Aptina™
IMAGING

AP0101AT High-Dynamic Range (HDR) Image Signal Processor (ISP)

AP0101AT Data Sheet

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

Features

- Up to 1.2Mp (1280x960) Aptina sensor support
- 45 fps at 1.2Mp, 60 fps at 720p
- Optimized for operation with HDR sensors.
- Color and gamma correction
- Auto exposure, auto white balance, 50/60 Hz flicker avoidance
- Adaptive Local Tone Mapping (ALTM)
- Test Pattern Generator
- Two-wire serial programming interface (Slave)
- Interface to low-cost Flash or EPROM through SPI bus (to config and load patches etc)
- High-level host command interface
- Standalone operation supported
- Up to 5 GPIO
- Fail-safe IO
- Multi-Camera synchronization support
- Dual Band IR filter support

Applications

- Surround, rear and front view cameras
- Blind spot / side mirror replacement cameras
- Automotive viewing/processing fusion cameras

Ordering Information

Table 1:   Available Part Numbers

Part Number	Description
AP0101AT2L00XPGA0-E	81-ball VFBGA Package Part Samples
AP0101AT2L00XPGA0	81-ball VFBGA Package Part
AP0101AT2L00XPGAD-E	Demo kit
AP0101AT2L00XPGAH-E	Headboard

Table 2:   Key Performance Parameters

Parameter		Value
Primary camera interface		Parallel
Primary camera input format		RAW12 Linear/Companded Bayer data
Output interface		Up to 16-bit Parallel
Output format		YUV422 8-bit,10-bit, and SMPTE296M 10-, 12-bit tone-mapped Bayer
Maximum resolution		1280x960 (1.2Mp)
Input clock range		6-30Mhz
Maximum frame rate[1]		45 fps at 1.2Mp, 60 fps at 720p
Maximum output clock frequency		Parallel clock up to 84 Mhz
Supply voltage	$V_{DD}IO\_S$	1.8 or 2.8V nominal
	$V_{DD}IO\_H$	1.8, 2.5, or 3.3V nominal
	V_{DD}_REG	1.8V nominal
	$V_{DD}IO\_OTPM$	2.5 or 3.3V nominal
Operating temperature (ambient - T_A)		−40°C to +105°C
Power consumption[2]		128.8 mW

Notes: 1. Maximum frame rate depends on output interface and data format configuration used.
2. 720p HDR 60 fps 74.25MHz YCbCr_422_16

Table of Contents

List of Figures

List of Tables

General Description

Aptina's AP0101AT is a high-performance, ultra-low power in-line, digital image processor optimized for use with HDR (High Dynamic Range) sensors. The AP0101AT provides full auto-functions support (AWB and AE) and ALTM (Adaptive Local Tone Mapping) to enhance HDR images and advanced noise reduction which enables excellent low-light performance.

Functional Overview

Figure 1 shows the typical configuration of the AP0101AT in a camera system. On the host side, a two-wire serial interface is used to control the operation of the AP0101AT, and image data is transferred using the parallel bus between the AP0101AT and the host. The AP0101AT interface to the sensor also uses a parallel interface.

Figure 1: **AP0101AT Connectivity**
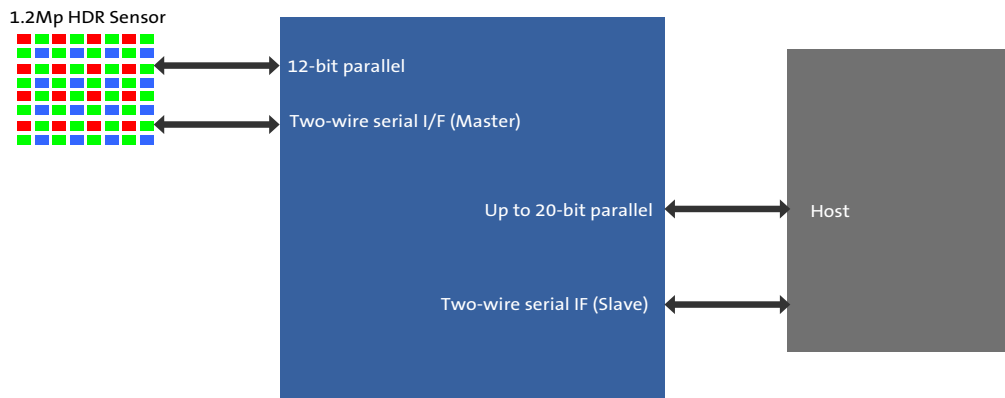
System Interfaces

Figure 2 on page 7 shows typical AP0101AT device connections.

All power supply rails must be decoupled from ground using capacitors as close as possible to the package.

The AP0101AT signals to the sensor and host interfaces can be at different supply voltage levels to optimize power consumption and maximize flexibility. Table 3 on page 8 provides the signal descriptions for the AP0101AT.

Figure 2:   Typical Configuration
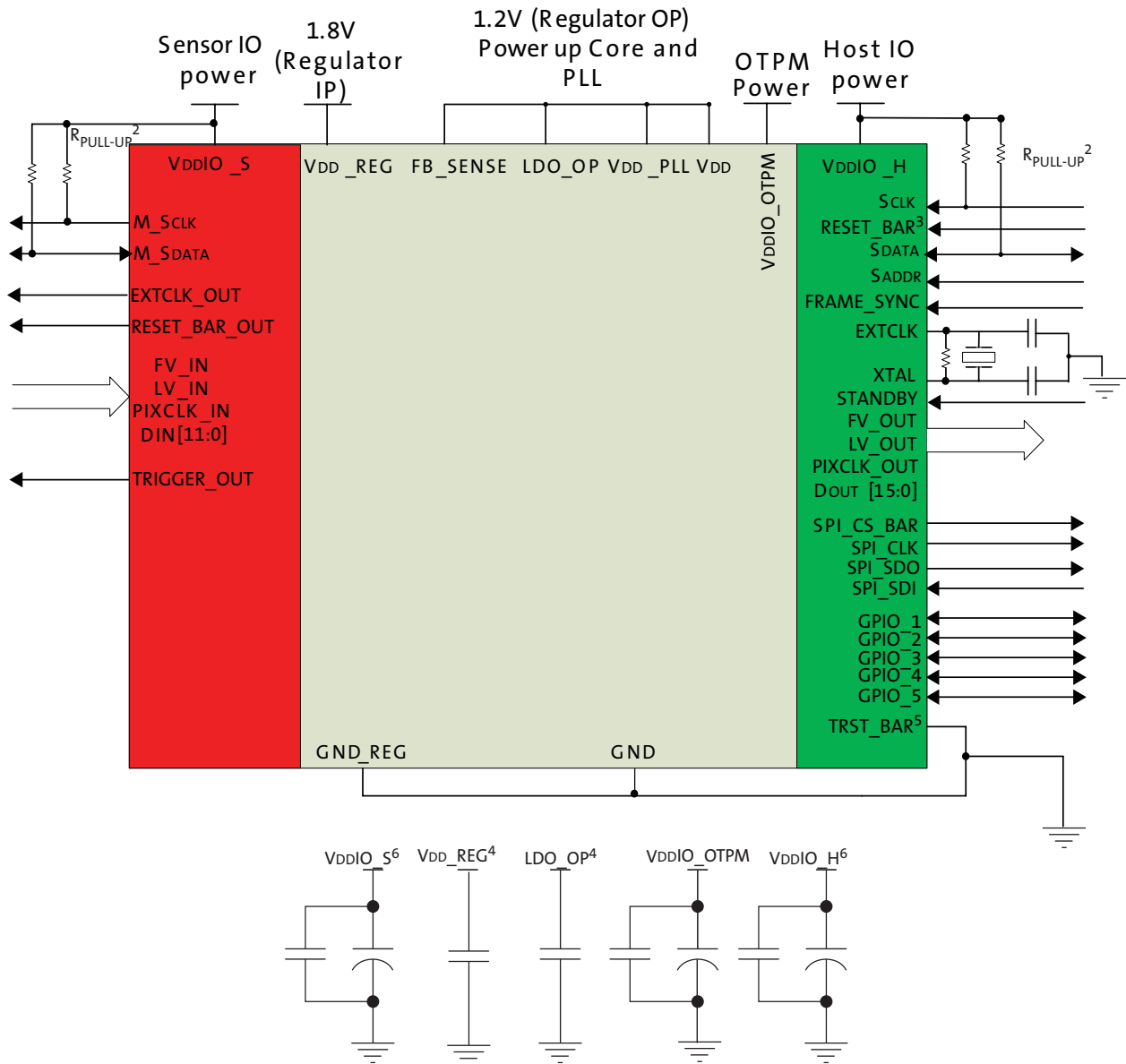
Note: 1. This typical configuration shows only one scenario out of multiple possible variations for this sensor.
2. Aptina recommends a 1.5kΩ resistor value for the two-wire serial interface $R_{PULL-UP}$; however, greater values may be used for slower transmission speed.
3. RESET_BAR has an internal pull-up resistor and can be left floating if not used.
4. The decoupling capacitors for the regulator input and output should have a value of 1.0uF. The capacitors should be ceramic and need to have X5R or X7R dielectric.
5. TRST_BAR connects to GND for normal operation.
6. Aptina recommends that 0.1μF and 1μF decoupling capacitors for each power supply are mounted as close as possible to the pin. Actual values and numbers may vary depending on layout and design consideration

Crystal Usage

As an alternative to using an external oscillator, a crystal may be connected between EXTCLK and XTAL. Two small loading capacitors and a feedback resistor should be added, as shown in Figure 3.

Aptina does not recommend using the crystal option for applications above 85°C. A crystal oscillator with temperature compensation is recommended for applications that require this.

Figure 3:   Using a Crystal Instead of an External Oscillator
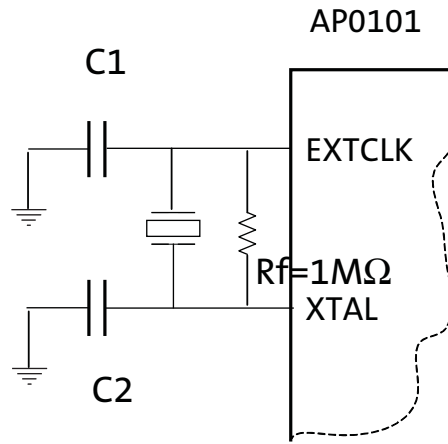
Rf represents the feedback resistor, an Rf value of $1M\Omega$ would be sufficient for AP0101AT. C1 and C2 are decided according to the crystal or resonator CL specification. In the steady state of oscillation, CL is defined as $(C1 \times C2)/(C1+C2)$. In fact, the I/O ports, the bond pad, package pin and PCB traces all contribute the parasitic capacitance to C1 and C2. Therefore, CL can be rewritten to be $(C1^* \times C2^*)/(C1^*+C2^*)$, where $C1^*=(C1+Cin,$ stray$)$ and $C2^*=(C2+Cout,$ stray$)$. The stray capacitance for the IO ports, bond pad and package pin are known which means the formulas can be rewritten as $C1^*=(C1+1.5pF+Cin,$ PCB$)$ and $C2^*=(C2+1.3pF+Cout,$ PCB$)$.

Table 3:   Pin Descriptions

Name	Type	Description
EXTCLK	Input	Master input clock. This can either be a square-wave generated from an oscillator (in which case the XTAL input must be left unconnected) or direct connection to a crystal.
XTAL	Output	If EXTCLK is connected to one pin of a crystal, the other pin of the crystal is connected to XTAL pin; otherwise this signal must be left unconnected.
RESET_BAR	Input/PU	Master reset signal, active LOW. This signal has an internal pull up.
S$_{CLK}$	Input	Two-wire serial interface clock (host interface).
S$_{DATA}$	I/O	Two-wire serial interface data (host interface).
S$_{ADDR}$	Input	Selects device address for the two-wire slave serial interface. When connected to GND the device ID is 0x90. When wired to V$_{DDIO\_H}$, a device ID of 0xBA is selected.
FRAME_SYNC	Input	This input can be used to set the output timing of the AP0101AT. The input buffer associated with this input is permanently enabled. This signal should be connected to GND if not used.
STANDBY	Input	Standby mode control, active HIGH.
SPI_S$_{CLK}$	Output	Clock output for interfacing to an external SPI flash or EEPROM memory.

Table 3:   Pin Descriptions (Continued)

Name	Type	Description
SPI_SDI	Input	Data in from SPI flash or EEPROM memory. When no SPI device is fitted, this signal is used to determine whether the AP0101AT should auto-configure: 0: Do not auto-configure; two-wire interface will be used to configure the device (host-config mode) 1: Auto-configure. This signal has an internal pull-up resistor.
SPI_SDO	Output	Data out to SPI flash or EEPROM memory.
SPI_CS_BAR	Output	Chip select out to SPI flash or EEPROM memory.
FV_OUT	Output	Host frame valid output (synchronous to PIXCLK_OUT)
LV_OUT	Output	Host line valid output (synchronous to PIXCLK_OUT)
PIXCLK_OUT	Output	Host pixel clock output.
D_{OUT}[15:0]	Output	Host pixel data output (synchronous to PIXCLK_OUT) D_{OUT}[15:0]. Note 20-bit output (SMPTE) also uses GPIO[5:2].
GPIO [5:1]	I/O	General purpose digital I/O. Note: 20-bit output (SMPTE) also uses GPIO[5:2]
TRST_BAR	Input	Must be tied to GND in normal operation.
EXT_CLK_OUT	Output	Clock to external sensor.
RESET_BAR_OUT	Output	Reset signal to external sensor.
M_S_{CLK}	Output	Two-wire serial interface clock (Master).
M_S_{DATA}	I/O	Two-wire serial interface clock (Master).
FV_IN	Input	Sensor frame valid input.
LV_IN	Input	Sensor line valid input.
PIXCLK_IN	Input	Sensor pixel clock input.
DIN[11:0]	Input	Sensor pixel data input D_{IN}[11:0]
TRIGGER_OUT	Output	Trigger signal for external sensor.
V_{DD}IO_S	Supply	Sensor I/O power supply.
GND	Supply	Ground for sensor IO, host IO, PLL, V_{DD}IO_OTPM, and V_{DD}.
VDD_REG	Supply	Input to on-chip 1.8V to 1.2V regulator.
LDO_OP	Output	Output from on chip 1.8V to 1.2V regulator. Note: The regulator on the AP0101AT must be used.
FB_SENSE	Output	On-chip regulator sense signal.
GND_REG	Supply	Ground for on-chip regulator
V_{DD}_PLL	Supply	PLL supply.
V_{DD}	Supply	Core supply.
V_{DD}IO_OTPM	Supply	OTPM power supply.
V_{DD}IO_H	Supply	Host I/O power supply.

