDR image. The T1, T2, and combined regions may be defined using the s2_threshold and s12_range parameters for linearization, as described in the linearization section. During linear combination, blue halo correction is applied. This algorithm will move some of the brightest pixels that would normally fall in the T1 region and move them to T2 to correct for blue halo artifacts. After a proprietary noise filtering algorithm, the sensor is capable of outputting the combined image, only T1 pixels, only T2 pixels, or interleaved T1 and T2 pixels. See “Data Selection and Interleaved Mode” on page 17 for more details.

Disabling Motion Correction

The motion correction algorithm may be disabled by resetting the motion_correction_enable bit (R0x318C[14] = 0). When disabling motion correction, blue halo and the smoothing filter should also be disabled. Setting the motion_detect_q1 value to 255 as described earlier disables motion detection, but does not bypass the motion correction algorithm. The motion_correction_enable bit should be used to completely disable the motion correction algorithm.

Linearization

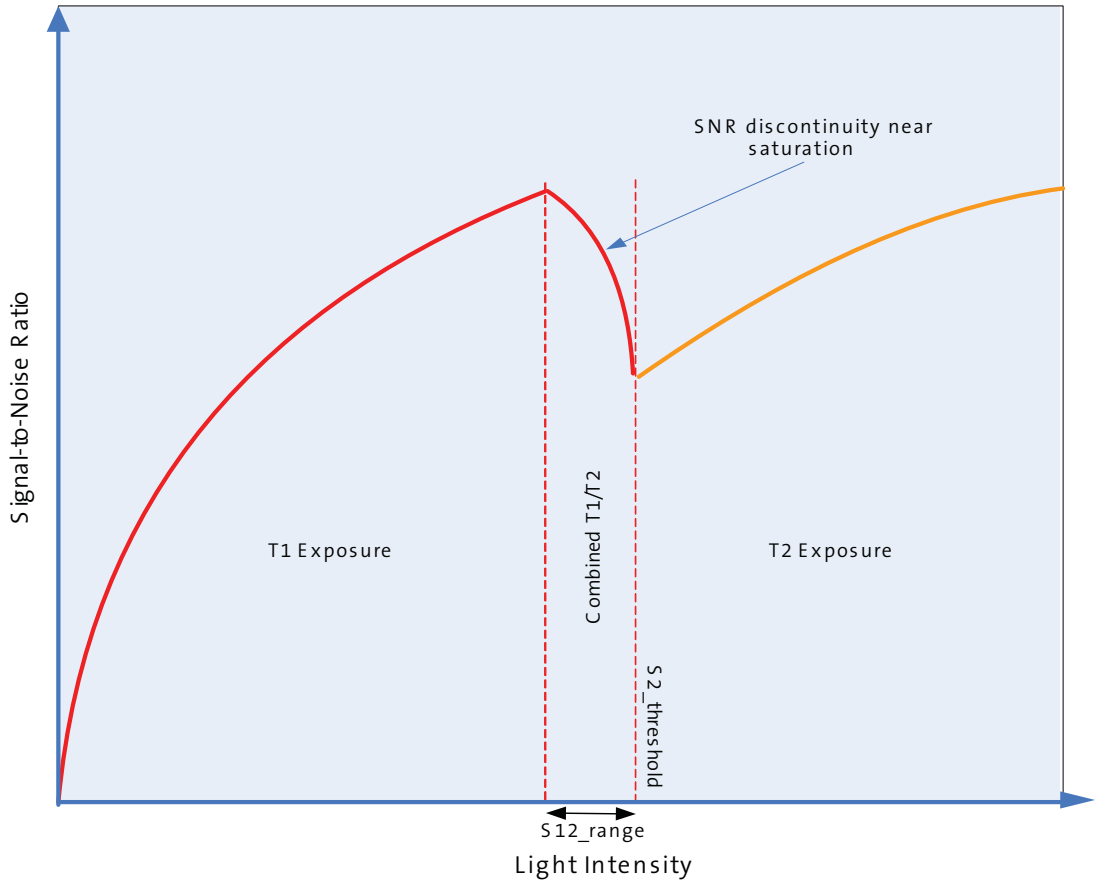
A separate linearization block is used if motion correction and DLO are disabled. Linearization is the process whereby the 12-bit T1 and T2 exposures are combined to form a 16-bit image with a higher dynamic range than the individual exposures. The longer T1 exposure is used for the darker regions of the image, and the shorter T2 exposure covers the brighter regions. If the linearization algorithm were to specify a hard transition between T1 and T2 exposures, a noisy boundary would be seen in the combined image between the T1 and T2 regions due to the SNR discontinuity as shown in Figure 6 on page 15. Two parameters are provided to configure when the transition

region between T1 and T2 begins: how wide of a region will see a combined T1/T2 image, and where the T2 image begins.

The s2_threshold (R0x318A) is used to set the point above which only pixel values from T2 are chosen. Setting this value too high may prevent one or more color channels from saturating. Setting s2_threshold too low will reduce the dynamic range of the sensor. The other parameter, s12_range (R0x3192[12:0]) defines the width of the transition region between all T1 and all T2 pixel values. Anything between S1 and S2 will result in a combination of T1 and T2 pixel values, based on a proprietary weighting function. If the transition region is too small, the boundary between T1 and T2 exposures may result in increased noise due to a sharper transition in signal-to-noise ratio. If s12_range is too large, the cleaner T1 region is reduced, while the combined portion of the image is increased. Since T2 is a noisier, shorter exposure, this will result in increased noise in the overall combined image.

Figure 6 shows the relationship between the s1_threshold, s12_range, and the T1 and T2 exposures.

Figure 6: S2_Threshold, S12_Range, and Relationship to T1 and T2 Exposures

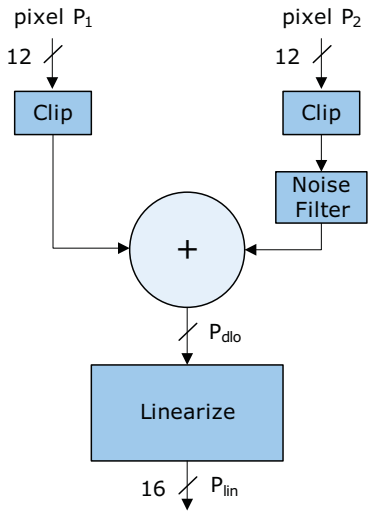


Digital Lateral Overflow (DLO)

DLO is an alternative to motion correction when motion is too fast, causing excessive artifacts in the HDR image. Color artifacts may result if a given pixel is bright, but not saturated in the T1 image, but surrounding pixels are replaced by T2 values. Instead of replacing pixels based on degree of motion, DLO will clip pixel values depending on a

user-defined threshold. When DLO is enabled, motion detection and correction, blue halo correction, and motion smoothing filter are all disabled.

Figure 7: DLO Block Diagram



Enabling DLO

Digital Lateral Overflow may be enabled by setting the pixel_build_dlo register (R0x3190[13] = 1). Enabling DLO will automatically disable the motion correction block and all associated algorithms.

Configuring DLO

Several registers are provided to customize the DLO algorithm. T2 noise filtering may be enabled by setting noise_filter_dlo_en (R0x3190[14] = 1). This will help smooth T2 noise in the combined image by clipping T2 values that fall below the s1_dlo threshold, and scaling pixel values that fall within the transition region. A threshold for noise filtering may be set by the noise_dlo_dis_threshold (R0x31A2[11:0]) register. If the pixel from T2 exceeds the value set in this register, the noise filtering is disabled. The s**2_dlo_threshold** (R0x319E[11:0]) and s12_dlo_range (R0x319E[15:12]) are set in a similar manner as the analogous s2_threshold and s12_range registers that were described in “Linearization” on page 14. The values at which the algorithm clips the T1 and T2 pixel values is defined by the T1_barrier (R0x3194[11:0]) and T2_barrier registers (R0x3196[11:0].

