Advanced Image Coprocessor for onsemi's Sensors

6.5 mm x 6.5 mm Package Data Sheet

AP1302

ORDERING INFORMATION

Table 1. AVAILABLE PART NUMBERS

Part Number	Description
AP1302CSSL00SMGA0	6.5x 6.5 x 0.8(mm) VFBGA Package Part
AP1302CSSL00SMGAH−E	Product demo board
AP1302CSSL00SMGAD3−E	Demo kit

Table 2. KEY PERFORMANCE PARAMETERS

Parameter		Value
Primary camera interface		Up to 4−lane MIPI (up to 1.2 Gbps/lane)
Primary camera input format		RAW6, RAW8, RAW10, RAW12
Output interface		Up to 4−lane MIPI (up to 1.2 Gbps/lane)
Output format		YUV422, YUV420, 888RGB, 565RGB, 555RGB, JPEG, RAW8, RAW10, RAW12, MJPEG
Secondary camera interface		Up to 3−lane MIPI (up to 1.2 Gbps/lane)
Secondary camera input format		RAW6, RAW8, RAW10, RAW12
Maximum resolution		4224x3156 (13 Mp)
Maximum frame rate		30 fps at 13Mp, 120 fps at 1080p
Maximum output clock frequency		MIPI clock up to 600 MHz (1200 Mbps DDR)
Maximum color processing frequency		450 Mpixels per second
Supply voltage	CORE	1.2V nominal ± 5%
	I/O	1.8V nominal ± 10%
	PLL	1.2V nominal ± 5%
	MIPI	1.2V ± 5%
Operating temperature		−30°C to +70°C (ambient) −30°C to +85°C (junction)
Process		55nm
ESD Susceptibility		2000V HBM

Features

- Optimized for operation with **onsemi**'s Clarity+™ Pixel technology sensors as well as Bayer pattern CMOS Image Sensors (CIS)
- Up to 13 Mp (4224x3156) sensor support
- Designed to support **onsemi** sensors that provide video interlaced HDR (iHDR).
- 450 Mp/second processing (13Mp at 30/1080p at 120/720p at 240 – subject to sensor limitations

Features

- Color gaining and gamma correction
- Frame rate reduction, image resizing and clipping
- Auto exposure, auto white balance, auto focus, auto flicker detection and mitigation
- Adaptive Tone Mapping, Local Tone Mapping
- Face detection
- Smooth digital zoom and panning
- OSD, special effects
- Test pattern generator
- Programmable JPEG encoder with EXIF header support
- SpeedTags™ Encode support
- Two−wire serial interface (I²C) for sensor and peripheral control and register access supporting Standard (100kbps), Fast mode (400kbps), FM+ (1Mbps) and HS (3.4 Mbps)
- Dual on−chip 32−bit RISC processor cores
- Dual sensor support (second camera or 3D bridge application)
- 6 MIPI data lanes shared between two sensor interfaces (4+2/ 3+3)
- Four−wire serial interface (SPI slave) for register access supporting up to 25 Mbps
- 12 GPIOs (shared functionality with SPI master and slave, second I²C master)
- Fail−safe IO, programmable slew−rate control

Applications

- IoT
- Drones
- Machine vision
- AR/VR/MR
- Video conferencing

Note: onsemi will only support its own image sensors on this device.

GENERAL DESCRIPTION

The **onsemi** AP1302 is a high−performance, ultra−low power in−line, digital image and video stream processor with advanced DSC and DVC features, supporting image resolutions up to 13 megapixels (4224x3156) and frame rates up to 13 Mp at 30 fps, 1080p at 120 fps, and 720p at 240 fps. AP1302 supports various advanced features such as face detection. It enables full auto− functions support (AWB, AE, AF) to produce DSC quality images. The AP1302 supports both sub−LVDS (MIPI) sensor (input) and host (output) interface.

AP1302 interfaces to CMOS imaging sensors and performs all the necessary operations required to capture state−of−the−art 13 Mp images including JPEG compression, capture 1080p video streams, and preview generation. AP1302 is optimized for operation with AR1335 and other **onsemi** sensors with MIPI interface (contact local **onsemi** FAE for details).

AP1302 is designed to support **onsemi**'s innovative and advanced sensors with Clarity+™ Pixel technology, which increases the low−light sensitivity and performance of the camera system. AP1302 is also designed to support **onsemi**'s sensors that provide video interlaced HDR (iHDR) capability (i.e.. AR1335), allowing capture of high dynamic range scenes at video recording speed.

An on−chip RISC processor and ROM allow it to be operated with minimal external control. All advanced features are supported with on−chip memory and do not require any external full or partial frame buffer.

FUNCTIONAL OVERVIEW

Figure 1 on page 2 shows the typical configuration of the AP1302 in a camera module. The AP1302 captures the data from primary or secondary sensor (one at a time) through two MIPI receivers, processes the image, and sends the processed data to the host. The AP1302 uses sensor interface 0 to connect to the primary sensor. Sensor interface 0 can use up to four high−speed sub− LVDS MIPI data lanes capable of transfer speed up to 1.2 Gbps each. The second optional sensor is connected through sensor interface 1 using up to three sub−LVDS MIPI lanes. The I²C master interface is used to control the primary and secondary sensor operation.

The host uses I²C (Standard, Fast Mode, FM+ and HS supported) or SPI Slave (up to 48 Mbps) interfaces to access the AP1302 internal registers and control the operation of the AP1302. Image data is transferred using four MIPI data lanes between the AP1302 and the host processor.

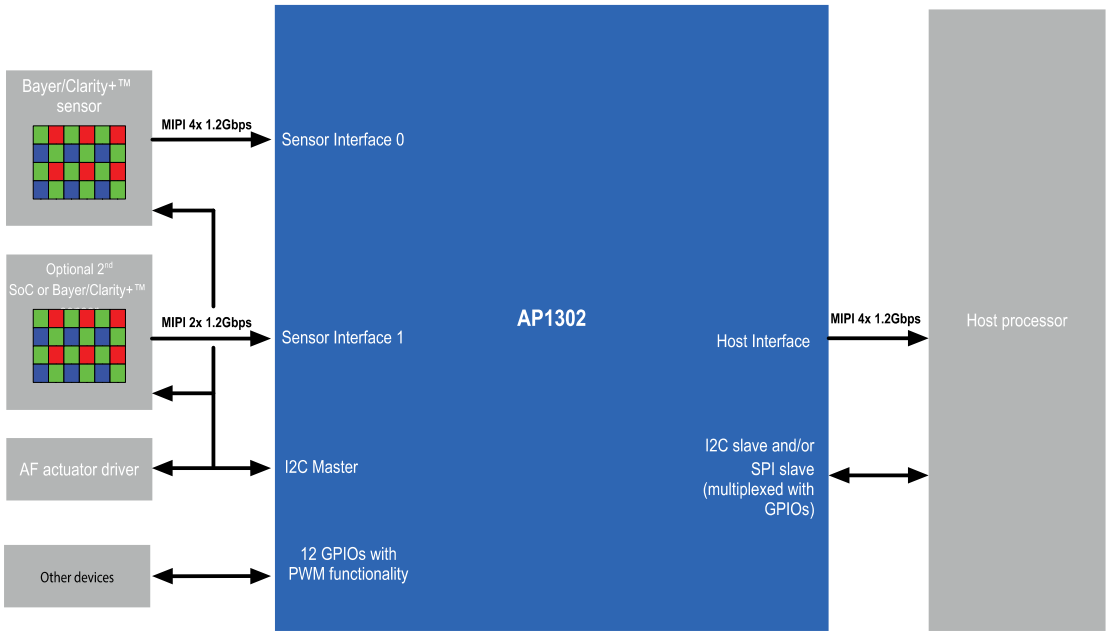


Figure 1. AP1302 Connectivity

Imaging sensors are controlled through the I²C Master, which could also be used to control various types of auto−focus (VCM) drivers. There are also 12 GPIO signals that can be used to control other optional devices. Please contact local FAE support for list of supported devices and possibility to add support for new devices.

Figure 2 on page 3 shows the 3D bridging application connectivity. In this application, the AP1302 is connected to two identical sensors processing two independent streams simultaneously and sending processed data to the host. The Max width of each sensor is 2112 pixels. The first sensor is connected to sensor interface 0 using up to 4 MIPI lanes and is used for single sensor snapshot or as a left sensor for 3D bridging application. The second sensor is only used as a right sensor for 3D bridging application. Note that the AP1302 only does the processing of two image streams from two sensors and does not merge the streams into one 3D video stream. 3D merging is done on the host; the AP1302 only performs bridging function. Two I²C master interfaces are used to control the two sensors in the 3D bringing

application. Both sensors need to be of the same type. The AP1302 will issue exactly same commands and send them synchronously via two I²C interfaces to the sensors. In this way the operation of the two sensors will be kept synchronous up to ±1 line.

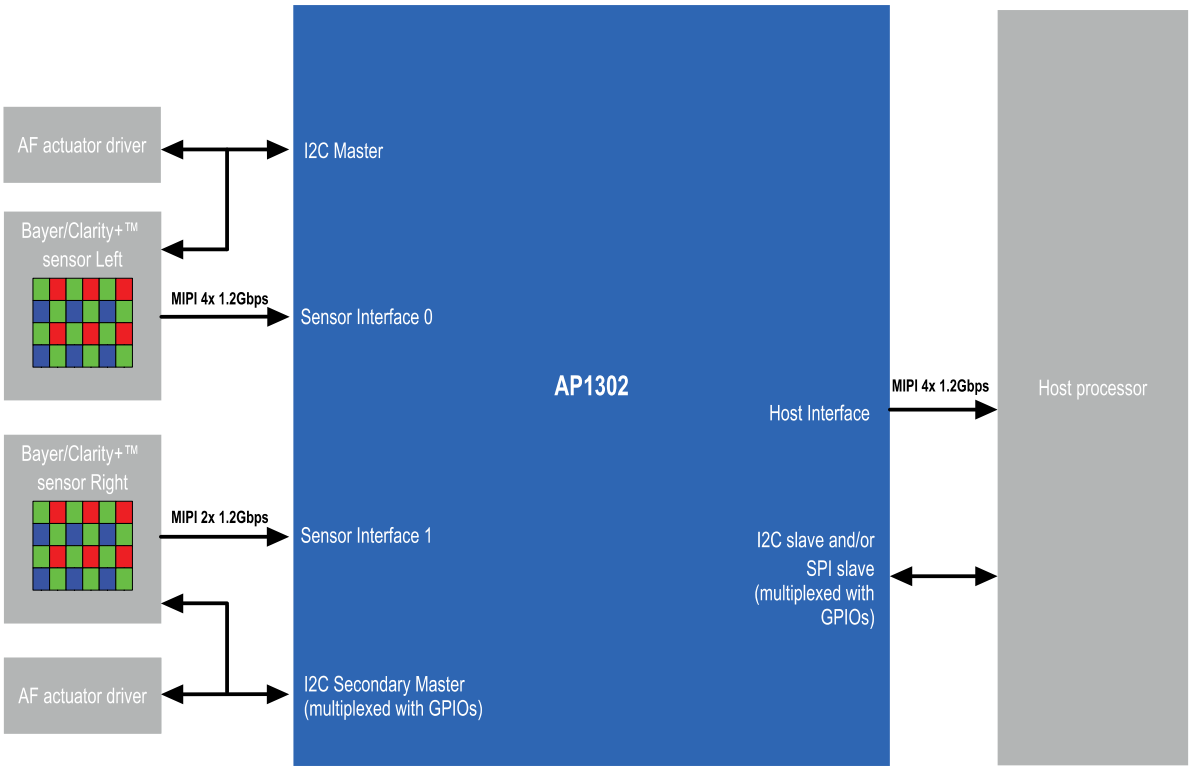


Figure 2. AP1302 3D−Assist Camera Connectivity

Embedded 32−bit CPU is used to run the automatic functions and system control implemented as firmware executed from on−chip ROM and/or RAM. Automatic functions are used for determination of proper exposure (AE − Auto Exposure), determination of optimal dynamic range mapping function (ATM − Adaptive Tone Mapping), detection and correction of the light source illuminant (AWB − Auto White Balance); detection of focal distance and the lens system control (AF − Auto Focus) and detection and compensation of flickering light source (FD − Flicker Detection). These auto functions are implemented in firmware running on the 32−bit CPU. System control firmware is used to control the hardware and system resources and provide an interface to the host processor for the configuration and control of the chip.

Advanced features such as face detection are executed using a combination of hardware and programmable engine to maximize performance and flexibility. Image data and statistics required are stored on−chip. Production calibration data can be stored in one−time programmable memory (OTPM) inside the sensor.

FUNCTIONAL DESCRIPTION

Figure 3 shows the internal structure of the AP1302:

- Image Processing sub−system
- Statistics engine
- Control processor subsystem
- Advanced features processor subsystem
- Sensor interface
- Test pattern generator
- Host interface
- Clock and Control
- Production test and debug

The image processing and image enhancement core of the design includes the image preprocessing, color processing and pixel processing engines, the statistics gathering engine, and the JPEG encode engine.

The advanced feature processor includes the 32−bit processor, as well as its associated program and data memories. The advanced feature processor is part of the chip data path and can operate on and manipulate pixel data.

The processor subsystem includes a 32−bit processor to perform all imaging sensor control, control of external focus and zoom actuators, all on−chip real−time control, and to present a simplified API to the host interface in form of a basic register set. All program code for the processor, as well as the settings for imaging sensors and actuators is stored on−chip ROM memory. The processor subsystem also includes a scratch pad RAM.

AP1302 supports sub−LVDS data interfaces to the sensor and the host. I²C master (also referred to as SIPM) is used to control the sensor and the AF driver while I²C slave (also referred to SIPS) allows for host control of AP1302. The I²C master interface may be optionally used to interface to an external EEPROM device. This device has 12 ports that may be configured as GPIO and can be used to implement PWM, to control mechanical shutter, zoom or the secondary sensor. All IO cells are fail−safe.

An on−chip, low−jitter PLL with input clock of 10 MHz – 54 MHz and a combination of clock dividers is used to generate all the clock signals and frequencies required for the chip core and interfaces. The core of the design is completely synchronous and operates as a single domain. Control circuitry and gating is employed as appropriate to control switching and therefore power dissipation. The maximum operational frequency of the device is 450/2 = 225 MHz.

AP1302 uses an external 1.8 V supply for its interfaces and a 1.2 V (nominal) external supply for the core.

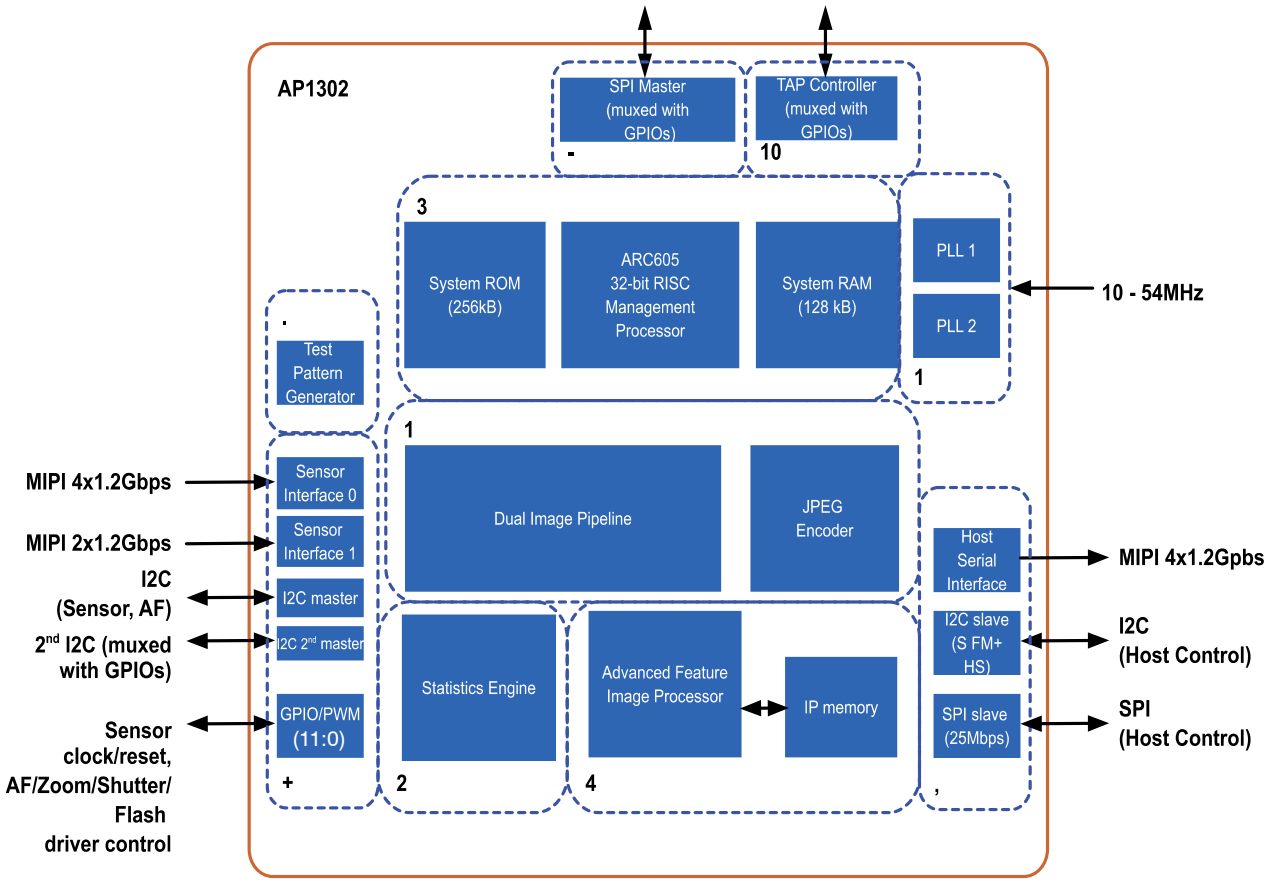


Figure 3. AP1302 Internal Block Diagram

Image Processing Pipeline and Image Statistics Engine

The image processing pipeline includes:

- CF – Color Processing: black level correction, RGB gains, lens shading and color shading correction, GrGb global correction
- BPC – Bad Pixel Correction: bad pixel correction and Bayer de−noise engine (two 5x5 kernels one for each T1/T2 iHDR plane)
- HDR – Interleaved High Dynamic Range: reconstruction of high dynamic range image (14 bits) out of two 10bit planes, one with long exposure time (T1) and one for short exposure time (T2), motion compensation module
- PP – Pixel Processor: GrGb local correction, de−mosaic, luminance de−noise and sharpening (7x7 kernel)
- Denoise/Sharpening: de−noise and sharpening in RGB/RCB domain (28x28 kernel for chrominance, 7x7 for luminance), chroma coring, anti−fringing
- PM – Pixel Mixer: flare removal, color correction, gamut compression, tone mapping
- Gamma: very precise gamma (exponential) engine
- Curves: post gamma S−curve LUT (32 points)
- SFX – Special Effects: special effects module for color transformations (like sepia) and edge based transformations (like emboss)

- PCR – Preferred Color Rendering: preferred color rendering
- Resample: scale up using bilinear interpolation and/or scale down using averaging algorithm. Max output size limitation:
  - Resample 0: 4224x3156
  - Resample 1: 2304x1296
  - Resample 2: 428x240
- CCONV – Color Conversion: conversion of spaces from YUV to RGB or JFIF version of YUV suitable for JPEG compression
- CF – Color Formatting: Formatting the output stream, RGB565, RGB555, RGB444, YUV422, different orders of color components

The image processing pipe including the JPEG engine is duplicated, which means that there are two instances of each processing block. Incoming frame is divided into left and right halves with 16 pixels of overlap. Each instance up to the JPEG engine does the processing of one half of the image. The image halves are then merged and the merged image is sent to the host interface directly or compressed in the JPEG engine. In case JPEG compression is used, the image is first reordered to form JPEG MCU blocks. All even blocks are processed by one JPEG instance and all odd blocks are processed by the other JPEG instance. Again the outputs from the two JPEG instances are merged before the JPEG stream is sent to host interface.