Table 4:   Package Pinout

	1	2	3	4	5	6	7	8	9
A	EXTCLK	XTAL	S$_{CLK}$	SPI_SDO	D$_{OUT}$[15]	D$_{OUT}$[13]	D$_{OUT}$[10]	D$_{OUT}$[9]	D$_{OUT}$[8]
B	V$_{DD}$	VDDIO_H	S$_{DATA}$	SPI_SDI	D$_{OUT}$[14]	D$_{OUT}$[12]	D$_{OUT}$[11]	D$_{OUT}$[7]	D$_{OUT}$[6]
C	EXT_CLK_OUT	VDDIO_S	S$_{ADDR}$	SPI_CS_BAR	GND	PIXCLK_OUT	FV_OUT	D$_{OUT}$[5]	D$_{OUT}$[4]
D	RESET_BAR_OUT	V$_{DD}$	GND	SPI_SCLK	GND	TRST_BAR	LV_OUT	D$_{OUT}$[3]	D$_{OUT}$[2]
E	D$_{IN}$[3]	D$_{IN}$[7]	GND	FB_SENSE	GND	GND	V$_{DD}$_PLL	D$_{OUT}$[1]	D$_{OUT}$[0]
F	D$_{IN}$[11]	D$_{IN}$[2]	LDO_OP	GND_REG	GND	GND	V$_{DD}$_PLL	V$_{DD}$_PLL	VDDIO_OTPM
G	D$_{IN}$[6]	D$_{IN}$[1]	D$_{IN}$[4]	V$_{DD}$_REG	VDDIO_S	V$_{DD}$	RESET_BAR	GPIO[4]	GPIO[5]
H	D$_{IN}$[10]	D$_{IN}$[0]	D$_{IN}$[8]	FV_IN	M_S$_{DATA}$	VDDIO_H	FRAME_SYNC	GPIO[2]	GPIO[3]
J	D$_{IN}$[5]	D$_{IN}$[9]	PIXCLK_IN	LV_IN	M_S$_{CLK}$	V$_{DD}$	STANDBY	TRIGGER_OUT	GPIO[1]

Power-Up and Down Sequence

Powering up and down the AP0101AT requires voltages to be applied in a particular order, as seen in Figure 4. The timing requirements are shown in Table 5. The AP0101AT includes a power-on reset feature that initiates a reset upon power up of the AP0101AT.

Figure 4:   Power-Up and Power-Down Sequence
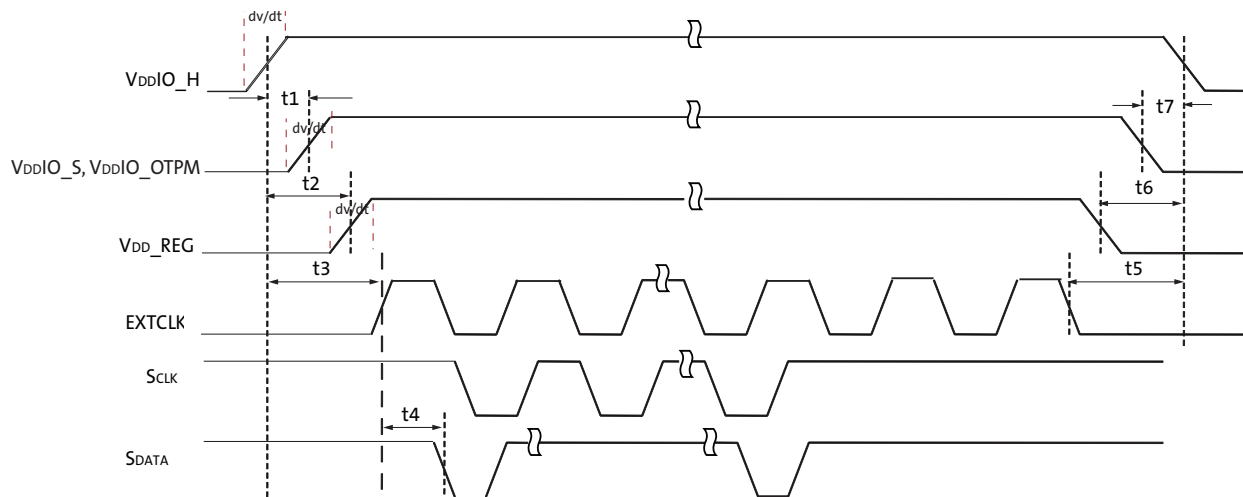
Table 5:   **Power-Up and Power-Down Signal Timing**

Symbol	Parameter	Min	Typ	Max	Unit
t1	Delay from V_{DDIO_H} to V_{DDIO_S}, V_{DDIO_OTPM}	0	–	50	ms
t2	Delay from V_{DDIO_H} to V_{DD_REG}	0	–	50	ms
t3	EXTCLK activation	t2 + 1	–	–	ms
t4	First serial command	100	–	–	EXTCLK cycles
t5	EXTCLK cutoff	t6	–	–	ms
t6	Delay from V_{DD_REG} to V_{DDIO_H}	0	–	50	ms
t7	Delay from V_{DDIO_S}, V_{DDIO_OTPM} to V_{DDIO_H}	0	–	50	ms
dv/dt	Power supply ramp time (slew rate)	–	–	0.1	V/µs

Note: If the system cannot support this power supply slew rate, then power supplies must be designed to overcome inrush currents in Table 29, "Inrush Current," on page 44.

Reset

The AP0101AT has 3 types of reset available:
- A hard reset is issued by toggling the RESET_BAR signal
- A soft reset is issued by writing commands through the two-wire serial interface
- An internal power-on reset

Hard Reset

The AP0101AT enters the reset state when the external RESET_BAR signal is asserted LOW, as shown in Figure 5. All the output signals will be in High-Z state.

Figure 5:   **Hard Reset Operation**
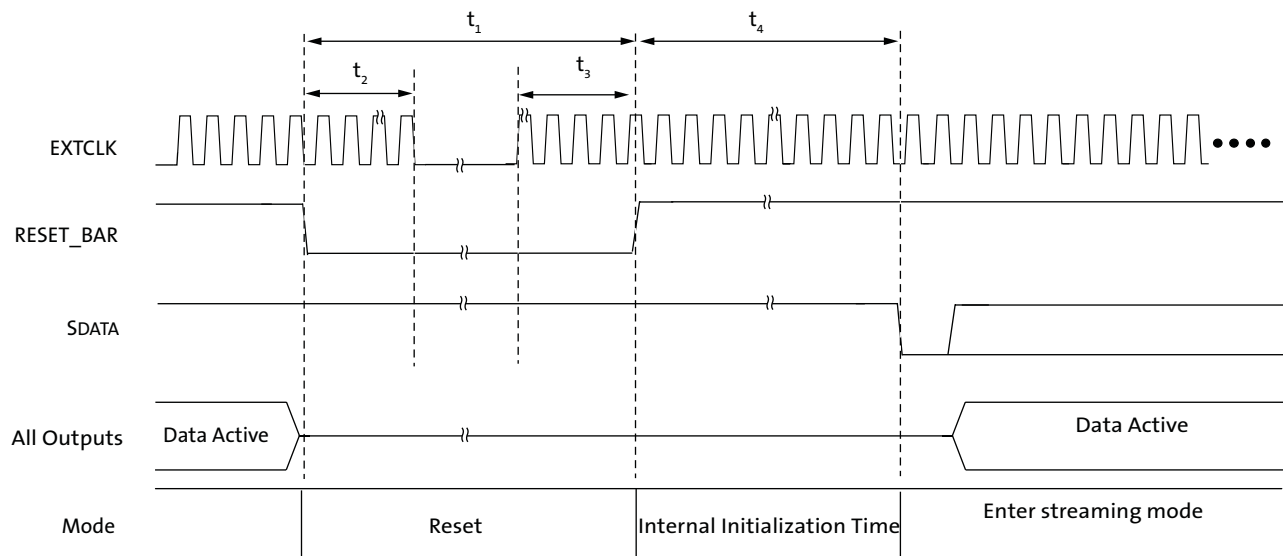
Note: This assumes auto-config.

Table 6: **Hard Reset**

Symbol	Definition	Min	Typ	Max	Unit
t_1	RESET_BAR pulse width	50	–	–	EXTCLK cycles
t_2	Active EXTCLK required after RESET_BAR asserted	10	–	–	
t_3	Active EXTCLK required before RESET_BAR de-asserted	10	–	–	
t_4	First two-wire serial interface communication after RESET_BAR is HIGH	100	–	–	

Soft Reset

A soft reset sequence to the AP0101AT can be activated by writing to a register through the two-wire serial interface.

Hard Standby Mode

The AP0101AT can enter hard standby mode by using the external STANDBY signal, as shown in Figure 6.

Entering Standby Mode

1. Assert STANDBY signal HIGH.

Exiting Standby Mode

1. De-assert STANDBY signal LOW.

Figure 6:   Hard Standby Operation
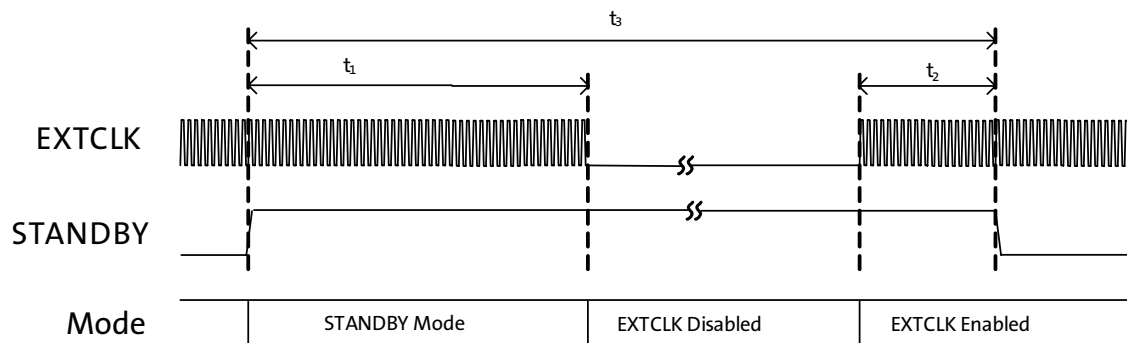
Table 7:   Hard Standby Signal Timing

Symbol	Parameter	Min	Typ	Max	Unit
t_1	Standby entry complete	900	–	1 Frame + 26	Lines
t_2	Active EXTCLK required after going into STANDBY mode	10	–	–	EXTCLKs
t_3	Active EXTCLK required before STANDBY de-asserted	10	–	–	EXTCLKs
t_4	STANDBY pulse width	1 Frame + 40	–		Lines

Multi-Camera Synchronization Support

The AP0101AT supports multi-camera synchronization via the FRAME_SYNC pin. The host (or controlling entity) 'broadcasts' a sync-pulse to all cameras within the system that triggers streaming start. The AP0101AT will propagate the signal to the TRIGGER_OUT pin to the sensor's TRIGGER pin.

The AP0101ATsupports two different trigger modes. The first mode supported is 'single-shot'; this is when the trigger pulse will cause one frame to be output from the image sensor and AP0101AT (see Figure 7).

Figure 7:   **Single-Shot Mode**
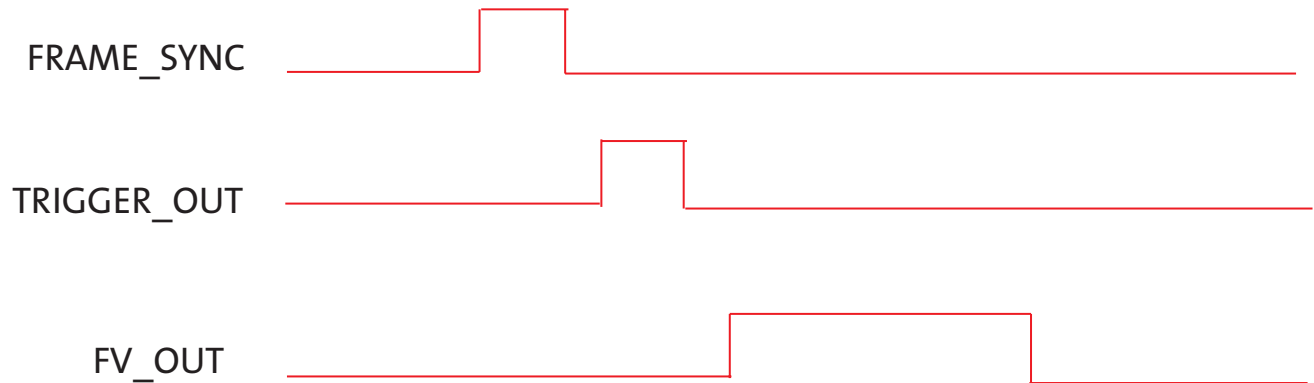
The second mode supported is called 'continuos'; this is when a trigger pulse will cause the part to continuously output frames, see Figure 8. This mode would be especially useful for applications which have multiple sensors and need to have their video streams synchronized (for example, surround view or panoramic view applications).