Setting the T1_barrier too high pushes the T1 exposure into the saturation region and may result in color artifacts. Setting T1_barrier too low increases T2 influence on the overall image and may introduce noise. The T1_barrier value may be set similar to S1_DLO. Setting the T2_barrier too high has minimal effect, but setting it too low will clip too much of the T2 exposure.

Disabling DLO

If DLO is disabled by resetting pixel_build_dlo (R0x3190[13] = 0), T1 and T2 pixel data is then linearized by the motion compensation block if motion compensation is enabled (R0x318C[14] = 1), or by the linearization block if motion compensation is disabled (R0x318C[14] = 0).

Data Selection and Interleaved Mode

The AR0331 sensor is capable of outputting the combined image, only T1 data, only T2 data, or interleaved T1 and T2 data. This selection is made in register R0x318E (see Table 3). When interleave mode is selected, alternating pixels from the T1 and T2 exposures are output. As the pixel data per row is doubled in interleave mode, the line_length_pck must be doubled to output a full row of image data.

Table 3: Motion Correction, Linearization, and DLO Registers

Register/Field Name	Address	Default Value	Description
HDR_MC_CTRL1	0x318A	0x0E10	s2_threshold Threshold level for the end point of the weighting transfer function. Pixel values above this level are chosen from T2 exposure only.
HDR_MC_CTRL2	0x318C[15]	0x0001	Smoothing filter Enable 0: Three-tab smoothing filter is disabled. 1: Smoothing filter is enabled.
	0x318C[14]	0x0001	Motion Correction Enable 0: Motion correction is disabled. 1: Motion correction is enabled.
	0x318C[2]	0x0000	all_motion_functions_force_on 1: All motion-related functions are forced on.
	0x318C[0]	0x0001	blue_halo_enable 0: Blue halo algorithm is disabled. 1: Blue halo algorithm is enabled.
HDR_MC_CTRL3	0x318E[15:14]	0x0000	bypass_pix_comb_cb Data select for context B. 0: HDR data. 1: T1 data. 2: T2 data. 3: Interleave mode(T1, T2, ...).
	0x318E[13:12]	0x0000	bypass_pix_comb Data select for context A. 0: HDR data. 1: T1 data. 2: T2 data. 3: Interleave mode (T1, T2, ...).
HDR_MC_CTRL4	0x3190[14]	0x0000	noise_filter_dlo_en Enables noise filtering of small pixel values in the digital lateral overflow pixel combination.
	0x3190[13]	0x0000	pixel_build_dlo 0: Digital Lateral Overflow Disabled 1: Use the digital lateral overflow method for combining t1and t2 data. This also overrides R0x318C[14], MOTION_CORRECT_EN which gets disabled.
HDR_MC_CTRL5	0x3192[12:0]	0x0400	s12_range Range for the HDR smooth combination weighting transfer function defined by S2-S1.
HDR_MC_CTRL6	0x3194[11:0]	0x0BB8	t1_barrier Barrier for clipping T1 data in the digital lateral overflow combination method.
HDR_MC_CTRL7	0x3196[11:0]	0x0DAC	t2_barrier Barrier for clipping T2 data in the digital lateral overflow combination method.

Table 3: Motion Correction, Linearization, and DLO Registers (continued)

Register/Field Name	Address	Default Value	Description
HDR_MC_CTRL8	0x3198[14:8]	0x0006	motion_detect_q2 Range for the motion detection algorithm.
	0x3198[7:0]	0x001E	motion_detect_q1 Lower threshold for the motion detection algorithm.
HDR_MC_CTRL9	0x319E[15:12]	0x0005	s12_dlo_range Range of code values for the noise filter weighting transfer function for digital lateral overflow defined by s2_dlo - s1_dlo. 4'b0000 = 1 4'b0001 = 2 4'b0010 = 4 4'b0011 = 8 4'b0100 = 16 4'b0101 = 32 4'b0110 = 64 4'b0111 = 128 4'b1000 = 256 4'b1001 = 512 4'b1010 = 1024 4'b1011 = 2048 4'b1100 = 4096 >= 4'b1101 = 8192 Setting the range to 8192 effectively sets s1_dlo to -4095 and s2_dlo to 4095.
	0x319E[11:0]	0x0040	s2_dlo_threshold Threshold level for end point of noise filter weighting transfer function for digital lateral overflow.
HDR_MC_CTRL11	0x31A2[11:0]	0x0BB8	noise_dlo_dis_threshold For the digital lateral overflow method, if either T1 data or T2 data is greater than this threshold, noise filtering is turned off. Evaluated on a single pixel.

Adaptive Color Difference (AdaCD) Noise Filtering

A good noise reduction filter will remove noise from an image while retaining as much image detail as possible. To retain image detail, the noise reduction filter must adapt to the image signal. To remove noise, the noise reduction filter must adapt to the noise level of the image signal. The key is to remove the appropriate amount of noise. Over-filtering will cause image blurring while under-filtering will leave noise in the image. The AdaCD algorithm relies on a noise model derived from characterization data to aid in separating noise from signal.

Description

The AdaCD algorithm performs pixel-by-pixel color noise correction for each of the red, blue, and green color planes. Each pixel is corrected based on surrounding pixel values on the same color plane and a noise model. The noise model is based on characterization data, and takes into account applied analog gain.

AdaCD Controls

The AdaCD algorithm may be enabled by setting bit R0x3200[1] = 1. Under low light conditions, bit R0x3200[0] may be set to further improve filtering in noisy situations. Enabling the low light AdaCD setting causes the filter to sample over a larger number of columns. This results in a smoother image under noisy conditions. Filtering strength may be adjusted using R0x3202. Figure 4 lists the registers, defaults, and descriptions for AdaCD controls.

AdaCD Tuning

In HDR mode, AdaCD operates on the T2 (short) exposure. It is recommended that AdaCD be enabled in HDR mode as T2 filtering will improve the combined image. AdaCD may also be used in linear mode, but the effect is more subtle. Increasing the filter strength (R0x3202) will decrease noise, but may also result in image blurring. Decreasing the filter strength will result in more image detail, but at the cost of higher noise. It is recommended to start with R0x3202 = 0x00A0 as a reference, and increase and/or decrease the value to find the optimal trade-off between image detail and noise. Under normal lighting conditions (R0x3200[0] = 0), it is recommended to set R0x3202 = 0x0064. If the low-light mode is enabled (R0x3200[0] = 1), setting R0x3202 = 0x00DC would be a good starting point. AdaCD filtering operates on a small region of pixels. At pixel locations at the edge of the array, the sensor runs out of pixels as input to the algorithm. As a result, increased noise may be observed on the edges of the field of view.

Table 4: AdaCD Registers

Register/Field Name	Address	Default Value	Description
adacd_control[1:0]	0x3200[1]	0x0001	adacd_filter_en 0: AdaCD algorithm is disabled. 1: AdaCD algorithm is enabled.
	0x3200[0]	0x0000	Low light 0: standard AdaCD algorithm. 1: AdaCD algorithm is adapted for low light.
adacd_noise_model1	0x3202[9:0]	0x00A0	Adjusts AdaCD filter strength

Auto Exposure Embedded Statistics Generation

Any auto exposure (AE) algorithm, be it implemented as part of the sensor device or in an off-chip companion device, will require statistical data from the image as input. The statistical data generated by the AR0331 sensor is described below for implementation of off-chip auto exposure algorithms.