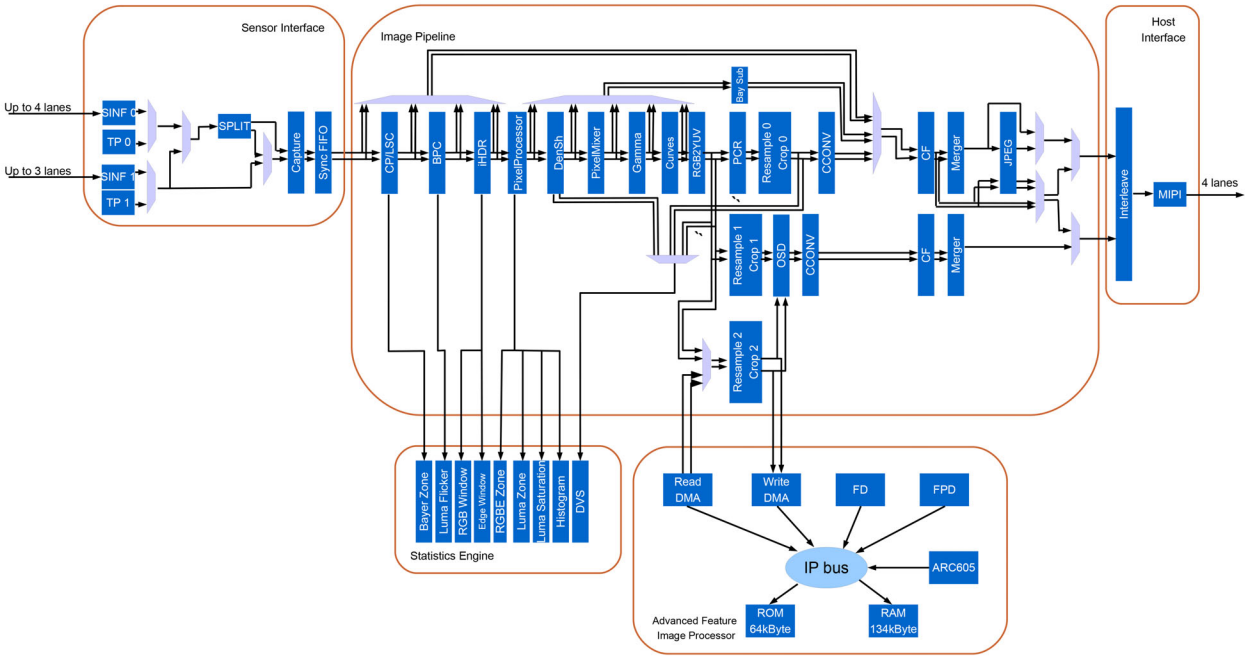


Figure 4. AP1302 Image Pipeline Block Diagram

Max frequency of the IPIPE engine and JPEG is 225 MHz, which means that maximum pixel rate into the IPIPE is 450 Mpix/sec (2 instances of the IPIPE working in parallel). Each block in the IPIPE has an acknowledge signal which can be used to stall the incoming stream of pixels coming from the block in front. Similarly each block in the IPIPE can be stalled by the block positioned later in the IPIPE. Blocks have different delays, blocks which has to form the kernels of pixels contains memory for holding several lines of the image (called line buffers) and this way the delay of such block can be several lines. The blocks up to resample blocks do not affect the data rate.

Resample then can reduce or increase the pixel rate on its output (depending whether it is operating in scale down or scale up mode). If operating in scale up mode and is the input pixel stream is at its maximum for a given frequency (meaning that every clock a pixel is received on the resample input), resample will then lower the acknowledge signal and this way block the incoming stream to be able to output the generated pixels. As a consequence, the Sync FIFO module will have to buffer the data which are coming from the sensor. Sync FIFO has capability to buffer one line of data.

This effectively means that it can buffer the data over one line time (which is measured from start on the line to the start of the next one). If sync FIFO gets overflowed, this is a recoverable error and it is reported in warning register (this means that ISP will still be running, but image is corrupted).

Another such bottleneck is the host MIPI interface, which can run at a different clock frequency than IPIPE and this

way the pixel rate can be lower (or higher, but this will not cause any issues in terms of data flow).

At the end of the image processing pipe there are three resample and crop engines. Resample engine 0 is used to produce primary (main) output image (snapshot or video image), resample 1 is used to produce secondary (auxiliary) output image (preview, postview, bubble or video image) and resample 2 is used to produce image for Advanced Image Processor.

Black Level (BL)

Sensors typically provide predefined data pedestal for zero signal, to prevent noise clipping. AP1302 can subtract that black level and optionally provide noise filtering (on negative data) to prevent clipping. Optionally, black level can vary based on the scene conditions, for example, a lower black level for low light conditions.

Lens Shading Correction

Lens vignetting and optical crosstalk can produce reduced signal levels near borders and/or corners of the image. The AP1302 lens shading correction (LSC) module generates a gain map for compensation of lens vignetting (shading) and pixel crosstalk. Four gain maps are generated, one for each CFA channel.

The gain map curve is obtained using interpolation between reference points−vertices. These are equidistantly placed along the horizontal and equidistantly along the vertical axis, forming an *n x m* vertex matrix. The *n* and *m* are adjustable and the maximum number of vertices is determined by hardware resources; typical setting would be: *n = 7, m = 7*. Some of the vertices fall outside of the image coordinates and can be adjusted to tune the corner gain map.

Bad Pixel Correction

To increase sensor production yield, certain defective pixels are tolerated. The bad pixel correction block uses a 5 x 5 kernel and is able to remove the following defects:

- Single physical pixels inside 3 x 3 physical (and limited amount of bad pixels inside 9 x 9 area)
- Single pixels in same color channel inside 5 x 5 physical area
- Double pixels in same color channel inside 5 x 5 physical area

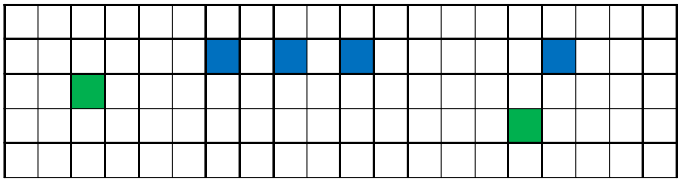


Figure 5. Single Physical Bad Pixel Defect Examples

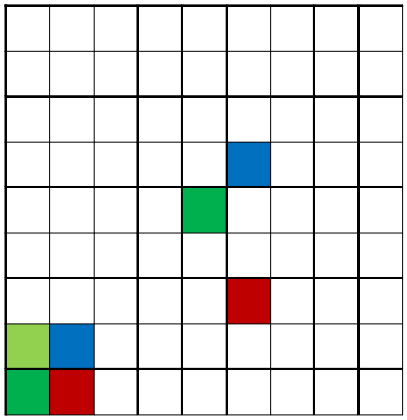


Figure 6. Single Same Color Channel Bad Pixel Defect Examples

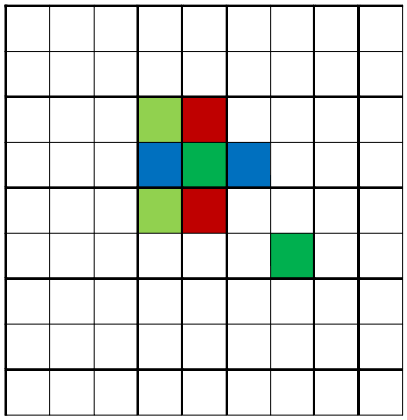


Figure 7. Double Same Color Channel Bad Pixel Defect Example

AP1302 provides several methods for defining the defective pixels:

- Automatic detection
- Single physical defects
- Single Bayer defects
- Double Bayer defects

Bad pixel detection strength is controlled by absolute and relative pixel value threshold along with bounded flat field detection. Furthermore detection of hot and cold defective pixels can be biased. All of these parameters can be adjusted based on the light level.

The detection is sensitive to gradients and can be configured to detect various natural image detail, including highlights. Detection and correction of single physical pixels is generally better than that for those in Bayer plane.

Pixel Processor

Due to crosstalk and other effects, green pixels in red and green pixels in blue rows can slightly differ. This step compensates for these imbalances locally, using 7 x 7 kernel. The algorithm is similar to flat field correction, but the AP1302 algorithm is also capable of compensating for the

GR−GB disparities efficiently on edges and high resolution detail.

Demosaicing

Demosaicing is one of the most important steps for achieving the high−quality images.

Due to the sensor's physical limitations, only one color channel is captured at the same pixel location. In almost all sensors today, the image input data is captured in Bayer color format:

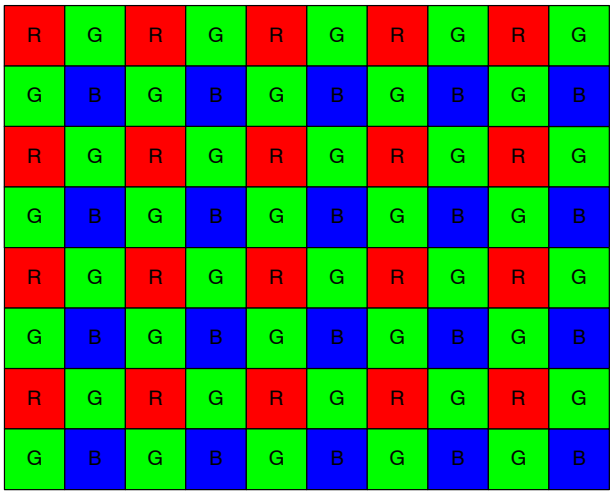


Figure 8. Bayer Array (CFA)

AP1302 uses a 7 x 7 region to interpolate the missing colors. The algorithm tries to preserve all the edges and details and can be tuned to handle various module inefficiencies, which could influence the demosaicing (crosstalk, chromatic aberration, and so on).

Denoise

The amount of random noise increases with shorter exposure times and higher analog or digital gains as well as module form factor. Effective noise removal is required to render visually pleasant images.

AP1302 has implemented the following denoise schemes:

- Bayer filtering (to remove sensor readout related noise and artifacts) occurs during BPC correction and is therefore named BPCDEN; it uses a 5 x 5 kernel. Use parameters starting with BPCDEN_ and BPCTRI_ to tune the module.
- Edge−adaptive multifrequency luminance sigma filter (to control the edge SFR). Uses 7 x 7 kernel. Use parameters starting with PM_ to tune the module.
- Luminance Edge detection and reconstruction in the denoise/sharpening − (DENSH) module. Uses 7 x 7 kernel.
- Edge−adaptive multifrequency chrominance filter. Uses 28 x 28 kernel. Use parameters starting with PM_ to tune the module.
- High performance edge−adaptive chroma filtering (to remove color random noise). Use DENSH_ parameters to tune the module. Uses 7 x 7 kernel.

White Balance

White balance adjust white point to match the sRGB white point.

Sharpening

The lens system together with crosstalk modify the edge SFR, visually reducing the sharpness. There are two blocks that try to compensate for such effects:

- Edge−adaptive multifrequency chrominance filter. Uses 28 x 28 kernel. Use parameters starting with PM_ to tune the module.
- High performance edge−adaptive chroma filtering. Use DENSH_ parameters to tune the module. Uses 5 x 5 kernel.

Illumination Based Color Correction and Saturation Control

This block ensures correct color rendering for the given scene using 3 x 3 illuminant−dependent color correction matrix. The dark areas and highlights are carefully rendered to provide visually pleasant saturated areas.

Tone Mapping

Tone mapping implements several techniques to capture and map the high dynamic range scenes to display:

- ATM (adaptive tone mapping) adjusts the tone mapping curve according to the scene
- LTM (local tone mapping) locally adjusts contrast in order to provide more details in dark and bright areas

Gamma Correction

This block converts the color space by adjusting the gamma in very high precision. Typically the conversion is done to sRGB color space.

Curves

To ensure custom gamma, tuning curves can be applied independently to each color channel. This block can also bypass gamma correction and provide 32 configurable (either equidistant or with higher density in low−key) interpolation points.

Preferred Colors

This block can remap colors based on a custom color map. This is especially useful for mapping colors like sky and foliage to their memory color locations.

Color Conversion

AP1302 uses 3x3 configurable color conversion matrix to output the image in the desired format. By default the following output formats are supported:

- sRGB (identity)
- JFIF YCbCr
- CCIR601 YCbCr

Resampler and Crop

This block will crop the input image so as to match the proper aspect ratio and settings. It will as well perform the appropriate downscaling or upscaling so as to match the target output size.

When upscaling this block will output more data than what is coming in, therefore must properly set the frame rate to prevent any overflow.

There are three instances of this block: two for primary and secondary image streams and additional one for advanced features block (e.g. face detection). Figure 9 on page 9 shows sample usages:

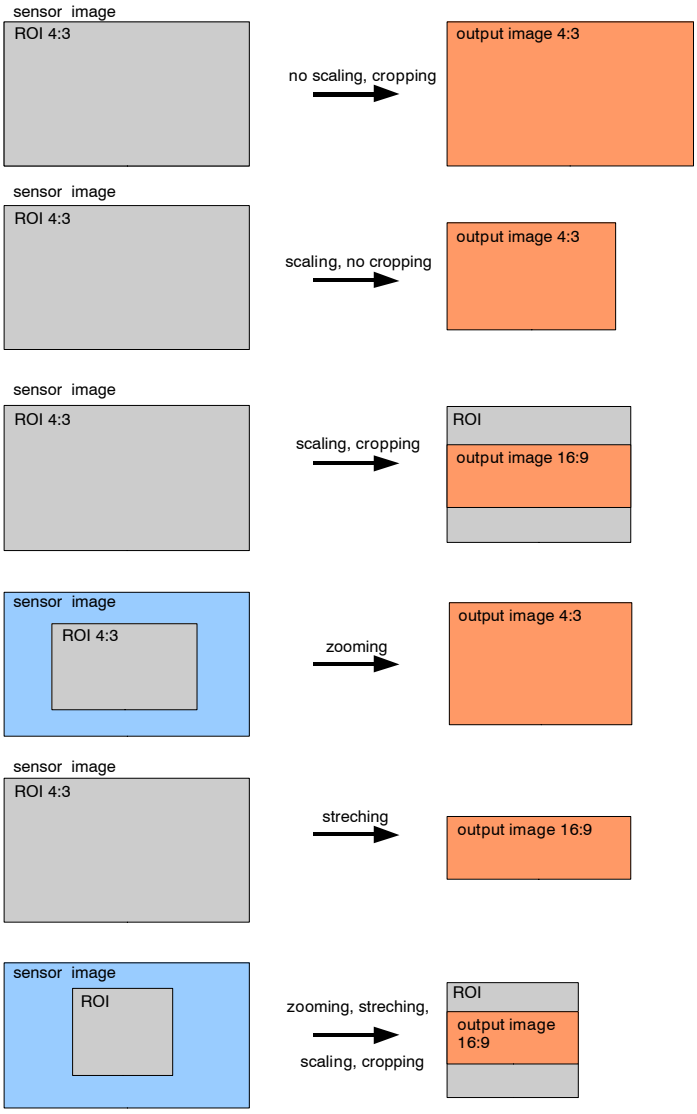


Figure 9. Sample Usages

JPEG Compression Engine

The JPEG engine in AP1302 is composed of two identical instances. An image is first broken to even and odd MCU blocks. All even MCU blocks are processed by one JPEG instance and all odd MCU blocks are processed by the other JPEG instance. Output streams from both engines are then merged together to form a proper JEPG stream.