Figure 8:   **Continuous Mode**
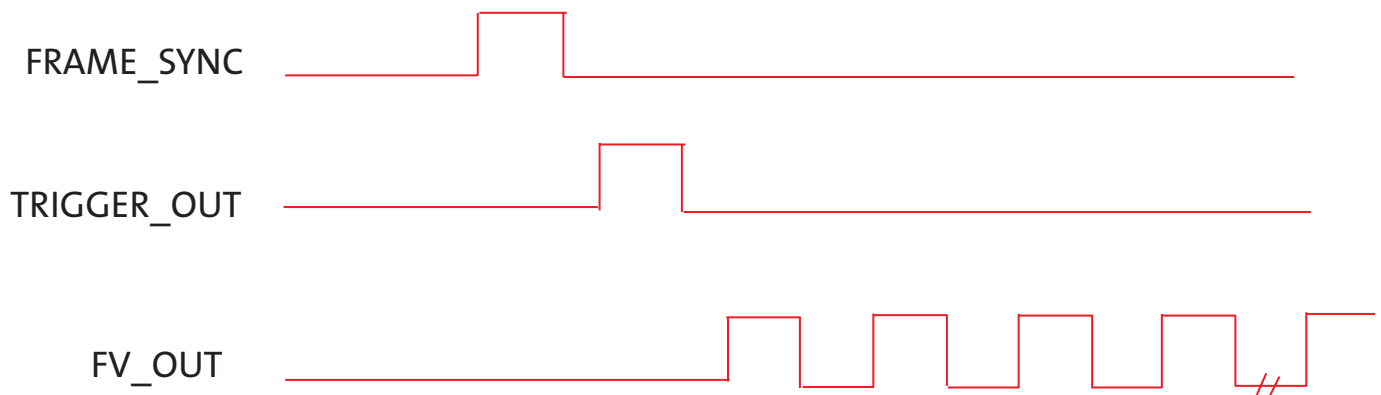
When two or more cameras have a signal applied to the FRAME_SYNC input at the same time, the respective FV_OUT signals would be synchronized within 5 PIXCLK_OUT cycles. This assumes that all cameras have the same configuration settings and that the exposure time is the same.

Image Flow Processor

Image and color processing in the AP0101AT is implemented as an image flow processor (IFP) coded in hardware logic. During normal operation, the embedded microcontroller will automatically adjust the operating parameters. For normal operation of the AP0101AT, streams of raw image data from the attached image sensor are fed into the color pipeline. The user also has the option to select a number of test patterns to be input instead of sensor data. The test pattern is fed to the IFP for testing the image pipeline without sensor operation.
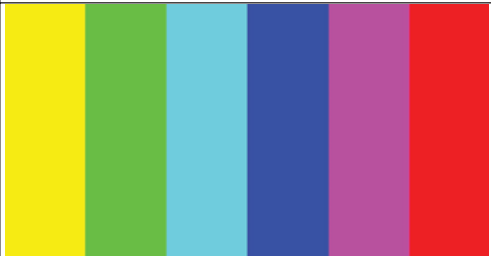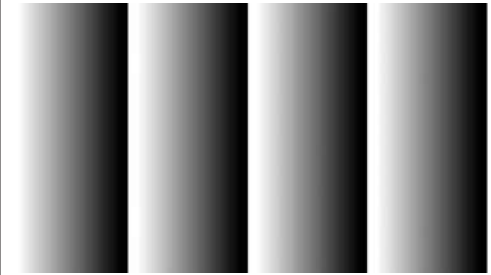
The test patterns can be selected by programming variables. To select enter test pattern mode, set R0xC88F to 0x02 and issue a Change- Config request; to exit this mode, set R0xC88F to 0x00.

Figure 9: **AP0101AT IFP**
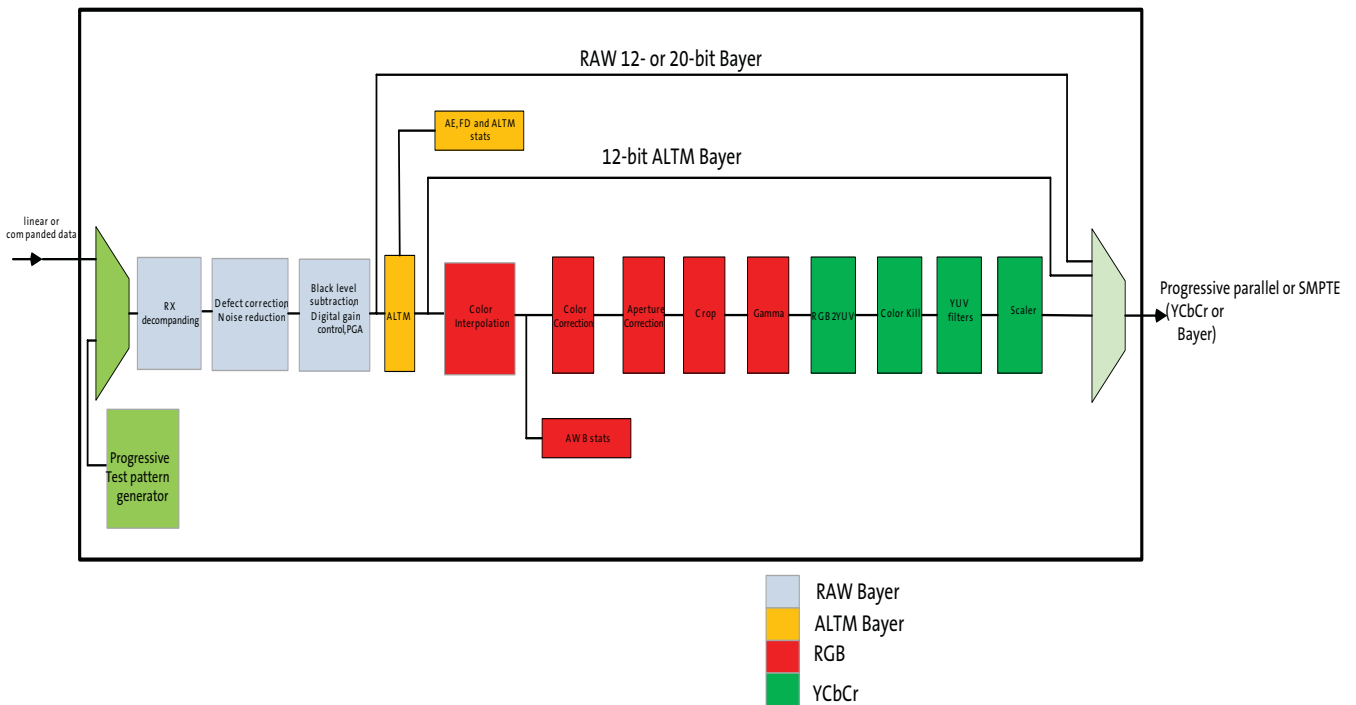
Test Patterns

Figure 10:   Color Bar Test Pattern

Test Pattern	Example

FLAT FIELD
REG= 0xC88C, 0x02   // CAM_MODE_SELECT
REG= 0xC88F, 0x01   // CAM_MODE_TEST_PATTERN_SELECT
REG= 0xC890, 0x000FFFFF   // CAM_MODE_TEST_PATTERN_RED
REG= 0xC894, 0x000FFFFF   // CAM_MODE_TEST_PATTERN_GREEN
REG= 0xC898, 0x000FFFFF   // CAM_MODE_TEST_PATTERN_BLUE
Load = Change-Config
Changing the values in R0xC890-R0xC898 will change the color of the test pattern.	

100% Color Bar
REG= 0xC88C, 0x02   // CAM_MODE_SELECT
REG= 0xC88F, 0x02   // CAM_MODE_TEST_PATTERN_SELECT
Load = Change-Config	

Pseudo-Random
REG= 0xC88C, 0x02   // CAM_MODE_SELECT
REG= 0xC88F, 0x05   // CAM_MODE_TEST_PATTERN_SELECT
Load = Change-Config	

Fade-to-Gray
REG= 0xC88C, 0x02   // CAM_MODE_SELECT
REG= 0xC88F, 0x08   // CAM_MODE_TEST_PATTERN_SELECT
Load = Change-Config	

Linear Ramp
REG= 0xC88C, 0x02   // CAM_MODE_SELECT
REG= 0xC88F, 0x09   // CAM_MODE_TEST_PATTERN_SELECT
Load = Change-Config	

Defect Correction

After data decompanding the image stream processing starts with defect correction.

To obtain defect free images, the pixels marked defective during sensor readout and the pixels determined defective by the defect correction algorithms are replaced with values derived from the non-defective neighboring pixels. This image processing technique is called defect correction.

AdaCD (Adaptive Color Difference)

Automotive applications require good performance in extremely low light, even at high temperature conditions. In these stringent conditions the image sensor is prone to higher noise levels, and so efficient noise reduction techniques are required to circumvent this sensor limitation and deliver a high quality image to the user.

The AdaCD Noise Reduction Filter is able to adapt its noise filtering process to local image structure and noise level, removing most objectionable color noise while preserving edge details.

Black Level Subtraction and Digital Gain

After noise reduction, the pixel data goes through black level subtraction and multiplication of all pixel values by a programmable digital gain. Independent color channel digital gain can be adjusted with registers. Black level subtract (to compensate for sensor data pedestal) is a single value applied to all color channels. If the black level subtraction produces a negative result for a particular pixel, the value of this pixel is set to 0.

Positional Gain Adjustments (PGA)

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing fixed pattern signal gradients in images captured by image sensors. The cumulative result of all these factors is known as image shading. The AP0101AT has an embedded shading correction module that can be programmed to counter the shading effects on each individual R, Gb, Gr, and B color signal.

The Correction Function

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

$$P_{corrected}(row, col) = P_{sensor}(row, col) \times f(row, col) \qquad \text{(EQ 1)}$$

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

Adaptive Local Tone Mapping (ALTM)

Real world scenes often have 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 object in a scene. In recent years many technologies have been developed to capture the full dynamic range of real world scenes. For example, the multiple exposure method is a widely adopted method for capturing high dynamic range images, which combines a series of low dynamic range images of the same scene taken under different exposure times into a single HDR image.

Even though the new digital imaging technology enables the capture of the full dynamic range, low dynamic range display devices are the limiting factor. Today's typical LCD monitor has contrast ratio around 1,000:1; however, it is not atypical for an HDR image having contrast ratio 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 called tone mapping.

Tone mapping methods can be classified into global tone mapping and local tone mapping. Global tone mapping methods apply the same mapping function to all pixels. While global tone mapping methods provide computationally simple and easy to use solutions, they often cause loss of contrast and detail. A local tone mapping is thus necessary in addition to global tone mapping for the reproduction of visually more appealing images that also reveal scene details that are important for automotive safety and surveillance applications. Local tone mapping methods use a spatially varying mapping function determined by the neighborhood of a pixel, which allows it to increase the local contrast and the visibility of some details of the image. Local methods usually yield more pleasing results because they exploit the fact that human vision is more sensitive to local contrast.

Aptina's ALTM solution significantly improves the performance over global tone mapping. ALTM is directly applied to the Bayer domain to compress the dynamic range from 20-bit to 12-bit. This allows the regular color pipeline to be used for HDR image rendering.

Color Interpolation

In the raw data stream fed by the sensor core to the IFP, each pixel is represented by a 20- or 12-bit integer number, which can be considered proportional to the pixel's response to a one-color light stimulus, red, green, or blue, depending on the pixel's position under the color filter array. Initial data processing steps, up to and including ALTM, preserve the one-color-per-pixel nature of the data stream, but after ALTM it must be converted to a three-colors-per-pixel stream appropriate for standard color processing. The conversion is done by an edge-sensitive color interpolation module. The module pads the incomplete color information available for each pixel with information extracted from an appropriate set of neighboring pixels. The algorithm used to select this set and extract the information seeks the best compromise between preserving edges and filtering out high frequency noise in flat field areas. The edge threshold can be set through register settings.

Color Correction and Aperture Correction

To achieve good color fidelity of the IFP output, interpolated RGB values of all pixels are subjected to color correction. The IFP multiplies each vector of three pixel colors by a 3 x 3 color correction matrix. The three components of the resulting color vector are all sums of three 10-bit numbers. The color correction matrix can be either programmed by

the user or automatically selected by the auto white balance (AWB) algorithm implemented in the IFP. Color correction should ideally produce output colors that are corrected for the spectral sensitivity and color crosstalk characteristics of the image sensor. The optimal values of the color correction matrix elements depend on those sensor characteristics and on the spectrum of light incident on the sensor. The color correction variables can be adjusted through register settings.

To increase image sharpness, a programmable 2D aperture correction (sharpening filter) is applied to color-corrected image data. The gain and threshold for 2D correction can be defined through register settings.

Gamma Correction

The gamma correction curve is implemented as a piecewise linear function with 33 knee points, taking 12-bit arguments and mapping them to 10-bit output. The abscissas of the knee points are fixed at 0, 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384, 448, 512, 640, 768, 896, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096. The 10-bit ordinates are programmable through variables.

The AP0101AT has the ability to calculate the 33-point knee points based on the tuning of cam_ll_gamma and cam_ll_contrast_gradient_bright. The other method is for the host to program the 33 knee point curve themselves.

Also included in this block is a Fade-to Black curve which sets all knee points to zero and causes the image to go black in extreme low light conditions.

Color Kill

To remove high-or low-light color artifacts, a color kill circuit is included. It affects only pixels whose luminance exceeds a certain preprogrammed threshold. The U and V values of those pixels are attenuated proportionally to the difference between their luminance and the threshold.

YUV Color Filter

As an optional processing step, noise suppression by one-dimensional low-pass filtering of Y and/or UV signals is possible. A 3- or 5-tap filter can be selected for each signal.

Camera Control and Auto Functions

Auto Exposure

The auto exposure algorithm optimizes scene exposure to minimize clipping and saturation in critical areas of the image. This is achieved by controlling exposure time and analog gains of the sensor core as well as digital gains applied to the image.

The auto exposure module analyzes image statistics collected by the exposure measurement engine, makes a decision, and programs the sensor and color pipeline to achieve the desired exposure. The measurement engine subdivides the image into 25 windows organized as a 5 x 5 grid.