Enabling Embedded Statistics

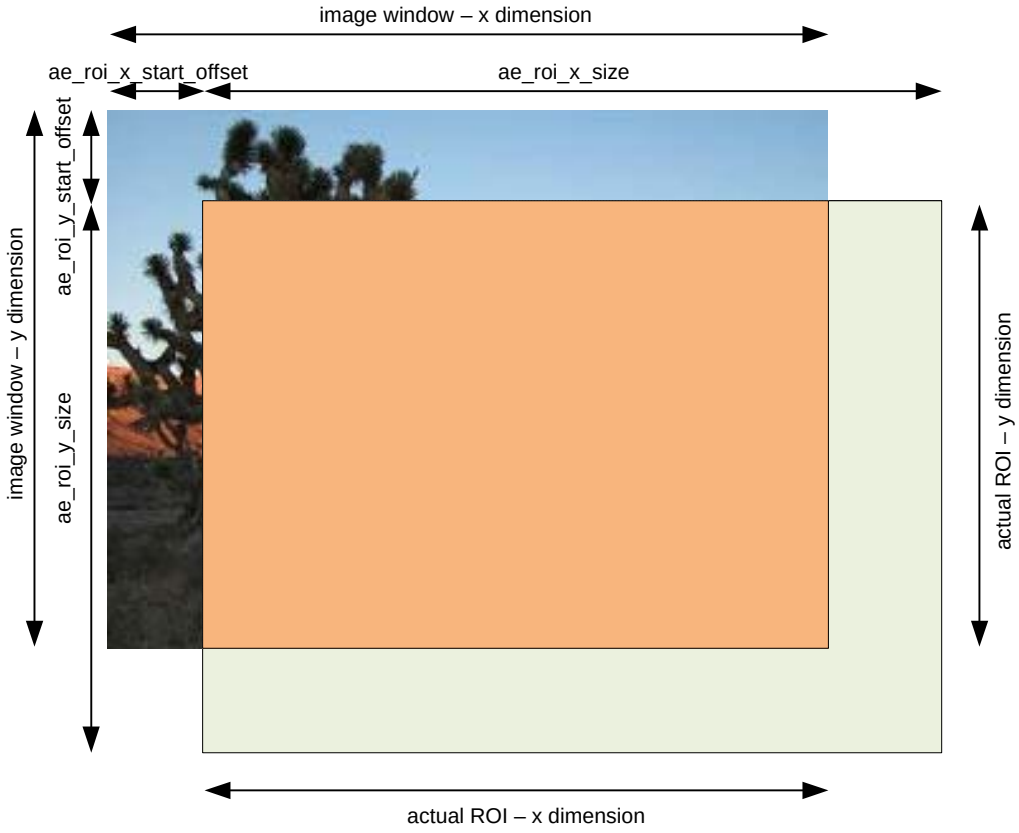
The generation of statistics for use by off-chip AE algorithms must be enabled by setting register R0x3064[7] = 1. Embedded statistics will not be output if this register is not set. Embedded data may also be enabled by setting register R0x3064[8] = 1, but is not necessary for statistics generation.

Data Used for Statistics Generation

Region of Interest for Statistics Gathering

The histogram is generated and statistics calculated based on the Gr pixels within a user-specified region of interest. The ROI is specified by four programmable register values: ae_roi_x_start_offset, ae_roi_y_start_offset, ae_roi_x_size, and ae_roi_y_size. The ae_roi_x_start_offset and ae_roi_y_start_offset values define the starting coordinate of the ROI with respect to the image window that is output, and the ae_roi_x_size and ae_roi_y_size values define the dimensions of the ROI. Each value must be an even number. If the requested ROI extends 'beyond' the image window then it will be restricted in size such that the final pixel of the ROI will be the final pixel of the image window, as illustrated in the example in Figure 8: “Example of Statistics ROI Definition,” on page 21.

Figure 8: Example of Statistics ROI Definition



Histogram

Data extracted from a specified region of interest (ROI) of an image frame is accumulated in a histogram. All statistics will be based upon the Gr pixels – that is, the green pixels on even rows (Gr, Red, Gr, Red, …) – for both monochrome and color sensors.

The AR0331 HDR histogram consists of 244 bins where:

- bins 0-63 cover the data range 0-(2^8-1), a range of 4 LSBs per bin
- bins 64-183 cover the data range 2^8 -(2^{12}-1), a range of 32 LSBs per bin
- bins 184-243 cover the data range 2^{12}-(2^{16}-1), a range of 1024 LSBs per bin

The AR0331 12-bit linear mode histogram is MSB aligned with the 16-bit HDR mode histogram:

- bins 0-63 cover the data range 0 - (2^4 - 1), a range of 1 LSB per bin (where 3 out of 4 bins are unused)
- bins 64 - 183 cover the data range 2^4 - (2^8 - 1), a range of 2 LSBs per bin
- bins 184 - 243 cover the data range 2^8 - (2^{12} - 1), a range of 64 LSBs per bin

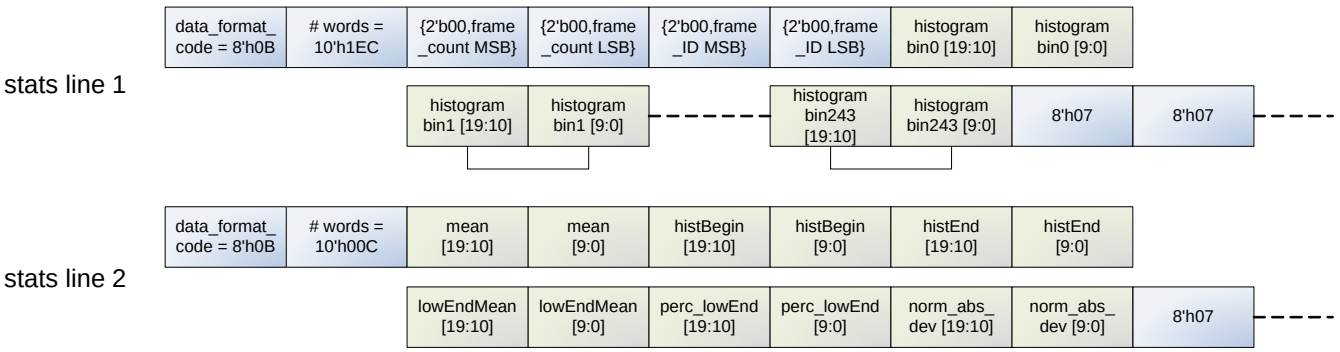
and each bin contains a count of the number of Gr pixels in the ROI that fall in to the range covered by that bin. The raw histogram data (that is, the count value of each bin) is embedded in the statistics rows (the two rows immediately after the last image row).

Statistics Output

Format of Embedded Statistics Rows

All the statistics data (including histogram data) is embedded in the two rows immediately following the image. The embedded statistics are output as shown in Figure 9. The first line contains histogram data. The second line contains statistics for backwards compatibility. It is recommended that auto exposure algorithms be developed based on the histogram data found in line 1. Since the AR0331 supports up to 16-bit pixel data, higher order bits for the output statistics and register data will be set to zero.

Figure 9: Embedding of Statistics Data



AE Guidelines

When controlling the exposure, in most cases it is best to adjust integration time first, followed by analog gain, and finally digital gain. There is a default integration step size of 4 rows times the exposure ratio. For a 16x ratio, the integration time cannot be changed by more than 64 rows per frame. Setting no_sh_jump_limit (R0x30B0[1] = 1 enables larger jumps. To maintain proper timing within a frame, the maximum jump in integration time is limited by the number of idle rows (vertical blanking). The number of idle rows should be equal to the number of rows in the target T2 exposure. For example, for a 16x exposure ratio, a T1 jump from 1 to 320 rows would require 320/16 = 20 extra rows of vertical blanking. A jump from 300 to 320 rows would also require 20 extra rows.