Restart markers can be enabled. Only restart marker interval of 1 MCU is supported, which means that restart markers are inserted after every MCU. SCALADO speed tags are also supported. When SCALADO speed tags option is turned on, restart marker insertion needs to be disabled.

In 3D bridging application, JPEG instances can be used to compress frames from both sensors independently forming two separated streams.

Statistics Capture

The statistics engine gathers various statistical data about the image from various parts of the image processing pipe. Most of the gathered data is transferred directly to the main CPU memory for further processing using DMA. Part of the data is stored in statistics engine registers and the CPU needs to read them. Statistics data is used by the auto function running on the main CPU. Typical statistical data include:

- AE statistics
- AWB statistics
- Flicker statistics
- RGB zone statistics

Face Detection

The face detection function detects faces and optimizes focus, color and exposure before capture. The detection function provides the following functionality:

- Static face detection
- Moving face detection and
- Face enumeration.

The faces are detected in frontal position and tracked in profile as long as the face is within the frame. Faces are tracked when the camera is rotated left or right. Once the face is “locked” momentary pose change to profile does not affect the tracking. The algorithm implemented in AP1302 works on the principle that while detecting faces that are visible in the field of view is critical, avoiding false positives (detecting faces when they are not present) is equally important. The face detection algorithm supports up to 60−degree range of rotation in the horizontal and vertical directions. Faces in the field of view can be detected in 33 ms (within one frame when operating at 30 fps).

The face detection works on the resolution of 320 x 240 pixels scaled down from any preview resolution. A box is drawn around the face once a minimum face size (20 pixels) is detected. The number of faces that can be detected and tracked is 10.

Once detected, the coordinates of the location of the face(s) are fed back to the embedded CPU where the auto functions (AE and AF) can track the faces. This ensures that the faces are well− exposed and optimally focused and skin colors reproduced accurately before capture.

Special Effects

The following effects can be optionally enabled in AP1302:

- Normal
- Alien
- Antique
- Black and White (B&W)
- Emboss
- Emboss color
- Grayscale
- Negative
- Blueish
- Greenish
- Reddish
- Posterize1
- Posterize2
- Sepia1
- Sepia2
- Sketch
- Solarize
- Vivid

Auto Image Control

Auto image control functions include the 3As (AE, AWB, AF), and flicker detection, adaptive− tone−mapping (ATM), local−tone−mapping (LTM), and so on.

Auto White Balance

In auto white balance mode, the algorithm uses on−chip RGB zone statistics to detect the illuminant or white point and compensate for it. There are 16 x 16 programmable zones that provide RGB statistics. The algorithm determines for each zone if the statistics would be acceptable or not.

Generally, the algorithm rejects those zones that are determined to be of strong color. To make this determination, the AWB algorithm uses a set of calibration data from the calibration profile, which consists of a set of measured white points from standard illuminants, and bounds for each illuminant that define the region of acceptable colors. Using RGB statistics from the accepted zones, the AWB algorithm estimates a white point for the image. For images where none of the zones are accepted, for example, images that consist entirely of strong colors, the algorithm would decide that there is no white point, and it will use history from previous frames as the white point for the current image. In addition, the algorithm uses information of local scene classes to help determine the white point.The algorithm can accept green and color temperature offset from the calibration profile for each of the standard illuminants. As a result, it is possible, for

example, to make the images “warm” for tungsten lighting and neutral for other illuminants.

In manual mode, illuminant or white point can be specified. Additionally ’measure white−point’ functionality is supported, which performs a gray−world algorithm on the input scene.

Both the auto and manual white settings control the color matrix. In the calibration profile, a calibrated color matrix is defined for each of the standard illuminants. The color matrix that is used for any particular moment in the pipeline is a result of interpolation based on the color temperature for that moment.

Auto Exposure (AE)

Images may be captured under various light conditions. Exposure time must match lighting conditions and has to put the sensor in the most sensitive acquisition range.

Both auto exposure and manual exposure are supported. A rectangle of interest may be specified for AE and is fully programmable. AP1302 supports different metering modes.

- Center−weighted metering mode where the scene brightness is calculated by taking the average luminance value at the center of the frame (divided into equal zones) and the objects in the center are properly exposed.
- Matrix metering mode splits the frame into many segments and measures exposure value based on the average luma of each segment and its respective weighting.
- Spot metering mode is predominantly utilized in high contrast scenes where the main subject in center of the frame is properly exposed. The window specified is smaller than that of the center− weighted mode.

The algorithm for AE is configurable and is executed by embedded CPU.

Flicker Detection

Flicker detection algorithm can automatically detect 50 Hz and 60 Hz flicker and compensate for it. The detection time is programmable.

The flicker can also be removed manually.

Auto Focus (AF)

If the module has an AF actuator, AP1302 can drive it through an external driver. The AF algorithm uses programmable zones to detect focus position in the scene and can support a wide variety of actuators, including voicecoil, MEMS, liquid lens, Heliomorph, and so on. The algorithm can perform a single or two−pass search to find the best focus position. In manual mode, arbitrary position can be set.

In case of reliable actuator AP1302 can also perform continuous AF.

SYSTEM OPERATION

This section describes the operation of the AP1302.

Boot Procedure

After power up host needs to load FW patch along with configuration, calibration and tuning settings to AP1302 in order to start frame processing. FW patch and settings are divided into high priority and low priority group:

- High priority group contains essential FW functions and settings needed for basic system operation and needs to be loaded before frame processing is started. Frames processed with only high priority group loaded are not exhibit best image quality, but the frames are of correct size and timing so host can utilize them order to synchronize the receiver to the AP1302 output stream.
- Low priority group contains patches and settings that are not influencing the basic operation of the chip, instead they contain functions and settings that ensure best possible frame processing with highest image quality. Low priority can therefore be loaded after frame processing is already in progress.

All this is compiled into single continuous binary called bootdata. Host does not need to be aware of these steps or the content of bootdata.

Boot Procedure Using Bootdata

Bootdata is a compressed binary that contains all the necessary configuration to get AP1302 up and running with highest image quality:

- Patches (high and low priority)
- Configuration (output size, format, etc.)
- Tuning
- Calibration

Bootdata is provided in hex format as part of the configuration XML (generated by build script provided by ON Semi). Bootdata is content of a section with a tag <dump> with the ”name” attribute ”bootdata”.

All data inside bootdata content is compressed in order to reduce the loading time. AP1302 decompresses and interprets data on the fly as it receives it. Since there is no need to send the register address when loading register values, this further reduces the loading time. Below is an example of bootdata XML content:

```
<dump name="bootdata" checksum="0xce40" pll_init_size="0x1108">
   0500010c 00000000 00000100

   6018     0840

   600c     0201

   6025     0002    00320000

   602d     0005    001f0101  00000000  4000

   6039     0005    002e0300  00000000  4000

   2051     0012    00010033  0001002c  00030021 00030020 00010015 00010015
   0001003d

   00000000 00030021
</dump>
```

Boot data content is loaded to basic address 0x8000 onwards. Once the host loads the address 0x9FFF, it goes back to address 0x8000 and continues loading from there. Host cycle around 0x9FFF−>0x8000 until all content is loaded.

Boot up procedure using bootdata is as follows:

1. After power up sequence, host should poll for model_id basic register until value 0x0265 is read.
2. Host loads the number of bytes as specified in “pll_init_size” attribute and then set boot−data_stage basic register to 0x0002. This will apply basic_init_hp settings and enable PLL.
3. Host should wait 1ms for PLL to lock.
4. Continue load the rest of bootdata content (starting at pll_init_size and observing the wrapping point of 0x9FFF −> 0x8000) until all bootdata is loaded.
5. Host write the 0xFFFF to bootdata_stage basic register. This indicates to AP1302 that the whole bootdata content was loaded.
6. Host waits for bootdata_checksum to become non−zero and compare it with ”checksum” attribute. If the read value does not match the value from the attribute, there was an error during bootdata transmission. Host should reset the AP1302 and repeat the boot up procedure.

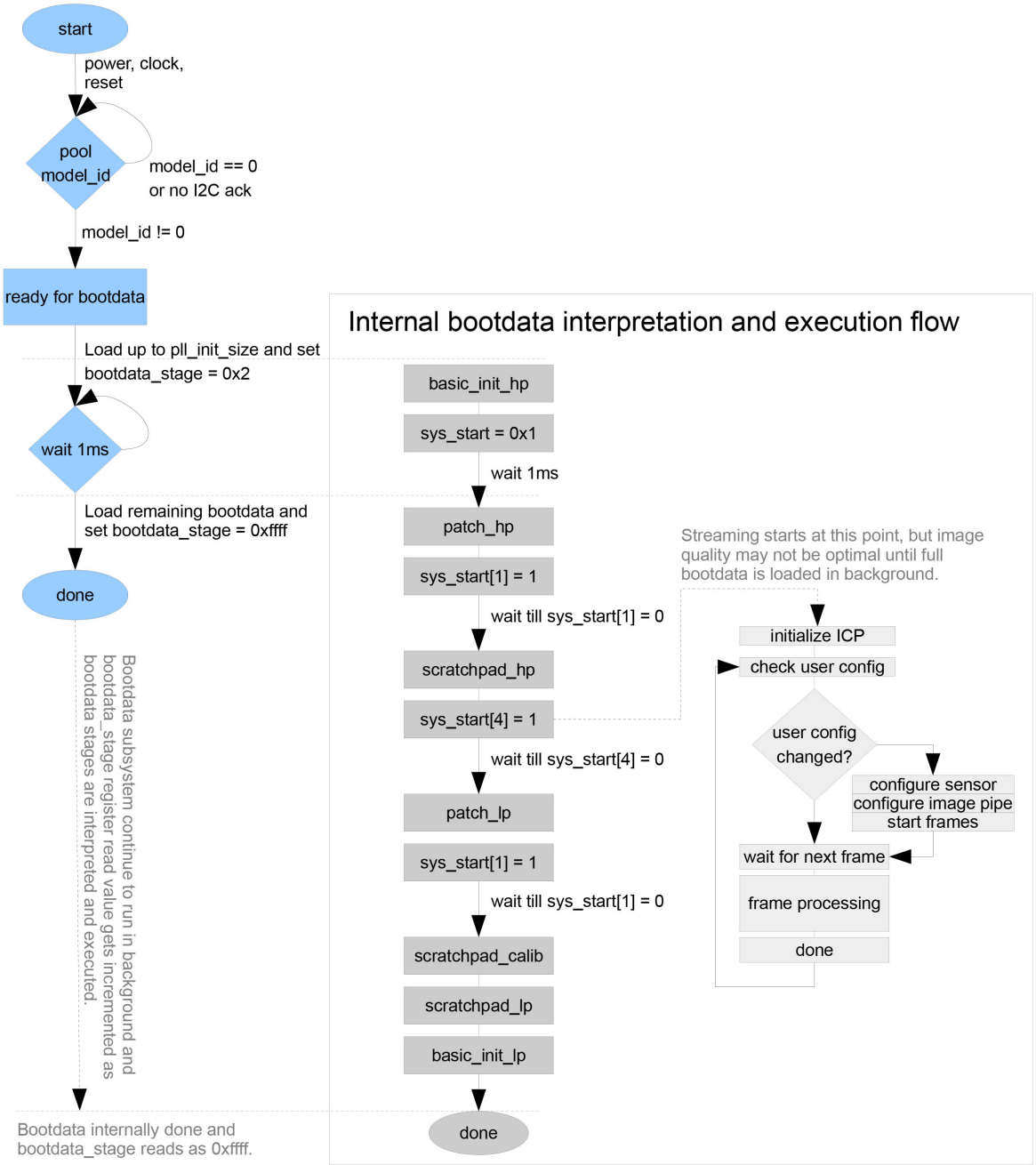


Figure 10. Boot Procedure Flow Chart

NOTE: User does not need to be concerned with flow in gray boxes.

Contexts

The AP1302 can operate in three contexts: Snapshot, Video and Preview. Each context can be independently preconfigured for:

- Divider settings for clock frequency settings
- Output image size and format
- ROI and aspect ratio
- Sensor operating mode
- Max frame rate
- Various other settings related to auto functions

The host can switch between the contexts by a single register write to ctrl register, cntx field.

Many configuration options described in this document apply to all contexts. In such cases register name is written as <cntx>_register_name, where <cntx> is either preview, snapshot or video. For example:

– <cntx>_out_fmt
– <cntx>_width
– <cntx>_height
– <cntx>_ss
– <cntx>_roi_x0
– <cntx>_roi_x1
– <cntx>_roi_y0
– <cntx>_roi_y1

ROI, Aspect Ratio, Zoom Factor, Output Size

There are many controls that define the size and field of view (FOV) of the output image. Each of these controls is context−dependent, except digital zoom factor, which is global. Figure 11 on page 14 shows the order how different controls are applied.

First the region of interest (ROI) is cropped out of the sensor native full FOV image. ROI is defined as the relative number based on full FOV image. Normal ratio is from 0 to 1. Sensor image can be binned before ROI is applied as specified in <cntx>_sensor_mode register.

Unless digital zoom is being used or <cntx>_roi_x0/y0/x1/y1 limits are used, AP1302 will assume that ROI is maximum FOV image which match the aspect ratio of requested output image size (that is, when working with 4:3 sensor producing 16:9 output image, the ROI will be reduced in Y direction to 75% of original Y dimension, so sensor will output 16:9 output image). Normally cropping will be done in sensor in order to reduce amount of data transferred between the sensor and the AP1302. In case sensor limitations prevent exact ROI cropping inside the sensor, then just part of size reduction will be done in the sensor and additional cropping will be done inside the AP1302.

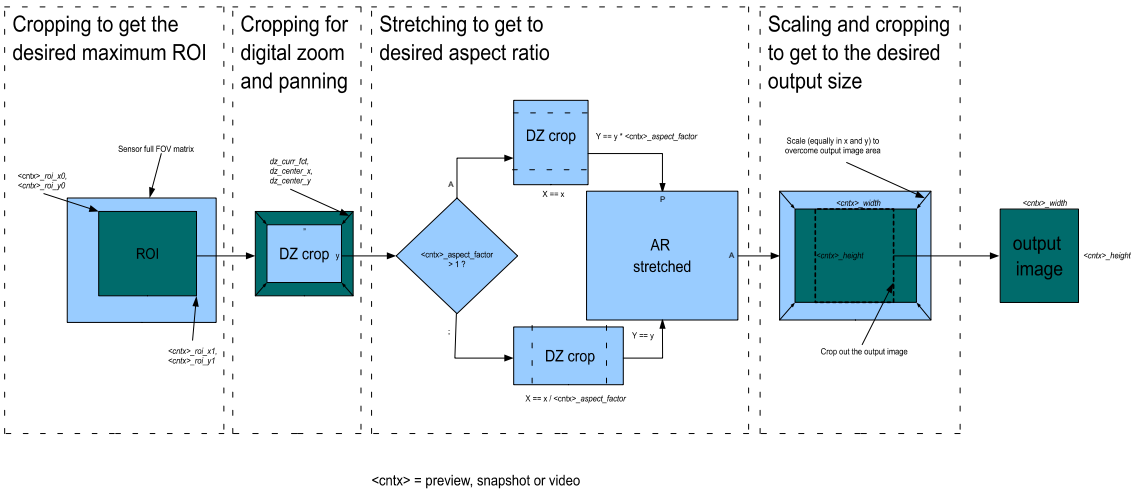


Figure 11. Output Image Size Calculation

In the next step digital zoom and panning is applied where out of requested ROI image the center region shrunk by a dz_curr_fct will be cropped out. The output will then be stretched to desired aspect ratio. With _aspect_factor register, host can configure AP1302 to do stretching and/or squeezing of the image (meaning scaling the image in just one direction while the other direction is not affected).

In the last step, image is scaled up or down such that the resulting image match (or just covers in case aspect ratio does not match) the desired output image size specified by <cntx>_width and height. Scaling is always done equally in both directions. In case resulting image after scaling is bigger than desired output image size, the image is cropped appropriately.

Bubble

Bubble is special type of image that can be generated by the AP1302 that can only be sent out as an auxiliary (second) stream. Host can configure bubble image stream to have its own output size and format. On top of that, ROI and/or independent zoom factor and pan location can be specified. In contrast to ROI for the snapshot, video, and preview context settings, bubble ROI is set relative to the output FOV. This means that the current context output image is used as a base for bubble FOV generation. This does not mean that the bubble size needs to be equal to or smaller than cropped output image specified by the ROI; it can as well be bigger. In this case the AP1302 will generate bubble from bigger sensor image if possible in order to preserve as much information in the bubble image as possible. If bubble output size is even bigger than corresponding sensor image region, then scale up feature will be used to generate bubble image.

Note that ROI settings for the bubble can also be negative or greater than 1 (this is to allow bubble to get out of current context FOV). In this case, AP1302 will extend the field of the current context ROI settings to accommodate bubble ROI. Of course this is only possible if current context ROI settings are occupying ROI smaller than full FOV of the sensor. If bubble ROI settings will extend over the full FOV of the sensor, the ROI will be automatically clipped appropriately.

Bubble is especially usable when the host wants to configure bubble to carve out the portion of the image which is occupied by the face reported by face detection. The resulting bubble image will have bigger resolution than the face in the output image has so it can be used for face recognition or other such applications.

Interleaving Two Streams

The AP1302 can send out two streams simultaneously using interleave. The primary output stream is always defined by the context currently selected. The auxiliary (second) stream is selected by <cntx>_out_fmt[iis] register. Table 3 shows the possible auxiliary stream options and suggested application usage.

Table 3. AUXILIARY STREAM OPTIONS

Context	Auxiliary Stream	Application
Snapshot	Preview	Postview, ZSL
	Video	Snapshot during video
	Bubble	Face recognition
Video	Preview	Video w/ preview
	Bubble	Face recognition
Preview	Bubble	Face recognition, AF check

Note that the auxiliary stream always adopts the output image size and format of the context (or bubble) specified (i.e. when preview is selected as an auxiliary stream during snapshot, auxiliary stream will be of size and format as specified by preview_width, preview_height and preview_out_fmt). But the ROI settings of the auxiliary stream will be ignored rather ROI settings of the current context will be used to generate the auxiliary stream.