Figure 11:   5 x 5 Grid
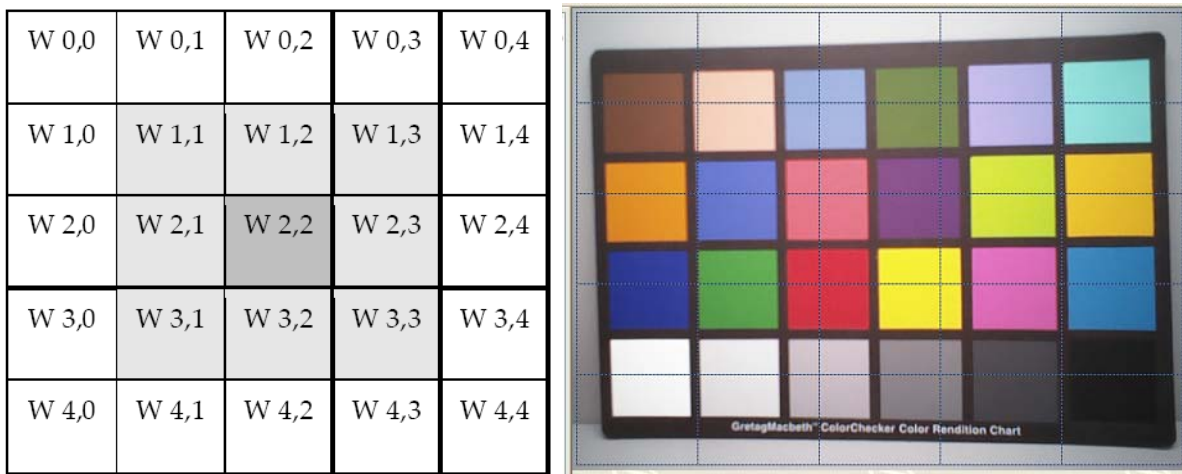
AE Track Driver

Other algorithm features include the rejection of fast fluctuations in illumination (time averaging), control of speed of response, and control of the sensitivity to small changes. While the default settings are adequate in most situations, the user can program target brightness, measurement window, and other parameters described above.

The driver changes AE parameters (integration time, gains, and so on) to drive scene brightness to the programmable target.

To avoid unwanted reaction of AE on small fluctuations of scene brightness or momentary scene changes, the AE track driver uses a temporal filter for luma and a threshold around the AE luma target. The driver changes AE parameters only if the difference between the AE luma target and the filtered luma is larger than the AE target step and pushes the luma beyond the threshold.

Auto White Balance

The AP0101AT has a built-in AWB algorithm designed to compensate for the effects of changing spectra of the scene illumination on the quality of the color rendition. The algorithm consists of two major parts: a measurement engine performing statistical analysis of the image and a driver performing the selection of the optimal color correction matrix and IFP digital gain. While default settings of these algorithms are adequate in most situations, the user can reprogram base color correction matrices, place limits on color channel gains, and control the speed of both matrix and gain adjustments. The AP0101AT AWB displays the current AWB position in color temperature, the range of which will be defined when programming the CCM matrixes.

The region of interest can be controlled through the combination of an inclusion window and an exclusion window.

Dual Band IRCF

For some applications a day/night filter would be switched in/out, this option is an additional cost to the camera system. The AP0101AT supports the use of dual band IRCF, which removes the need for the switching day/night filter. Tuning support is provided for this usage case. Refer to the AP0101AT developer guide for details.

Exposure and White Balance Modes

AP0101AT supports auto and manual exposure and white balance modes. In addition, it will operate within synchronized multi-camera systems. In this use case, one camera within the system will be the 'master', and the others 'slaves'. The master is used to calculate the appropriate exposure and white balance. This is then applied to all slaves concurrently under host control.

Auto Mode

In Auto Exposure mode the AE algorithm is responsible for calculating the appropriate exposure to keep the desired scene brightness, and for applying the exposure to the underlying hardware. In Auto White Balance mode the AWB algorithm is responsible for calculating the color temperature of the scene and applying the appropriate red and blue gains.

Triggered Auto mode

The Triggered Auto Exposure and Triggered Auto White Balance modes are intended for the multi-camera use cases, where a host is controlling the exposure and white balance of a number of cameras. The idea is that one camera is in triggered-auto mode (the master), and the others in host-controlled mode (slaves). The master camera must calculate the exposure and gains, the host then copies this to the slaves, and all changes are then applied at the same time.

Manual Mode

Manual mode is intended to allow simple manual exposure and white balance control by the host. The host needs to set the CAM_AET_EXPOSURE_TIME_MS, CAM_AET_EXPOSURE_GAIN and CAM_AWB_COLOR_TEMPERATURE controls, the camera will calculate the appropriate integration times and gains.

Host Controlled

The Host Controlled mode is intended to give the host full control over exposure and gains.

Flicker Avoidance

Flicker occurs when the integration time is not an integer multiple of the period of the light intensity. The AP0101AT can be programmed to avoid flicker for 50 or 60 Hertz. For integration times less than the light intensity period (10ms for 50Hz environment), flicker cannot be avoided. The AP0101AT supports an indoor AE mode, that will ensure flicker-free operation.

Output Formatting

The AP0101AT can output pixel data as an 8 or 10 bit word, over one or two clocks per pixel. AP0101AT supports parallel output & SMPTE modes.

Uncompressed YCbCr Data Ordering

The AP0101AT supports swapping YCbCr mode, as illustrated in Table 8.

Table 8:   YCbCr Output Data Ordering

Mode	Data Sequence			
Default (no swap)	Cbi	Yi	Cri	Yi+1
Swapped CrCb	Cri	Yi	Cbi	Yi+1
Swapped YC	Yi	Cbi	Yi+1	Cri
Swapped CrCb, YC	Yi	Cri	Yi+1	Cbi

The data ordering for the YCbCr output modes for AP0101AT are shown in Table 8:

Table 9:   YCbCr Output Modes (cam_port_parallel_msb_align=0x1)

Mode	Byte	Pixel i	Pixel i+1	Notes
YCbCr_422_8_8	Odd (Dout [15:8])	Cbi	Cri	Data range of 0-255 (Y=16-235 and C=16-240)
	Even (Dout [15:8])	Yi	Yi+1	
YCbCr_422_10_10	Odd (Dout [15:6])	Cbi	Cri	Data range of 0-1023 (Y=64-940 and C=64-960)
	Even (Dout [15:6])	Yi	Yi+1	
YCbCr_422_16	Single (Dout [15:0])	Cbi_Yi	Cri_Yi+1	Data range of 0-255 (Y=16-235 and C=16-240)

Note: Odd means first cycle; even means second cycle.

Table 10:   YCbCr Output Modes (cam_port_parallel_msb_align=0x0)

Mode	Byte	Pixel i	Pixel i+1	""Notes""
YCbCr_422_8_8	Odd (Dout[7 :0])	Cbi	Cri	Data range of 0-255 (Y=16-235 and C=16-240)
	Even (Dout [7:0]	Yi	Yi+1	
YCbCr_422_10_10	Odd (Dout [9:0])	Cbi	Cri	Data range of 0-1023 (Y=64-940 and C=64-960)"
	Even (Dout [9:0])	Yi	Yi+1	
YCbCr_422_16	Single (Dout [15:0])	Cbi_Yi	Cri_Yi+1	Data range of 0-255 (Y=16-235 and C=16-240)

Figure 12: **8- bit YCbCr Output (YCbCr_422_8_8)**

Pixel Clock
Frame Valid
Line Valid
Data[15:8]
Data[7:0]

Porch – 0-255 cycles
00
Cr
HBlank | Image | HBlank | Image | HBlank

Pixel Clock
Frame Valid
Line Valid
Data[15:8]
Data[7:0]

Porch – 0-255 cycles
00
Cr
HBlank | Image | HBlank | Image | HBlank

Active Video

Pixel Clock
Frame Valid
Line Valid
Data[15:8]
Data[7:0]

Porch – 0-255 cycles
00
Cr
Image | Vblank

Pixel Clock
Frame Valid
Line Valid
Data[15:8]
Data[7:0]

Porch – 0-255 cycles
00
Cr
Vblank | Image

Vertical Blanking

Note: Cb Y Cr Y by default.

Figure 13: **10-bit YCbCr Output (YCbCr_422_10_10)**

Active Video

Vertical Blanking
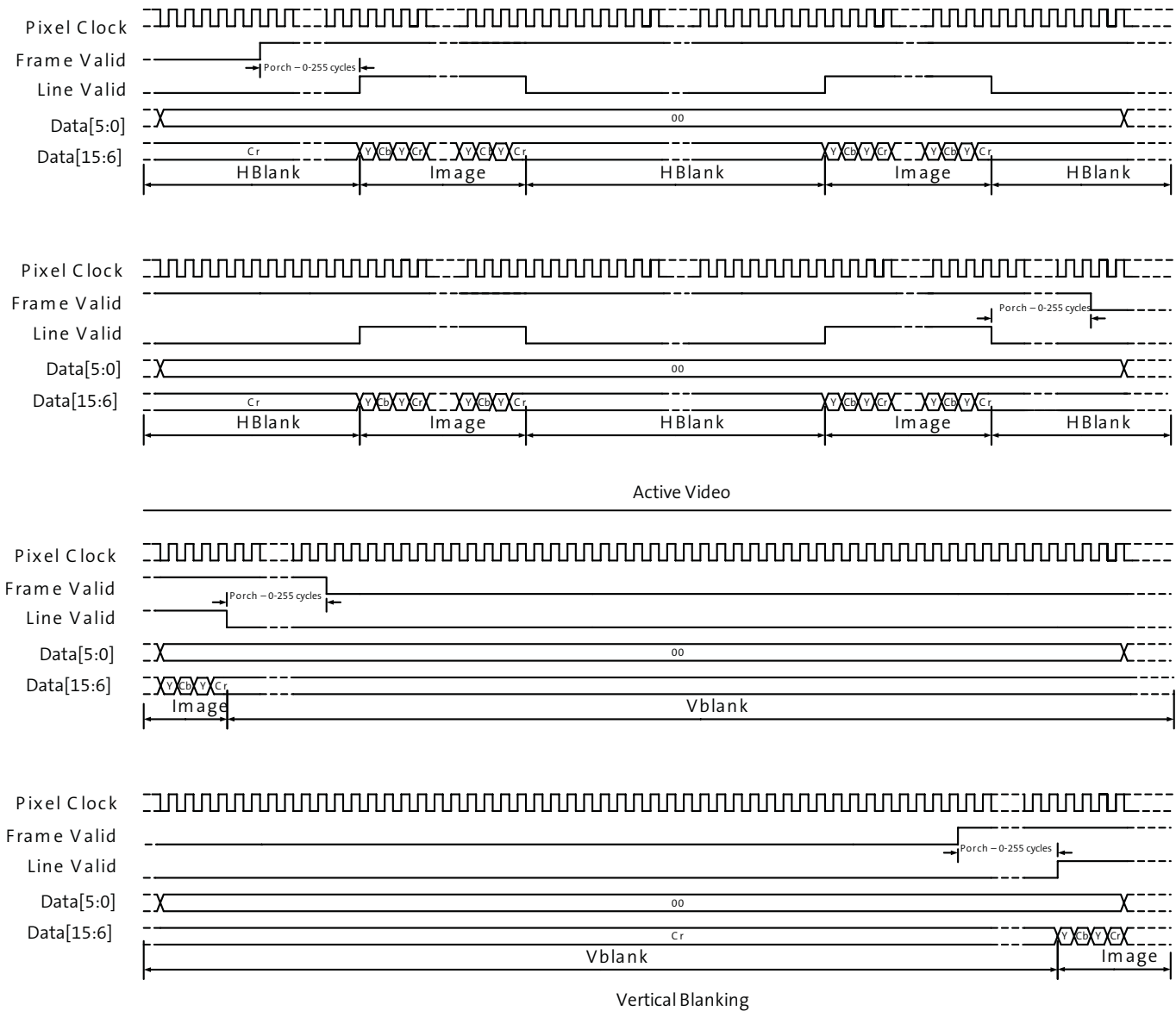
Note: Cb Y Cr Y by default.