Sample AE Algorithm

Introduction

Auto exposure (AE) consists of 3 components: 1) AE algorithm which determines the right amount of exposure for the scene; 2) AE tracking which takes care of the dynamics of the exposure to make sure the exposure moves smoothly; 3) exposure setting which finds the optimal combination of integration time, analog gain and digital gain considering the trade-off among frame rate, image sharpness, and noise.

The following guidelines are for an HDR AE algorithm as it differs from a linear mode algorithm. Guidelines for AE tracking and exposure settings are not provided since there is no difference between HDR and linear modes for those two components.

HDR AE Algorithm

The HDR AE algorithm is used to align an average luminance value (WeightedAvgY) to a target luminance (AvgYTarget) while limiting the amount of highlight pixel clipping within a specified tolerance. There are two exposures to estimate: 1) Exposure to control the average brightness to reach a target value (AvgY_Exp); 2) Exposure to control the highlight clipping (HistAvoidClipping_Exp).

Average Luminance Exposure (AvgY_Exp)

The exposure value AvgY_Exp is computed as:

$$Avg\_Exp = \log_2(AvgYTarget / WeightedAvgY) \quad \text{(EQ 2)}$$

Here, AvgYTarget is a programmable value. It can be an output from a lookup table based on the estimated scene brightness value. WeightedAvgY is the average luminance value. It can be calculated using the histogram which is output from the AR0331 in the first line of the embedded statistics.

Assume the bin number of the embedded histogram is num_bin. The boundary of the bin is represented by his_boundary(i), and the pixel number within each bin is represented by his(i). WeightedAvgY is calculated by the following formula:

$$WeightedAvgY = \sum_{i=1}^{num\_bin} (his\_boundary(i) \cdot his(i)) / M \quad \text{(EQ 3)}$$

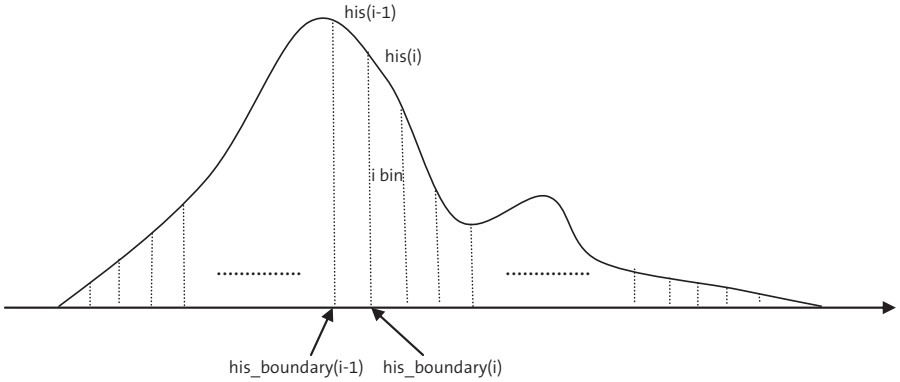
Where M is the total number of pixels.

In general, using a $\log_2$ average can provide better performance. The calculation for WeightedAvgY then becomes:

$$WeightedlogAvgY = \sum_{i=1}^{num\_bin} (his\_boundary(i) \cdot \log_2(his(i))) / M \quad \text{(EQ 4)}$$

$$WeightedAvgY = 2^{WeightedlogAvgY} \quad \text{(EQ 5)}$$

Figure 10: Histogram



Exposure to Avoid High Light Clipping (HistAvoidClipping_Exp)

Some scenes will have excessive clipping in highlight regions. To ensure that highlight clipping is within a tolerance, the AE algorithm tries to place a histogram high end percentile point (HistHighPercPoint) below a programmable target value (HistHighTarget). Assume the HistHighPercPoint is calculated from the histogram for the programmable percentile parameter, HistHighPerc. The exposure value (HisHighPerc_Exp) for matching HistHighPercPoint to HistHighTarget is:

$$HistHighPerc\_Exp = \log_2\left(\frac{HistHighTarget}{HistHighPercPoint}\right) \quad \text{(EQ 6)}$$

When there are a lot of highlight clipping pixels, the value of histogram high percentile point is calculated from a clipped histogram and the resulting value is not reliable.

In order for the AE to settle quickly, a formula to decrease exposure to eliminate highlight clipping based on the amount of clipping can be derived from the image database. Assume the clipping percentage is denoted as HighClipPerc. The exposure value (HighClip_Exp) needed to bring the clipping percentage down to 0 is calculated as follows:

$$HighClip\_Exp = -k \cdot HighClipPerc \quad \text{(EQ 7)}$$

Here, k is selected empirically.

The exposure (HistAvoidClipping_Exp) to avoid highlight clipping is determined based on the percentage of highlight clipping pixels. For a high percentage of clipping, the minimum of HighClip_Exp and HistAvoidClipping_Exp is selected. Otherwise, HistHighPerc_Exp is selected. The pseudo code is:

```
if (HighClipPerc > 2*(1.0 - HistHighPerc)
  HistAvoidClipping_Exp = min (HighClip_Exp, HistHighPerc_Exp);
else
  HistAvoidClipping_Exp = HistHighPerc_Exp;
```

Final Exposure (Exp_Final)

The final exposure value is based on the following rules:

1. If AvgY_Exp < HistAvoidClipping_Exp, then use AvgY_Exp as the final exposure. This exposure is for low dynamic range scenes to have the average luminance reach the luminance target while keeping highlight clipping under control.
2. If AvgY_Exp ≥ HistAvoidClipping_Exp, and the difference between those two exposures is smaller than a programmable maximum exposure adjustment value, MaxHistHighExpAdjust, then use HistAvoidClipping_Exp as the final exposure. This is for mid dynamic range scenes where the average luminance needs to be below the average luminance target to keep the histogram high percentile point from reaching the histogram high percentile target, and to keep clipping under control.
3. Otherwise, use AvgY_Exp - MaxHistHighExpAdjust as the final exposure. This is for high dynamic range scenes where the reduction of average luminance value is limited to prevent darker regions of the scene from underexposure and clipping, while allowing some clipping at the brighter regions.

The above rules can be expressed in the form of pseudo code as follows:

```
if (AvgY_Exp < HistAvoidClipping_Exp)
  Exp_Final = AvgY_Exp;
else
{
  if ((AvgY_Exp - HistAvoidClipping_Exp) < MaxHistHighExpAdjust)
  Exp_Final = HistAvoidClipping_Exp;
  else
  Exp_Final = AvgY_Exp - MaxHistHighExpAdjust;
}
```

Adding Digital Gain for Optimal SNR Output

In HDR mode, when a pixel value from the long exposure is saturated, the value from the short exposure will be used. Depending on the exposure ratio, pixels from the short exposure could be much noisier than those from the longer exposure. Pixels from the shorter T2 exposure in Figure 12 (left) are noisier than pixels from the longer T1 exposure in Figure 12 (right). (Figure 11 shows the area of interest.) To maximize SNR, the number of pixels taken only from the short exposure should be limited.

Figure 13 shows a histogram of image luminance values. The T1 saturation point is the saturation value for the long (T1) exposure. Pixel values greater than the T1 saturation point are taken from the short (T2) exposure. If the exposure is reduced such that the histogram valley point is below the T1 saturation point, those pixels with values between

T1 saturation point and the valley point are now taken from the long exposure (Figure 14). Noise in the pixels moved from the T2 exposure will be significantly reduced. Since HDR pixel combination takes place after the analog gain stage and before digital gain, the exposure reduction in Figure 14 can be compensated by applying more digital gain. After applying digital gain, the histogram will look the same as in Figure 13 and the overall image brightness will remain the same.