Output Formats

The AP1302 sends out frames that can contain more than one image (snapshot image can be interleaved with postview image). Image can be of the following formats:

- YUV4:2:2
- YUV 4:2:0 (YYYY.../UYVYUYVY)
- RGB565
- RGB555
- JPEG (based on YUV4:2:2)
- JPEG (based on YUV4:2:0)
- RAW

JPEG images can be equipped with header and footer to form JFIF/EXIF file. When two images are interleaved, the two images can be of different output format.

Along with the image data, AP1302 can append the status structure at the end of the frame, containing metadata of the current frame.

Status Structure

Status structure, when enabled, contains various metadata about the image and it is appended to the end of every frame sent out. Which metadata is part of the status structure and its format can be configured by the host using a simple description list. A status structure description list consists of entries, each entry contains a base address of a first register to be copied into the status structure and the number of bytes to be copied. Each entry also defines whether the base address is part of the Basic register address space or Advanced register address space. In case entry describes the Basic register space, the address is 16bit and if the entry describes the Advances register space, the address is 32 bit. Description list ends with the entry containing all zeroes.

There can be up to four different lists defined. Status structure description lists should be placed into the Scratchpad memory. The address of the first entry for each list should be put into a one of the four ss_head_pointer[4] basic registers. preview_ss[enable], snapshot_ss[enable] and video_ss[enable] basic registers control which description lists are active in a given context. This allows the host to configure different data to be sent out as part of the status structure dependent on a context. Figure 12 on page 16 shows the status structure description list organization.

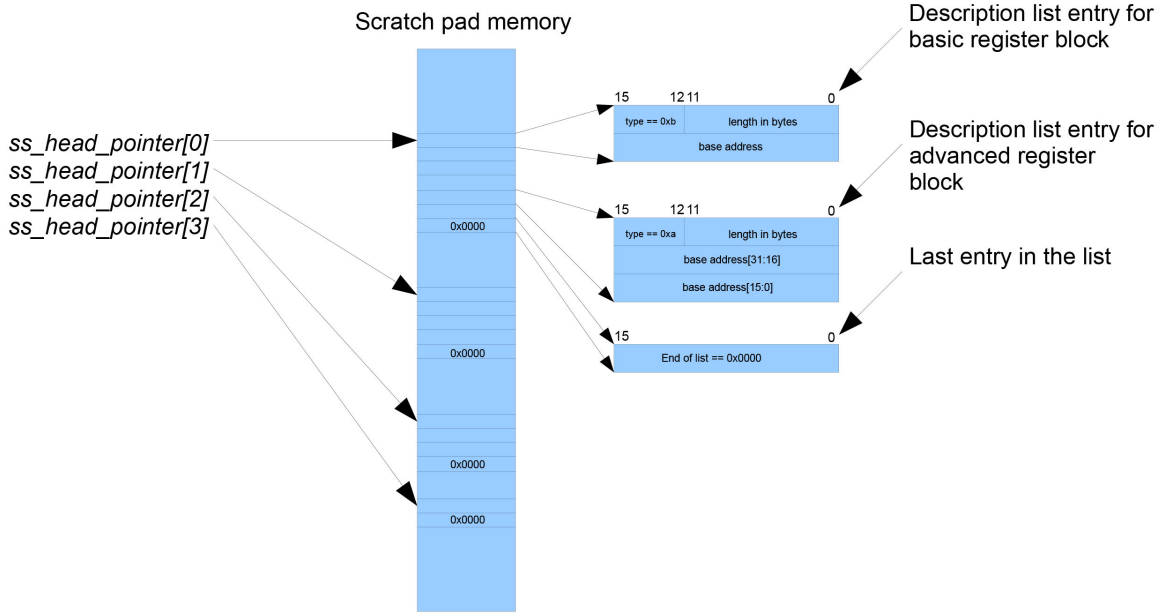
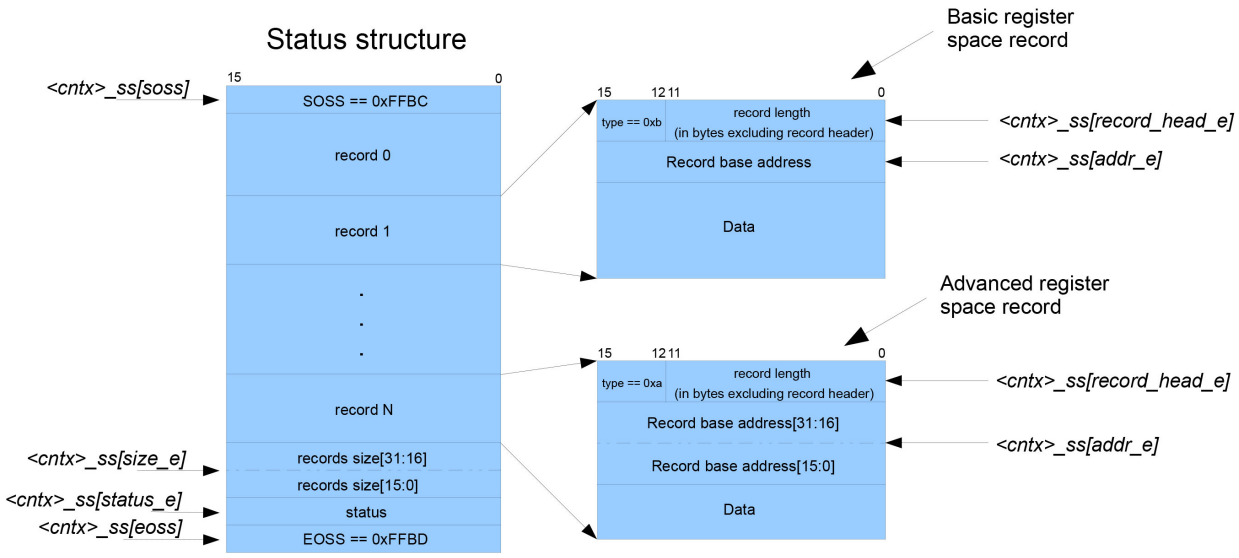


Figure 12. Status Structure Description Lists

Based on the status structure descriptor lists and other settings in <cntx>_ss registers, the status structure is built. Every entry in a given list creates one record in the status structure. The records are concatenated one after another as they are organized in the lists and lists are organized from the ss_head_pointer[0] to ss_head_pointer[3] (depending on which list is enabled in <cntx>_ss[enable] field). Additionally all records can preceded with SOSS marker (if enabled in <cntx>_ss[soss]) and followed by records size (if enabled in <cntx>_ss[size_e]), status filed (if enabled in _ss[status_e]) and EOSS marker (if enabled in _ss[eoss]). Status structure organization is shown in Figure 13 on page 17.

NOTE: <cntx> throughout this document stands for snapshot, preview, or video context.



<cntx> = preview, snapshot or video

Figure 13. Status Structure

In any given output format, status structure is always aligned to the end of the frame (meaning that the last byte out of AP1302 is last byte of the status structure). In spoof modes, this means that certain padded data are inserted between the image data and the status structure if needed.

The AP1302 has a few preconfigured three status structure description lists. Those lists reside in the ROM.

These preconfigured lists can be overridden by the host by writing different description lists in the scratchpad memory and reassign the ss_head_pointer[4] pointers.

Spoof Mode

Host interface supports spoof mode. Spoof mode enforces the width and height of the outgoing image (or two interleaved images if one virtual channel is used to send the data out) as it is defined by <cntx>_hinf_spoof_w and <cntx>_hinf_spoof_h. This means that when spoof mode is enabled, the mipi packets are all of the same length as defined by <cntx>_hinf_spoof_w and the number of packets sent out for one frame is always equal to <cntx>_hinf_spoof_h (except when <cntx>_hinf_spoof_h is set to zero, in which case the number of packets sent out will depend of the number of data to be sent). This means that if the number of bytes in the frame (which can compose of one or two images plus the status structure, if enabled) is lower than the size of the spoof frame is, the missing bytes will be padded to form the whole spoof frame (in case <cntx>_hinf_spoof_h is set to zero, then the last packet will be padded to obey the spoof width as specified in <cntx>_hinf_spoof_w). Padded bytes will be positioned at the end of the spoof frame, or, if status structure is enabled, between the last image data byte and first status structure data byte. This will ensure that status structure is always aligned to the end of the spoof frame, so host will always be able to find the status structure in the incoming data buffer. Figure 14 shows the example of a spoof frame with two interleaved images and status structure.

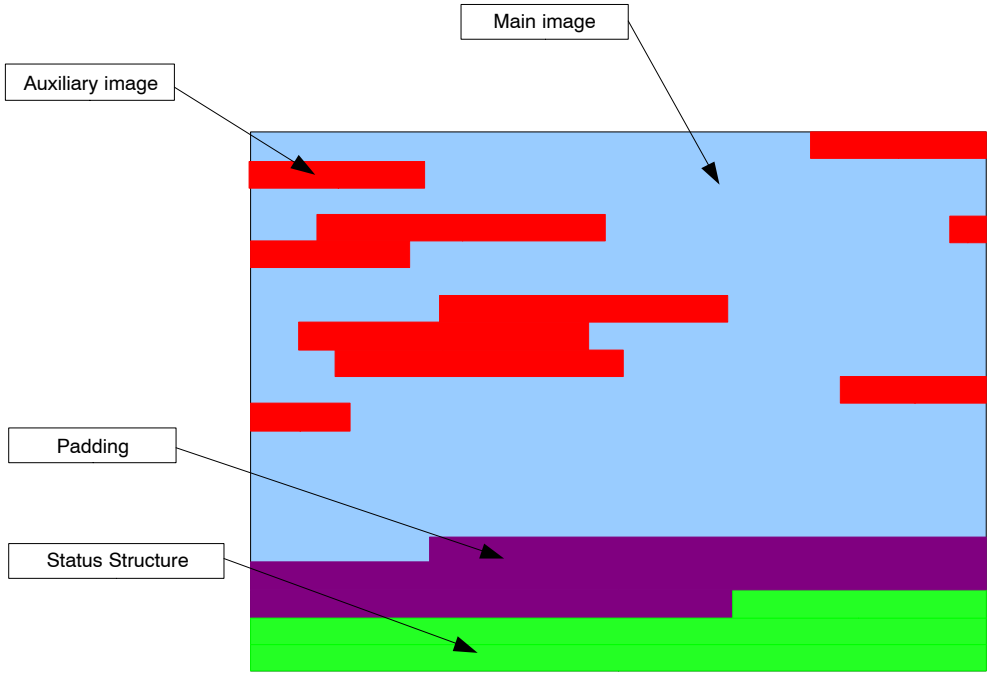


Figure 14. Spoof Frame Example with Two interleaved images and Status Structure

In case frame data (one or two images along with the status structure, if enabled) will exceed the size of the spoof frame size, the image data will be truncated so that the spoof frame size will be maintained. In case status structure is enabled, the status structure data is not truncated, rather the image data is truncated appropriately so that the complete status structure is sent out (still being aligned to the end of the spoof frame). This ensures that the status structure data is always correct and host will be able to detect the spoof frame overflow by examining warning[4] and error regis− ters if they are sent as part of status structure.

RAW Bypass

RAW bypass is achieved by selecting the RAW output format in <cntx>_out_fmt. This mode is only meant for production testing and debugging purposes. All autofunctions and majority of features (such as face detection, status structure, and postview image) are not possible in this mode.

INTERFACES

The AP1302 has the following interfaces to interact with the external devices:

- Sensor Serial MIPI interfaces
  - Up to 4x 1.2 Gbps data lanes to receive data from primary sensor
  - Up to 3x 1.2 Gbps data lanes to receive data from secondary sensor
- Host serial MIPI interface (4x1.2 Gbps)
- I2C slave interface for host to control AP1302
- SPI slave interface for host to control AP1302 (optional)
- Two I2C master interfaces for AP1302 to control sensors, AF and other devices
- SPI master interface for future use
- GPIO/PWM controls

Sensor Interfaces

The sensor interface consists of two MIPI receiver ports. Each receiver has its own clock lane and they share 6 data lanes as shown in the Figure 14. Each lane supports up to 1.2 Gbps bandwidth. In normal operation mode, the primary and optional secondary sensor can be connected to the AP1302. The primary sensor can use up to 4 (in the order: S0_D0, S0_D1, S0_D2, S01_D32) data lanes and secondary sensor can use up to 3 data lanes – the last when not used by the primary sensor (in the order S1_D0, S1_D1, S01_D32).

In 3D bridging application, two sensors of the same type can be connected to the two receivers with the same order of data lanes. The left sensor is connected to the receiver 0 and the right sensor is connected to the receiver 1. In this case, the secondary sensor I²C master is used to control the right sensor. This ensures that all the register writes to the sensor are executed synchronously while there is still opportunity to read the registers from the two sensors (like LSC configuration data from OTPM inside the two sensors).

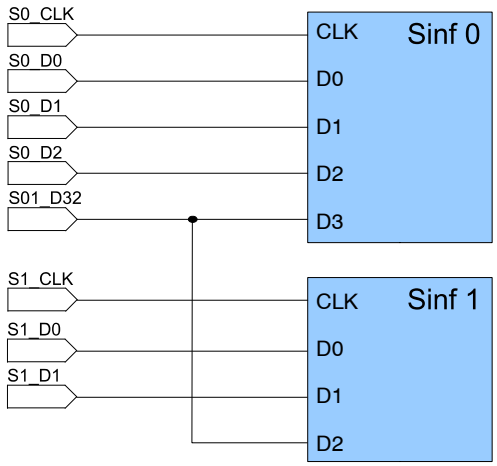


Figure 15. Data Lane Sharing Between Two Sensor Interfaces

Host Interface

The host interface consists of one clock and four data MIPI lanes capable of 1.2 Gbps each. When less than 4 lanes are required, data lanes needs to be disabled in order of 3,2,1 (that is, if just two lanes need to be used, these two lanes are 0 and 1).

The host interface sources the data from resample 0 as main image frame (JPEG compressed or uncompressed). Optionally auxiliary image frame from resample 1 can be interleaved with main image using interleave engine. Interleaving can be done in two ways:

- Using two MIPI virtual channels
  In this mode, every MIPI packet hold the data of only one image frame and the virtual channel ID in the MIPI header tells the host which frame the data in the packet correspond to. Similarly as image data, the optional status structure is also sent out using MIPI packets of programmable virtual channel ID. In this mode FS/FE short packets are sent out for each stream. FS short packet does not support frame counting. LS/LE packets are not supported.

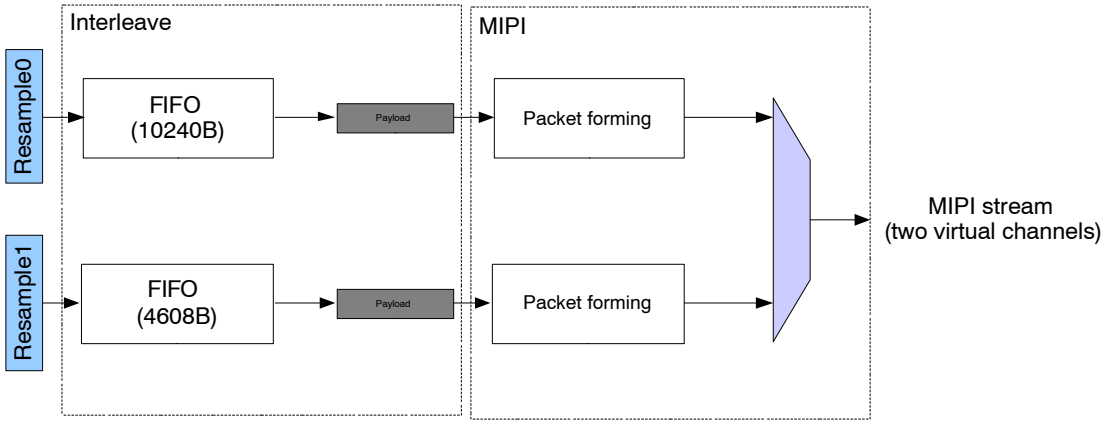


Figure 16. Interleaving Using MIPI Virtual Channels

- Using single MIPI channel and proprietary packets. In this mode, the data from one stream are put into packets, each packet consisting of header and payload. Header contains ID (ID==0: resample0, ID==1: resample1) and size of the payload in bytes. Packets are then sent out using MIPI packets of single virtual channel. One MIPI packet can carry data of one or both image frames (MIPI packets does not align with proprietary packets). In this mode only one MIPI FS/FE synchronization packet is sent per frame. LS/LE packets are not supported.

Interleave module has two FIFOs to buffer the data from the two streams. Main branch can buffer up to 10240 bytes of data, which means that maximum packet (MIPI or proprietary) can consists of 10240 bytes. Similarly auxiliary branch can buffer up to 4608 bytes of data, which is enough to form a packet of 4608.