Figure 14: **16-bit YCbCr Output (YCbCr_422_16)**

Active Video

Vertical Blanking
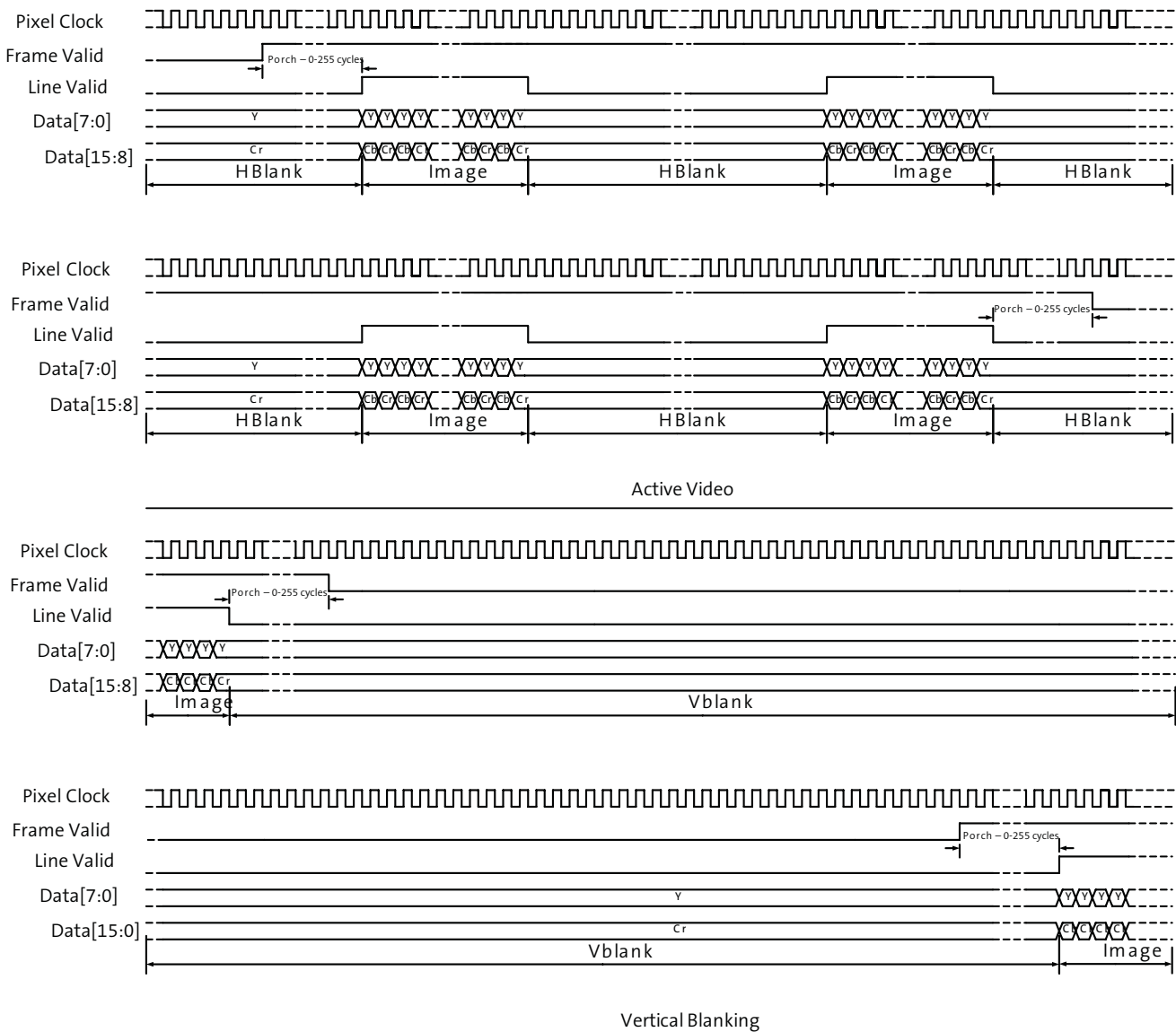
SMPTE Output

The data ordering for the SMPTE output mode for AP0101AT shown in Table 10:

Table 11:   SMPTE Output Mode

Mode	Byte	Pixel i	Pixel i+1	Notes
SMPTE	Single{Dout[15:8],GPIO[5:4]}-->Cb/Cr {Dout[7:0],GPIO[3:2]} --->Y	Cbi_Yi	Cri_Yi+1	Data range of 4-1019 (Y=64-940 and C=64-960)

Figure 15:   SMPTE296M Output

ALTM Bayer Output

The data ordering for the ALTM Bayer output modes for AP0101AT are shown in Table 12.

Table 12: ALTM Bayer Output Modes

Mode	Byte	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
ALTM_Bayer_10	Single	0	0	0	0	0	0	D9	D98	D7	D6	D5	D4	D3	D2	D1	D0
ALTM_Bayer_12	Single	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Table 12 and Table 13 show LSB aligned data; it is possible by using a register setting to obtain MSB aligned data.

The data ordering for the Bayer output modes for AP0101AT are shown in Table 13.

Table 13: Bayer Output Modes

Mode	Byte	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Notes
Bayer_12	Single	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	RAW Bayer data

Note: Bayer_12 can be selected by setting cam_mode_select = 0x1.

Sensor Embedded Data

The AP0101AT is capable of passing sensor embedded data in Bayer output mode only. The AP0101AT Statistics are available through the serial interface. Refer to the Developer Guide for details.

Slave Two-Wire Serial Interface

The two-wire slave serial interface bus enables read/write access to control and status registers within the AP0101AT.

The interface protocol uses a master/slave model in which a master controls one or more slave devices.

Protocol

Data transfers on the two-wire serial interface bus are performed by a sequence of low-level protocol elements, as follows:
• a start or restart condition
• a slave address/data direction byte
• a 16-bit register address
• an acknowledge or a no-acknowledge bit
• data bytes
• 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.

The SADDR pin is used to select between two different addresses in case of conflict with another device. If SADDR is LOW, the slave address is 0x90; if SADDR is HIGH, the slave address is 0xBA. See Table 14 below. The user can change the slave address by changing a register value.

Table 14:   Two-Wire Interface ID Address Switching

SADDR	Two-Wire Interface Address ID
0	0x90
1	0xBA

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.

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 AP0101AT are 0x90 (write address) and 0x91 (read address). Alternate slave addresses of 0xBA (write address) and 0xBB (read address) can be selected by asserting the SADDR input signal.

Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data. The protocol used is outside the scope of the two-wire serial interface specification.

Acknowledge Bit

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

No-Acknowledge Bit

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

Stop Condition

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

Typical Operation

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 a WRITE will 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 will then transfer the 8-bit or 16-bit data, as one or two 8-bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. 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, just as in the 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, 8 bits at a time. The master generates an acknowledge bit after each 8-bit transfer. The data transfer is stopped when the master sends a no-acknowledge bit.

Single READ from Random Location

Figure 16 shows the typical READ cycle of the host to the AP0101AT. The first two bytes sent by the host are an internal 16-bit register address. The following 2-byte READ cycle sends the contents of the registers to host.

Figure 16:   Single READ from Random Location
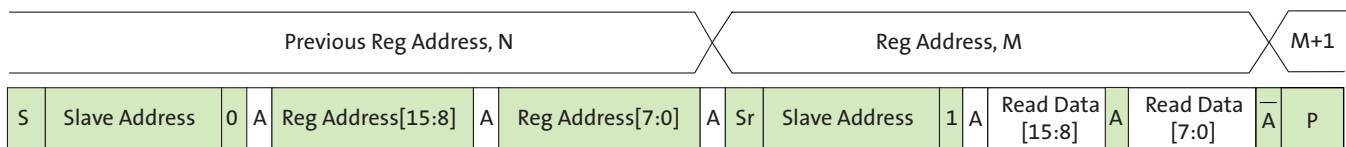
S = start condition
P = stop condition
Sr = restart condition
A = acknowledge
Ā = no-acknowledge

□ slave to master
▨ master to slave

Single READ from Current Location

Figure 17 shows the single READ cycle without writing the address. The internal address will use the previous address value written to the register.

Figure 17:   **Single Read from Current Location**

Sequential READ, Start from Random Location

This sequence (Figure 18) starts in the same way as the single READ from random location (Figure 16 on page 29). 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 18:   **Sequential READ, Start from Random Location**
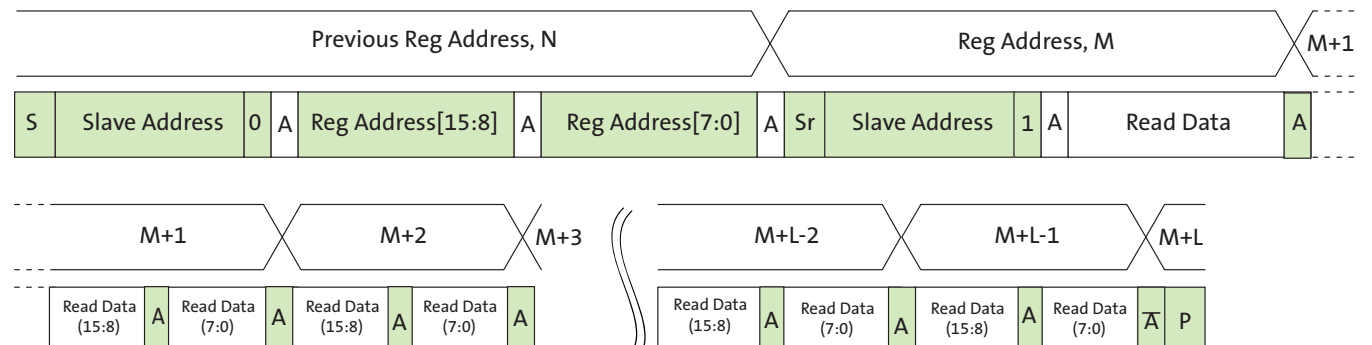
Sequential READ, Start from Current Location

This sequence (Figure 19) starts in the same way as the single READ from current location (Figure 17). 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 19:   **Sequential READ, Start from Current Location**
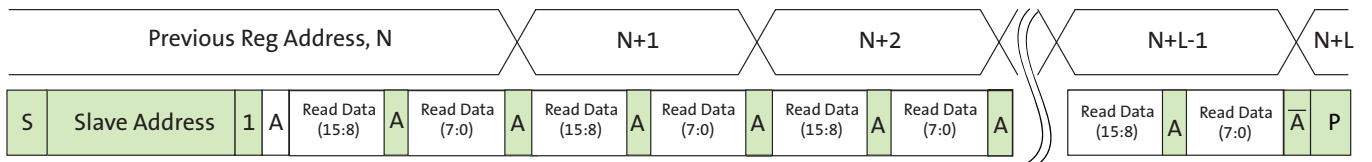
Single Write to Random Location

Figure 20 shows the typical WRITE cycle from the host to the AP0101AT.The first 2 bytes indicate a 16-bit address of the internal registers with most-significant byte first. The following 2 bytes indicate the 16-bit data.

Figure 20:   Single WRITE to Random Location
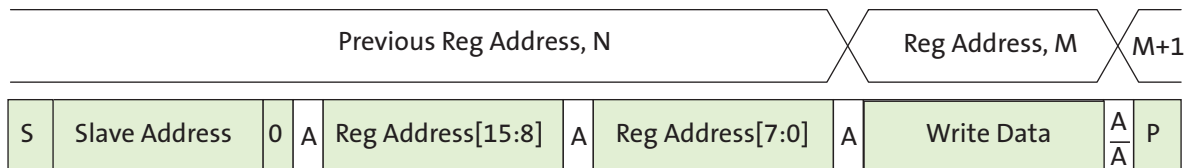
Sequential WRITE, Start at Random Location

This sequence (Figure 21) starts in the same way as the single WRITE to random location (Figure 20). 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 21:   Sequential WRITE, Start at Random Location
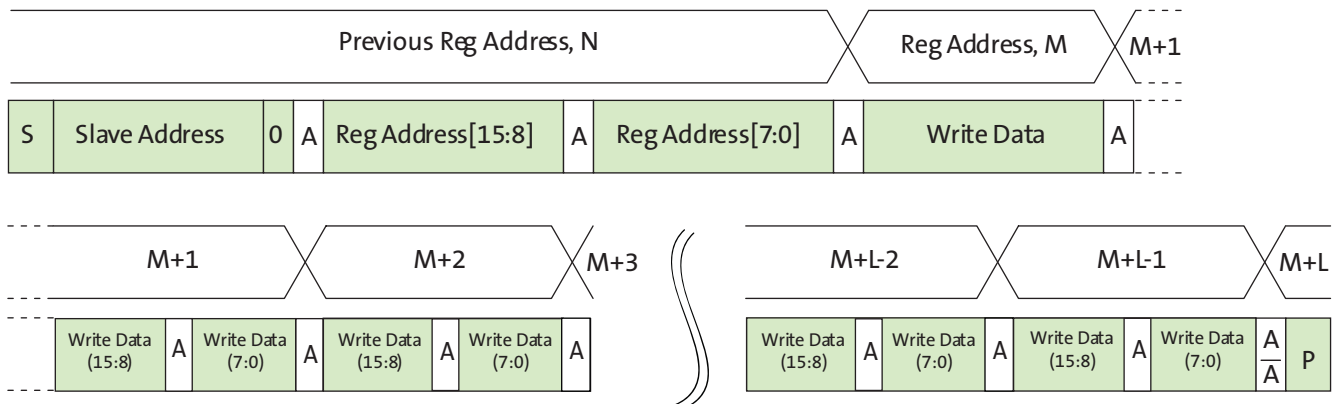
Device Configuration

After power is applied and the device is out of reset (either the power on reset, hard or soft reset), it will enter a boot sequence to configure its operating mode. There are essentially three configuration modes: Flash/EEPROM Config, Auto Config, and Host Config.

The AP0101AT firmware supports a System Configuration phase at start-up. This consists of four sub-phases of execution:

1. Flash detection, then one of:
   a. Flash Config
   b. Auto Config
   c. Host Config

The System Configuration phase is entered immediately following power-up or reset. Then the firmware performs Flash Detection.

Flash Detection attempts to detect the presence of an SPI Flash or EEPROM device:
- If no device is detected, the firmware then samples the SPI_SDI pin state to determine the next mode:
  – If SPI_SDI is low, then it enters the Host-Config mode.
  – If SPI_SDI is high, then it enters the Auto-Config mode.
- If a device is detected, the firmware switches to the Flash-Config mode.

In the Flash-Config mode, the firmware interrogates the device to determine if it contains valid configuration records:
- If no records are detected, then the firmware enters the Auto-Config mode.
- If records are detected, the firmware processes them. By default, when all Flash records are processed the firmware switches to the Host-Config mode. However, the records encoded into the Flash can optionally be used to instruct the firmware to proceed to auto-config, or to start streaming (via a Change-Config).

In the Host-Config mode, the firmware performs no configuration, and remains idle waiting for configuration and commands from the host. The System Configuration phase is effectively complete and the AP0101AT will take no actions until the host issues commands.

In the Auto-Config mode, the part will start streaming with the default settings.

Supported SPI Devices

Table 15 lists supported EEPROM/Flash devices. Devices not compatible will require a firmware patch. Contact Aptina for additional support.

Table 15:   SPI Flash Devices

Manufacturer	Device	Type	Size	Autodetected	ManuID
Atmel	AT26DF081A	Flash	1Mbyte	Yes	0x1f4501
Atmel	AT25DF161	Flash	2Mbyte	Yes	0x1f4602
Sanyo	LE25FW806	Flash	1Mbyte	Yes	0x622662
ST	M25P05A	Flash	64kbyte	Yes	0x202010
ST	M25P16	Flash	2Mbyte	Yes	0x202015
ST	M95040	EEPROM	512byte	No	0x20ffff
ST	M95020	EEPROM	256byte	No	0x20ffff
ST	M95010	EEPROM	128byte	No	0x20ffff
ST	M95M01	EEPROM	128kbyte	No	0x20ffff
Microchip	M25AA080	EEPROM	1kbyte	No	0x29ffff
Microchip	M25LC080	EEPROM	1kbyte	No	0x29ffff

Note: The Sanyo LE25FW806 has been obsoleted.