Figure 11: Image Showing Area of Interest

Figure 12: Image Without Digital Gain Applied (left) — Image With Digital Gain Applied (right)

Figure 13: Image Luminance Histogram Before Reducing Analog Gain

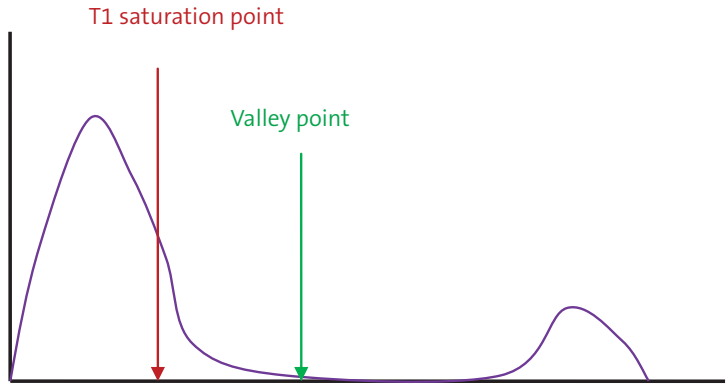
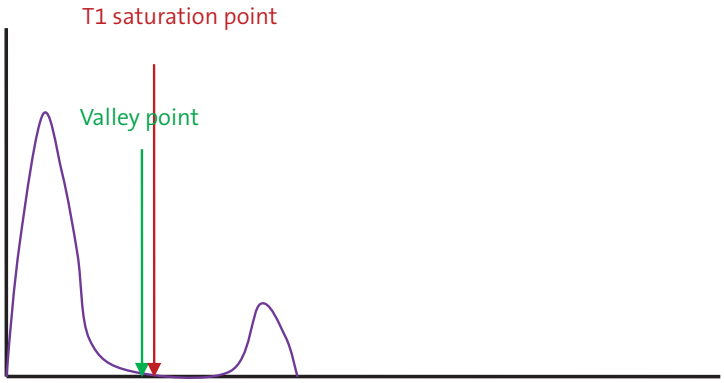


Figure 14: Image Luminance Histogram After Reducing Analog Gain



In order to have pixels with values below the histogram valley point use the long exposure, the applied digital gain (MinDigitalGain) has to be greater than the ratio of the histogram valley point (HistValleyPoint) over a programmable target value (T1PixelTarget) and less than pre-selected limited gain (MaxDigitalGain). T1PixelTarget should be less than or equal to the T1 saturation point. T1PixelTarget is the middle point of the T1 and T2 pixel smoothing range. It is set to (S2_threshold - S12_range/2) where S2_threshold (R0x318A) is the threshold for T1 pixel saturation point and S12_range (R0x3192) is the range of pixel values where T1 and T2 pixel values are combined.

$$MinDigitalGain = min(HistValleyPoint/(T1PixelTarget, MaxDigitalGain)) \quad \text{(EQ 8)}$$

The minimum digital gain is used to trade analog gain for digital gain. The noise degradation from trading analog gain for digital gain is small compared to the noise improvement obtained from moving pixels from the shorter T2 exposure to the longer T1 exposure. In principle, the minimum digital gain could also be applied to trade integration time for digital gain, but this is not recommended as noise degradation could be significant. Finally, it needs to be mentioned that applying digital gain may reduce the dynamic range of the captured image.

Table 5: Auto Exposure Registers

Register/Field Name	Address	Description
ae_roi_x_start_offset [10:0]	0x3140	Starting column of region of interest with respect to the starting column of the image window (b0 will be ignored/treated as 0, since the start column must always be an even column). Note: If statistics are being gathered from a scaled image, then the 'number of pixels' value must be the number of scaled pixels. Default = 0.
ae_roi_y_start_offset [9:0]	0x3142	Starting row of region of interest with respect to the starting row of the image window (b0 will be ignored/treated as 0, since the start row must always be an even row) - start row is with respect to the frame before any scaling. Default = 0.
ae_roi_x_size [10:0]	0x3144	Width (in columns) of ROI (b0 will be ignored/treated as 0, since the number of columns must always be even). Default = 0x804 (2052).
ae_roi_y_size [9:0]	0x3146	Height (in rows) of ROI (b0 will be ignored/treated as 0, since the number of rows must always be even). Default = 0x614 (1556).
ae_hist_begin_perc[15:0]	0x3148	Defines the percentage of Gr pixels that must have values below hist_begin. Specified as a number <1 = 0.xxxx...xxxx. Default = 0.
ae_hist_end_perc[15:0]	0x314A	Defines the percentage of Gr pixels that must have values below hist_end. Specified as a number <1 = 0.xxxx...xxxx. Default = 0xFFFF (100% - all 1's value may be expected to equal as 65535/65536, but is treated as a special case).
ae_hist_div[15:0]	0x314C	Defines the point at which the histogram is divided into low and high end. Specifies the boundary as hist_div*16. Default = 0x0100 (making the code value for the boundary between the histogram low and high end = 256*16 = 4096).
ae_norm_width_min[15:0]	0x314E	Defines the minimum histogram width normalization factor (norm_width_min*16), for norm_abs_dev calculation. A value of all 1s turns off the norm_width_min option, i.e., the absolute deviation is normalized by histEnd-histBegin. Default = 0x0020 (making the normalization factor = 32*16 = 512).
ae_mean_h[19:16]	0x3150	The 4 MSBs of the 'true mean' of all Gr pixel values in the ROI. Read only.
ae_mean_l[15:0]	0x3152	The 16LSBs of the 'true mean' of all Gr pixel values in the ROI. Read only.
ae_hist_begin_h[19:16]	0x3154	The 4 MSBs of the code value corresponding to the histogram bin below which hist_begin_perc of pixels exist. Read only.
ae_hist_begin_l[15:0]	0x3156	The 16 LSBs of the code value corresponding to the histogram bin below which hist_begin_perc of pixels exist. Read only.
ae_hist_end_h[19:16]	0x3158	The 4 MSBs of the code value corresponding to the histogram bin below which hist_end_perc of pixels exist. Read only.
ae_hist_end_l[15:0]	0x315A	The 16 LSBs of the code value corresponding to the histogram bin below which hist_end_perc of pixels exist. Read only.

Table 5: Auto Exposure Registers (continued)

Register/Field Name	Address	Description
ae_hist_end_mean_h[19:16]	0x315C	The 4 MSBs of the true mean of all Gr pixel value that fall into the low end of the histogram (as specified by hist_div). Read only.
ae_hist_end_mean_l[15:0]	0x315E	The 16 LSBs of the true mean of all Gr pixel value that fall into the low end of the histogram (as specified by hist_div). Read only.
ae_perc_low_end[15:0]	0x3160	Percentage of Gr pixels in ROI that fall into the low end of the histogram. Specified as a number <1 = 0.xxxx...xxxx. Read only.
ae_norm_abs_dev[15:0]	0x3162	The normalized absolute deviation of Gr pixel data in the ROI. Specified as a number <1 = 0.xxxx...xxxx. Read only.

Combi Mode

The AR0331 sensor ships with two sequencers to control internal timing, one optimized for linear mode and a second one optimized for HDR mode. Switching from true linear mode to HDR mode (and vice versa) will require the appropriate sequencer be loaded for optimal performance. This will result in one bad frame being output during the transition.

To avoid losing one frame, the AR0331 sensor can operate in combi mode. In this mode, the HDR sequencer is loaded for both linear and HDR modes. The user can then switch seamlessly between linear and HDR modes through a register write.