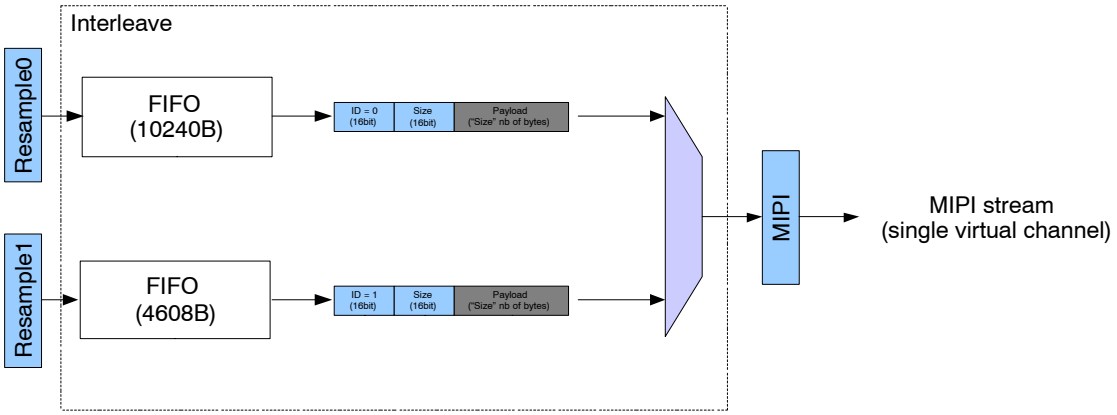


Figure 17. Interleaving Using Single MIPI Virtual Channels and Proprietary Packets

I²C Slave Interface

The AP1302 is controlled by the host through the I²C or SPI interface where host processor is a master and AP1302 is a slave on the bus. The I²C slave interface of the AP1302 can support Standard−mode (up to 100 kbps), Fast−mode (up to 400 kbps), Fast−mode Plus (up to 1 Mbps) and High speed mode (up to 3.4 Mbps).

The I2C slave ID can be configured as:

- 0x78 (write) and 0x79 (read) or
- 0x7A (write) and 0x7B (read)

To select the ID AP1302 checks GPIO[11] right after reset:

- GPIO[11] == 0: ID = 0x78/0x79
- GPIO[11] == 1: ID = 0x7A/0x7B

Note that GPIO[11] can still be used for other purposes after I²C ID was latched after reset. If used for other purposes, pull up/down resistor needs to be used to pull the GPIO[11] line up/down during and immediately after reset.

The I²C slave has a digital glitch filter. This filter is controlled by SIPS_FILTER register. Filter can filter glitches out of SCLK signal based on median 3 filter (two input needs to be “1” to output “1”; otherwise output is “0”). SDATA signal has a programmable delay in order to avoid race conditions. Filter uses SYS clock which is beforehand divided. Filter is disabled after reset. Figure 18 shows the I²C glitch filter scheme.

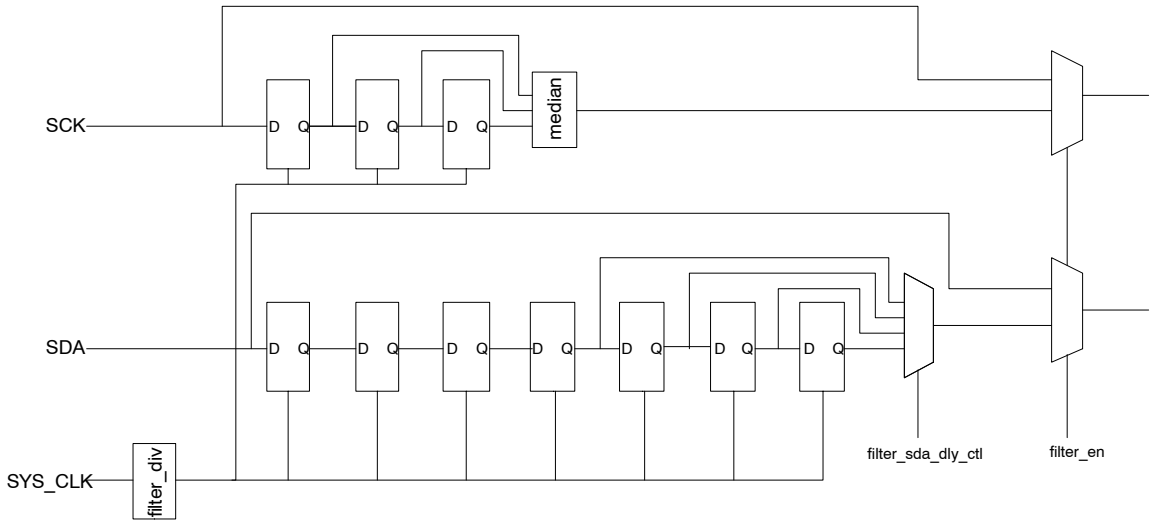


Figure 18. I²C Slave Digital Glitch Filter

Host sees the AP1302 through a 16−bit address space called Basic register space. Every I²C access starts with a 16−bit address, following by a read or write data transaction. Data transaction can consists of one or more bytes (each consecutive byte will be read from or written to incremented address from the address that was specified in the address transaction).

Advanced Register Space

Host can access Advanced register space indirectly through the Basic register space. 4 kBytes of Basic register space is mapped to Advanced register space as shown in Figure 19 on page 22.

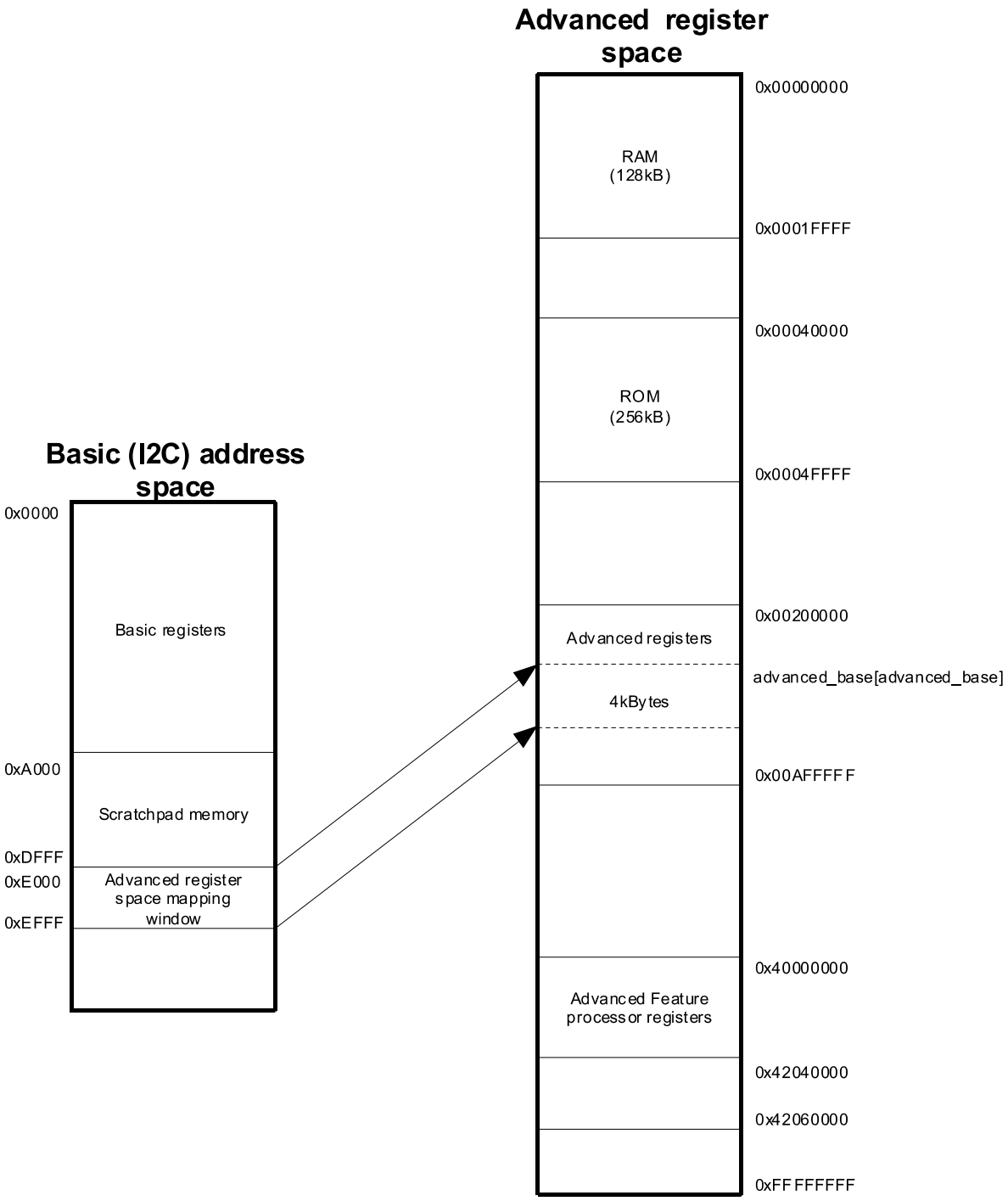


Figure 19. Advanced Register Space to Basic Register Space Mapping

Base address inside Advanced register space to which this 4 kb window is mapped is defined in Advanced space base pointer register.

Table 4. ADVANCED SPACE BASE POINTER REGISTER

BITS	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIEL D	Reserved												advanced_base			
RST	0x0												0x0			
R/W	rw												rw			
ADD R	0xF038															
SYN C	none															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	advanced_base															
RST	0x0															
R/W	rw															
ADDR	0xF03A															
SYNC	none															

Len	Bits	Name	Reset	Description
16	19:0	advanced_base	0x0	This field defines the Advanced system address that Basic page starting at 0xE000 is mapped to (used to map advanced address space).

```
Example of setting GPIO3 to 1 by setting ADV_GPIO_DIR[3] and ADV_GPIO_DO[3] bits:
REG= 0xF038, 0x002A0000 // ADVANCED_BASE [ADV_GPIO_DIR]
REG= 0xE004, 0x0000069A // ADV_GPIO_DIR
REG= 0xE000, 0x0000049A // ADV_GPIO_DO
```

SPI Slave Interface

Host can control AP1302 either through the I²C or the SPI bus. In both cases, the host acts as master and AP1302 acts as a slave. When SPI slave interface is used, the host sees the same Basic register set through the SPI interface as it is visible through the I²C interface.

Every access on SPI bus starts with the Basic register space 16−bit address, followed by an 8−bit control byte (holding 1 for WRITE and 0 for READ) and then one or more data bytes. Figure 20 shows write and read cycle waveforms. All data are transferred MSB first. SPI mode 0 is adopted.

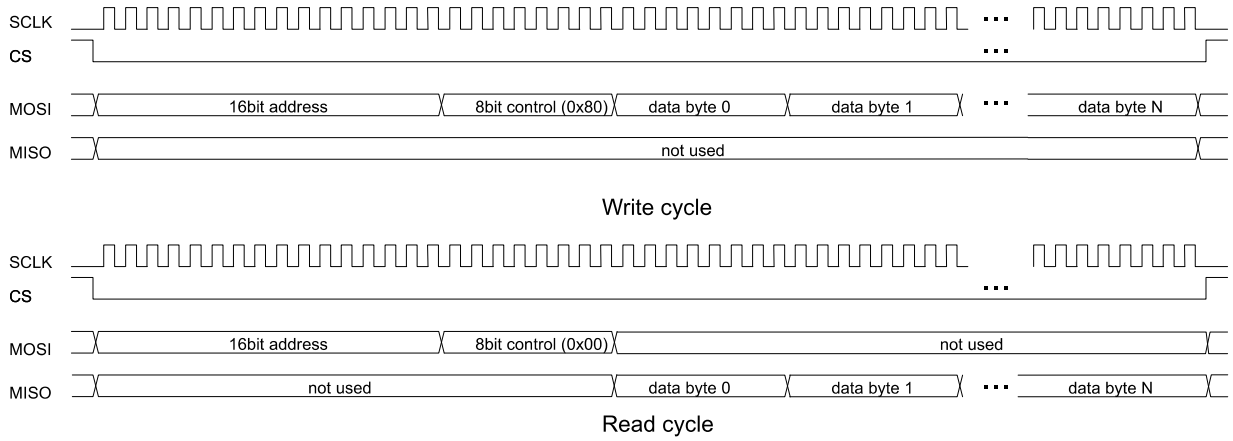


Figure 20. SPI Write and Read Data Access

SPI Slave interface of AP1302 is by default disabled. It can be enabled through advanced registers (please refer to Advanced Register Space). These are the advanced register writes necessary to enable SPI Slave interface:

```
// SPI Slave interface: Enabling.
REG= 0xF038, 0x00200000 // ADVANCED_BASE [ADV_SYS_RST_EN_0]
REG= 0xE014, 0x0000003D // ADV_SYS_RST_EN_0
REG= 0xE00C, 0x0000001C // ADV_SYS_CLK_EN_0
REG= 0xE008, 0x0000039D // ADV_SYS_DIV_EN
REG= 0xF038, 0x002B0000 // ADVANCED_BASE [ADV_SYS_STBY_GPIO_OSEL]
REG= 0xE020, 0x03333001 // ADV_SYS_STBY_GPIO_OSEL
REG= 0xE028, 0x19919111 // ADV_SYS_STBY_GPIO_IN_MASK
```

SPIS clock has a limitation depending on SYS clock. SYS clock, which internal system bus is running on, needs to be fast enough to make sure that write data are written before the new write access comes over SPIS bus. This means that write access needs to finish before then next 16 bits are received over SPIS bus (assuming burst writes). Similarly, on read access the data needs to be read before they need to be sent over the SPI bus back to the host, which means that there are 7 SPI cycles available to read the data over system bus. Since the system bus operation in not entirely predictable (when SPIS, which is master on internal bus, asserts bus request, the CPU or any other master just started a long access over the bus, which means that the SPI will have to wait), worst case is assumed, which is 20 SYS clock cycles for access. Since SPI might have to wait for one access that just started and then to finish its own access, in worst case 40 SYS cycles needs to be compared against 7 SPIS cycles, which gives a minimum ratio of SYS/SPIS >= 40/7 =~ 6.

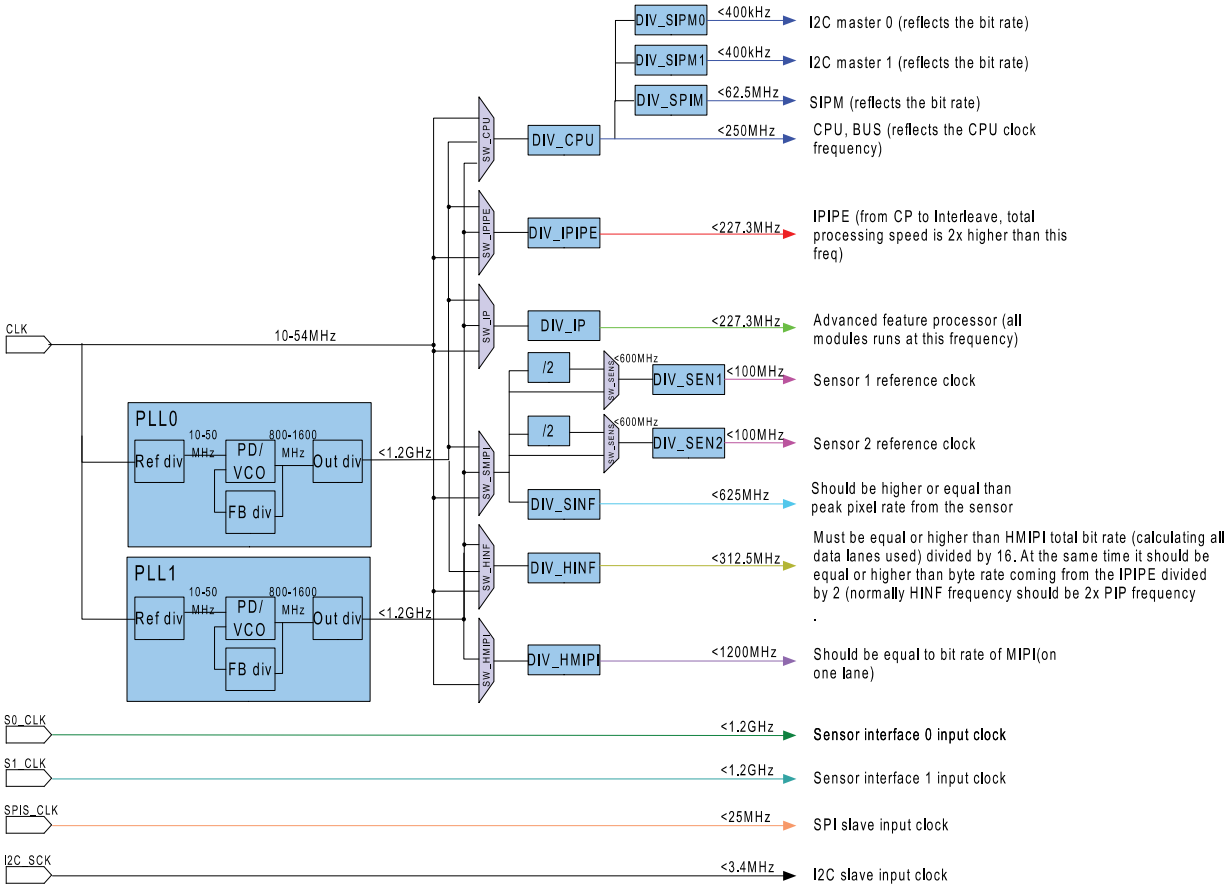
While the above holds true for normal operation, the conditions are different when the system boots up. The CPU or any other masters are not performing any access over the system bus at that time. Also, when loading the bootdata code, the host only performs writes. In this particular case the SPIS clock can match the SYS clock, so minimum ration can be lowered to SYS/SPIS >= 1.

Clocking Scheme

AP1302 has two independent PLLs to generate internal clocks. All internal clocks can source the clock from any of the two PLLs as well as reference clock pin directly, except HINF clock, which can only use PLL1 or CLK input as the clock source. This provides enough configuration freedom that all meaningful clocking ratios are possible. Normally PLL0 will be used to generate SYS and SINF. In case the requirements can be met with just one PLL, PLL0 can be turned off to save power.

The following figure shows the clock scheme and maximum frequencies for all clock nodes. Clocks of same color are synchronous to each other, but most of the clock domains are asynchro− nous to each other. Note that PLL0 can go up to 1.25 GHz while PLL1 can only go to 1.2GHz (1.25 is used when CPU needs to run at its max speed 250MHz, so division of 5 is possible).