Host Command Interface

The AP0101AT has a mechanism to write higher level commands, the Host Command Interface (HCI). Once a command has been written through the HCI, it will be executed by on chip firmware and the results are reported back. EEPROM or Flash memory is also available to store commands for later execution.

Figure 22:   Interface Structure
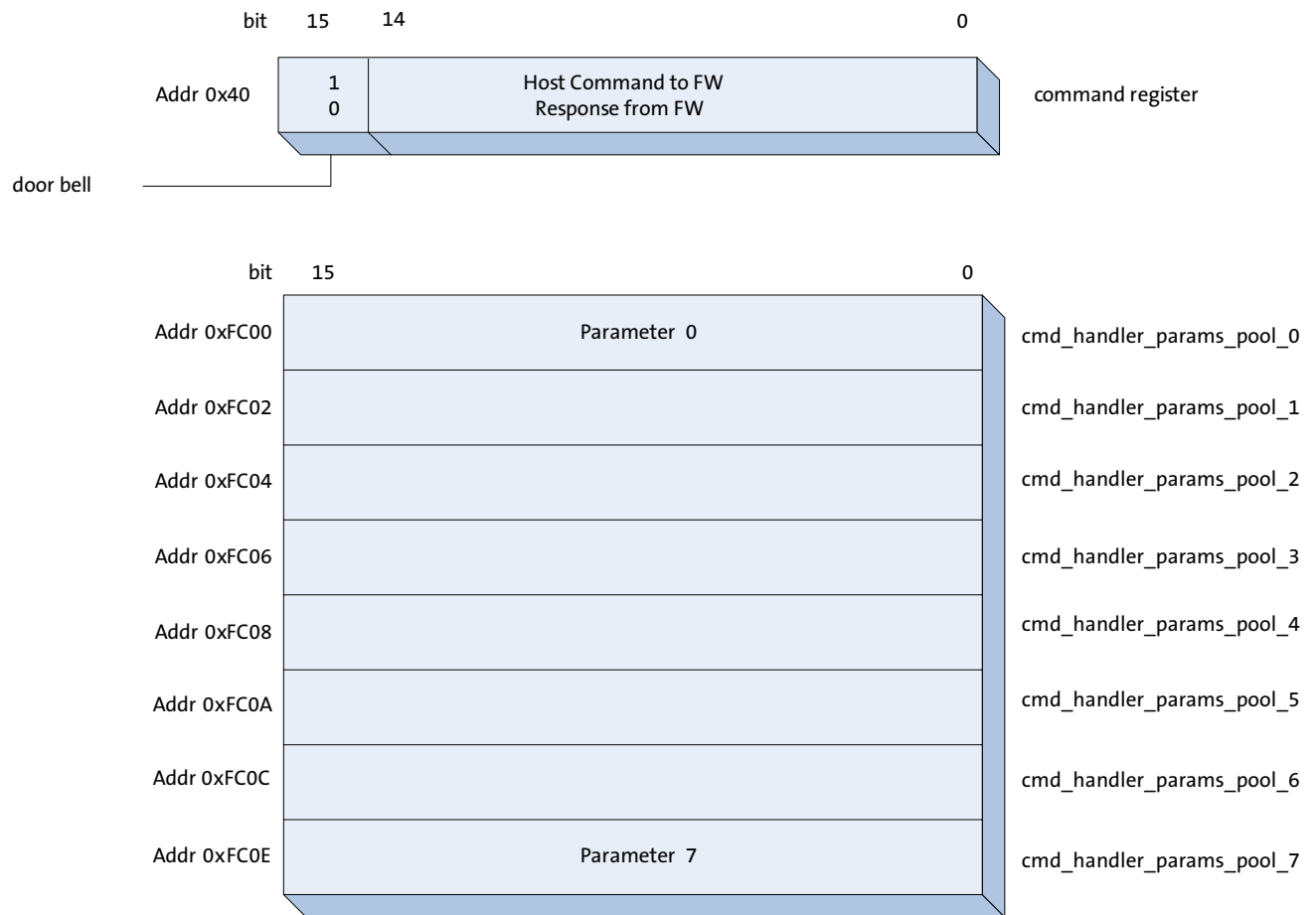
Command Flow

The host issues a command by writing (through the two wire interface) to the Command Register. All commands are encoded with bit 15 set, which automatically initiates processing of the command.

Assuming initial conditions, the host first writes the command parameters (if any) to the Parameters Pool (in the Command Handler's shared-variable page), then writes the command to Command Register.

If the host wishes to determine the outcome of the command, it must poll the Command Register waiting for the doorbell bit to become cleared. This indicates that the command processing is complete. The contents of the Command Register indicate the command's result status. If the command generated response parameters, the host can now retrieve these from the Parameters Pool.

The host must not write to the Parameters Pool, nor issue another command, until the previous command completes. This is true even if the host does not care about the result of the previous command. It is strongly recommended that the host tests that doorbell bit is clear before issuing a command.

Synchronous Command Flow

The typical 'flow' for synchronous commands is:
1. The host issues a 'request' command to perform an operation.
2. The operation is performed.
3. The host retrieves the command result value, and any associated command response parameters.

Asynchronous Command Flow

The typical 'flow' for asynchronous commands is:
1. The host issues a 'request' command to start an operation.
2. If the command was acceptable and the operation initiated, then the command result status will indicate no error.
3. The host retrieves the command return value – if it is not ENOERR the host knows that the command was not accepted and is not in progress.
4. Subsequently, the host issues an appropriate 'get status' command to both poll whether the command has completed, and if so, retrieve any associated response parameters.
5. The host must re-issue the 'get status' command until it does not receive the EBUSY response.

Asynchronous commands exist to allow the Host to issue multiple commands to the various subsystems without having to wait for each command to complete. This prevents the host command interface from being blocked by a long-running command. Therefore, each asynchronous command has a "Get Status" (or similar) command to allow the Host to determine when the asynchronous command completes.

Start-up Host Command Lock-out

The AP0101AT rejects all host commands during the start-up configuration processing with a result status of EBUSY. The time to do this is dependent upon the configuration mechanism. It is recommended that the Host poll the device with the System Manager Get State command until ENOERR is returned.

Once the Host can send serial commands it should perform the following sequence.
1. POLL command_register[15] until it clears (This is called the doorbell bit).
2. Continuously issue the SYSMGR_GET_STATE command (0x8101) until the result status is not EBUSY

Below is some pseudocode that a host could use to implement the above sequence:

```
def systemWaitReadyFollowingReset(numRetries=10):
    """API function: waits for the system to be ready following reset (or powerup)
        - first wait for the doorbell bit to clear - this indicates that the device can
          accept host commands.
        - then wait until the system has completed its configuration phase; the system is
          ready when the SYSMGR_GET_STATE command does not return EBUSY.
        - note the time for the system to be ready is dependent upon the active system
          configuration mode.
        - numRetries is the number of retries before timing-out
        - returns result status code
    """

    # Wait for doorbell bit to clear (indicates device can receive host commands)
    retries = numRetries
```

```
    while (0 != retries):
        if (reg.COMMAND_REGISTER.DOORBELL.uncached_value == 0): break   # ready to receive
commands
        retries -= 1

    if (0 == retries):
        # device failed to respond in time
        return printError(ResultStatus.EIO, 'systemWaitReadyFollowingReset failed (doorbell
failed to clear)')

    # Wait for the System Manager to complete the System Configuration phase
    retries = numRetries
    while (0 != retries):

        res, currentState = sysmgrGetState()

        if (ResultStatus.ENOERR == res): break   # we're done
        if (ResultStatus.EBUSY != res):
            return printError(res, 'systemWaitReadyFollowingReset failed (sysmgrGetState
failed)')

        retries -= 1

    if (0 == retries):
        # device failed to respond in time
        return printError(ResultStatus.EAGAIN, 'systemWaitReadyFollowingReset failed (device
busy)')

    return res
```

Host Commands

Overview

The AP0101AT supports a number of functional modules or processing subsystems. Each module or subsystem exposes commands to the host to control and configure its operation. The intention is that the Host Interface is extensible – commands are not 'hard-coded'. This permits firmware patches to be loaded that offer new (or extended) commands.

Command Parameters

Command parameters are written to the Parameters Pool shared-variables by the host prior to invoking the command. Similarly, any Command Response parameters are also written back to the Parameters Pool by the firmware.

Result Status Codes

Table 16 shows the result status codes that are written by the Command Handler to the Host Command register, in response to a command.

Table 16:   Result Status Codes

Value	Mnemonic	Typical Interpretation – each command may re-interpret
0x00	ENOERR	No error – command was successful
0x01	ENOENT	No such entity
0x02	EINTR	Operation interrupted
0x03	EIO	I/O failure
0x04	E2BIG	Too big
0x05	EBADF	Bad file/handle
0x06	EAGAIN	Would-block, try again
0x07	ENOMEM	Not enough memory/resource
0x08	EACCES	Permission denied
0x09	EBUSY	Entity busy, cannot support operation
0x0A	EEXIST	Entity exists
0x0B	ENODEV	Device not found
0x0C	EINVAL	Invalid argument
0x0D	ENOSPC	no space/resource to complete
0x0E	ERANGE	parameter out-of-range
0x0F	ENOSYS	operation not supported
0x10	EALREADY	already requested/exists

Note: Any unrecognized host commands will be immediately rejected by the Command Handler, with result status code ENOSYS.

Summary of Host Commands

Table 17 on page 38 through Table 23 on page 39 show summaries of the host commands. The commands are divided into the following sections:

- System Manager
- GPIO
- Flash Manager - Flash/EEPROM Non-Volatile Memory
- Sequencer
- Patch Loader
- Miscellaneous
- Event Monitor

Following is a summary of the Host Interface commands. The description gives a quick orientation. The "Type" column shows if it is an asynchronous or synchronous command. For a complete list of all commands including parameters, consult the Host Command Interface Specification document.

Table 17:   System Manager Host Command

System Manager Host Command	Value	Type	Description
Set State	0x8100	Synchronous	Request the system enter a new state
Get State	0x8101	Synchronous	Get the current state of the system

Table 18:   GPIO Host Command

GPIO Host Command	Value	Type	Description
Set GPIO Property	0x8400	Synchronous	Set a property of one or more GPIO pins
Get GPIO Property	0x8401	Synchronous	Retrieve a property of a GPIO pin
Set GPIO State	0x8402	Synchronous	Set the state of a GPO pin or pins
Get GPIO State	0x8403	Synchronous	Get the state of a GPI pin or pins
Set GPI Association	0x8404	Synchronous	Associate a GPI pin state with a Command Sequence stored in SPI NVM
Get GPI Association	0x8405	Synchronous	Retrieve a GPIO pin association

Table 19:   Flash Manager Host Command

Flash Mgr Host Command	Value	Type	Description
Get Lock	0x8500	Asynchronous	Request the Flash Manager access lock
Lock Status	0x8501	Synchronous	Retrieve the status of the access lock request
Release Lock	0x8502	Synchronous	Release the Flash Manager access lock
Config	0x8503	Synchronous	Configure the Flash Manager and underlying SPI NVM subsystem
Read	0x8504	Asynchronous	Read data from the SPI NVM
Write	0x8505	Asynchronous	Write data to the SPI NVM
Erase Block	0x8506	Asynchronous	Erase a block of data from the SPI NVM
Erase Device	0x8507	Asynchronous	Erase the SPI NVM device
Query Device	0x8508	Asynchronous	Query device-specific information
Flash Status	0x8509	Synchronous	Obtain status of current asynchronous operation
Config Device	0x850A	Synchronous	Configure the attached SPI NVM device

Table 20:   Sequencer Host Command

Sequencer Host Command	Value	Type	Description
Refresh	0x8606	Asynchronous	Refresh the automatic image processing algorithm configuration
Refresh Status	0x8607	Synchronous	Retrieve the status of the last Refresh operation

Table 21:   Patch Loader Host Command

Patch Loader Host Command	Value	Type	Description
Load Patch	0x8700	Asynchronous	Load a patch from SPI NVM and automatically apply
Status	0x8701	Synchronous	Get status of an active Load Patch or Apply Patch request
Apply Patch	0x8702	Asynchronous	Apply a patch (already located in Patch RAM)
Reserve RAM	0x8706	Synchronous	Reserve RAM to contain a patch

Table 22:   Miscellaneous Host Command

Miscellaneous Host Command	Value	Type	Description
Invoke Command Seq	0x8900	Synchronous	Invoke a sequence of commands stored in SPI NVM
Config Command Seq Processor	0x8901	Synchronous	Configures the Command Sequence processor
Wait for Event	0x8902	Synchronous	Wait for a system event to be signalled

Table 23:   Event Monitor Host Command

Event Monitor Host Command	Value	Type	Description
Event Monitor Set Association	0x8C00	Synchronous	Associate a system event with a Command Sequence stored in NVM
Event Monitor Get Association	0x8C01	Synchronous	Retrieve an event association

Usage Modes

How a camera based on the AP0101AT will be configured depends on what features are used. In the simplest case, an AP0101AT operating in Auto-Config mode with no customized settings might be sufficient.

In the simplest case no EEPROM or Flash memory or µC is required, as shown in Figure 23.

Figure 23:   Auto-Config Mode

AP0101AT + image sensor

Auto-Config Mode

Digital Out
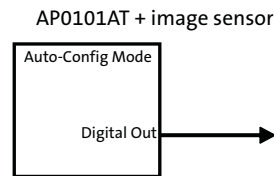
The AP0101AT can be configured by a serial EEPROM or Flash through the SPI Interface.

Figure 24:   Flash Mode

AP0101AT
+ image sensor

Serial
EEPROM/Flash

SPI

Figure 25:   Host Mode with Flash

8/16bit µC

AP0101AT
+ image sensor

Serial
EEPROM/Flash

System Bus

two-wire

SPI

In this configuration all settings are communicated to the AP0101AT and sensor through the microcontroller.