Enabling Combi Mode

To use combi mode, the sensor should be first configured as HDR mode using the HDR sequencer (parallel or HiSPi). Combi mode is enabled by setting bit 8 of the digital_ctrl register (0x30BA[8] = 1), and is enabled by default when selecting an HDR mode from the AR0331 initialization file.

Switching Between Linear and HDR in Combi Mode

To switch between linear and HDR mode when combi mode is enabled, set R0x3082[1:0] = 0x0 for HDR mode, or set R0x3082[1:0] = 0x1 for linear mode. Because the pixel data is MSB-aligned, there may be situations in which digital gain must be applied to maintain the same intensity between linear and HDR images. For example, a low-light HDR scene will have the majority of the pixel data in the low-order bits of a 16-bit word. A linear image will also have most pixel data in the low-order bits, but because the output data is 12-bit MSB-aligned, the linear mode image will have a higher intensity level. To maintain a constant intensity level, digital gain will need to be applied to the HDR image to shift the pixel data to the equivalent of the linear mode image. For example, switching from a 16-bit HDR image with a 16x exposure ratio to a linear mode image will require applying 16x digital gain to the HDR image to maintain the same intensity level as the linear mode image with 1x digital gain.

Difference Between Combi-Linear and Pure Linear Modes

HDR images are the same whether or not combi mode is enabled. If linear (combi linear) mode is enabled (with combi mode bit set), the HDR sequencer is used, and the T1 image is output. Images in combi linear mode may have slight differences in dark shading, row noise, and column FPN when compared to the pure linear mode (i.e., using the linear mode sequencer). These differences are due to the fact that pure linear mode uses a sequencer optimized for linear mode.

Register Settings

Table 6: Combi-Mode Registers

Register/Field Name	Address	Default Value	Description
digital_ctrl	0x30BA[8]	0x1	combi_mode 0: combi_mode disabled. 1: combi_mode enabled. Operation mode can be switched between HDR and Linear mode with no bad frames.

Table 6: Combi-Mode Registers

Register/Field Name	Address	Default Value	Description
operation_mode_ctrl	0x3082[1:0]	0x0	operation_mode 00: HDR mode. 01: Linear mode.

Global Reset Release (GRR) Mode

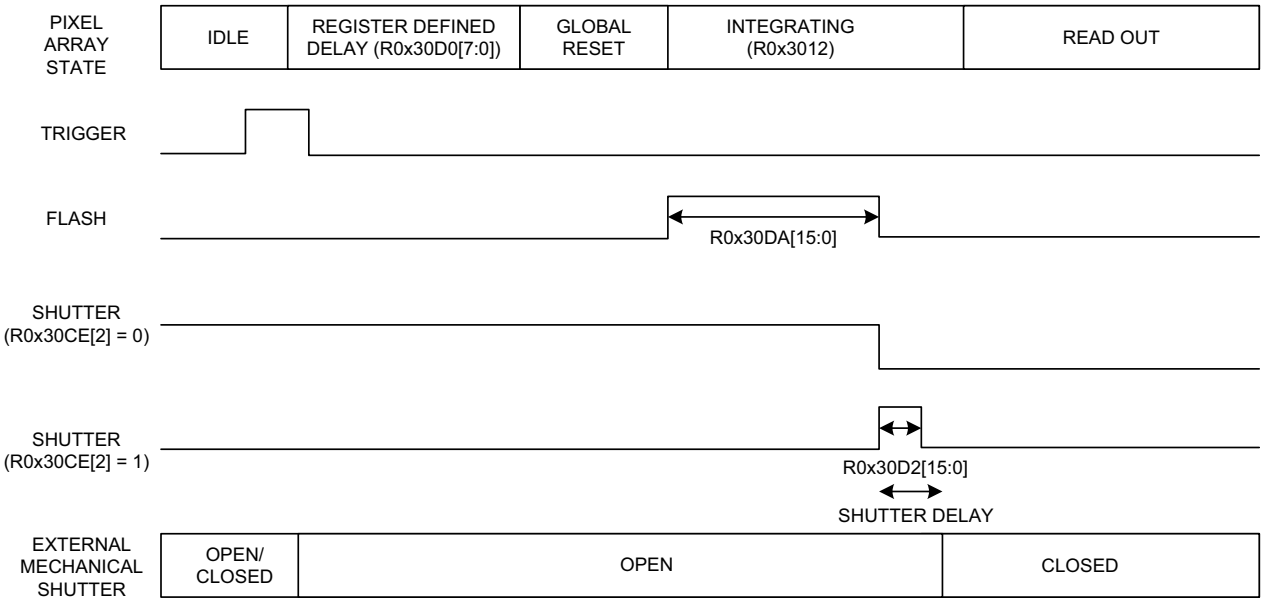
AR0331 provides a Global Reset Release (GRR) mode to accommodate external shutter control. GRR mode is supported only for the linear mode of operation.

The sequence of operation is:

1. GRR mode is started with the sensor in standby. The PLL may be disabled, but due to the PLL startup and lock times, it is recommended that the PLL continue to run. The serial links and analog array are in a low power mode, and no read out or shutter is in progress.
2. The sensor is put in GRR mode by a register write (R0x30CE = 0x1). The sensor logic will assert the signal SHUTTER depending on configuration (See Table 1: GRR Mode Registers). It is required that the coarse integration (R0x3012) time be set to a value greater than the desired exposure time + mechanical shutter closure time.
3. A pulse (minimum 3 external clock periods) is applied to the pad TRIGGER, or streaming mode is enabled by register write (0x301A[2] = 0x1).
4. The sensor transitions to streaming mode and enables the analogue circuitry.
5. A global reset operation is performed after a user defined delay (R0x30D0[7:0]) following the trigger pulse. After completion of the global reset operation, FLASH is asserted and integration begins.
6. FLASH and SHUTTER signals will be de-asserted depending on register settings.
7. The sensor will wait until the integration time has finished and then start frame readout.
8. If the GRR mode was initiated by a TRIGGER pulse, the sensor returns to standby.

GRR Mode

Figure 15: GRR Timing Diagram



Register Writes to Do One GRR Capture

1. Sensor is in standby and R0x30CE = 0x0000.
2. Set bit R0x301A[11] to force the PLL to run in standby mode.
3. Set R0x30DA to the desired exposure time in number of rows.
4. Set integration time (R0x3012) to desired exposure time plus mechanical shutter delay to close (could be few msec).
5. Increase the frame length lines (R0x300A) by the value (R0x30DA + 1).
6. Enable flash (R0x3046) if desired.
7. Enable GPI pins by R0x301A[8] = 1 (if the frame will be generated via TRIGGER pulse).
8. Enter GRR mode by setting R0x30CE[0] = 1. The output SHUTTER will go high (default setting R0x30CE[2] = 0).
9. Wait for the mechanical shutter to open.
10. Apply a TRIGGER pulse or set 0x301A[2] = 1 to enable streaming.
11. The FLASH pin will be asserted for the exposure time set in register R0x30DA. Depending on register setting, the SHUTTER pin will deasserted once the desired exposure is complete, or SHUTTER will be pulsed with the specified pulse width (R0x30D2[15:0]) to signal closure of the mechanical shutter.
12. Frame readout will begin.
13. If the operation was started by a register write, return to standby mode (R0x301A[2] = 0) before frame_valid is deasserted.
14. When frame valid is low (sensor is in standby), write R0x30CE = 0x0000. This resets the GRR logic.
15. Repeat for subsequent frame captures.