Dividers can produce division step of 0.5, staring with mind division of 1 (which means bypass of divider). When rational number is used, divider will use negative edge of input clock to produce 50−50 duty cycle on its output.



For easier conversion between wanted CPU/IPIPE/IP/SINF/HINF/sensor frequencies and values for div field in <cntx>_div_* basic registers follow this procedure:

1. Divide selected PLL output frequency with wanted CPU/IPIPE/IP/SINF/HINF/sensor frequency.
2. Subtract 1 from integer part of value from (1).
3. Multiply value from (2) with 2.
4. If non integer value from (1) is 0.5 add 1 to value from (3).

For easier conversion between values from div field in <cntx>_div_* basic registers to actual CPU/IPIPE/IP/SINF/HINF/sensor frequencies follow this procedure:

1. Divide value of div field with 2.
2. Add 1 to value from (1).
3. If LSB bit of div field is 1 add 0.5 to value from (2).
4. Divide selected PLL output frequency with value from (3).

Figure 21. Clocking Scheme

POWER MODES

The AP1302 can be in one of the following power modes:

- Power down
- Core power down
- Standby
- Normal operation

Power Up

AP1302 is in power down state if all core power and IO power supplies are removed. AP1302 can maintain I²C slave and GPIO interface IOs in HighZ state during power down state exercising its IO fail−safe feature. To bring the chip in normal operational mode, apply power to the chip and assert RESET. Figure 22 shows power−up sequence that guarantees fail−safe functionality.

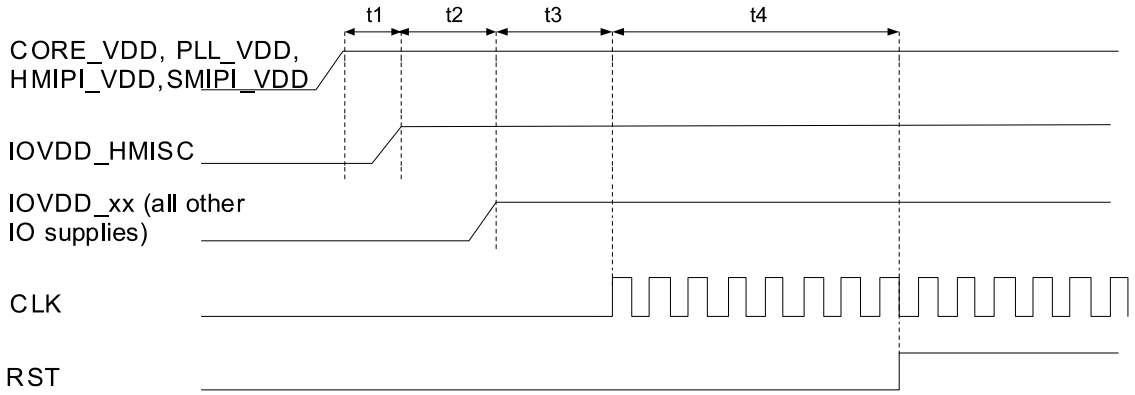


Figure 22. Power−Up Sequence (1.2V −> 1.8V), Fail−safe Guaranteed

Table 5. POWER UP SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	CORE_VDD to IOVDD_MISC delay		200	−	500000	μs
t2	IOVDD_MISC to IOVDD_xx delay (Note 1)		200	−	500000	μs
t3	Power stable to clock active delay		200	−	500000	μs
t4	Clock active to reset release delay		1	−	500000	μs

1. Supplying IOVDD_HMISC all others IO supplies (and keeping it after all other IO supplies) ensures that the logic, which is controlling the GPIOs and I²C IOs, is properly reset before IO power is supplied (or when IO power is cut off) as IOVDD_HMISC is supplying the RESET input pad. If the fail−safe IO option is not required, then IOVDD_HMISC can be ramped down together with other IOVDD supplies, which might cause glitches when power up/down sequence is on.

The voltage ramp up order in the power−up sequence can also be reversed if the STANDBY pin is asserted before voltage ramp−up and de−asserted after voltage stabilizes and RESET is applied as shown in Figure 23 on page 26. Note that when reversed power up sequence is used, the fail−safe functionality is not guaranteed.

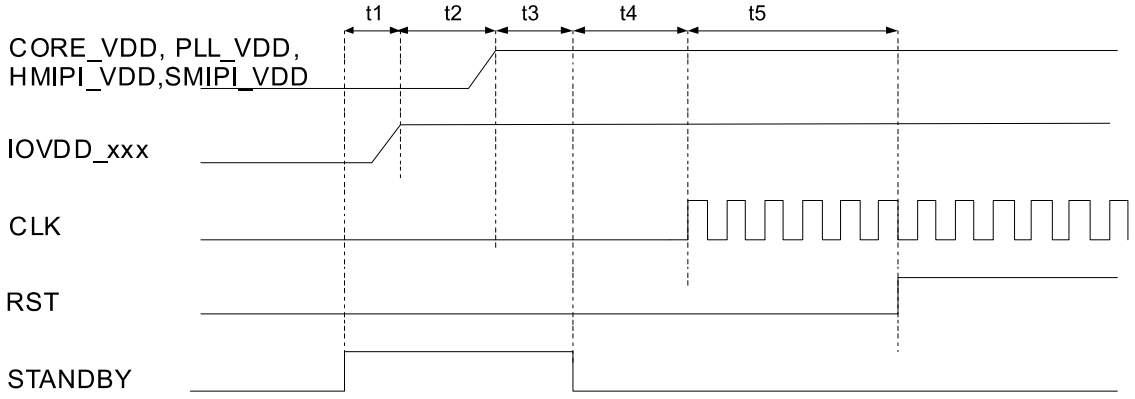


Figure 23. Reversed Power−Up Sequence (1.8V −> 1.2V), Fail−safe Not Guaranteed

Table 6. REVERSED POWER−UP SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	STANDBY ON to IOVDD_xxx		200	–	500000	μs
t2	IOVDD_xxx to CORE_VDD delay		200	–	500000	μs
t3	Power stable to STANDY OFF delay		200	–	500000	μs
t4	STANDBY OFF to clock active delay		200	–	500000	μs
t5	Clock active to reset release delay		1	–	500000	μs

Power Down

Figure 24 shows the power down sequence.

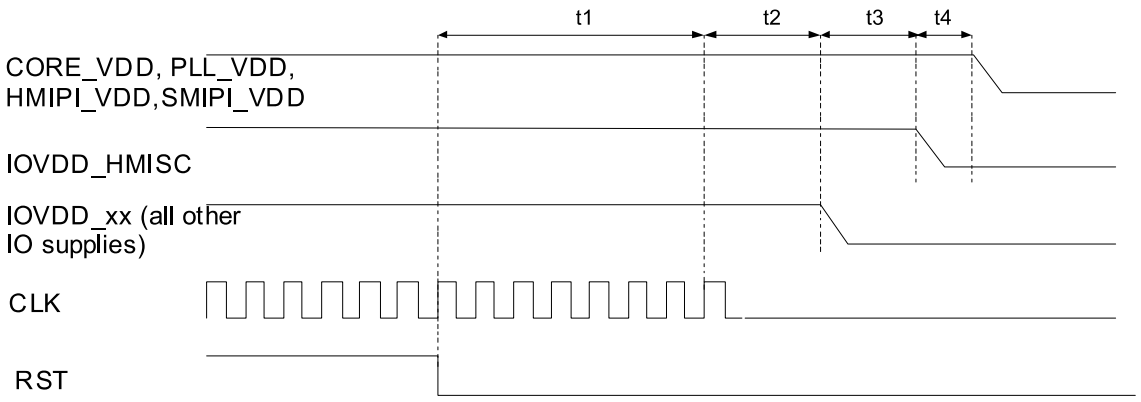


Figure 24. Power−Down Sequence (1.8 V −> 1.2 V), Fail−safe Guaranteed

Table 7. POWER−DOWN SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	Reset active to clock nonactive delay		0	–	500000	ns
t2	Clock nonactive to IOVDD_xx cutoff		1	–	500000	μs
t3	IOVDD_xx to IOVDD_HMISC cut off delay (Note 2)		200	–	500000	μs
t4	IOVDD_HMISC to CORE_VDD cut off delay		200	–	500000	μs

2. Supplying IOVDD_HMISC before all other IO supplies (and keeping it after all other IO supplies) ensures that the logic, which is controlling the GPIOs and I²C IOs, is properly reset before power for these IOs is supplied (or during power for these IOs is cut off) as IOVDD_HMISC is supplying the RESET input pad. If fail−safe IO functionality is not required, then IOVDD_HMISC can be ramp up/down together with the rest IOVDD supplies, which might cause glitches on the I²C and GPIOs when the power up or power down sequence is on.

Voltage ramp down order can also be reversed in power down sequence, but in this case the STANDBY signal needs to be asserted. Note that fail−safe functionality is not guaranteed with reversed power down sequence.

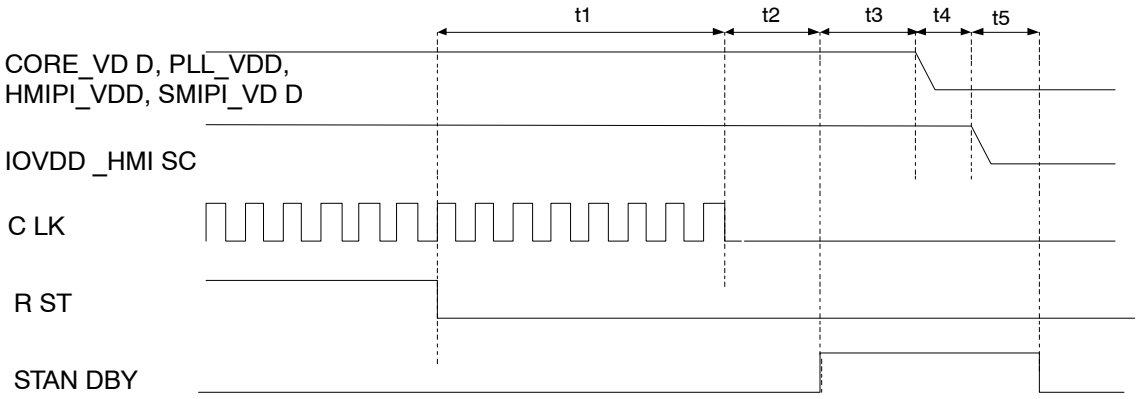


Figure 25. Reversed Power−Down Sequence (1.2 V −> 1.8 V), Fail−safe Not Guaranteed

Table 8. REVERSED POWER−DOWN SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	Reset active to clock nonactive delay	–	0	–	500000	ns
t2	Clock inactive to STANDBY ON	–	1	–	500000	µs
t3	STANDBY ON to CORE_VDD cut off delay (Note 3)	–	200	–	500000	µs
t4	CORE_VDD to IOVDD_xxx cut off delay	–	200	–	500000	µs
t5	IOVDD_xxx to STANDBY OFF delay	–	200	–	500000	µs

3. Supplying IOVDD_HMISC before all others IO supplies (and kept it after all others IO supplies), ensures that the logic, which is controlling the GPIOs and I²C IOs is properly reset before power these IOs is supplied (or during power for these IOs is cut off) as IOVDD_HMISC is supplying the RST input pad. If fail− safe IO functionality is not required, then IOVDD_HMISC can be ramp up/down together with rest IOVDD supplies, which might cause glitches on the I²C and GPIOs when the power up or power down sequence is on.

Core Power Down

I²C and GPIO interfaces are using fail−safe IOs. This means that I²C and GPIO signals, which are in HighZ state before power down sequence is started (when reset is applied), maintain HighZ state during the power down state. When the AP1302 is brought back to operational mode, the GPIO signals, which are in HighZ state during reset (refer to Table 1 for GPIO signal reset state), and the I²C signals will maintain HighZ state during the power up sequence. I²C signals maintain HighZ state until the host performs the first access via I²C.

The AP1302 enters core power down state if core power is removed. IOs are still powered during core power down state in order for GPIO signals to maintain configured state (HighZ/driving 1/ driving 0). This is achieved by asserting STANDBY signal just before removing core power, which will latch in the IO configuration state inside the IO voltage domain. This way, the GPIO signals will maintain the state that they were assuming just before asserting the STANDBY signal for the core power down state duration.

To enter core power down state host must:

1. Configure the GPIO in the desired state.
2. Make sure that AP1302 is not being accessed on the SIPS bus.
3. Enter standby mode by asserting STANDBY signal.
4. Cut off core power.

In core power down state, the chip consumes the least amount of power. Core partition of the chip is in power down state and all configuration data is lost. Reference clock signal CLK may be disabled by the host in this state.

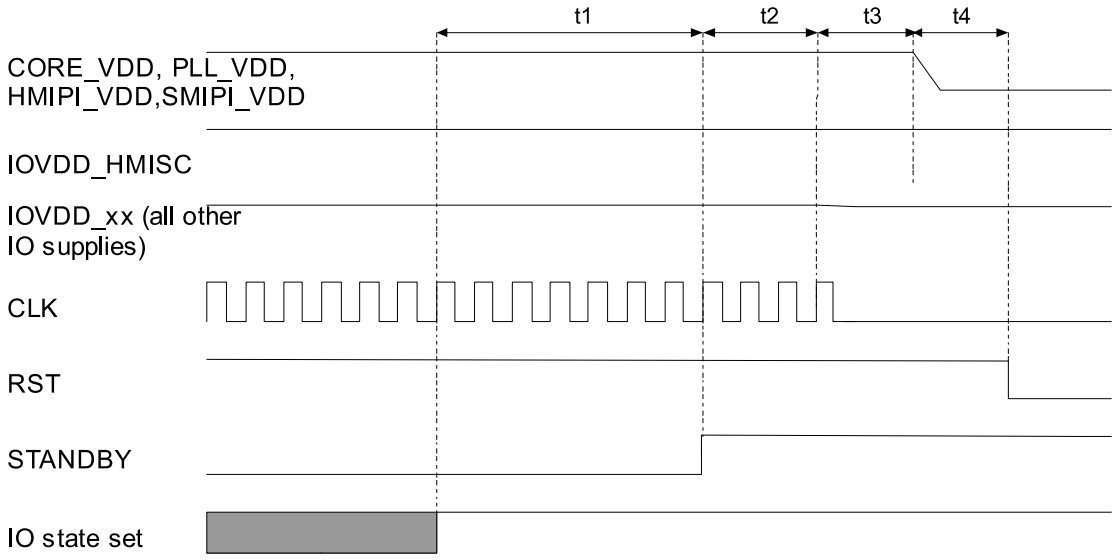


Figure 26. Entering Core Power Down State

Table 9. CORE POWER−DOWN SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	IO state setting to STANDBY delay		1	–	500000	μs
t2	STANDBY to clock non−active delay		10	–	500000	μs
t3	Clock non−active to CORE_VDD cut−off delay		1	–	500000	μs
t4	CORE_VDD cut−off to RESET delay		200	–	500000	μs

When the chip is brought from core power down state by applying the core power, the clock must be enabled and reset must be applied so that internal logic is set to a known state. Note that when the reset condition is applied, the GPIO signals will enter reset state, which might differ from the state specified by the host before STANDBY was asserted (refer to Table 1 for GPIO reset states). Host must do the initialization of the chip including the GPIO state again.

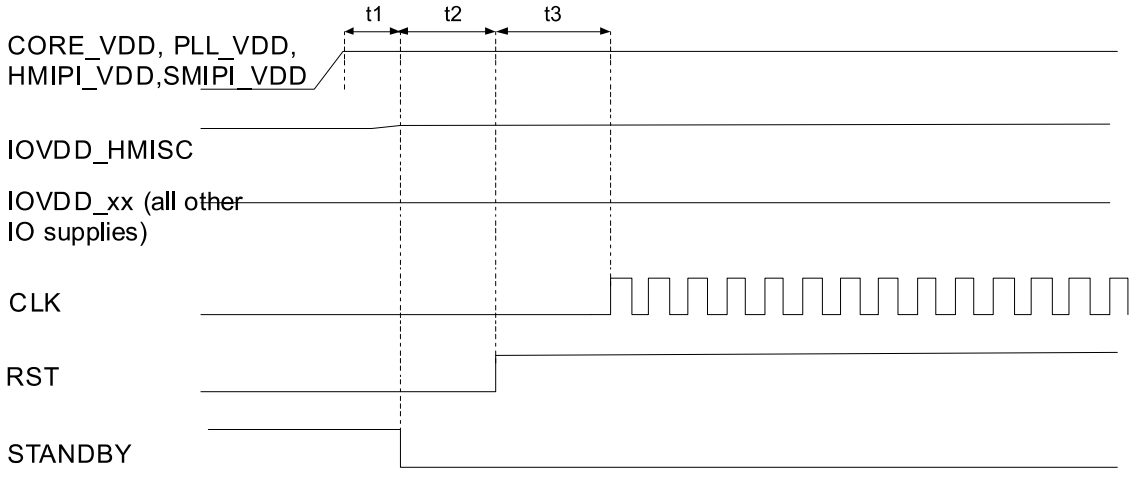


Figure 27. Exiting Core Power Down State

Table 10. EXITING CORE POWER−DOWN SEQUENCE TIMING

Symbol	Parameter	Condition	Value			Unit
			Min	Typ	Max	
t1	CORE_VDD stable to STANDBY off delay	–	200	–	500000	μs
t2	STANDBY off to RESET off delay	–	0	–	500000	μs
t3	RESET off to clock active delay	–	0	–	500000	μs

Standby

The AP1302 enters standby state by asserting STANDBY pin or setting the bit in standby basic register. Both options perform the following actions:

- Reset all the engines inside the AP1302 except the GPIO engine, which stays unaffected (maintaining the GPIO state).
- Gate clocks to all the system in AP1302.