Figure 26:   Host Mode

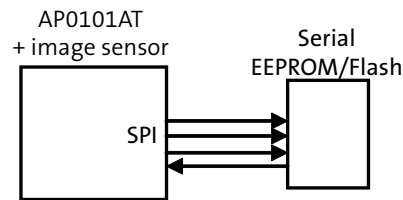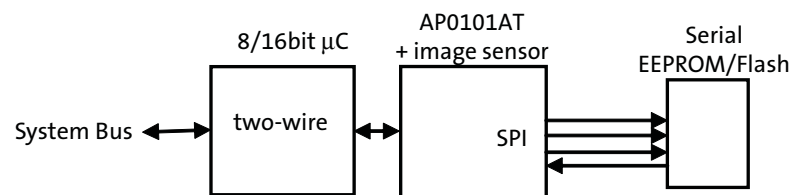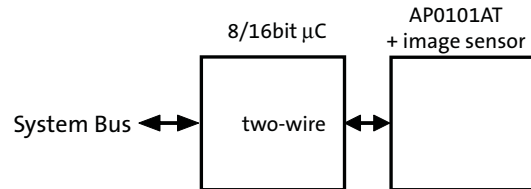**Caution** **Stresses greater than those listed in Table 24 may cause permanent damage to the device. This is a stress rating only, and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.**

Table 24:   Absolute Maximum Ratings

Symbol	Parameter	Rating		Unit
		Min	Max	
V_{DD}_REG	Digital power (1.8V)	-0.3	4.95	V
$V_{DD}IO$_H	Host I/O power (1.8V,2.5V,3.3V)	1.7	5.4	V
$V_{DD}IO$_S	Sensor I/O power (1.8V, 2.8V)	1.7	5.4	V
$V_{DD}IO$_OTPM	OTPM power	2.25	5.4	V
V_{IN}	DC Input Voltage	-0.3	$V_{DD}IO$_*+0.3	V
V_{OUT}	DC Output Voltage	-0.3	$V_{DD}IO$_*+0.3	V
T_{STG}	Storage temperature	-50	150	°C

Table 25:   Electrical Characteristics and Operating Conditions

Parameter	Condition	Min	Typ	Max	Unit
Supply input to on-chip regulator (V_{DD}_REG)		1.62	1.8	1.98	V
Host IO voltage ($V_{DD}IO$_H)		1.7	1.8/2.5/3.3	3.6	V
Sensor IO voltage ($V_{DD}IO$_S)		1.7	1.8/2.8	3.1	V
OTPM power supply ($V_{DD}IO$_OTPM)		2.25	2.5/3.3	3.6	V
Functional operating temperature (ambient - T_A)		-40		105	°C
Storage temperature		-55		150	°C

Figure 27: **Parallel Digital Output I/O Timing**
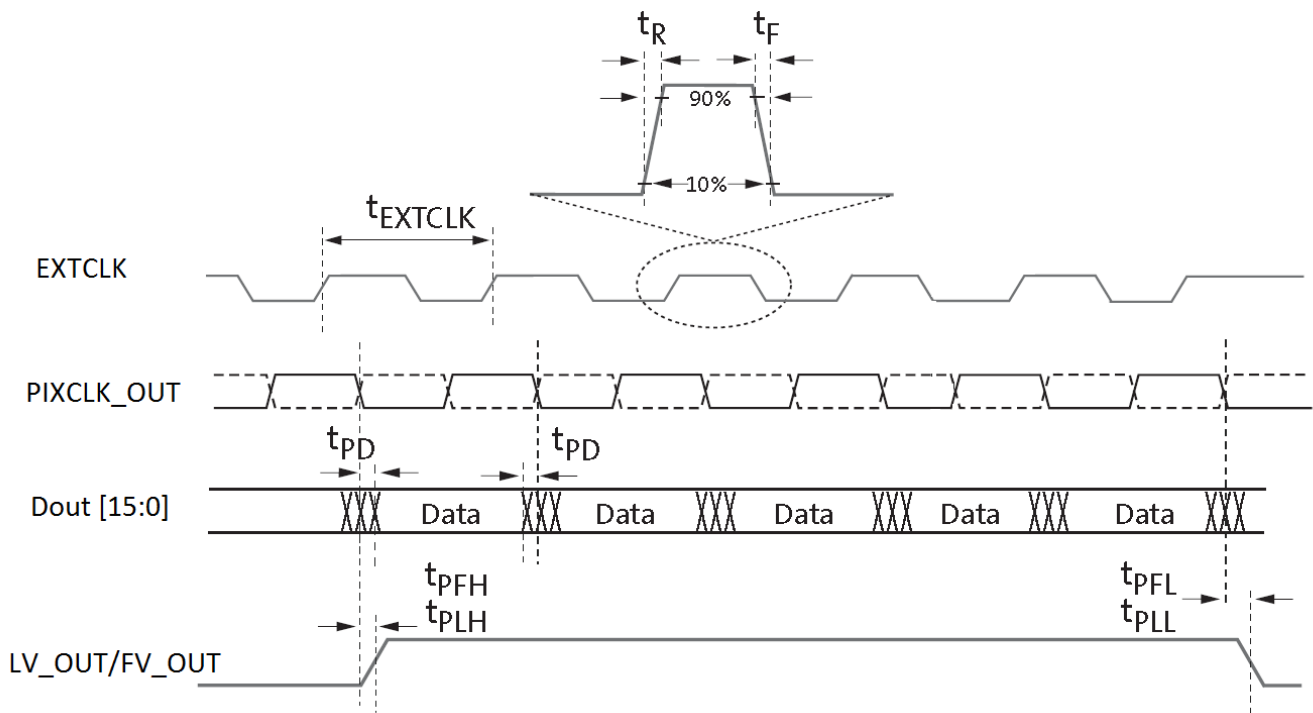
Table 26: **AC Electrical Characteristics**

Default Setup Conditions: f_{EXTCLK}= 27 MHz, f_{PIXCLK} = 74.125 MHz or f_{PIXCLK} = 84 MHz, V_{DDIO}_H = V_{DD}_OTPM = 2.8V, V_{DD}_REG = V_{DDIO}_S = 1.8V, T_A = 25°C unless otherwise stated

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	Notes
f_{EXTCLK}	External clock frequency		6		30	MHz	1
t_R	External input clock rise time	10%-90% V_{DDIO}_H	–	2	5	ns	2
t_F	External input clock fall time	90%-10% V_{DDIO}_H	–	2	5	ns	2
D_{EXTCLK}	External input clock duty cycle		40	50	60	%	
t_{JITTER}	External input clock jitter		–	500	–	ps	
f_{PIXCLK}	Pixel clock frequency (one-clock/pixel)		6		74.125	MHz	
	Pixel clock frequency (two-clocks/pixel)		6		84	MHz	
$t_{RPIXCLK}$	Pixel clock rise time	C_{LOAD}=35pf	–	3	5	ns	
$t_{FPIXCLK}$	Pixel clock fall time	C_{LOAD}=35pf	–	3	5	ns	
t_{PD}	PIXCLK to data valid		–	3	5	ns	
t_{PFH}	PIXCLK to FV HIGH		–	3	5	ns	
t_{PLH}	PIXCLK to LV HIGH		–	3	5	ns	
t_{PFL}	PIXCLK to FV LOW		–	3	5	ns	
t_{PLL}	PIXCLK to LV LOW		–	3	5	ns	

Notes: 1. V_{IH}/V_{IL} restrictions apply.
2. This is applicable only a when the PLL is bypassed. When the PLL is being used then the user should ensure that V_{IH}/V_{IL} is met.

Table 27: **Operating Current Consumption**

Default Setup Conditions: f_{EXTCLK} = 27 MHz, f_{PIXCLK} = as below, VDD_REG=1.8V; VDDIO_H not included in measurement VDDIO_S= 2.8V, VDDIO_OTPM=3.3V, T_A =85°C unless otherwise stated

Symbol	Conditions	Min	Typ	Max	Unit
VDD_REG		1.62	1.8	1.98	V
VDDIO_H	VDDIO_H=1.8V	1.62	1.8	1.98	V
	VDDIO_H=2.5V	2.25	2.5	2.75	V
	VDDIO_H=3.3V	3	3.3	3.6	V
VDDIO_S	VDDIO_S=1.8V	1.7	1.8	1.9	V
	VDDIO_S=2.8V	2.5	2.8	3.1	V
VDDIO_OTPM	VDDIO_OTPM=2.5V	2.25	2.5	2.75	V
	VDDIO_OTPM=3.3V	3	3.3	3.6	V
IDD_REG	960p HDR 30 fps 37.125MHz YCbCr_422_16		42.41		mA
	800p HDR 30 fps 84 MHz YCbCr_422_10_10 or YCbCr_422_8_8		35.83		mA
	720p HDR 60 fps 74.25MHz YCbCr_422_16		64.04		mA
	720p HDR 30 fps 37.125MHz YCbCr_422_16		32.78		mA
	720p HDR 30 fps 74.25 MHz YCbCr_422_10_10 or YCbCr_422_8_8		32.52		mA
IDDIO_S	960p HDR 30 fps 37.125 MHz YCbCr_422_16		4.37		mA
	800p HDR 30 fps 84 MHz YCbCr_422_10_10 or YCbCr_422_8_8		4.3		mA
	720p HDR 60 fps74.25 MHz YCbCr_422_16		4.54		mA
	720p HDR 30 fps 37.125 MHz YCbCr_422_16		4.25		mA
	720p HDR 30 fps 74.25 MHz YCbCr_422_10_10 or YCbCr_422_8_8		4.25		mA
IDDIO_OTPM	960p HDR 30 fps 37.125 MHz YCbCr_422_16		0.25		mA
	800p HDR 30 fps 84 MHz YCbCr_422_10_10 or YCbCr_422_8_8		0.25		mA
	720p HDR 60 fps 74.25 MHz YCbCr_422_16		0.25		mA
	720p HDR 30 fps 37.125 MHz YCbCr_422_16		0.25		mA
	720p HDR 30 fps 74.25 MHz YCbCr_422_10_10 or YCbCr_422_8_8		0.25		mA
Total power consumption[1]	960p HDR 30 fps 37.125 MHz YCbCr_422_16		89.40		mW
	800p HDR 30 fps 84 MHz YCbCr_422_10_10 or YCbCr_422_8_8		77.36		mW
	720p HDR 60 fps 74.25 MHz YCbCr_422_16		128.81		mW
	720p HDR 30 fps 37.125 MHz YCbCr_422_16		71.73		mW
	720p HDR 30 fps 74.25 MHz YCbCr_422_10_10 or YCbCr_422_8_8		71.26		mW

Table 28: **Standby Current Consumption**

f_{EXTCLK} = 27 MHz, VDD_REG =1.8V, VDDIO_S=1.8V, VDDIO_OTPM=VDDIO_H=3.3V, TA = 85°C, excludes VDDIO_H current

Symbol	Parameter	Condition	Typ	Max	Unit
Hard standby	Total standby current when asserting the STANDBY signal		1.63		mA
Standby power			2.93		mW
Soft standby (clock on)	Total standby current	f_{EXTCLK} = 27 MHz	2.10		mA
Standby power			3.80		mW

Table 29:   Inrush Current

Supply	Max. Current
V$_{DD}$_REG (1.8V)	150mA
V$_{DD}$IO_H (2.5/3.3V)	80mA
V$_{DD}$IO_H (1.8V)	110mA
V$_{DD}$IO_S (2.8V/1.8V)	110mA
V$_{DD}$IO_OTPM (2.5/3.3V)	170mA

Figure 28:   Trigger
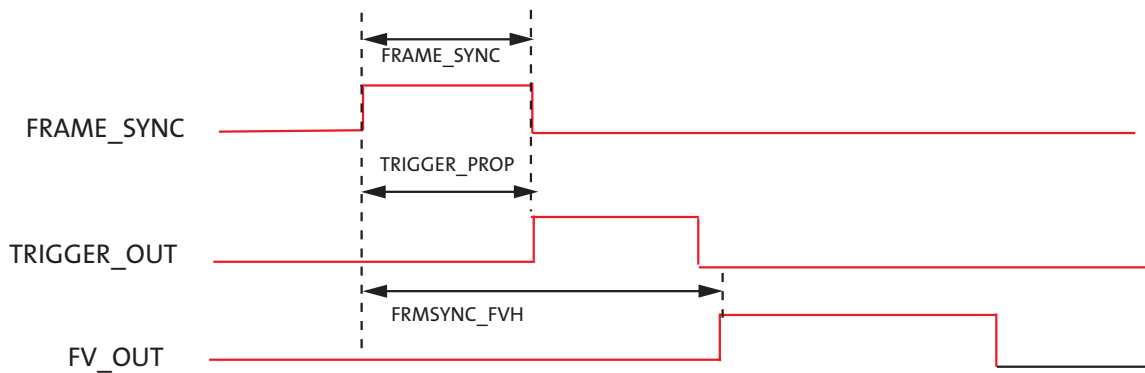
Table 30:   Trigger Timing

Parameter	Name	Conditions	Min	Typ	Max	Unit
FRAME_SYNC to FV_OUT	t$_{FRMSYNC\_FVH}$		8 lines+ exposure time + sensor delay	–	–	Lines
FRAME_SYNC to TRIGGER_OUT	t$_{TRIGGER\_PROP}$		–	–	9	ns
t$_{FRAME\_SYNC}$	t$_{FRAMESYNC}$		3	–	–	EXTCLK cycles

Figure 29:   Reset to AE/AWB Stable Image
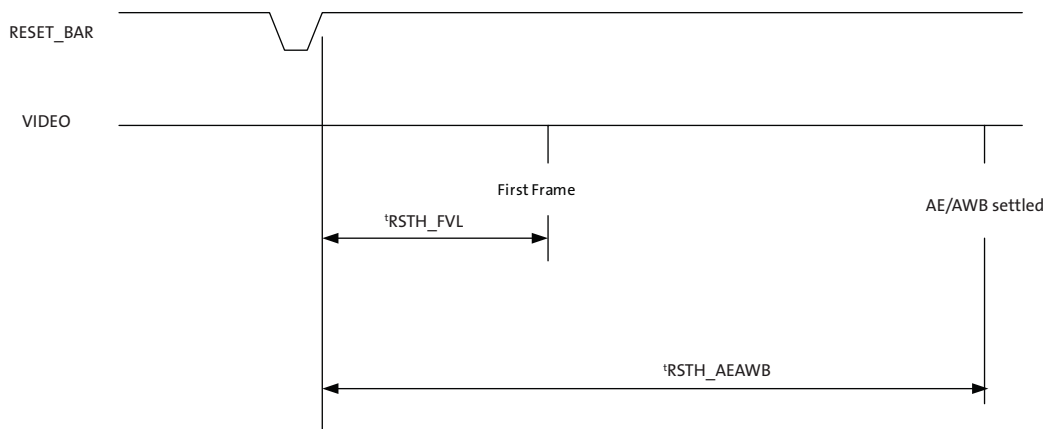
Table 31:   RESET_BAR Delay Parameters

Parameter	Name	Condition	Min	Typ	Max	Unit
RESET_BAR HIGH to FRAME_VALID	t_{RSTH_FVL}		–	–	–	ms
RESET_BAR HIGH to AE/AWB settled	$t_{RSTH\_AEAWB\_Settled}$		–	400	–	ms

Two-Wire Serial Register Interface

The electrical characteristics of the slave two-wire serial register interface (SCLK, SDATA) are shown in Figure 30 and Table 32.