Table 7: GRR Mode Registers

Register/Field Name	Register Address	Description
grr_mode	0x30CE[0]	0: Normal ERS mode 1: Enable GRR mode
ext_shut_pulse	0x30CE[2]	0: SHUTTER goes high when GRR mode enabled and transitions low after integration is complete 1: SHUTTER is pulsed after integration is complete with pulse width specified in R0x30D2[15:0]
shutter_disable	0x30CE[6]	Disable mechanical shutter control; output pad SHUTTER is high impedance.
shutter_always_open	0x30CE[7]	Always open mechanical shutter; output pad SHUTTER is high.
gr_delay	0x30D0[7:0]	Number of rows to wait before starting global reset.
ext_shut_pulse_width[15:0]	0x30D2[15:0]	Width of the shutter pulses (when ext_shut_pulse = 1), in clock cycles
ext_shut_delay[15:0]	0x30DA[15:0]	Delay before the shutter is closed, in number of rows

Subsampling

Introduction

The AR0331 supports subsampling. Subsampling allows the sensor to read out a smaller set of active pixels by either skipping or binning pixels within the readout window. The AR0331 supports several skip factors, as well as various binning and summing modes.

Enabling Subsampling

To enable subsampling, a skip factor in the x and y dimensions must be specified. This is done by setting x_odd_inc (R0x30A2) for the x-dimension, and y_odd_inc (R0x30A6) for the y-dimension. Legal values are 0x01, 0x03, and 0x05 for no skip, skip 2, and skip 3. No other values are supported.

For subsampling modes, the window size should adhere to the following:

Horizontal:

$$\frac{(x\_addr\_end - x\_addr\_start + 1)}{(x\_odd\_inc + 1)/2} = \text{an even number} \quad \text{(EQ 9)}$$

Vertical:

$$\frac{(y\_addr\_end - y\_addr\_start + 1)}{(y\_odd\_inc + 1)/2} = \text{an even number} \quad \text{(EQ 10)}$$

Color subsampling is the default. This will perform subsampling based on color plane. Monochrome subsampling may be enabled by setting mono_chrome_operation (R0x30B0[7] = 1) and is only supported with x_odd_inc = y_odd_inc = 0x3.

Subsampling Modes

Row Subsampling

In color subsampling modes, pixels of the same color will be binned in the analog domain. Because of this, the binning is done on the column. When this is done, multiple source followers may be enabled at once. If the signal level of one pixel is significantly higher than the others being binned, the source follower of the lower signal level pixels may be turned off. Because of this, a true summing operation may not be possible, and the SNR benefit from binning may not be achieved.

In monochrome subsampling modes, adjacent pixels are binned on the FD (floating diffusion) node. This allows for a true summing operation, and will allow for a boost in SNR.

Row binning is enabled by setting read_mode_row_bin (R0x3040[12] = 1). Skipping is enabled by setting a skip factor in x_odd_inc, but leaving read_mode_row_bin low (R0x3040[12] = 0x0).

Column Subsampling

By setting the appropriate bit, averaging or summing may be done in the digital domain on the columns. To enable averaging, set R0x3040[13] = 1. To enable summing, set both R0x3040[13] = 1 and R0x3040[5] = 1. Similar to row subsampling, column skipping is enabled by setting a skip factor in y_odd_inc, but leaving the summing (0x3040[5]) and binning (0x3040[13]) bits deselected.

Table 8: Subsampling Registers

Register/Field Name	Address	Default Value	Description
read_mode	0x3040[13]	0x0	read_mode_col_bin 0: Normal operation. 1: Pixels averaged in digital domain for context A.
	0x3040[12]	0x0	read_mode_row_bin 0: Normal operation. 1: Analog row binning for context A.
	0x3040[11]	0x0	read_mode_col_bin_cb 0: Normal operation. 1: Pixels averaged in digital domain for context B.
	0x3040[10]	0x0	read_mode_row_bin_cb 0: Normal operation. 1: Analog row binning for context B.
	0x3040[5]	0x0	read_mode_col_sum 0: Normal operation. 1: Pixel values are summed in digital domain.
x_odd_inc	0x30A2	0x1	Specifies x-dimension skip factor for subsampling. Legal values: [1, 3, 5].

Table 8: Subsampling Registers

Register/Field Name	Address	Default Value	Description
y_odd_inc	0x30A6	0x1	Specifies y-dimension skip factor for subsampling. Legal values: [1, 3, 5].
mono_chrome_operation	0x30B0[7]	0x0	Monochrome operation 0: Normal operation. 1: Subsampling will be performed as if using a monochrome sensor.

Analog Shading Correction

Because of overlapping sampling and readout activities, digital switching noise may affect image quality. The impact of switching noise is largely dependent on the timing relationship between sampling and readout. To counteract the effect of switching noise, the AR0331 sensor includes an analog shading correction feature. The analog shading correction circuit compensates for linear mode shading, as well as T1 shading in HDR mode. Analog shading correction is not effective in combi-linear mode.

Enabling Analog Shading Correction

Analog shading correction may be enabled by setting ana_sreg_shading_en (R0x3ED2[15] = 1). Setting R0x3ED2[15] = 0 will disable the analog shading correction feature.

Tuning Analog Shading Correction

There are two registers available to tune the analog shading correction, ana_sreg_shading_on_shs (R0x3ED2[14]), and ana_sreg_shading_curr (R0x3ED2[13:9]). Setting R0x3ED2[14] = 1 will apply analog shading correction on shs (signal), while setting R0x3ED2[14] = 0 will apply analog shading correction on shr (reset). This effectively acts as a polarity switch. Setting R0x3ED2[13:9] adjusts the magnitude of the analog shading correction.

Shading across the field of view can be affected by many variables such as linear or HDR mode, line_length_pck, gain, and temperature. Because of this variation, it is recommended that settings be customized for the particular application and operating conditions. The on-chip temperature sensor may be used to adjust analog shading correction settings based on temperature readings.

Table 9: Analog Shading Correction Registers

Register/Field Name	Address	Default Value	Description
ana_sreg_shading_en	0x3ED2[15]	0x1	Analog shading correction enable 0: Analog shading correction disabled. 1: Analog shading correction enabled.
ana_sreg_shading_on_shs	0x3ED2[14]	0x0	Analog shading correction polarity 0: Analog shading correction applied on SHR. 1: Analog shading correction applied on SHS.
ana_sreg_shading_curr	0x3ED2[13:9]	0x1F	Analog shading correction current Legal values: [0, 31].

Black Level Correction

The AR0331 HDR sensor has built in controls for black level (delta dark) correction and calibration. By utilizing optically dark rows, the magnitude of dark current can be measured and used to compensate for its effect on black level in the visible array.

Enabling Black Level Correction

Black level correction is enabled by default, but may be manually enabled or disabled by writing to the delta_dk_sub_en (R0x3180[15]) register. Gradient removal may be enabled by setting delta_dk_gradient_removal (R0x3180[10]). This is useful in GRR (external mechanical shutter) mode when a top-to-bottom shading effect may be seen as the bottom rows have more exposure to dark current.

Controlling Black Level Correction

The number of dark rows may be set in delta_dk_rows (R0x3180[7:4]). The valid values are 2, 4, or 8 rows. The more rows, the more accurate the dark current compensation. The dark current is automatically recalculated on changes in integration time. It may be forced to recalculate every frame or by asserting a control register. To force dark current calculations every frame, set delta_dk_every_frame (R0x3180[14]) = 0x1. Similarly, gradient removal may be triggered every frame by setting delta_dk_gradient_every_frame (R0x3180[9]) = 0x1. Recalculating every frame is not recommended as it may result in oscillations. Setting delta_dk_recalc (R0x3180[13]) = 0x1 allows manual retriggering of the dark current calculations. This allows periodic recalculations from a watchdog timer on an ISP or FPGA. It also enables recalculation based on readings from the on chip temperature sensor. The delta_dk_recalc register will reset to zero after one calculation is performed.