On top of the above, the STANBY pin also latches the state of all GPIOs in IO voltage domain to detach the IO from core power domain (this is to maintain the GPIOs' state during core power down possible, as described in section 2.8). Reference clock to AP1302 can be gated during standby.

Standby does not erase internal memories, so all Basic registers stay unchanged.

GPIOs

GPIOs are used to control various external devices mostly related to sensor modules, like mechanical shutters, optical zoom stepper motors, flash devices and similar. There are 12 GPIOs all together in AP1302, but not all of them are available in all package options. GPIOs are sharing the pins with some external interfaces as listed in Table 11.

Table 11. GPIO 2ND FUNCTION MAPPING

GPIO	6.5 mm package	Reset Value
0	SEN_CLK1	HighZ
1	SHUTDOWN1	0
2	SEN_CLK2	HighZ
3	SHUTDOWN2, SPIS_CLK	HighZ
4	SPIS_MISO	1
5	SPIS_MOSI, 2ND_I2C_SCL	HighZ
6	SPIS_CS, 2ND_I2C_SDA	HighZ
7	SPIM_CLK	1
8	SPIM_MISO	HighZ
9	SPIM_MOSI	0
10	SPIM_CS	1
11	I2C slave address select	HighZ

All GPIOs are capable of PWM functions. There are three independent PWM engines, each can drive four GPIOs. Each engine can have up to eight states, which means it can do a sequence of eight different GPIO states with four GPIOs and it can automatically cycle through the sequence.

GPIOs are not controlled by the FW when they are not used as interfaces in the 2nd function. This way host has direct control over the GPIOs and can set the direction, read the value when GPIO is configured as input, and set the driving value when GPIO is configured as output. This way the host can use GPIOs as the reset signal for sensors, driving LED flash and similar functions independently of the AP1302 FW control loop.

GPIOs have the ability to retain state during Standby and Core power down mode.GPIO also have fail−safe functionality, which means that they will remain in the High−Z state when the chip will be powered down.

GENERAL PHYSICAL SPECIFICATIONS

- Package type: VFBGA
- Package thickness: 0.80 mm (maximum)
- Package size: 6.5 ±0.10mm x 6.5 ±0.10mm
- Ball count: 120
- Ball diameter: 0.3 ±0.05mm
- Min. ball pitch: 0.50mm

Table 12 describes all signals of the AP1302.

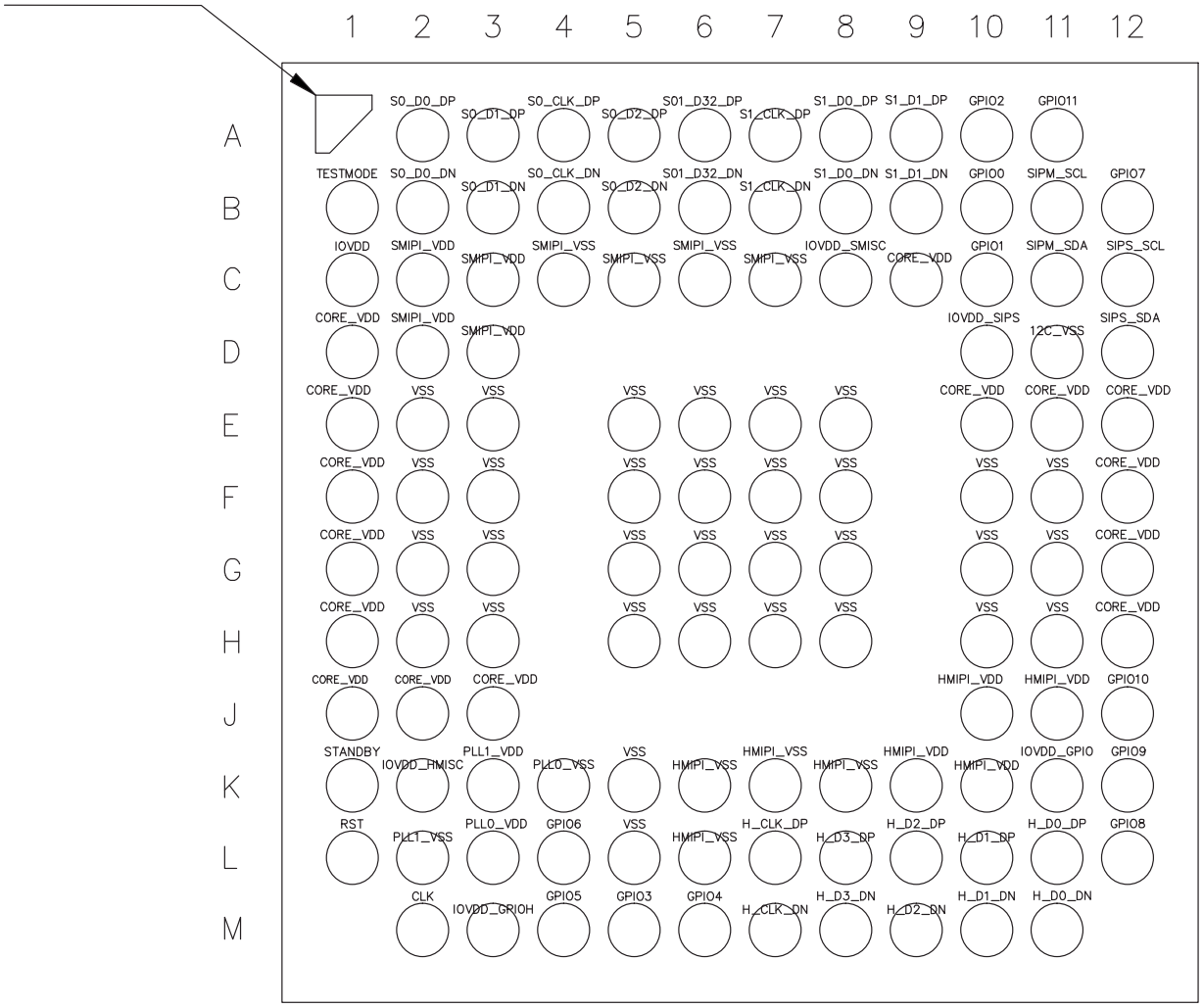
Table 12. SIGNAL DESCRIPTIONS

Signal	Ball Assignments	Direction	Description
CLK	M2	Input	Master Clock input.
RST	L1	Input	System Reset; Asynchronous assertion, synchronously released
TESTMODE	B1	Input	Test mode.
STANDBY	K1	Input	Device Stand−by mode enable input.
S1_D1_DP	A9	Input	Secondary sensor serial interface pixel data input − Lane 1 (sub−LVDS)
S1_D1_DN	B9	Input	Secondary sensor serial interface pixel data input − Lane 1 (sub−LVDS)
S1_CLK_DP	A7	Input	Secondary Sensor Serial Interface Clock input (sub−LVDS)
S1_CLK_DN	B7	Input	Secondary Sensor Serial Interface Clock input (sub−LVDS)
S1_D0_DP	A8	Input	Secondary Sensor Serial Interface Pixel Data input− Lane 0 (sub−LVDS)
S1_D0_DN	B8	Input	Secondary Sensor Serial Interface Pixel Data input t− Lane 0 (sub− LVDS)
S0_CLK_DP	A4	Input	Primary Sensor Serial Interface Clock input (sub−LVDS)
S0_CLK_DN	B4	Input	Primary Sensor Serial Interface Clock input (sub−LVDS)
S0_D0_DP	A2	Input	Primary Sensor Serial Interface Pixel Data input − Lane 0 (sub−LVDS)
S0_D0_DN	B2	Input	Primary Sensor Serial Interface Pixel Data input − Lane 0 (sub−LVDS)
S0_D1_DP	A3	Input	Primary Sensor Serial Interface Pixel Data input − Lane 1 (sub−LVDS)
S0_D1_DN	B3	Input	Primary Sensor Serial Interface Pixel Data input − Lane 1 (sub−LVDS)
S0_D2_DP	A5	Input	Primary Sensor Serial Interface Pixel Data input − Lane 2 (sub−LVDS)
S0_D2_DN	B5	Input	Primary Sensor Serial Interface Pixel Data input − Lane 2 (sub−LVDS)
S01_D32_DP	A6	Input	Primary Sensor Serial Interface Pixel Data input − Lane 3 (sub−LVDS)
S01_D32_DN	B6	Input	Primary Sensor Serial Interface Pixel Data input − Lane 3 (sub−LVDS)
GPIO[11]	A11	Bi−Directional	General Purpose IO
GPIO[10]	J12	Bi−Directional	General Purpose IO
GPIO[9]	K12	Bi−Directional	General Purpose IO
GPIO[8]	L12	Bi−Directional	General Purpose IO
GPIO[7]	B12	Bi−Directional	General Purpose IO
GPIO[6]	L4	Bi−Directional	General Purpose IO
GPIO[5]	M4	Bi−Directional	General Purpose IO
GPIO[4]	M6	Bi−Directional	General Purpose IO
GPIO[3]	M5	Bi−Directional	General Purpose IO
GPIO[2]	A10	Bi−Directional	General Purpose IO
GPIO[1]	C10	Bi−Directional	General Purpose IO
GPIO[0]	B10	Bi−Directional	General Purpose IO

Table 12. SIGNAL DESCRIPTIONS

Signal	Ball Assignments	Direction	Description
VSS	E2, E3, E5, E6, E7, E8, F2, F3, F5, F6, F7, F8, F10, F11, G2, G3, G5, G6, G7, G8, G10, G11, H2, H3, H5, H6, H7, H8, H10, H11, L5, K5	Power	Common Ground
HMIPI_VSS	K6, K7, K8, L6	Power	Host MIPI VSS
SMIPI_VSS	C4, C5, C6, C7	Power	Sensor MIPI VSS
PLL0_VSS	K4	Power	PLL0 VSS
PLL1_VSS	L2	Power	PLL1 VSS
CORE_VDD	C9, D1, E1, E10, E11, E12, F1, F12, G1, G12, H1, H12, J1, J2, J3	Power	Core Power Supply; 1.2 V Nominal
PLL0_VDD	L3	Power	PLL0 Power Supply; 1.2 V Nominal
PLL1_VDD	K3	Power	PLL1 Power Supply; 1.2V Nominal
IOVDD_GPIOH	M3	Power	GPIO Power; 1.8V Nominal
IOVDD_GPIO	K11	Power	GPIO Power, 1.8V nominal
IOVDD_HMISC	K2	Power	Host MISC Interface IO Power; 1.8V Nominal
SMIPI_VDD	C2, C3, D2, D3	Power	Sensor MIPI Power Supply; 1.2 V Nominal
IOVDD_SMISC	C8	Power	Sensor MISC interface IO Power; 1.8VNominal
IOVDD	C1	Power	IO Power Supply 1.8 V Nominal
HMIPI_VDD	J10, J11, K9, K10	Power	Host MIPI IO Power Supply, 1.2 V Nominal
H_CLK_DP	L7	Output	Host Serial Interface Clock Output
H_CLK_DN	M7	Output	Host Serial Interface Clock Output
H_D0_DP	L11	Output	Host Serial Interface Data Output − Lane 0
H_D0_DN	M11	Output	Host Serial Interface Data Output − Lane 0
H_D1_DP	L10	Output	Host Serial Interface Data Output − Lane 1
H_D1_DN	M10	Output	Host Serial Interface Data Output − Lane 1
H_D2_DP	L9	Output	Host Serial Interface Data Output − Lane 2
H_D2_DN	M9	Output	Host Serial Interface Data Output − Lane 2
H_D3_DP	L8	Output	Host Serial Interface Data Output − Lane 3
H_D3_DN	M8	Output	Host Serial Interface Data Output − Lane 3
SIPS_SCL	C12	Bi−Directional	I2C Slave Clock
SIPS_SDA	D12	Bi−Directional	I2C Slave Data
SIPM_SCL	B11	Bi−Directional	I2C Master Clock
SIPM_SDA	C11	Bi−Directional	I2C Master Data
I2C_VSS	D11	Power	I2C Slave ground
IOVDD_SIPS	D10	Power	I2C Slave interface IO Power; 1.8 V Nominal
Total 120 Balls			

4. IO power supplies are grouped.
   Group A.) IOVDD_SIPS
   Group B.) IOVDD, IOVDD_HOST and IOVDD_HMISC Group C.) IOVDD_GPIO, IOVDD_GPIOH, IOVDD_SMISC.
5. The supplies within one group should have the same voltage, but can be powered on/off independently.



Top View

Figure 28. Ball Map Diagram

POWER CONSUMPTION MEASUREMENT SUMMARY

Table 13. TYPICAL POWER CONSUMPTION

Ambient Temperature	25°C	Sensor Paired	1.2V Current (mA)	1.8 Current (mA)	Power (mW)
Modes					
8M_30_FPS_JPEG		8M	443	3	536
8M_30_FPS_Bayer		8M	413	3	500
8M_15_FPS_JPEG		8M	276	3	336
8M_15_FPS_Bayer		8M	256	3	312
8M_1080p_30_FPS_HQ		8M	357	3	433
8M_1080p_30_FPS_HQ_Bayer		8M	340	3	413
8M_720p_30_FPS_LQ		8M	180	3	221
8M_720p_30_FPS_LQ_Bayer		8M	161	3	197
8M_1080p_15_FPS_HQ		8M	179	3	219

Table 13. TYPICAL POWER CONSUMPTION

Ambient Temperature	25°C	Sensor Paired	1.2V Current (mA)	1.8 Current (mA)	Power (mW)
Modes					
8M_1080p_15_FPS_HQ_Bayer		8M	159	3	196
8M_VGA_30_FPS_LQ		8M	130	3	160
8M_VGA_30_FPS_LQ_Bayer		8M	126	3	155
13M_30_FPS_JPEG		13M	645	3	778
13M_30_FPS_Bayer		13M	598	3	722
13M_15_FPS_JPEG		13M	385	3	466
13M_15_FPS_Bayer		13M	350	3	425
13M_1080p_30_FPS_HQ_JPEG		13M	498	3	603
13M_1080p_30_FPS_HQ_Bayer		13M	496	3	600
13M_720p_30_FPS_LQ_JPEG		13M	223	3	273
13M_720p_30_FPS_LQ_Bayer		13M	207	3	253
13M_1080p_60_FPS_LQ_JPEG		13M	337	3	409
13M_1080p_60_FPS_LQ_Bayer		13M	308	3	375
13M_1080p_30_FPS_LQ_JPEG		13M	225	3	275
13M_1080p_30_FPS_LQ_Bayer		13M	205	3	251
13M_VGA_30_FPS_LQ_JPEG		13M	145	3	179
13M_VGA_30_FPS_LQ_Bayer		13M	141	3	174
LQ: sensor does binning					
HQ: near−full size from sensor, ICP3 does scaling					
Standby Power Consumption					<10

ELECTRICAL CHARACTERISTICS

DC Electrical Characteristics

Table 14. DC ELECTRICAL DEFINITIONS AND CHARACTERISTICS

Power Source	Standard Voltage	Minimum Voltage	Typical Voltage	Maximum Voltage	Unit
CORE_VDD	Core digital circuit power supply	1.14	1.2	1.26	V
SMIPI_VDD	Power for Sensor MIPI interface	1.14	1.2	1.26	V
PLL_VDD	PLL circuit power supply	1.14	1.2	1.26	V
IOVDD_SMISC	Power for sensor I/O interfaces	1.62	1.8	1.98	V
IOVDD_HMISC	Power for host I/O interfaces	1.62	1.8	1.98	V
IOVDD_GP_3_0	Power for GPIO interfaces	1.62	1.8	1.98	V
IOVDD_SIPS	Power for the I^2C interfaces	1.62	1.8	1.98	V
IOVDD_HOST	Power for the host I/O interfaces	1.62	1.8	1.98	V
IOVDD	Power for general I/O interfaces	1.62	1.8	1.98	V
HMIPI_VDD	Power for the Host MIPI interfaces	1.14	1.2	1.26	V

Table 15. ABSOLUTE MAXIMUM RATINGS

Parameter	Value	Unit
Operating temperature (ambient)	−30 to +70	°C
Operating temperature (junction)	−30 to +85	°C
Storage temperature	−50 to +150	°C
Max voltage (1.2 V domain)	1.6	V
Max voltage (1.8 V domain)	3.6	V

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

Master Clock Input (MCLK) Characteristics

Table 16. MCLK CHARACTERISTICS

Symbol	Parameter	Min	Max	Unit
Fextclk	Input Clock Freq	10	54	MHz
I_{IH}	Input High Leakage Current	0	5	μA
V_{IH}	Input High Voltage	0.7*IOV_{DD}	–	V
V_{IL}	Input Low Voltage	–	0.3*IOVDD	V

Table 17. DC ELECTRICAL CHARACTERISTICS OF CONTROL SIGNALS (RST, STANDBY, CLK)

Symbol	Parameter	Min	Typ	Max	Unit
VIH	Input High voltage	0.7*IOVDD_HMISC			V
VIL	Input Low voltage			0.3*IOVDD_HMISC	V

I²C Signal AC/DC Specifications

Table 18. I²C SIGNAL SPECIFICATIONS

Parameter	Description	Min	Max	Unit
fSCL	SCL clock frequency	100	400	kHz
tSDHR	Data hold time	0	–	ns
tSDSR	Data setup time	0	–	ns
tSRTH	Setup time for start condition	0.6	–	us
tr_sclk	SCLK rise time	20	300	ns
tf_sclk	SCLK fall time	100	300	ns
tr_sdata	SDATA rise time	20	300	ns
tf_sdata	SDATA fall time	100	300	ns
VOL	Low level data output voltage	–	0.2 * IOVDD	V
VIL_sclk	SCLK input low voltage	–	0.3 * IOVDD	V
VIH_sclk	SCLK input high voltage	0.7 * IOVDD	–	V
VIL_sdata	SDATA input low voltage	–	0.3 * IOVDD	V
VIH_sdata	SDATA input high voltage	0.7 * IOVDD	–	V