Figure 30:   Slave Two Wire Serial Bus Timing Parameters (CCIS)
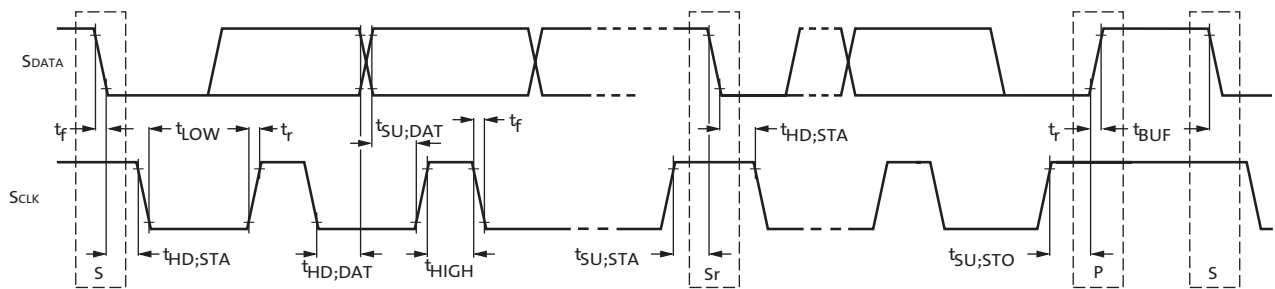
Table 32:   Slave Two-Wire Serial Bus Characteristics (CCIS)

Default Setup Conditions: f_{EXTCLK} = 27 MHz, f_{PIXCLK} = 74.125 MHz, VDDIO_H = VDD_OTPM = 2.8V, VDD_REG = VDDIO_S = 1.8V, Tj = 25°C unless otherwise stated

Parameter	Symbol	Standard-Mode		Fast-Mode		Unit
		Min	Max	Min	Max	
SCLK Clock Frequency	f_{SCL}	0	100	0	400	KHz
Hold time (repeated) START condition.						
After this period, the first clock pulse is generated	$t_{HD;STA}$	4.0	-	0.6	-	µS
LOW period of the SCLK clock	t_{LOW}	4.7	-	1.2	-	µS
HIGH period of the SCLK clock	t_{HIGH}	4.0	-	0.6	-	µS
Set-up time for a repeated START condition	$t_{SU;STA}$	4.7	-	0.6	-	µS
Data hold time	$t_{HD;DAT}$	0^2	3.45^3	0	0.9^3	µS
Data set-up time	$t_{SU;DAT}$	250	-	100	-	nS
Rise time of both SDATA and SCLK signals	t_r	-	1000	20 + 0.1Cb	300	nS
Fall time of both SDATA and SCLK signals	t_f	-	300	$20 + 0.1Cb^4$	300	nS
Set-up time for STOP condition	$t_{SU;STO}$	4.0	-	0.6	-	µS
Bus free time between a STOP and START condition	t_{BUF}	4.7	-	1.3	-	µS
Capacitive load for each bus line	Cb	-	400	-	400	pF
Serial interface input pin capacitance	C_{IN_SI}	-	3.3	-	3.3	pF
SDATA max load capacitance	C_{LOAD_SD}	-	30	-	30	pF
SDATA pull-up resistor	R_{SD}	1.5	4.7	1.5	.7	KΩ

Note: 1. All values referred to VIHmin = 0.9 VDD and VILmax = 0.1VDD levels. Sensor EXCLK = 27 MHz.
2. A device must internally provide a hold time of at least 300 ns for the SDATA signal to bridge the undefined region of the falling edge of SCLK.

3. The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the LOW period (t_{LOW}) of the S$_{CLK}$ signal.
4. Cb = total capacitance of one bus line in pF.

The electrical characteristics of the master two-wire serial register interface (M_S$_{CLK}$, M_S$_{DATA}$) are shown in Figure 31 and Table 33.

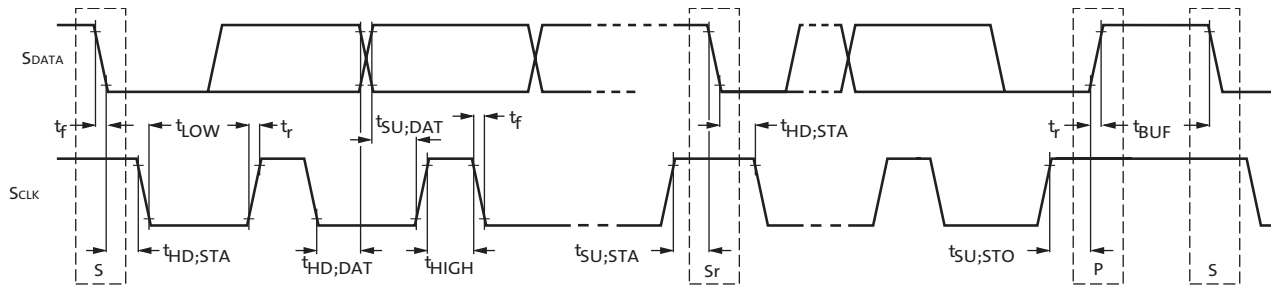**Figure 31:   Master Two Wire Serial Bus Timing Parameters (CCIM)**

**Table 33:   Master Two-Wire Serial Bus Characteristics (CCIM)**
Default Setup Conditions: f$_{EXTCLK}$ = 27 MHz, f$_{PIXCLK}$ = 74.125 MHz, V$_{DDIO}$_H = V$_{DD}$_OTPM = 2.8V, V$_{DD}$_REG = V$_{DDIO}$_S = 1.8V, Tj = 25°C unless otherwise stated

Parameter	Symbol	Standard-Mode		Fast-Mode		Unit
		Min	Max	Min	Max	
M_S$_{CLK}$ Clock Frequency	f$_{SCL}$	0	100	0	400	KHz
Hold time (repeated) START condition.						
After this period, the first clock pulse is generated	t$_{HD;STA}$	4.0	-	0.6	-	μS
LOW period of the M_S$_{CLK}$ clock	t$_{LOW}$	4.7	-	1.3	-	μS
HIGH period of the M_S$_{CLK}$ clock	t$_{HIGH}$	4.0	-	0.6	-	μS
Set-up time for a repeated START condition	t$_{SU;STA}$	4.7	-	0.6	-	μS
Data hold time	t$_{HD;DAT}$	0[2]	3.45[3]	0	0.9[3]	μS
Data set-up time	t$_{SU;DAT}$	250	-	100	-	nS
Rise time of both M_S$_{DATA}$ and M_S$_{CLK}$ signals	t$_r$	-	1000	20 + 0.1Cb	300	nS
Fall time of both M_S$_{DATA}$ and M_S$_{CLK}$ signals	t$_f$	-	300	20 + 0.1Cb[4]	300	nS
Set-up time for STOP condition	t$_{SU;STO}$	4.0	-	0.6	-	μS
Bus free time between a STOP and START condition	t$_{BUF}$	4.7	-	1.3	-	μS
Capacitive load for each bus line	Cb	-	400	-	400	pF
Serial interface input pin capacitance	C$_{IN\ SI}$	-	3.3	-	3.3	pF
M_S$_{DATA}$ max load capacitance	C$_{LOAD\_SD}$	-	30	-	30	pF
M_S$_{DATA}$ pull-up resistor	R$_{SD}$	1.5	4.7	1.5	.7	KΩ

Note: 1. All values referred to VIHmin = 0.9 V$_{DD}$ and VILmax = 0.1V$_{DD}$ levels. Sensor EXCLK = 27 MHz.
2. A device must internally provide a hold time of at least 300 ns for the M_S$_{DATA}$ signal to bridge the undefined region of the falling edge of M_S$_{CLK}$.
3. The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the LOW period (t_{LOW}) of the M_S$_{CLK}$ signal.
4. Cb = total capacitance of one bus line in pF.

Package and Die Options

Figure 32:   Package Diagram
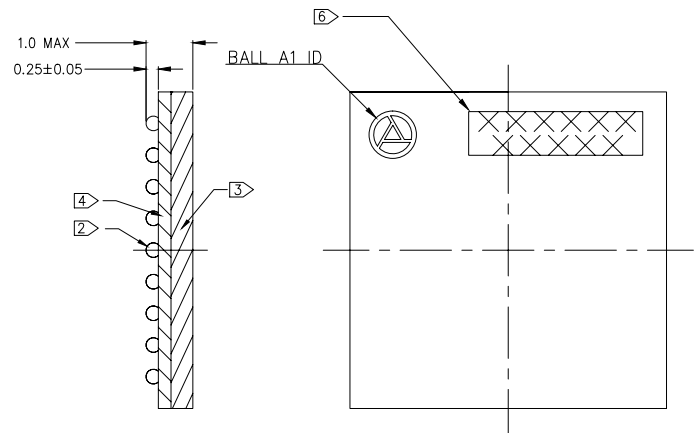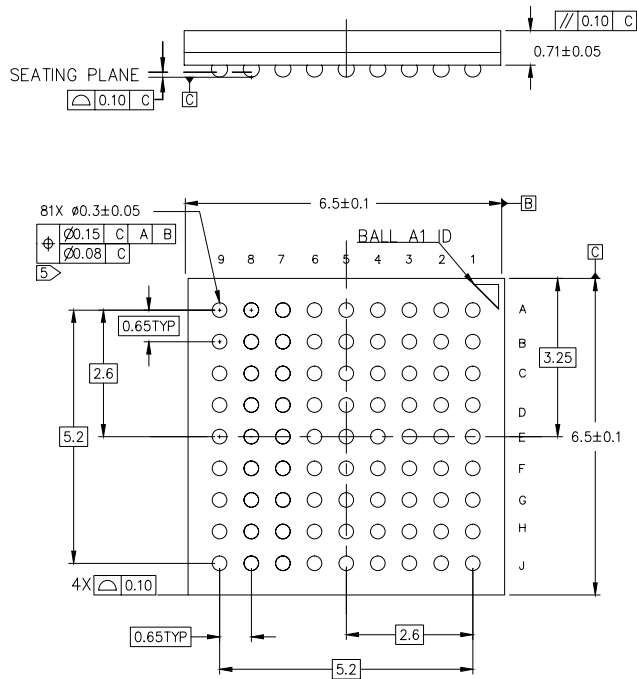
NOTES

1   DIMENSIONS ARE IN MM. DIMENSIONS IN ( ) ARE FOR REFERENCE ONLY
2   SOLDER BALL MATERIAL: SAC305
3   ENCAPSULANT: EPOXY MOLDING COMPOUND
4   SUBSTRATE MATERIAL: EPOXY LAMINATE 0.26 THICKNESS
5   DIMENSIONS APPLY TO SOLDER BALLS POST REFLOW
    PRE-REFLOW BALL IS Ø0.3 ON A Ø0.27 SMD BALL PAD
6   THIS AREA RESERVED FOR LASER MARKING

Revision History

Rev. B **9/20/12**

- Updated to Production
- Updated "Applications" on page 1
- Updated Table 2, "Key Performance Parameters," on page 1
- Updated description of D_OUT in Table 3, "Pin Descriptions," on page 8
- Updated Figure 6: "Hard Standby Operation," on page 13
- Updated "Multi-Camera Synchronization Support" on page 14
- Updated "Image Flow Processor" on page 15
- Updated Figure 9: "AP0101AT IFP," on page 15
- Updated "AE Track Driver" on page 20
- Added "Dual Band IRCF" on page 21
- Updated "Flicker Avoidance" on page 22
- Updated "Output Formatting" on page 22
- Replaced the table "YCbCr Output Modes" with Table 9, YCbCr Output Modes (cam_port_parallel_msb_align=0x1) and Table 10, "YCbCr Output Modes (cam_port_parallel_msb_align=0x0)," on page 22
- Added "SMPTE Output" on page 26 with Table 11, "SMPTE Output Mode," on page 26
- Added "ALTM Bayer Output" on page 27
- Updated "Typical Operation" on page 29
- Updated "Summary of Host Commands" on page 38
- Updated Table 19, "Flash Manager Host Command," on page 39
- Updated Table 25, "Electrical Characteristics and Operating Conditions," on page 41
- Updated Table 27, "Operating Current Consumption," on page 43
- Updated Table 28, "Standby Current Consumption," on page 43
- Updated Table 30, "Trigger Timing," on page 44
- Updated "Two-Wire Serial Register Interface" on page 45
- Updated Figure 27: "Parallel Digital Output I/O Timing," on page 42

Rev. A **7/30/12**

- Initial release