Table 10: Black Level Control Registers

Register/Field Name	Address	Default Value	Description
delta_dk_control	0x3180[15]	0x0001	delta_dk_sub_en 0: Disables delta dark correction. 1: Enables delta dark correction.
	0x3180[14]	0x0000	delta_dk_every_frame 0: Dark current is not recalculated every frame. 1: Dark current is recalculated every frame.
	0x3180[13]	0x0000	delta_dark_recalc 0: Manual delta dark recalculation is disabled. 1: Force recalculation of dark current.
	0x3180[10]	0x0000	delta_dk_gradient_removal 0: Gradient removal is disabled. 1: Gradient removal is enabled.
	0x3180[9]	0x0000	delta_dk_gradient_every_frame 0: Gradient removal is not recalculated every frame. 1: Gradient removal is recalculated every frame.
	0x3180[7:4]	0x0008	delta_dk_rows Sets the number of rows for dark current calculations. Legal values: [0, 2, 4, 8].

Temperature Sensor

The AR0331 sensor has a built-in PTAT-based (Proportional To Absolute Temperature) temperature sensor, accessible through registers, that is capable of measuring die junction temperature. The temperature sensor can be enabled by writing R0x30B4[0]=1 and R0x30B4[4]=1. After this, the temperature sensor output value can be read from R0x30B2[9:0].

The value read out from the temperature sensor register is an ADC output value that needs to be converted downstream to a final temperature value in degrees Celsius. Since the PTAT device characteristic response is quite linear in the temperature range of operation required, a simple linear function as in Equation 11 can be used to convert the ADC output value to the final temperature in degrees Celsius.

$$Temperature = slope \times R0x30B2[9:0] + T0 \quad (EQ\ 11)$$

For this conversion, a minimum of 2 known points are needed to construct the line formula by identifying the slope and y-intercept “T0”. These calibration values can be read from registers R0x30C6 and R0x30C8 which correspond to values read at 70°C and 55°C respectively. Once read, the slope and y-intercept values can be calculated and used in the above equation.

Example: What is the temperature in degrees Celsius when R0x30B2 = 0x1A2 (418)?

For this particular sensor, the 70°C calibration data reads R0x30C6 = 0x01D2 (466), and the 50°C calibration data reads R0x30C8 = 0x01BD (445). From these values, the correct temperature reading can be found as follows:

$$slope = (70 - 55) / (466 - 445) = (15/21) = 0.714 \quad (EQ\ 12)$$

From here, the intercept T0 can be found:

$$55 = (0.714)\ x\ (445) + T0 \quad (EQ\ 13)$$

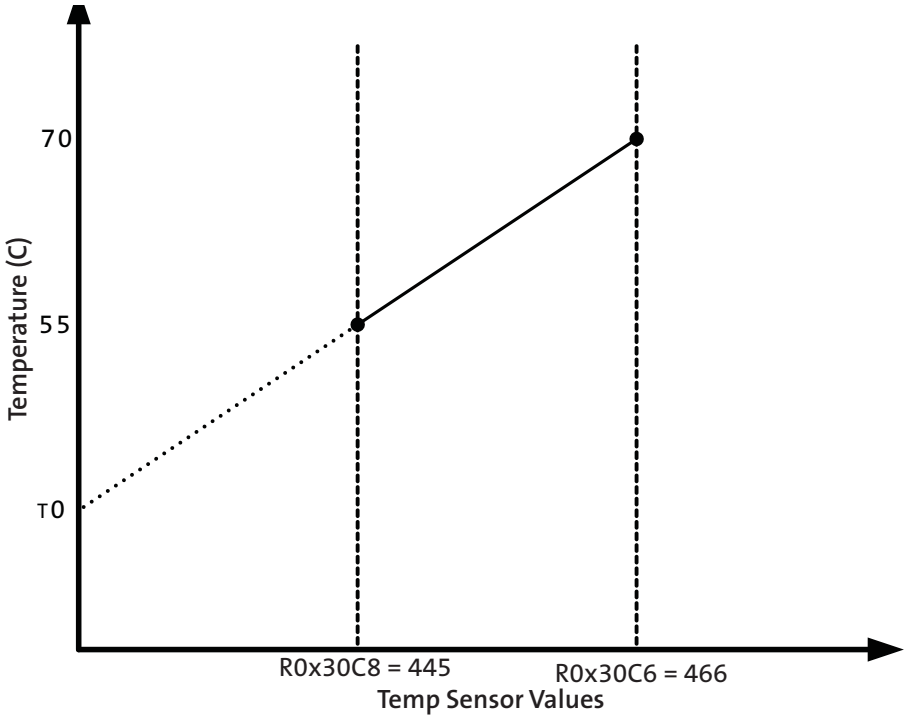
$$T0 = -262.73 \quad (EQ\ 14)$$

Now, the temperature corresponding to a register reading of 0x1A2 can be determined:

$$Temperature = (0.714)\ x\ (418) - 262.73 \quad (EQ\ 15)$$

$$Temperature = 35.7°C \quad (EQ\ 16)$$

Figure 16: Calculating Temperature Sensor Value

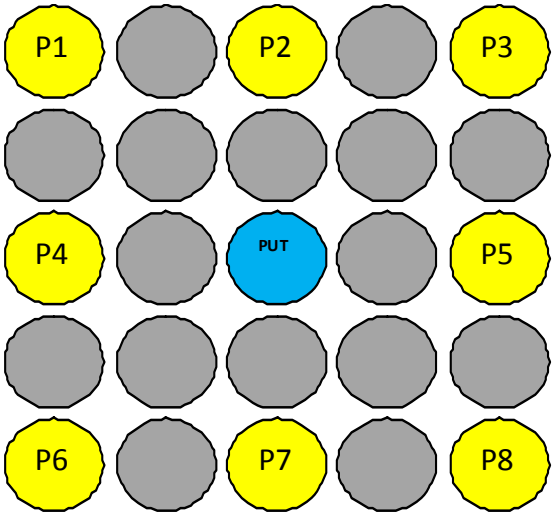


For more information on the temperature sensor registers, refer to the AR0331 Register Reference.

2d Single Pixel Defect Correction

The AR0331 is capable of correcting single pixel defects in two dimensions. 2d single pixel defect correction may be enabled by setting register 0x31E0[9] = 1. The default power up state is for defect correction to be disabled, but it is then enabled by most presets in the initialization file provided with DevWare.

Figure 17: Pixels Used by Defect Correction Algorithm



The on-the-fly single pixel defect correction is based on the maximum and minimum values of pixels P1 through P8 (see Figure 17) surrounding the pixel under test (PUT). The correction algorithm is implemented as follows:

1. For each PUT, form an 8-neighboring pixel set as shown in Figure #.
2. Sort {P1,P2,…P8}, and determine max and min pixel values
3. Detect and Correct:
   3a. If PUT>max, PUT is replaced by max
   3b. If PUT<min, PUT is replaced by min
   3c. Else, PUT is unchanged

The first two and last two rows/columns of the array are not corrected.

Revision History

Rev. B 3/25/13

- Updated “Using ALTM” on page 8
- Updated “Enabling ALTM” on page 8
- Updated “Configuring Motion Compensation” on page 13
- Added “Global Reset Release (GRR) Mode” on page 31, including Figure 15: “GRR Timing Diagram,” on page 32, and Table 7, “GRR Mode Registers,” on page 33
- Added “Temperature Sensor” on page 38, including Figure 16: “Calculating Temperature Sensor Value,” on page 39
- Added “2d Single Pixel Defect Correction” on page 39, including Figure 17: “Pixels Used by Defect Correction Algorithm,” on page 40

Rev. A 11/21/11

- Initial release

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