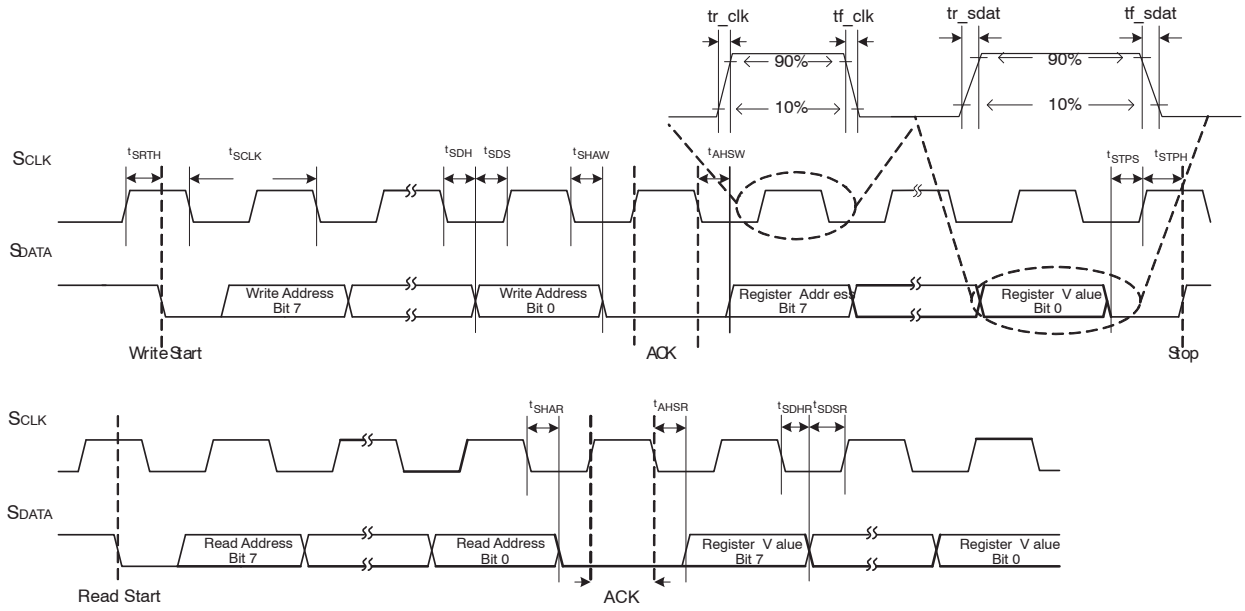


Figure 29. Two Wire Serial Bus Timing Parameters

Table 19. FOR I²C MASTER BUS READ AND WRITE MODES

Measurement conditions: fSCL = 400 kHz; IOVDD=1.8 V; CORE_VDD=1.2 V; T_J = 55°C; Output load < 100 pF

Parameter	Description	Min	Typical	Max	Unit
fSCL	SCL clock frequency	100	400	400	kHz
tLOW	Low period of SCL		1.26	–	µs
tHIGH	High period of SCL		1.23	–	µs
tF	Fall time of SDA and SCL		2.34	–	ns

Table 19. FOR I²C MASTER BUS READ AND WRITE MODES

Measurement conditions: fSCL = 400 kHz; IOVDD=1.8 V; CORE_VDD=1.2 V; T_J = 55°C; Output load < 100 pF

Parameter	Description	Min	Typical	Max	Unit
tR	Rise time of SDA and SCL		91.41	–	ns
VHU:DAT	Data hold time	0	344.20	–	ns
VSU:DAT	Data setup time	100	854.87	–	ns

MIPI Electrical Characterization

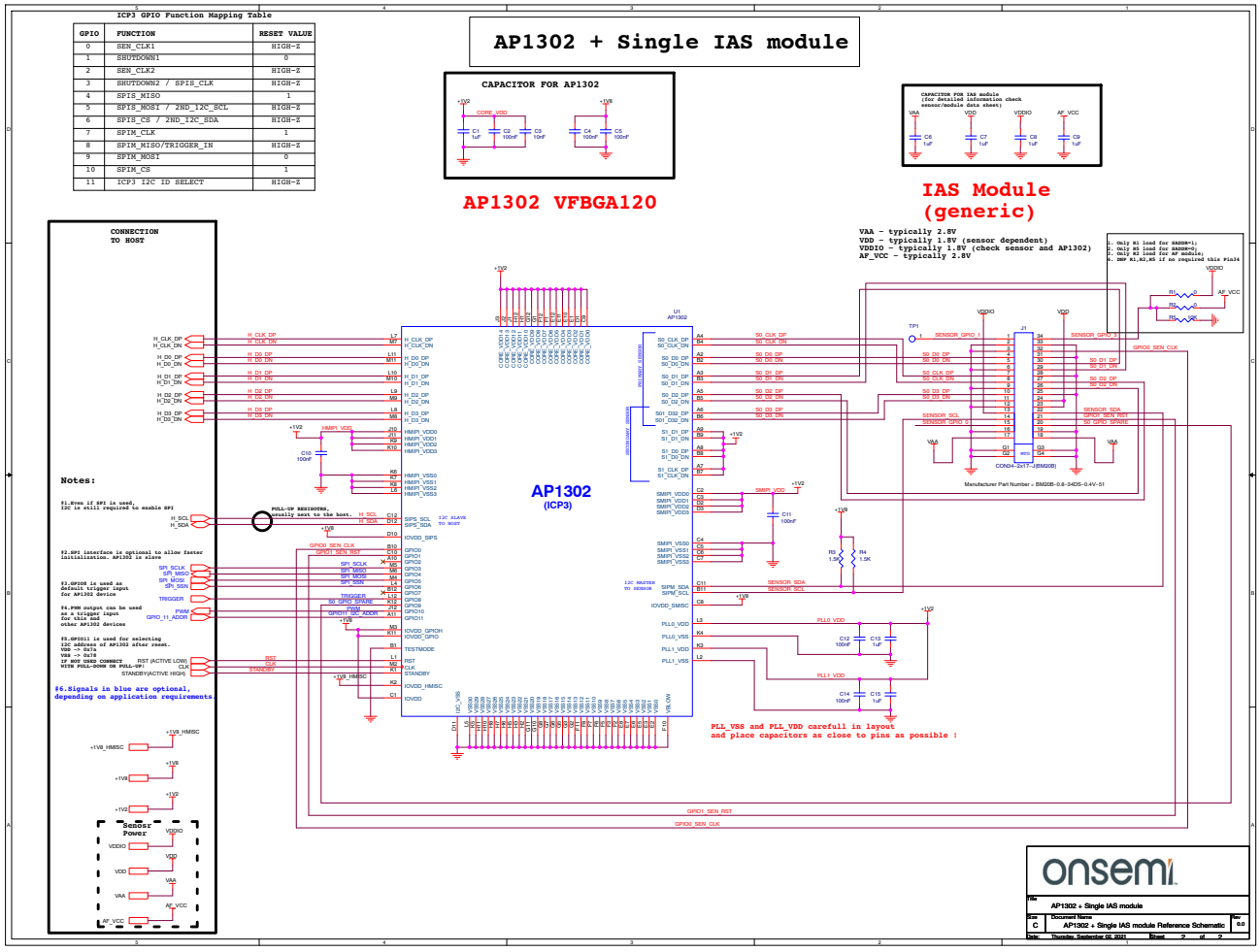
Table 20. MIPI ELECTRICAL CHARACTERIZATION RESULTS

Parameter	Notes	Typ	Unit
VCMTX	HS transmit static common−mode voltage	195	mV
$\lvert \Delta V_{CMTX}(1,0) \rvert$	V_{CMTX} mismatch when output is differential−1 or differential−0	1	mV
$\lvert V_{OD} \rvert$	HS transmit differential voltage	195	mV
$\lvert \Delta V_{OD} \rvert$	V_{OD} mismatch when output is differential−1 or differential−0	5	mV
VOHHS	HS output high voltage	295	mV
ZOS	Single ended output impedance	52	Ω
ΔZ_{OS}	Single ended output impedance mismatch	5	%
ΔVCMTX(HF)	Common level variations 50~450 MHz	5	mVRMS
Δ VCMTX(LF)	Common level variations above 450 MHz	8	mVPEAK
t_R	HS: 20%~80% rise time	350	ps
t_F	HS: 20%~80% fall time	350	ps
TRLP/TFLP	LP: 15%~85% Rise time and fall time	21	ns
TREOT	LP: 30%~85% Rise time and fall time	27	ns
TLP−PULSE−TX	Pulse width of the first LP exclusive−OR clock	170	ns
TLP−PULSE−TX	Pulse width of others	170	ns
TLP−PER−TX	Period of the LP exclusive−OR clock	360	ns

NOTE: MIPI timing diagrams will be supplied. For definitions of these signals, please look up in the MIPI specifications.

APPENDIX A: AP1302 REFERENCE SCHEMATICS USING STANDARD IMAGER ACCESS SYSTEM (IAS) MODULE SENSOR FOR REFERENCE

Note: Check your sensor part documentation for details on sensor schematics.



GPIO	FUNCTION	RESET VALUE
0	SEN_CLK1	HIGH-Z
1	SHUTDOWN1	0
2	SEN_CLK2	HIGH-Z
3	SHUTDOWN2 / SPIS_CLK	HIGH-Z
4	SPIS_MISO	1
5	SPIS_MOSI / 2ND_I2C_SCL	HIGH-Z
6	SPIS_CS / 2ND_I2C_SDA	HIGH-Z
7	SPIM_CLK	1
8	SPIM_MISO/TRIGGER_IN	HIGH-Z
9	SPIM_MOSI	0
10	SPIM_CS	1
11	ICP3 I2C ID SELECT	HIGH-Z

Figure 30. AP1302 Reference Schematics (using standard single IAS (Imager Access System) connector for reference)

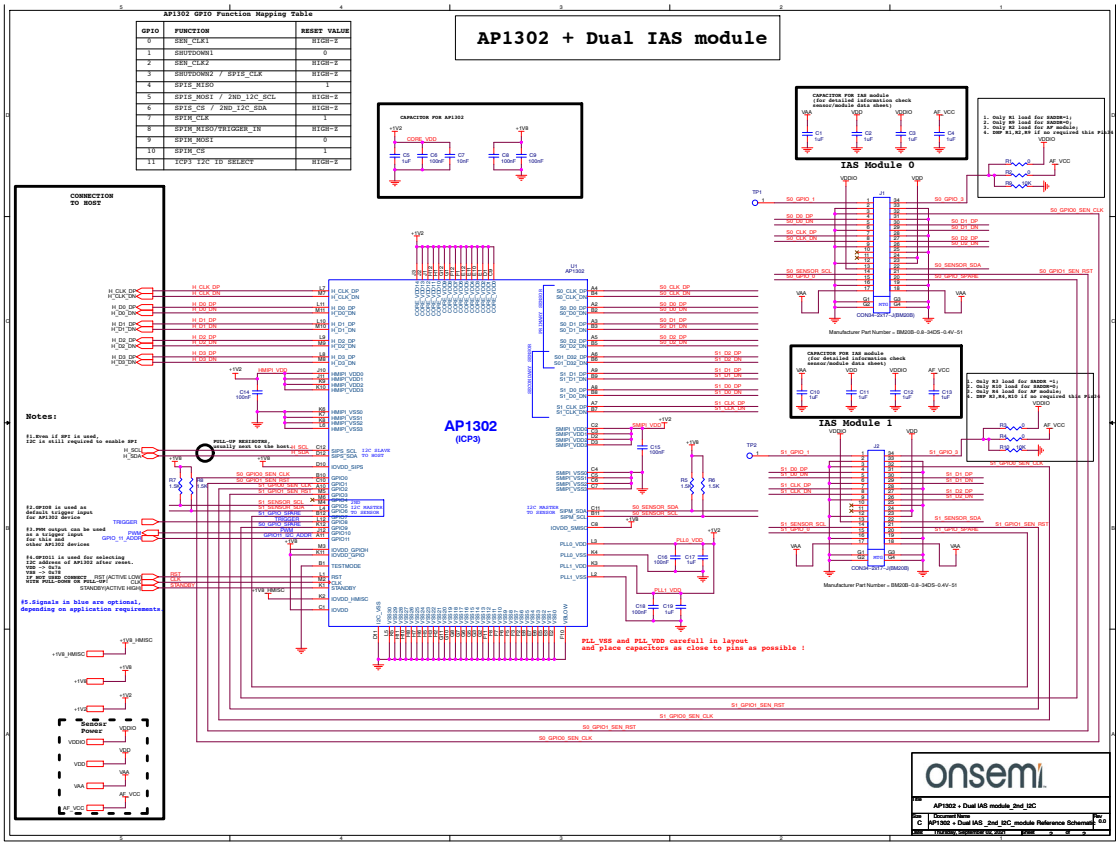
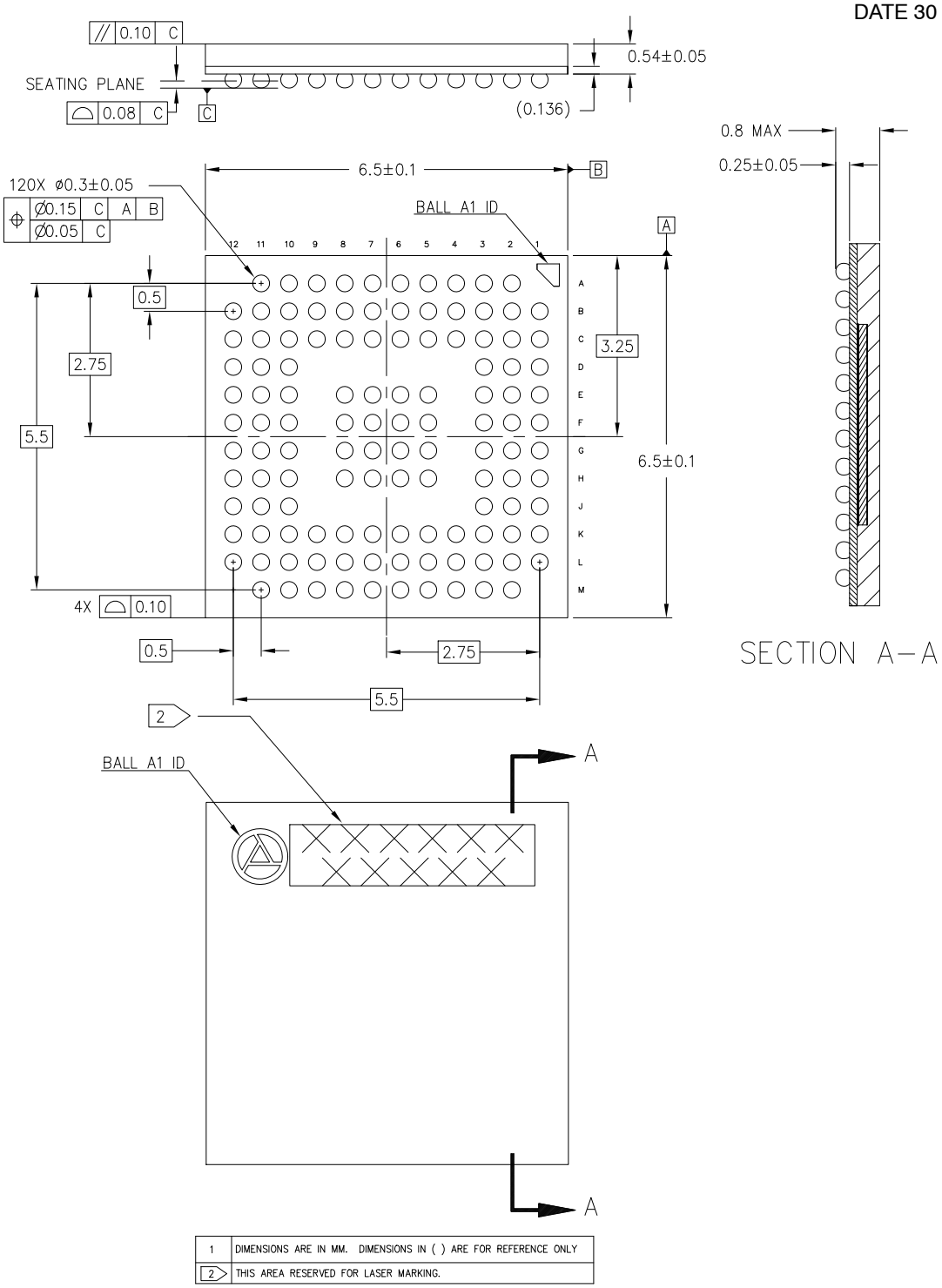


Figure 31. AP1302 Reference Schematics (using dual standard IAS (Imager Access System) connectors for reference)

VFBGA120 6.5x6.5
CASE 138AK
ISSUE O

DATE 30 DEC 2014



1	DIMENSIONS ARE IN MM. DIMENSIONS IN ( ) ARE FOR REFERENCE ONLY
2	THIS AREA RESERVED FOR LASER MARKING.

DOCUMENT NUMBER:	**98AON93702F**	Electronic versions are uncontrolled except when accessed directly from the Document Repository. Printed versions are uncontrolled except when stamped "CONTROLLED COPY" in red.
DESCRIPTION:	**VFBGA120 6.5X6.5**	**PAGE 1 OF 1**

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Email Requests to: orderlit@onsemi.com

onsemi Website: www.onsemi.com

TECHNICAL SUPPORT
North American Technical Support:
Voice Mail: 1 800−282−9855 Toll Free USA/Canada
Phone: 011 421 33 790 2910

Europe, Middle East and Africa Technical Support:
Phone: 00421 33 790 2910
For additional information, please contact your local Sales Representative

◊